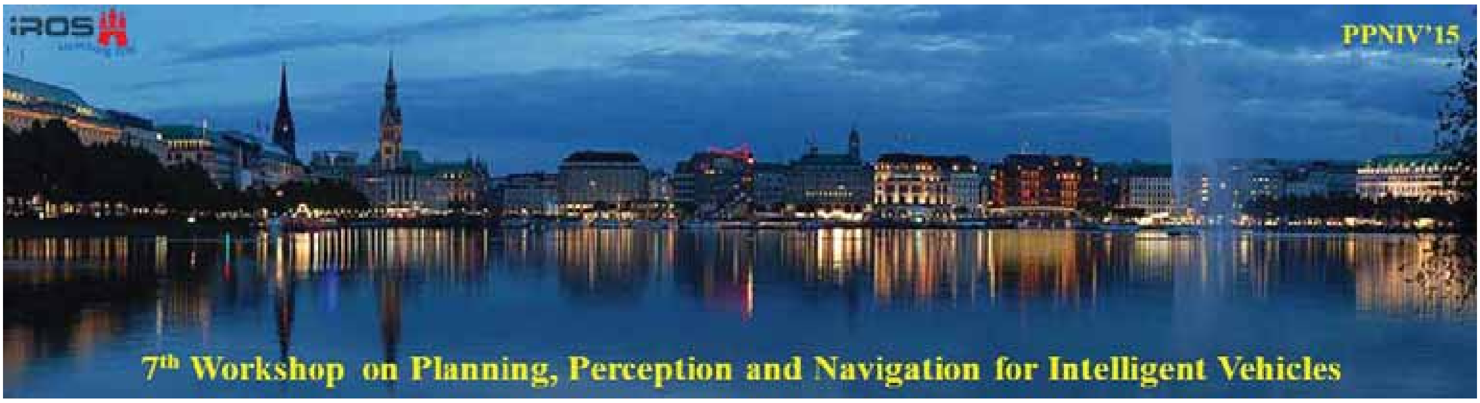

iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

IROS15

7th International workshop on

Planning, Perception and Navigation for Intelligent Vehicles

Full Day Workshop

September 28th, 2015 Hamburg, Germany

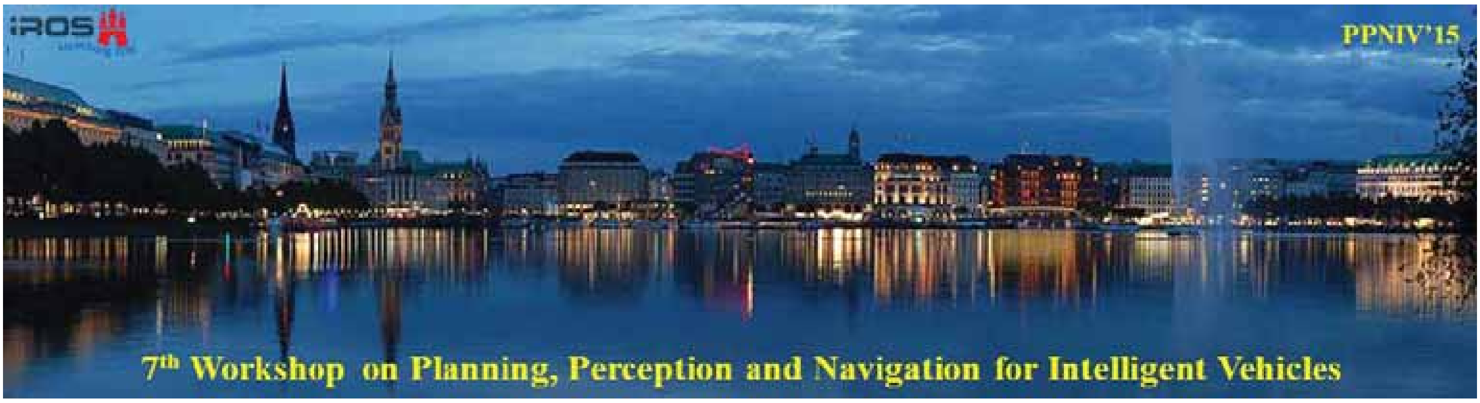
http://ppniv15.irccyn.ec-nantes.fr/

Organizers

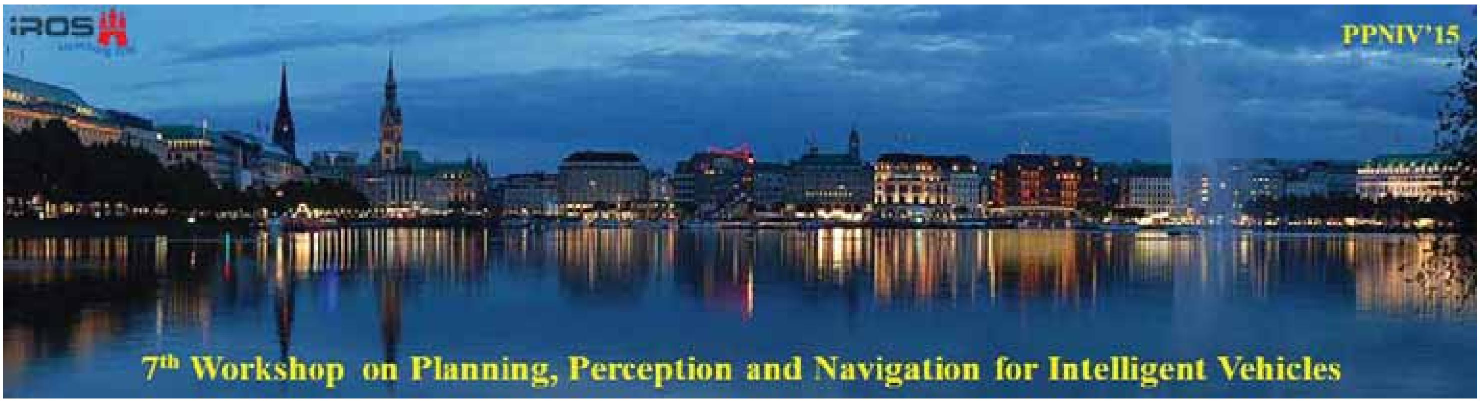
Pr Christian Laugier (INRIA, France),
Pr Philippe Martinet (IRCCYN, France),
Pr Urbano Nunes (ISR, Portugal)
Pr Christoph Stiller (KIT, Germany)

Contact

Professor Philippe Martinet
Ecole Centrale de Nantes,
IRCCyN-CNRS Laboratory
1 rue de la Noë, BP 92101, 44321 Nantes Cedex - FRANCE
Phone: +33 240 376 975, Sec : +33 240 376 934, Fax : +33 240 376 6930
Email: Philippe.Martinet@irccyn.ec-nantes.fr
Home page: http://www.irccyn.ec-nantes.fr/~martinet


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Foreword

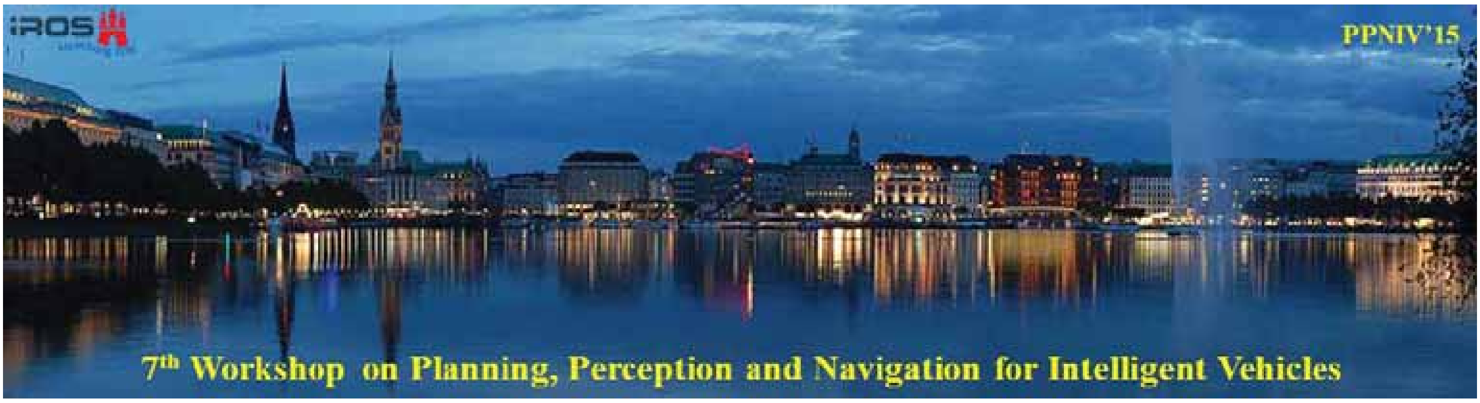
The purpose of this workshop is to discuss topics related to the challenging problems of autonomous navigation and of driving assistance in open and dynamic environments. Technologies related to application fields such as unmanned outdoor vehicles or intelligent road vehicles will be considered from both the theoretical and technological point of views. Several research questions located on the cutting edge of the state of the art will be addressed. Among the many application areas that robotics is addressing, transportation of people and goods seem to be a domain that will dramatically benefit from intelligent automation. Fully automatic driving is emerging as the approach to dramatically improve efficiency while at the same time leading to the goal of zero fatalities. This workshop will address robotics technologies, which are at the very core of this major shift in the automobile paradigm. Technologies related to this area, such as autonomous outdoor vehicles, achievements, challenges and open questions would be presented. Main topics include: Road scene understanding, Lane detection and lane keeping, Pedestrian and vehicle detection, Detection, tracking and classification, Feature extraction and feature selection, Cooperative techniques, Collision prediction and avoidance, Advanced driver assistance systems, Environment perception, vehicle localization and autonomous navigation, Real-time perception and sensor fusion, SLAM in dynamic environments, Mapping and maps for navigation, Real-time motion planning in dynamic environments, Human-Robot Interaction, Behavior modeling and learning, Robust sensor-based 3D reconstruction, Modeling and Control of mobile robot.

Previously, several workshops were organized in the near same field. The 1st edition PPNIV'07 of this workshop was held in Roma during ICRA'07 (around 60 attendees), the second PPNIV'08 was in Nice during IROS'08 (more than 90 registered people), the third PPNIV'09 was in Saint-Louis (around 70 attendees) during IROS'09, the fourth edition PPNIV'12 was in Vilamoura (over 95 attendees) during IROS'12, the fifth edition PPNIV'13 was in Vilamoura (over 135 attendees) during IROS'13, and the sixth edition PPNIV'14 was in Chicago (over 100 attendees) during IROS14.

In parallel, we have also organized SNODE'07 in San Diego during IROS'07 (around 80 attendees), MEPPC08 in Nice during IROS'08 (more than 60 registered people), SNODE'09 in Kobe during ICRA'09 (around 70 attendees), RITS'10 in Anchrorage during ICRA'10 (around 35 attendees), PNAVHE11 in San Francisco during the last IROS11 (around 50 attendees), and the last one WMEPC14 in Hong Kong during the last ICRA14 (around 65 attendees),

This workshop is composed with 4 invited talks and 15 selected papers (10 selected for oral presentation and 5 selected for interactive session. Five sessions have been organized:

- Session I: Localization & Mapping
- Session II: Perception & Situation awareness
- Session III: Interactive session
- Session IV: Planning & Navigation
- Session V: Sensing

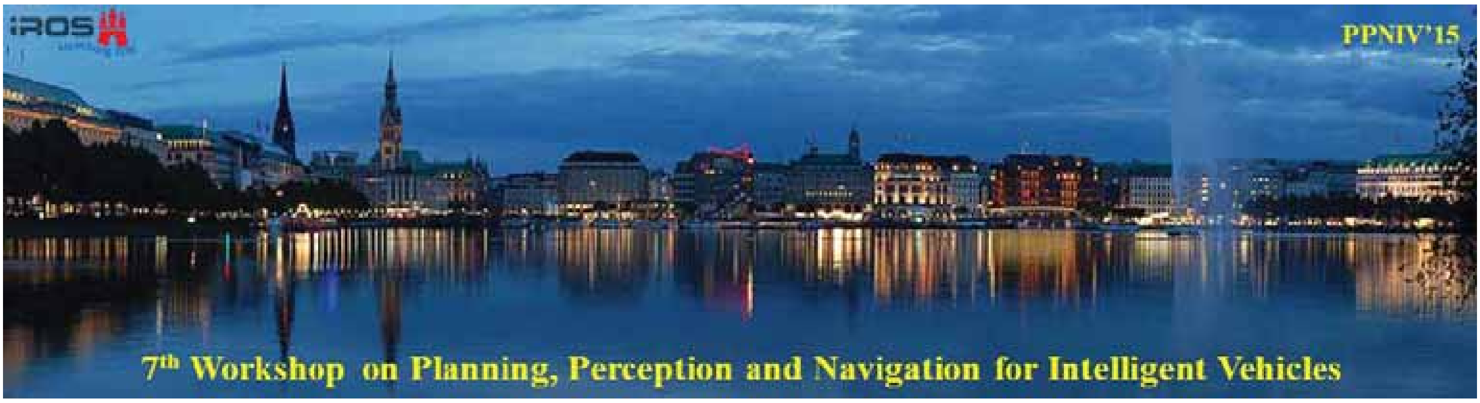

iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Intended Audience concerns researchers and PhD students interested in mobile robotics, motion and action planning, robust perception, sensor fusion, SLAM, autonomous vehicles, human-robot interaction, and intelligent transportation systems. Some peoples from the mobile robot industry and car industry are also welcome.

This workshop is made in relation with IEEE RAS: RAS Technical Committee on "Autonomous Ground Vehicles and Intelligent Transportation Systems" (http://tab.ieee-ras.org/).

Christian Laugier, Philippe Martinet, Urbano Nunes and Christoph stiller


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session I

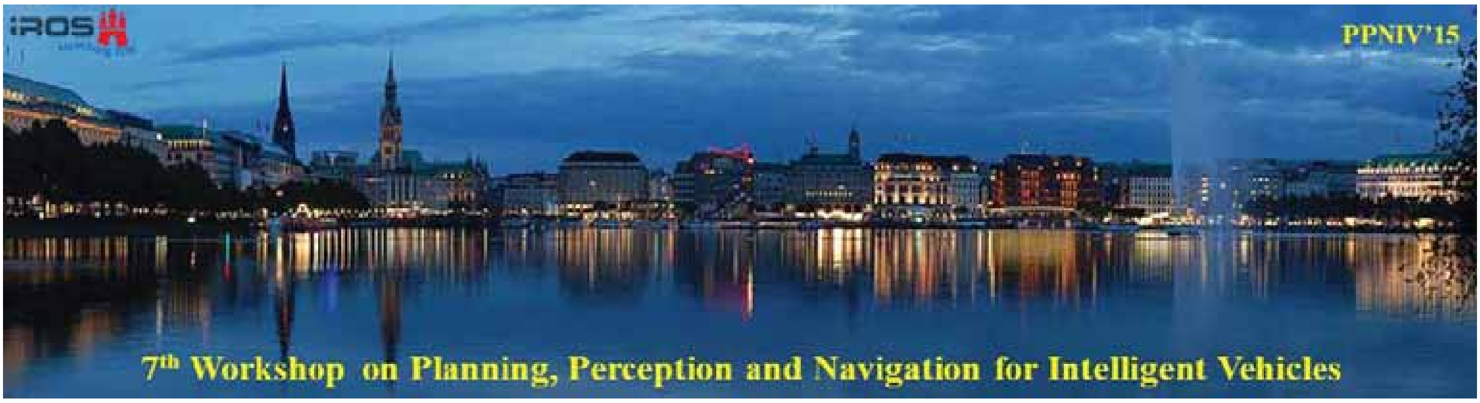
Localization & Mapping

- **Keynote speaker: Philippe Bonnifait (Heudiasyc, France)**
  Title: Autonomous Integrity Monitoring of Navigation Maps on board Vehicles

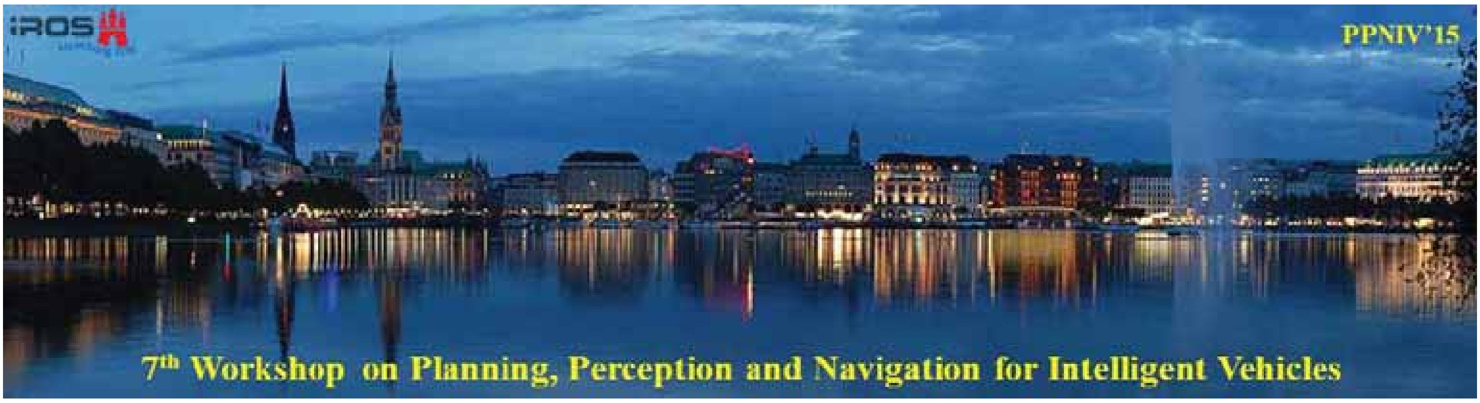
- **Title: Collaborative Visual SLAM Framework for a Multi-Robot System**
  Authors: Nived Chebrolu, David Marquez-Gamez and Philippe Martinet

- **Title: Improving Vision-based Topological Localization by Combining Local and Global Image Features**
  Authors: Shuai Yang and Han Wang

- **Title: PML-SLAM: a solution for localization in large-scale urban environments**
  Authors: S. Alsayed, G. Bresson, F. Nashashibi, A. Verroust Blondet


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems


IROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session I

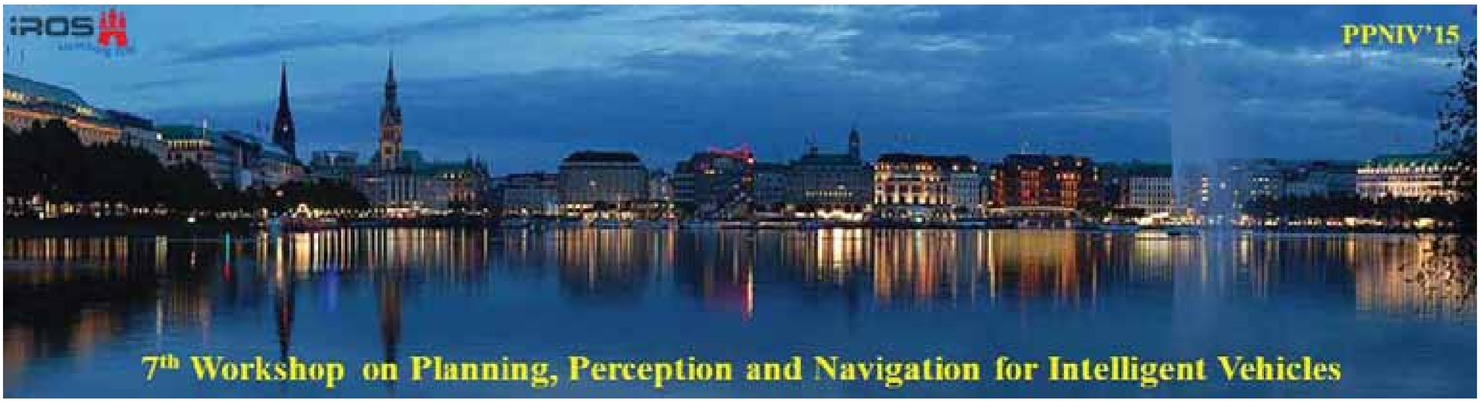
Keynote speaker: **Philippe Bonnifait**
(Heudiasyc, France)

Autonomous Integrity Monitoring of Navigation Maps on board Vehicles

Abstract: This talk addresses the problem of monitoring navigation systems on board passenger vehicles by using a Fault Detection, Isolation, and Adaptation (FDIA) paradigm. The aim is to prevent malfunctions in systems such as advanced driving assistance systems and autonomous driving functions that use data provided by the navigation system. The integrity of the estimation of the vehicle position provided by the navigation system is continuously monitored and assessed. The proposed approach uses an additional estimate of vehicle position that is independent of the navigation system and based on data from standard vehicle sensors. First, fault detection consists in comparing the two estimates using a sequential statistical test to detect discrepancies despite the presence of noise. Second, fault isolation and adaptation is introduced to identify faulty estimates and to provide a correction where necessary. The FDIA framework presented here utilizes repeated trips along the same roads as a source of redundancy. Relevant properties of this formalism are given and verified experimentally using an equipped vehicle in rural and urban conditions and with various map faults. Real results show that sequential FDIA performed well, even in difficult GNSS conditions.

Biography: Philippe Bonnifait is a professor in the Computer Science and Engineering department of the Université de Technologie de Compiègne (UTC) in France. He obtained his Ph.D. in automatic control and computer science at the École Centrale de Nantes in 1997. Since 1998, he has been with Heudiasyc UMR 7253, a joint research unit between UTC and CNRS. His research interests are Intelligent Vehicles, high integrity positioning and map-matching for mobile robot navigation in structured outdoor environments.

The attached paper is going to be published in IEEE Transactions on Intelligent Transportation Systems.


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Autonomous Integrity Monitoring of Navigation Maps on board Vehicles

Philippe Bonnifait

Professor at the
Université de Technologie de Compiègne
Heudiasyc UMR 7253 CNRS, France

In collaboration with Clément Zinoune and Javier Ibanez-Guzman
Renault S.A.S.

PPNIV 2015, Hamburg, 28 September 2015

Outline

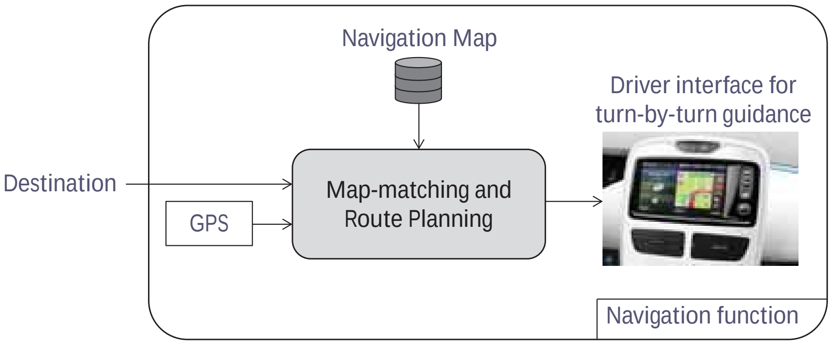
Context and Problem Statement

Fault Detection Isolation and Adaptation Principles

Adaptation to noisy data

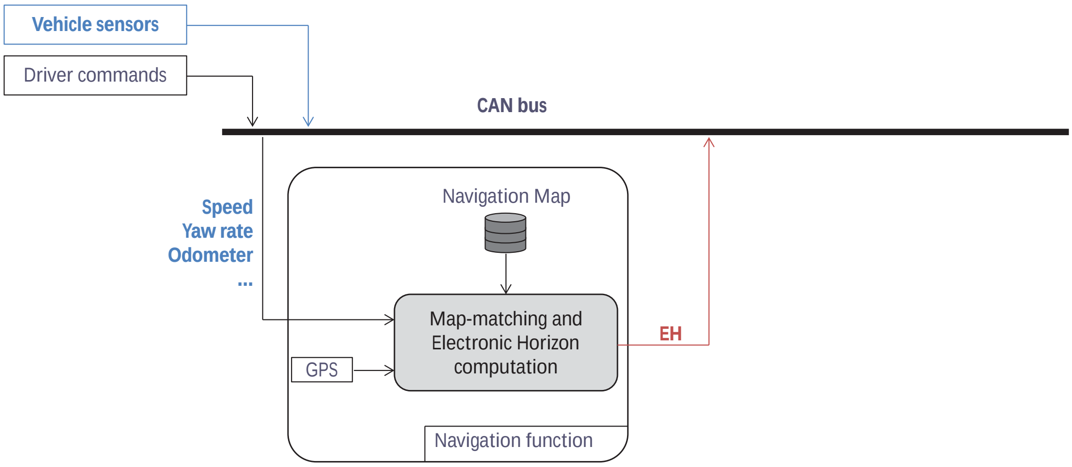
Conclusions and Perspectives

Turn-by-turn navigation system

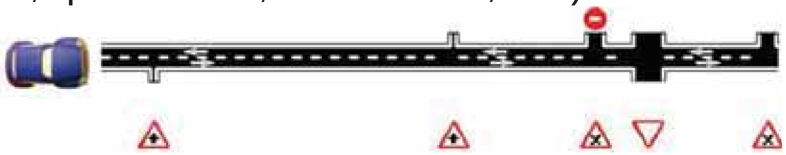

Navigation Map
Driver interface for turn-by-turn guidance
Destination
GPS
Map-matching and Route Planning
Navigation function

Map-Aided ADAS

Example: Intersection Warning

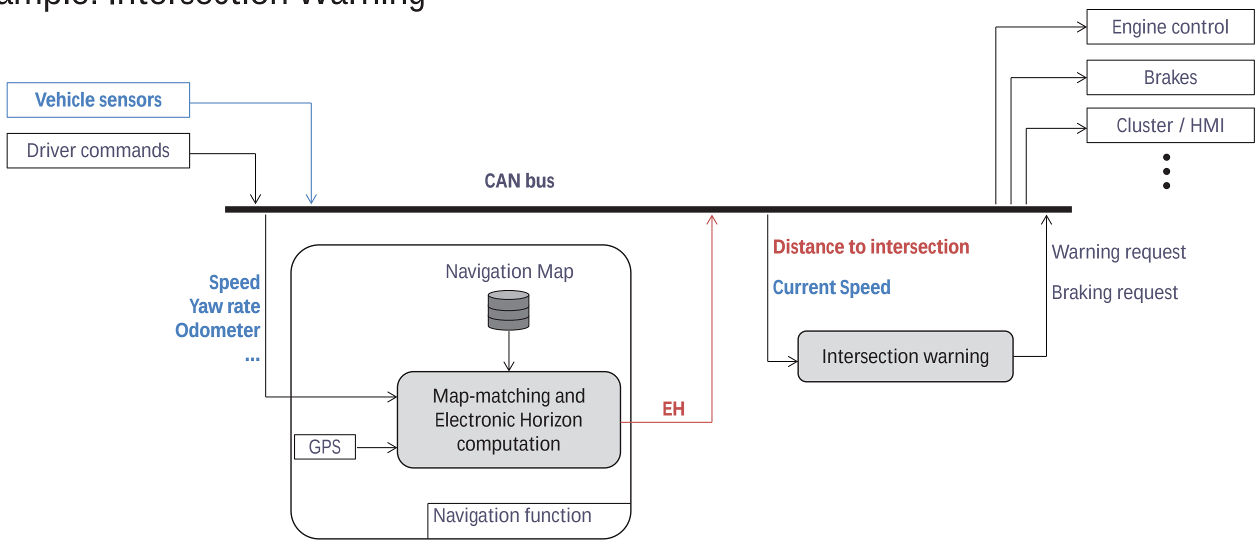


- Electronic Horizon (EH): representation of oncoming context events (e.g., curve, speed limits, intersection, etc.)



Map-Aided ADAS

Example: Intersection Warning


Vehicle sensors

Driver commands

CAN bus

Engine control

Brakes

Cluster / HMI

Speed
Yaw rate
Odometer
...

Navigation Map

Map-matching and Electronic Horizon computation

GPS

Navigation function

EH

Distance to intersection

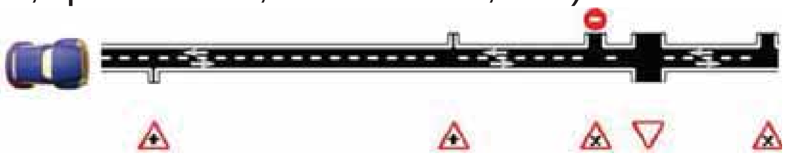
Current Speed

Intersection warning

Warning request

Braking request

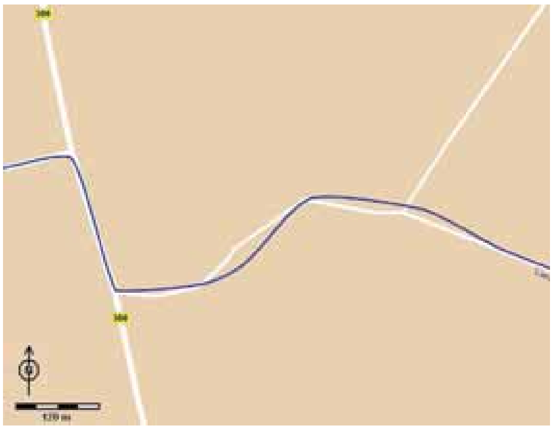
- Electronic Horizon (EH): representation of oncoming context events (e.g., curve, speed limits, intersection, etc.)



Problem Statement

- Map errors may be due to:
  - Errors during the mapping process.
  - Evolution of road network.

- What happens if the map is wrong ?
  - Uncomfortable and unsafe situations.
  - Repetitive ADAS malfunctions.

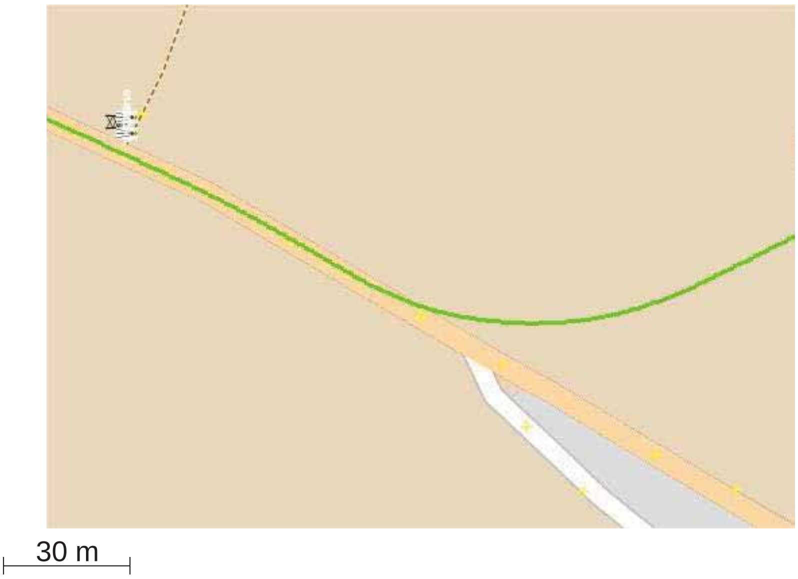


Curve warning system

Navigation map

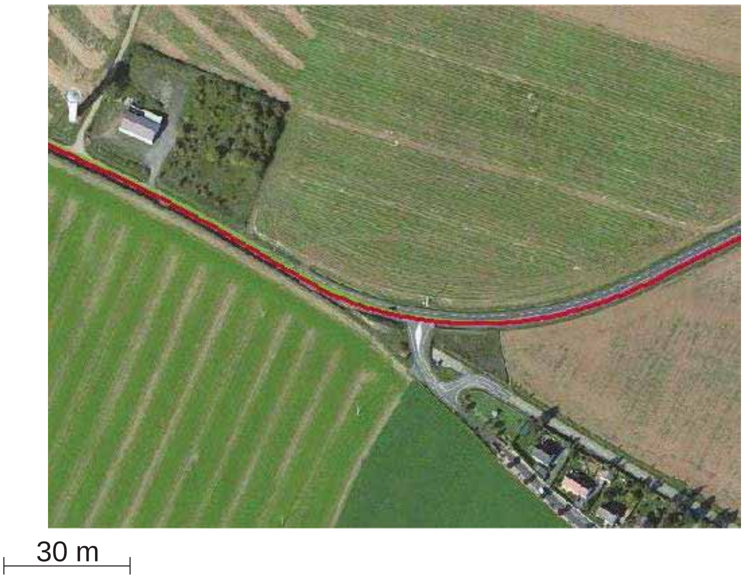
Curve warning system

GPS logs on top of the vehicle navigation map



Curve warning system

Missed detection of the road bend.



Problem Statement

1. Evaluate navigation system integrity in real-time.
2. Provide a correction when necessary.
3. Use only on board vehicle sensors.

Outline

Context and Problem Statement

Fault Detection Isolation and Adaptation

System architecture

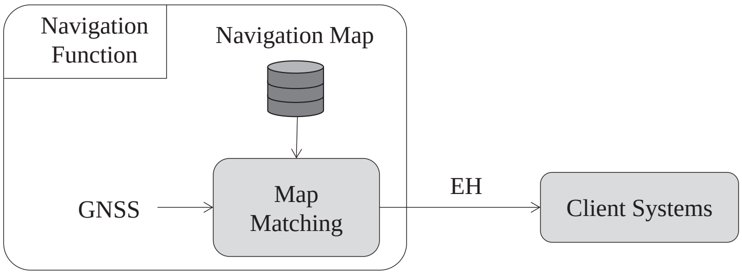
Methods for structural and geometrical faults

Experimental results

Adaptation to noisy data

Conclusions and Perspectives

Autonomous integrity monitoring

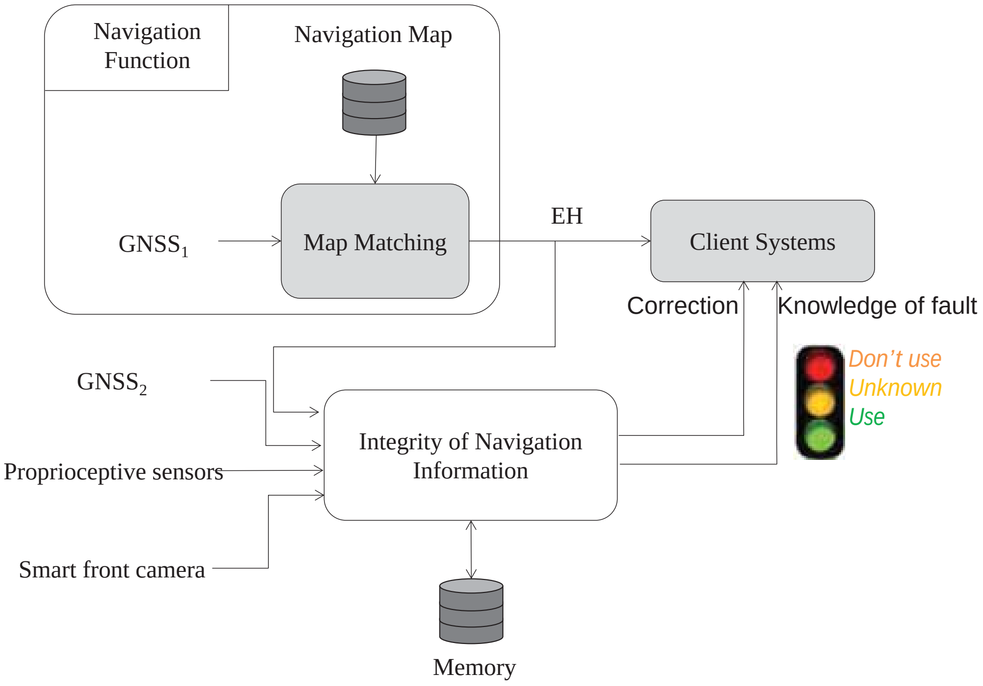


Ability of the vehicle to assess the confidence associated to navigation information using **redundant** information from on board sensors.

➔ every trip on the same road adds redundancy

To provide a reliable confidence indicator to avoid client systems malfunctions.

Architecture for Autonomous Integrity Monitoring

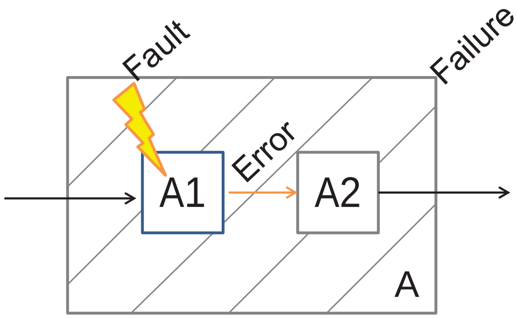


Definitions

Fault: Error generative process.

Error: Discrepancy between measured value and true value.

Failure: Time when a function exceeds the acceptable value.



Case of Navigation

Fault:

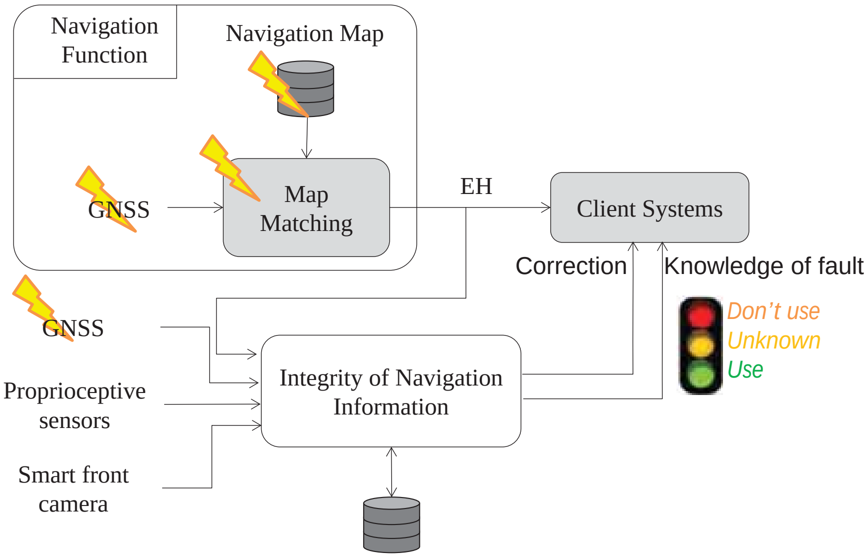
GNSS multipath;

Wrong road candidate selected by map-matching ;

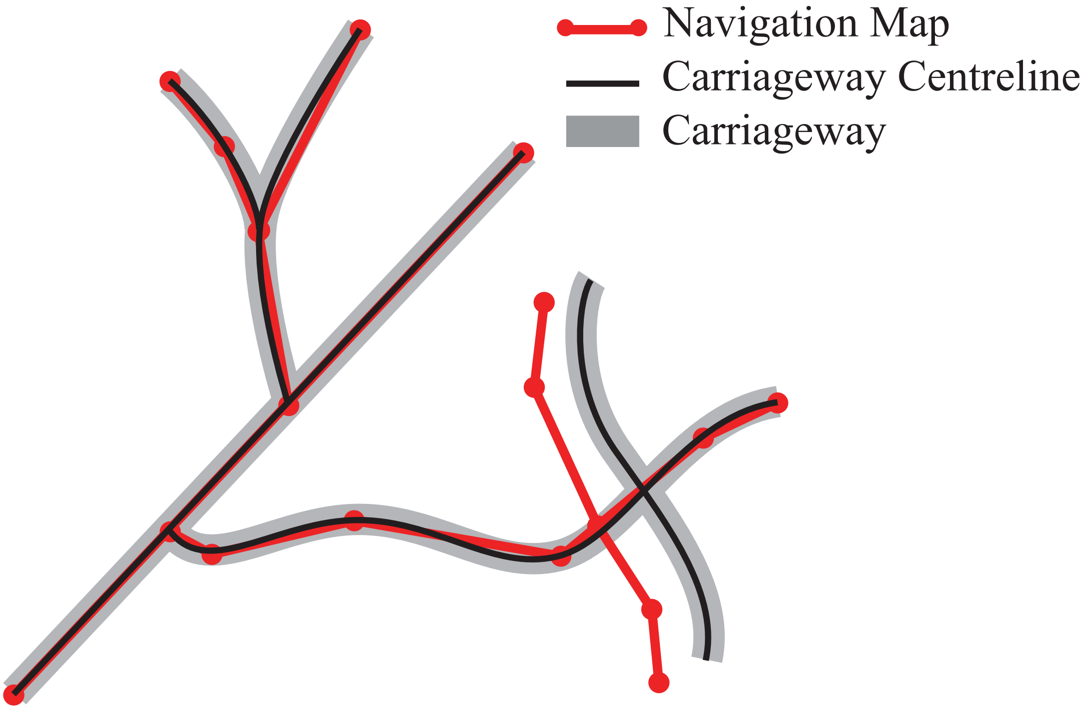
Wrong representation of the road network.

Error: Discrepancy between value in the EH and true value.

Failure: Dysfunction of a client ADAS or autonomous driving function.



Map Geometric Faults

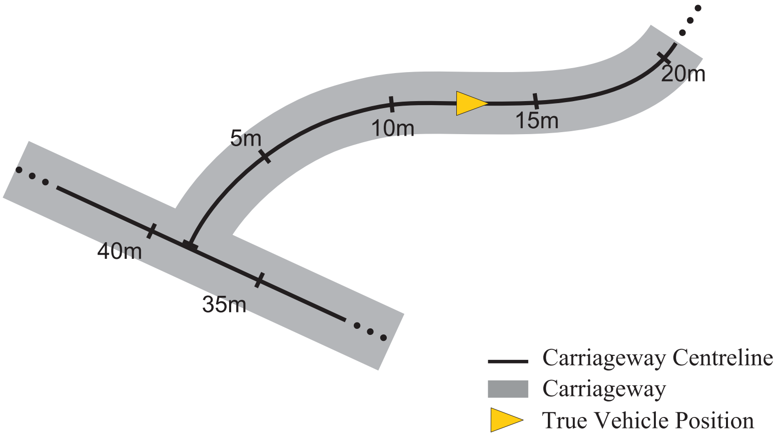


Geometric Fault Detection, Isolation and Adaptation

The vehicle position is encoded with:

The curvilinear abscissa **s**

The trip number **k**

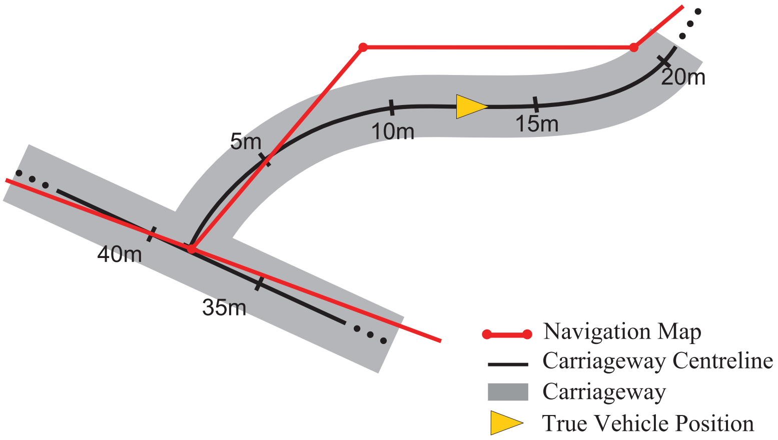


Geometric Fault Detection, Isolation and Adaptation

The vehicle position is encoded with:

The curvilinear abscissa **s**

The trip number **k**

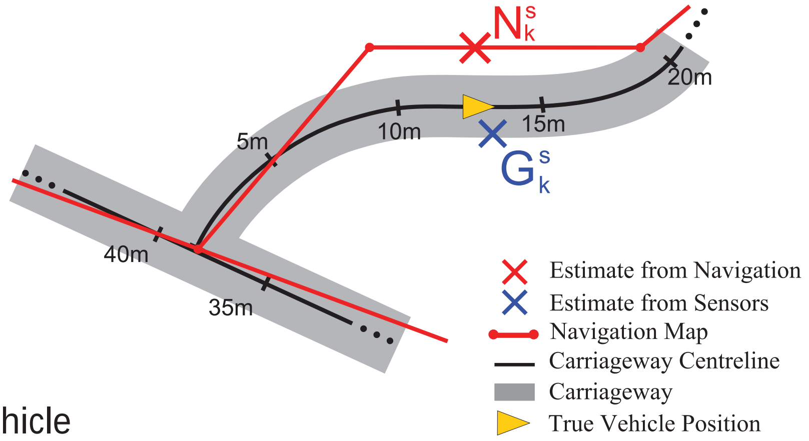


Geometric Fault Detection, Isolation and Adaptation

The vehicle position is encoded with:

The curvilinear abscissa **s**

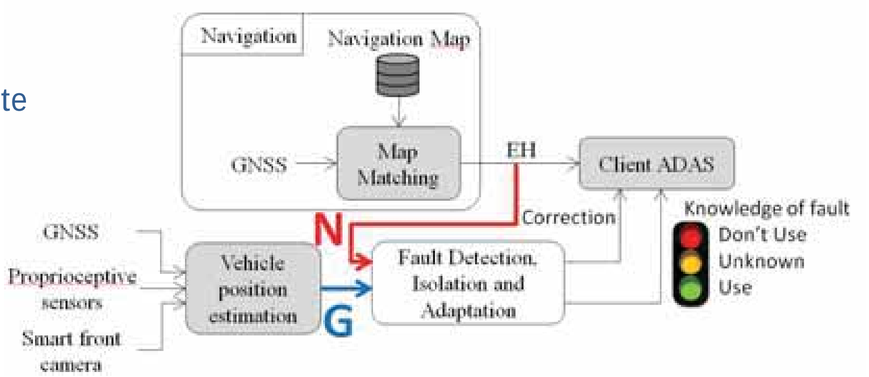
The trip number **k**



FDIA is based on the comparison of vehicle position estimates:

G from vehicle sensors

N from the Navigation function estimate



Geometric fault detection, isolation and adaptation

Detection: Determine whether an estimate is affected by a fault

Isolation: Determine which estimate is affected by a fault

Adaptation: Identify a fault free estimate to provide it to client systems

Assumptions

When travelling several times on a road, the vehicle follows the same path with small deviations

At a given abscissa:

- Faulty vehicle position estimates from sensors are different from one trip to the other.
- Faulty vehicle position estimates from the navigation are always the same.
- Faults on the vehicle position estimates from sensors and from the navigation are different from each other.

Method

- First vehicle trip
- Two independent estimates of the vehicle position:
  - G_1 (from vehicle sensors)
  - N_1 (from navigation system)
- Observed residual: $r_{G_1N_1} = \begin{cases} 0 & if\ G_1 = N_1 \\ 1 & if\ G_1 \neq N_1 \end{cases}$
- G_1 affected by a fault: $f_{G_1} = 1$
- N_1 affected by a fault: $f_{N_1} = 1$

Faults and residuals

Possible outcomes

- $G_1 = N_1$ $\quad (r_{G_1N_1} = 0)$

  Both estimates are fault-free

  $f_{G_1} = 0$ and $f_{N_1} = 0$

- $G_1 \neq N_1$ $\quad (r_{G_1N_1} = 1)$
  - One estimate is faulty

    $f_{G_1} = 1$ and $f_{N_1} = 0$

    $f_{G_1} = 0$ and $f_{N_1} = 1$
  - Both estimates are faulty

    $f_{G_1} = 1$ and $f_{N_1} = 1$

Faults on estimates from sensors and from the navigation are assumed to be different from each other

The residual is therefore the result of a Boolean OR:

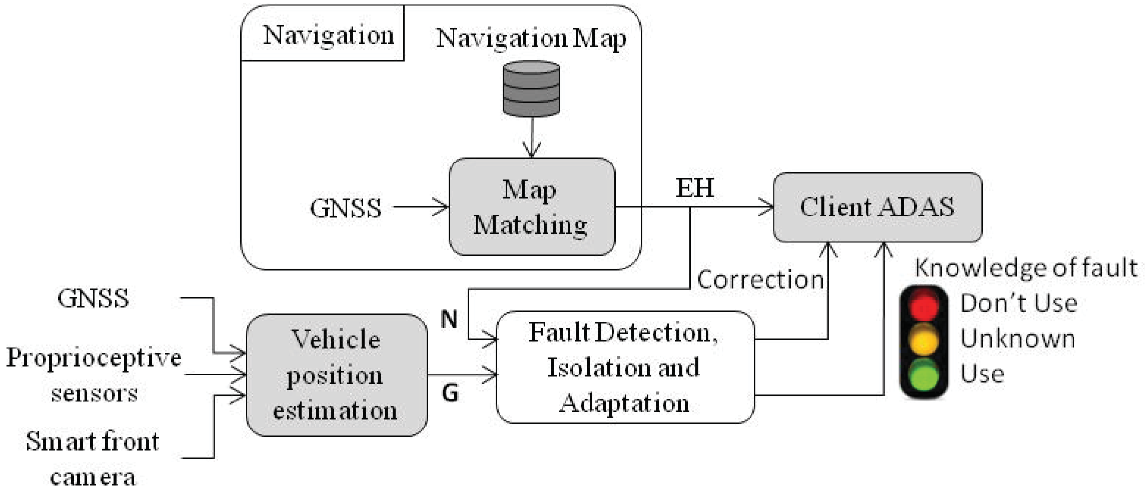
Sets of faults $e_{K,n}$			Residuals
	f_{G_1}	f_{N_1}	$r_{G_1N_1} = f_{G_1} \vee f_{N_1}$
$e_{1,1}$	0	0	0
$e_{1,2}$	1	0	1
$e_{1,3}$	0	1	1
$e_{1,4}$	1	1	1

Method

Compute the residual based on the available estimates

Find this residual in the truth table

Provide the knowledge of fault to client systems



Navigation

Navigation Map

GNSS

Map Matching

EH

Client ADAS

Correction

Knowledge of fault

Don't Use

Unknown

Use

GNSS

Proprioceptive sensors

Smart front camera

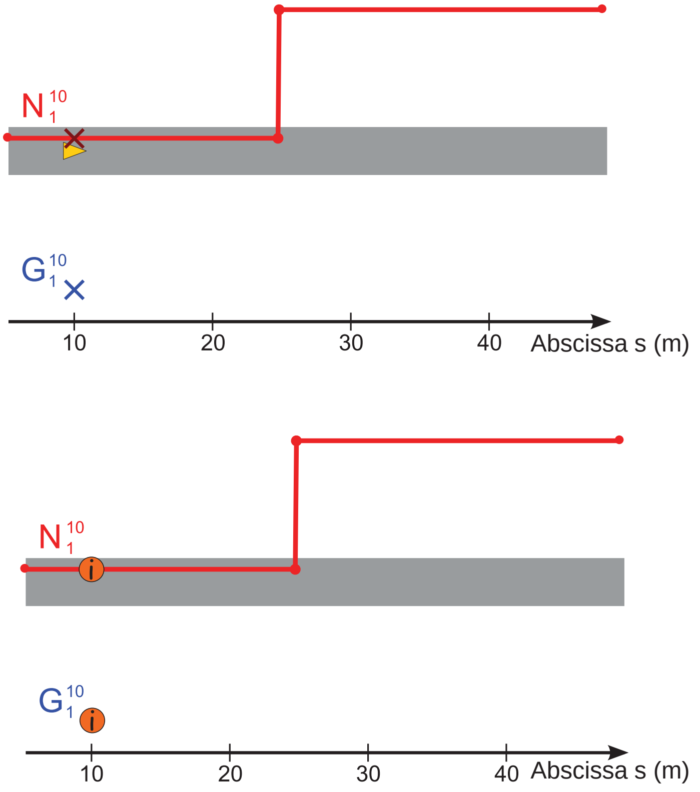
Vehicle position estimation

N

G

Fault Detection, Isolation and Adaptation

Illustrative example: First trip



s = 10m

$G_1 \neq N_1$

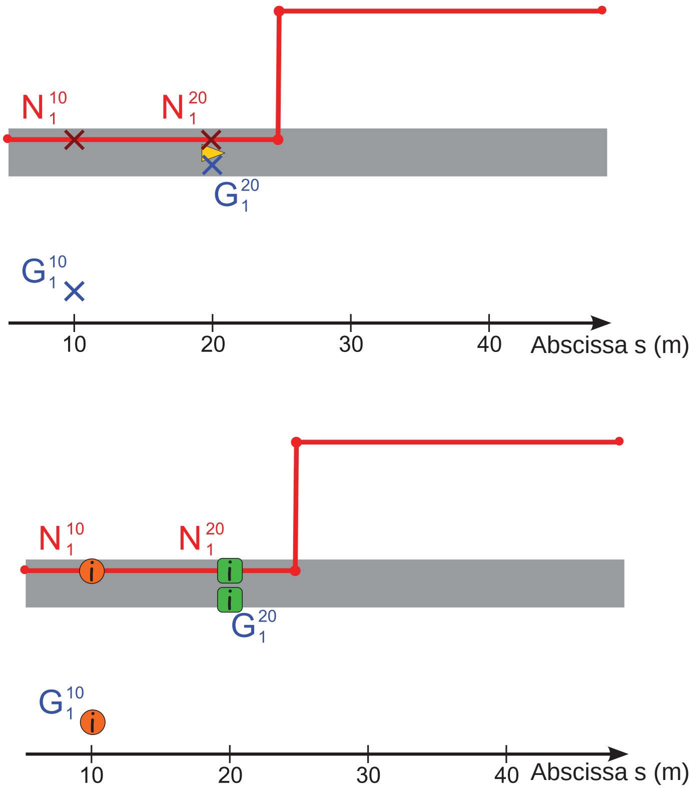
$$r_{G_1^{10} N_1^{10}} = 1$$

Sets of faults $e_{K,n}$			Residuals
	f_{G_1}	f_{N_1}	$r_{G_1 N_1} = f_{G_1} \vee f_{N_1}$
$e_{1,1}$	0	0	0
$e_{1,2}$	1	0	1
$e_{1,3}$	0	1	1
$e_{1,4}$	1	1	1

Both estimates are possibly faulty
A fault is **detected** but **not isolated**
The method returns *Unknown*

⚠ *Don't use*
● *Unknown*
🟩 *Use*

Illustrative example: First trip



s = 20m

$G_1 = N_1$

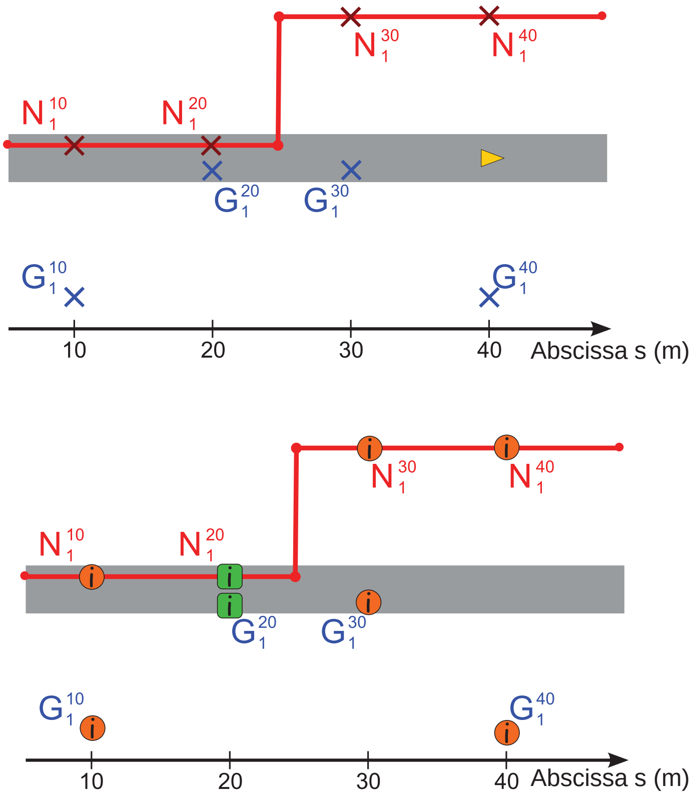
$$r_{G_1^{20} N_1^{20}} = 0$$

Sets of faults $e_{K,n}$			Residuals
	f_{G_1}	f_{N_1}	$r_{G_1 N_1} = f_{G_1} \vee f_{N_1}$
$e_{1,1}$	0	0	0
$e_{1,2}$	1	0	1
$e_{1,3}$	0	1	1
$e_{1,4}$	1	1	1

There is no fault
The method returns *Use*

⚠ *Don't use*
● *Unknown*
🟩 *Use*

Illustrative example: First trip



s = 40m

$G_1 \neq N_1$

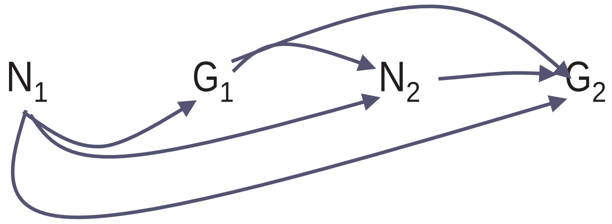
Sets of faults $e_{K,n}$			Residuals
	f_{G_1}	f_{N_1}	$r_{G_1N_1} = f_{G_1} \vee f_{N_1}$
$e_{1,1}$	0	0	0
$e_{1,2}$	1	0	1
$e_{1,3}$	0	1	1
$e_{1,4}$	1	1	1

Both estimates are possibly faulty
A fault is **detected** but **not isolated**
The method returns *Unknown*

⚠ *Don't use*
ⓘ *Unknown*
🟩 *Use*

Using several trips

- Second vehicle trip
- Two new estimates of the vehicle position at the same abscissa
  - G_2 (from vehicle sensors)
  - N_2 (from navigation system)



- Observed residual vector:

$$R = \left[\; r_{N_2G_2} \quad r_{G_2G_1} \quad r_{N_1G_2} \quad r_{N_2G_1} \quad r_{N_1N_2} \quad r_{N_1G_1} \; \right]$$

Faults and residuals

Possible outcomes when comparing two estimates from sensors G_1 and G_2

$G_1 = G_2$ $(r_{G_1G_2} = 0)$

Both estimates are fault-free

$f_{G_1} = 0$ and $f_{G_2} = 0$

- $G_1 \neq G_2$ $(r_{G_1G_2} = 1)$
  - One estimate is faulty

    $f_{G_1} = 1$ and $f_{G_2} = 0$

    $f_{G_1} = 0$ and $f_{G_2} = 1$
  - Both estimates are faulty

    $f_{G_1} = 1$ and $f_{G_2} = 1$

Errors on estimates from sensors are assumed to be different from one trip to the other

The residual is therefore the result of a Boolean OR: $r_{G_1G_2} = f_{G_1} \vee f_{G_2}$

Faults and residuals

Possible outcomes when comparing two estimates from Navigation N_1 and N_2

$N_1 = N_2$: Map faults $(r_{N_1N_2} = 0)$

Both estimates are fault-free

$f_{N_1} = 0$ and $f_{N_2} = 0$

Both estimates are faulty

$f_{N_1} = 1$ and $f_{N_2} = 1$

- $N_1 \neq N_2$ Matching faults $(r_{N_1N_2} = 1)$
  - One estimate is faulty

    $f_{N_1} = 0$ and $f_{N_2} = 1$

    $f_{N_1} = 1$ and $f_{N_2} = 0$

Errors on the vehicle position estimates from the navigation are always the same

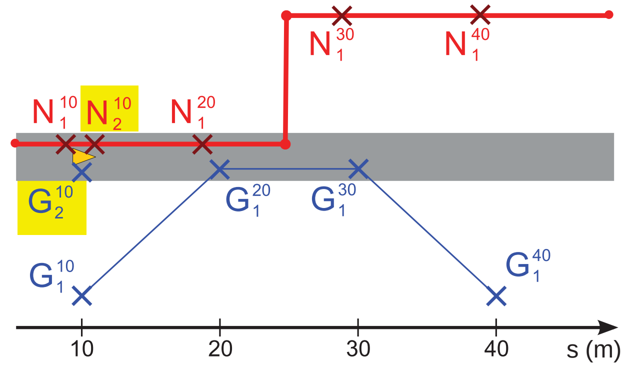
The residual is therefore the result of a Boolean Exclusive OR: $r_{N_1N_2} = f_{N_1} \oplus f_{N_2}$

Truth table for two trips

	Sets of faults $e_{K,n}$				Residuals					
	f_{G_2}	f_{N_2}	f_{G_1}	f_{N_1}	$r_{N_2G_2}$	$r_{G_2G_1}$	$r_{N_1G_2}$	$r_{N_2G_1}$	$r_{N_1N_2}$	$r_{N_1G_1}$
$e_{2,1}$	0	0	0	0	0	0	0	0	0	0
$e_{2,2}$	1	0	0	0	1	1	1	0	0	0
$e_{2,3}$	0	1	0	0	1	0	0	1	1	0
$e_{2,4}$	1	1	0	0	1	1	1	1	1	0
$e_{2,5}$	0	0	1	0	0	1	0	1	0	1
$e_{2,6}$	1	0	1	0	1	1	1	1	0	1
$e_{2,7}$	0	1	1	0	1	1	0	1	1	1
$e_{2,8}$	1	1	1	0	1	1	1	1	1	1
$e_{2,9}$	0	0	0	1	0	0	1	0	1	1
$e_{2,10}$	1	0	0	1	1	1	1	0	1	1
$e_{2,11}$	0	1	0	1	1	0	1	1	0	1
$e_{2,12}$	1	1	0	1	1	1	1	1	0	1
$e_{2,13}$	0	0	1	1	0	1	1	1	1	1
$e_{2,14}$	1	0	1	1	1	1	1	1	1	1
$e_{2,15}$	0	1	1	1	1	1	1	1	0	1
$e_{2,16}$	1	1	1	1	1	1	1	1	0	1

Illustrative example: Second trip

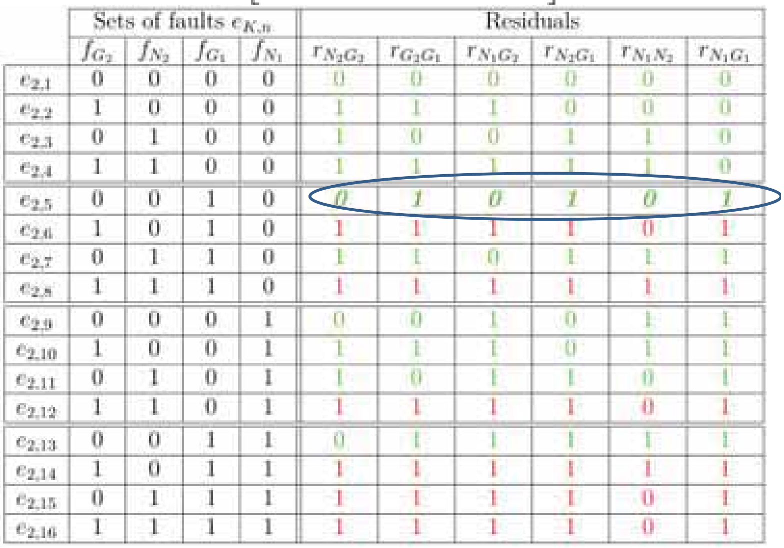
s = 10m



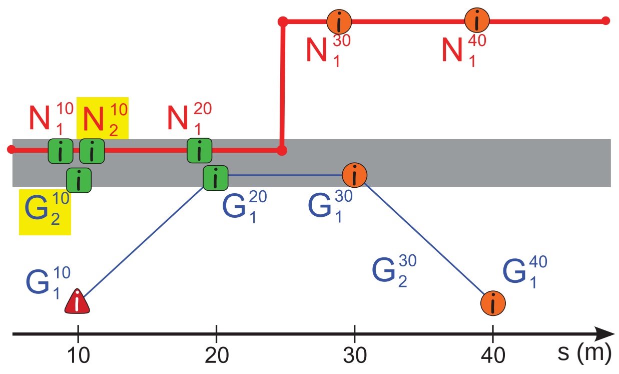
$$N_2^{10} = G_2^{10}\ , \quad G_2^{10} \neq G_1^{10}\ , \quad N_1^{10} = G_2^{10}$$

$$N_2^{10} \neq G_1^{10}\ , \quad N_1^{10} = N_2^{10}\ , \quad N_1^{10} \neq G_1^{10}$$

$$\Rightarrow R_2^{10} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$



	Sets of faults $e_{K,n}$				Residuals					
	f_{G_2}	f_{N_2}	f_{G_1}	f_{N_1}	$r_{N_2G_2}$	$r_{G_2G_1}$	$r_{N_1G_2}$	$r_{N_2G_1}$	$r_{N_1N_2}$	$r_{N_1G_1}$
$e_{2,1}$	0	0	0	0	0	0	0	0	0	0
$e_{2,2}$	1	0	0	0	1	1	1	0	0	0
$e_{2,3}$	0	1	0	0	1	0	0	1	1	0
$e_{2,4}$	1	1	0	0	1	1	1	1	1	0
$e_{2,5}$	0	0	1	0	*0*	*1*	*0*	*1*	*0*	*1*
$e_{2,6}$	1	0	1	0	1	1	1	1	0	1
$e_{2,7}$	0	1	1	0	1	1	0	1	1	1
$e_{2,8}$	1	1	1	0	1	1	1	1	1	1
$e_{2,9}$	0	0	0	1	0	0	1	0	1	1
$e_{2,10}$	1	0	0	1	1	1	1	0	1	1
$e_{2,11}$	0	1	0	1	1	0	1	1	0	1
$e_{2,12}$	1	1	0	1	1	1	1	1	0	1
$e_{2,13}$	0	0	1	1	0	1	1	1	1	1
$e_{2,14}$	1	0	1	1	1	1	1	1	1	1
$e_{2,15}$	0	1	1	1	1	1	1	1	0	1
$e_{2,16}$	1	1	1	1	1	1	1	1	0	1

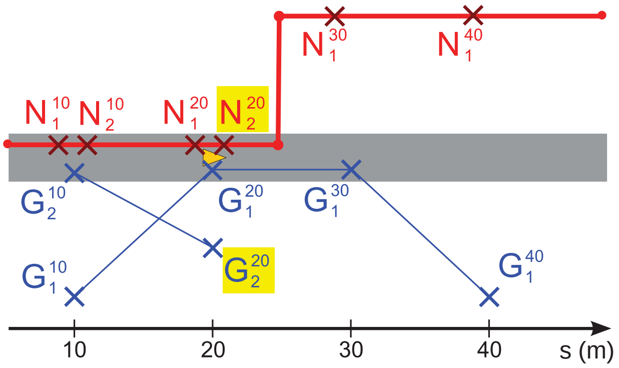


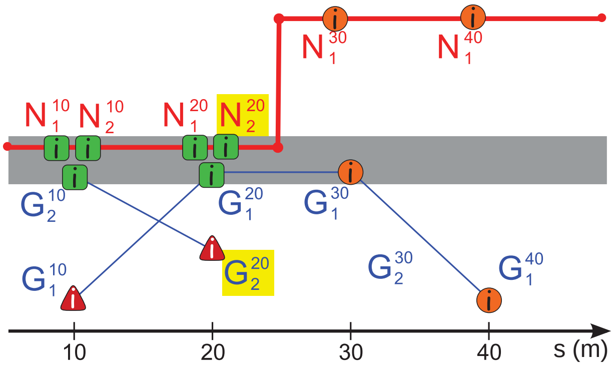
▲ *Don't use*
● *Unknown*
🟩 *Use*

This residual is unique in the table, **isolation** is done
The current navigation is found **not faulty**
The output *Use* is provided to client systems

Illustrative example: Second trip

s = 20m





$$N_2^{20} \neq G_2^{20}\,,\quad G_2^{20} \neq G_1^{20}\,,\quad N_1^{20} \neq G_2^{20}$$

$$N_2^{20} = G_1^{20}\,,\quad N_1^{20} = N_2^{20}\,,\quad N_1^{20} = G_1^{20}$$

$$\Rightarrow R_2^{20} = \left[\begin{array}{cccccc} 1 & 1 & 1 & 0 & 0 & 0 \end{array}\right]$$

	Sets of faults $e_{K,n}$				Residuals					
	f_{G_2}	f_{N_2}	f_{G_1}	f_{N_1}	$r_{N_2G_2}$	$r_{G_2G_1}$	$r_{N_1G_2}$	$r_{N_2G_1}$	$r_{N_1N_2}$	$r_{N_1G_1}$
$e_{2,1}$	0	0	0	0	0	0	0	0	0	0
$e_{2,2}$	1	0	0	0	*1*	*1*	*1*	*0*	*0*	*0*
$e_{2,3}$	0	1	0	0	1	0	0	1	1	0
$e_{2,4}$	1	1	0	0	1	1	1	1	1	0
$e_{2,5}$	0	0	1	0	0	1	0	1	0	1
$e_{2,6}$	1	0	1	0	1	1	1	1	0	1
$e_{2,7}$	0	1	1	0	1	1	0	1	1	1
$e_{2,8}$	1	1	1	0	1	1	1	1	1	1
$e_{2,9}$	0	0	0	1	0	0	1	0	1	1
$e_{2,10}$	1	0	0	1	1	1	1	0	1	1
$e_{2,11}$	0	1	0	1	1	0	1	1	0	1
$e_{2,12}$	1	1	0	1	1	1	1	1	0	1
$e_{2,13}$	0	0	1	1	0	1	1	1	1	1
$e_{2,14}$	1	0	1	1	1	1	1	1	1	1
$e_{2,15}$	0	1	1	1	1	1	1	1	0	1
$e_{2,16}$	1	1	1	1	1	1	1	1	0	1

This residual is unique in the table, **isolation** is done

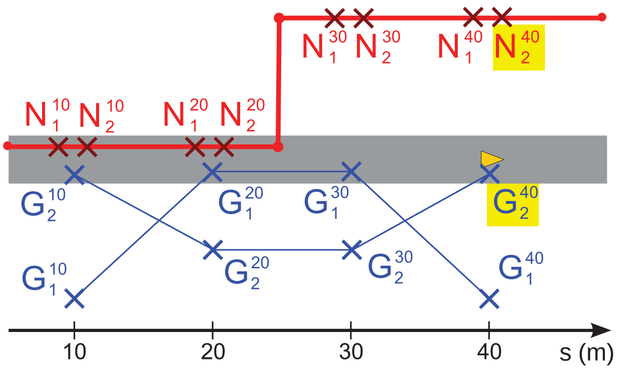
The current navigation is found **not faulty**

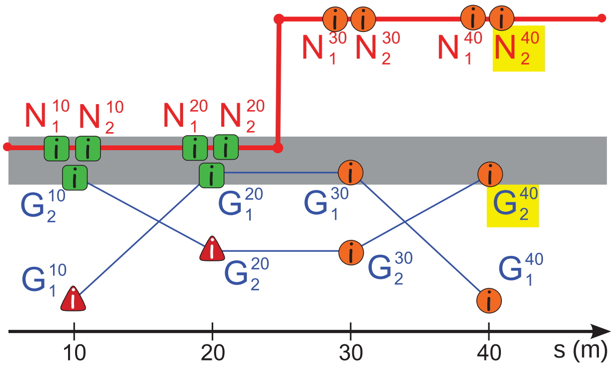
The output *Use* is provided to client systems

Don't use
Unknown
Use

Illustrative example: Second trip

s = 40m





$$N_2^{40} \neq G_2^{40}\,,\quad G_2^{40} \neq G_1^{40}\,,\quad N_1^{40} \neq G_2^{40}$$

$$N_2^{40} \neq G_1^{40}\,,\quad N_1^{40} = N_2^{40}\,,\quad N_1^{40} \neq G_1^{40}$$

$$\Rightarrow R_2^{40} = \left[\begin{array}{cccccc} 1 & 1 & 1 & 1 & 0 & 1 \end{array}\right]$$

	Sets of faults $e_{K,n}$				Residuals					
	f_{G_2}	f_{N_2}	f_{G_1}	f_{N_1}	$r_{N_2G_2}$	$r_{G_2G_1}$	$r_{N_1G_2}$	$r_{N_2G_1}$	$r_{N_1N_2}$	$r_{N_1G_1}$
$e_{2,1}$	0	0	0	0	0	0	0	0	0	0
$e_{2,2}$	1	0	0	0	1	1	1	0	0	0
$e_{2,3}$	0	1	0	0	1	0	0	1	1	0
$e_{2,4}$	1	1	0	0	1	1	1	1	1	0
$e_{2,5}$	0	0	1	0	0	1	0	1	0	1
$e_{2,6}$	1	0	1	0	*1*	*1*	*1*	*1*	*0*	*1*
$e_{2,7}$	0	1	1	0	1	1	0	1	1	1
$e_{2,8}$	1	1	1	0	1	1	1	1	1	1
$e_{2,9}$	0	0	0	1	0	0	1	0	1	1
$e_{2,10}$	1	0	0	1	1	1	1	0	1	1
$e_{2,11}$	0	1	0	1	1	0	1	1	0	1
$e_{2,12}$	1	1	0	1	*1*	*1*	*1*	*1*	*0*	*1*
$e_{2,13}$	0	0	1	1	0	1	1	1	1	1
$e_{2,14}$	1	0	1	1	1	1	1	1	1	1
$e_{2,15}$	0	1	1	1	*1*	*1*	*1*	*1*	*0*	*1*
$e_{2,16}$	1	1	1	1	*1*	*1*	*1*	*1*	*0*	*1*

This residual is four times in the table, fault is **detected** but **not isolated**

The output *Unknown* is provided to client systems

Don't use
Unknown
Use

Formalism properties

	Sets of faults $e_{K,n}$		Residuals
	f_{G_1}	f_{N_1}	$r_{G_1N_1} = f_{G_1} \vee f_{N_1}$
$e_{1,1}$	0	0	0
$e_{1,2}$	1	0	1
$e_{1,3}$	0	1	1
$e_{1,4}$	1	1	1

	Sets of faults $e_{K,n}$				Residuals					
	f_{G_2}	f_{N_2}	f_{G_1}	f_{N_1}	$r_{N_2G_2}$	$r_{G_2G_1}$	$r_{N_1G_2}$	$r_{N_2G_1}$	$r_{N_1N_2}$	$r_{N_1G_1}$
$e_{2,1}$	0	0	0	0	0	0	0	0	0	0
$e_{2,2}$	1	0	0	0	1	1	1	0	0	0
$e_{2,3}$	0	1	0	0	1	0	0	1	1	0
$e_{2,4}$	1	1	0	0	1	1	1	1	1	0
$e_{2,5}$	0	0	1	0	0	1	0	1	0	1
$e_{2,6}$	1	0	1	0	1	1	1	1	0	1
$e_{2,7}$	0	1	1	0	1	1	0	1	1	1
$e_{2,8}$	1	1	1	0	1	1	1	1	1	1
$e_{2,9}$	0	0	0	1	0	0	1	0	1	1
$e_{2,10}$	1	0	0	1	1	1	1	0	1	1
$e_{2,11}$	0	1	0	1	1	0	1	1	0	1
$e_{2,12}$	1	1	0	1	1	1	1	1	0	1
$e_{2,13}$	0	0	1	1	0	1	1	1	1	1
$e_{2,14}$	1	0	1	1	1	1	1	1	1	1
$e_{2,15}$	0	1	1	1	1	1	1	1	0	1
$e_{2,16}$	1	1	1	1	1	1	1	1	0	1

- Guaranteed detection of faults

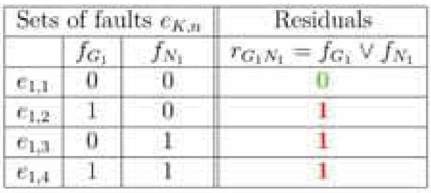
Formalism properties

	Sets of faults $e_{K,n}$		Residuals
	f_{G_1}	f_{N_1}	$r_{G_1N_1} = f_{G_1} \vee f_{N_1}$
$e_{1,1}$	0	0	0
$e_{1,2}$	1	0	1
$e_{1,3}$	0	1	1
$e_{1,4}$	1	1	1

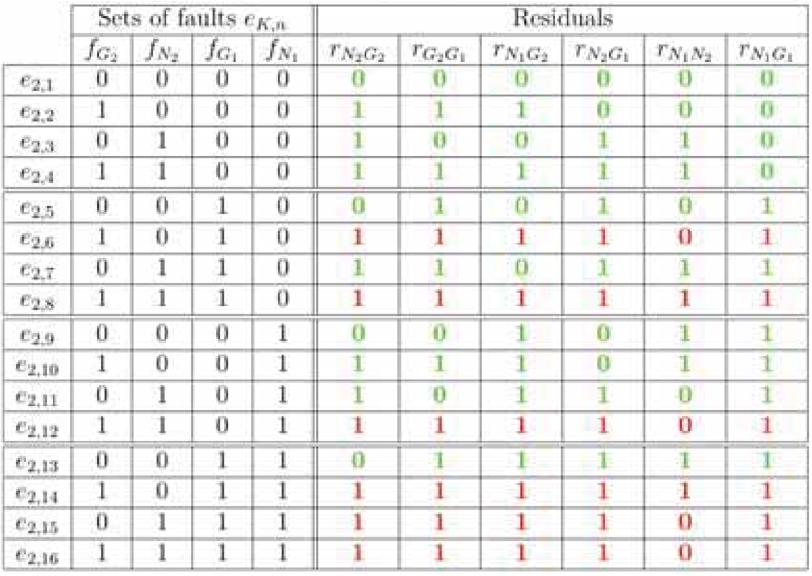
	Sets of faults $e_{K,n}$				Residuals					
	f_{G_2}	f_{N_2}	f_{G_1}	f_{N_1}	$r_{N_2G_2}$	$r_{G_2G_1}$	$r_{N_1G_2}$	$r_{N_2G_1}$	$r_{N_1N_2}$	$r_{N_1G_1}$
$e_{2,1}$	0	0	0	0	0	0	0	0	0	0
$e_{2,2}$	1	0	0	0	1	1	1	0	0	0
$e_{2,3}$	0	1	0	0	1	0	0	1	1	0
$e_{2,4}$	1	1	0	0	1	1	1	1	1	0
$e_{2,5}$	0	0	1	0	0	1	0	1	0	1
$e_{2,6}$	1	0	1	0	1	1	1	1	0	1
$e_{2,7}$	0	1	1	0	1	1	0	1	1	1
$e_{2,8}$	1	1	1	0	1	1	1	1	1	1
$e_{2,9}$	0	0	0	1	0	0	1	0	1	1
$e_{2,10}$	1	0	0	1	1	1	1	0	1	1
$e_{2,11}$	0	1	0	1	1	0	1	1	0	1
$e_{2,12}$	1	1	0	1	1	1	1	1	0	1
$e_{2,13}$	0	0	1	1	0	1	1	1	1	1
$e_{2,14}$	1	0	1	1	1	1	1	1	1	1
$e_{2,15}$	0	1	1	1	1	1	1	1	0	1
$e_{2,16}$	1	1	1	1	1	1	1	1	0	1

- Guaranteed detection of faults
- Conservation of residual isolability

Formalism properties



Sets of faults $e_{K,n}$			Residuals
	f_{G_1}	f_{N_1}	$r_{G_1N_1} = f_{G_1} \vee f_{N_1}$
$e_{1,1}$	0	0	0
$e_{1,2}$	1	0	1
$e_{1,3}$	0	1	1
$e_{1,4}$	1	1	1



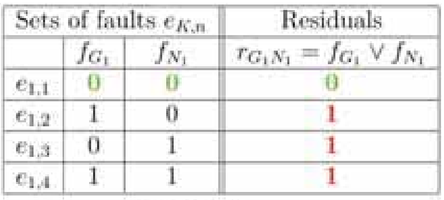
	Sets of faults $e_{K,n}$				Residuals					
	f_{G_2}	f_{N_2}	f_{G_1}	f_{N_1}	$r_{N_2G_2}$	$r_{G_2G_1}$	$r_{N_1G_2}$	$r_{N_2G_1}$	$r_{N_1N_2}$	$r_{N_1G_1}$
$e_{2,1}$	0	0	0	0	0	0	0	0	0	0
$e_{2,2}$	1	0	0	0	1	1	1	0	0	0
$e_{2,3}$	0	1	0	0	1	0	0	1	1	0
$e_{2,4}$	1	1	0	0	1	1	1	1	1	0
$e_{2,5}$	0	0	1	0	0	1	0	1	0	1
$e_{2,6}$	1	0	1	0	1	1	1	1	0	1
$e_{2,7}$	0	1	1	0	1	1	0	1	1	1
$e_{2,8}$	1	1	1	0	1	1	1	1	1	1
$e_{2,9}$	0	0	0	1	0	0	1	0	1	1
$e_{2,10}$	1	0	0	1	1	1	1	0	1	1
$e_{2,11}$	0	1	0	1	1	0	1	1	0	1
$e_{2,12}$	1	1	0	1	1	1	1	1	0	1
$e_{2,13}$	0	0	1	1	0	1	1	1	1	1
$e_{2,14}$	1	0	1	1	1	1	1	1	1	1
$e_{2,15}$	0	1	1	1	1	1	1	1	0	1
$e_{2,16}$	1	1	1	1	1	1	1	1	0	1

- Guaranteed detection of faults
- Conservation of residual isolability
- Isolation convergence

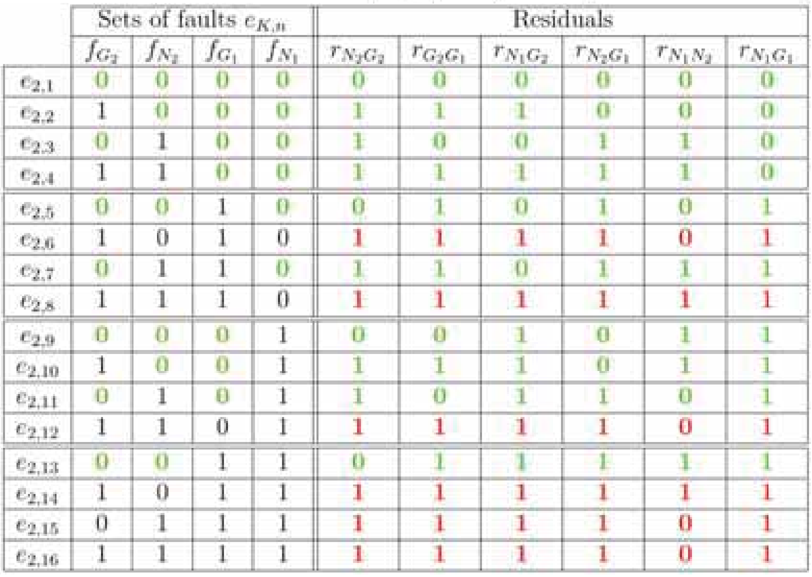
Ratio of adverse residuals

- One trip: q(1) = 3/4
- Two trips: q(2) = 6/16=3/8
- Infinity: q(Inf) → 0

Formalism properties



Sets of faults $e_{K,n}$			Residuals
	f_{G_1}	f_{N_1}	$r_{G_1N_1} = f_{G_1} \vee f_{N_1}$
$e_{1,1}$	0	0	0
$e_{1,2}$	1	0	1
$e_{1,3}$	0	1	1
$e_{1,4}$	1	1	1



	Sets of faults $e_{K,n}$				Residuals					
	f_{G_2}	f_{N_2}	f_{G_1}	f_{N_1}	$r_{N_2G_2}$	$r_{G_2G_1}$	$r_{N_1G_2}$	$r_{N_2G_1}$	$r_{N_1N_2}$	$r_{N_1G_1}$
$e_{2,1}$	0	0	0	0	0	0	0	0	0	0
$e_{2,2}$	1	0	0	0	1	1	1	0	0	0
$e_{2,3}$	0	1	0	0	1	0	0	1	1	0
$e_{2,4}$	1	1	0	0	1	1	1	1	1	0
$e_{2,5}$	0	0	1	0	0	1	0	1	0	1
$e_{2,6}$	1	0	1	0	1	1	1	1	0	1
$e_{2,7}$	0	1	1	0	1	1	0	1	1	1
$e_{2,8}$	1	1	1	0	1	1	1	1	1	1
$e_{2,9}$	0	0	0	1	0	0	1	0	1	1
$e_{2,10}$	1	0	0	1	1	1	1	0	1	1
$e_{2,11}$	0	1	0	1	1	0	1	1	0	1
$e_{2,12}$	1	1	0	1	1	1	1	1	0	1
$e_{2,13}$	0	0	1	1	0	1	1	1	1	1
$e_{2,14}$	1	0	1	1	1	1	1	1	1	1
$e_{2,15}$	0	1	1	1	1	1	1	1	0	1
$e_{2,16}$	1	1	1	1	1	1	1	1	0	1

- Guaranteed detection of faults
- Conservation of residual isolability
- Isolation convergence

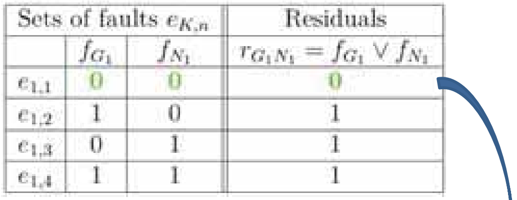
Ratio of adverse residuals

- One trip: q(1) = 3/4
- Two trips: q(2) = 3/8
- Infinity: q(Inf) → 0

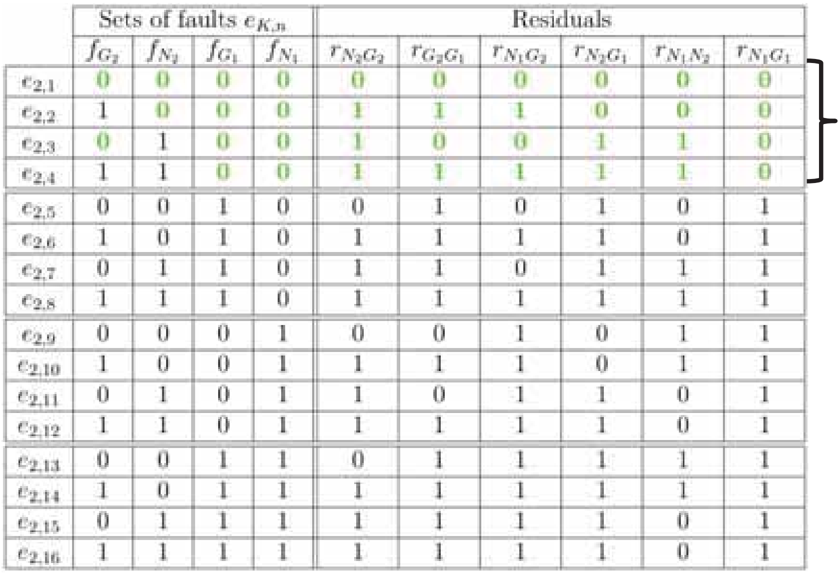
- Adaptation

There is at least one non faulty estimate in isolable sets of faults

Formalism properties



Sets of faults $e_{K,n}$			Residuals
	f_{G_1}	f_{N_1}	$r_{G_1N_1} = f_{G_1} \vee f_{N_1}$
$e_{1,1}$	0	0	0
$e_{1,2}$	1	0	1
$e_{1,3}$	0	1	1
$e_{1,4}$	1	1	1



	Sets of faults $e_{K,n}$				Residuals					
	f_{G_2}	f_{N_2}	f_{G_1}	f_{N_1}	$r_{N_2G_2}$	$r_{G_2G_1}$	$r_{N_1G_2}$	$r_{N_2G_1}$	$r_{N_1N_2}$	$r_{N_1G_1}$
$e_{2,1}$	0	0	0	0	0	0	0	0	0	0
$e_{2,2}$	1	0	0	0	1	1	1	0	0	0
$e_{2,3}$	0	1	0	0	1	0	0	1	1	0
$e_{2,4}$	1	1	0	0	1	1	1	1	1	0
$e_{2,5}$	0	0	1	0	0	1	0	1	0	1
$e_{2,6}$	1	0	1	0	1	1	1	1	0	1
$e_{2,7}$	0	1	1	0	1	1	0	1	1	1
$e_{2,8}$	1	1	1	0	1	1	1	1	1	1
$e_{2,9}$	0	0	0	1	0	0	1	0	1	1
$e_{2,10}$	1	0	0	1	1	1	1	0	1	1
$e_{2,11}$	0	1	0	1	1	0	1	1	0	1
$e_{2,12}$	1	1	0	1	1	1	1	1	0	1
$e_{2,13}$	0	0	1	1	0	1	1	1	1	1
$e_{2,14}$	1	0	1	1	1	1	1	1	1	1
$e_{2,15}$	0	1	1	1	1	1	1	1	0	1
$e_{2,16}$	1	1	1	1	1	1	1	1	0	1

- Guaranteed detection of faults
- Conservation of residual isolability
- Isolation convergence

Ratio of adverse residuals

- One trip: q(1) = 3/4
- Two trips: q(2) = 3/8
- Infinity: q(Inf) → 0

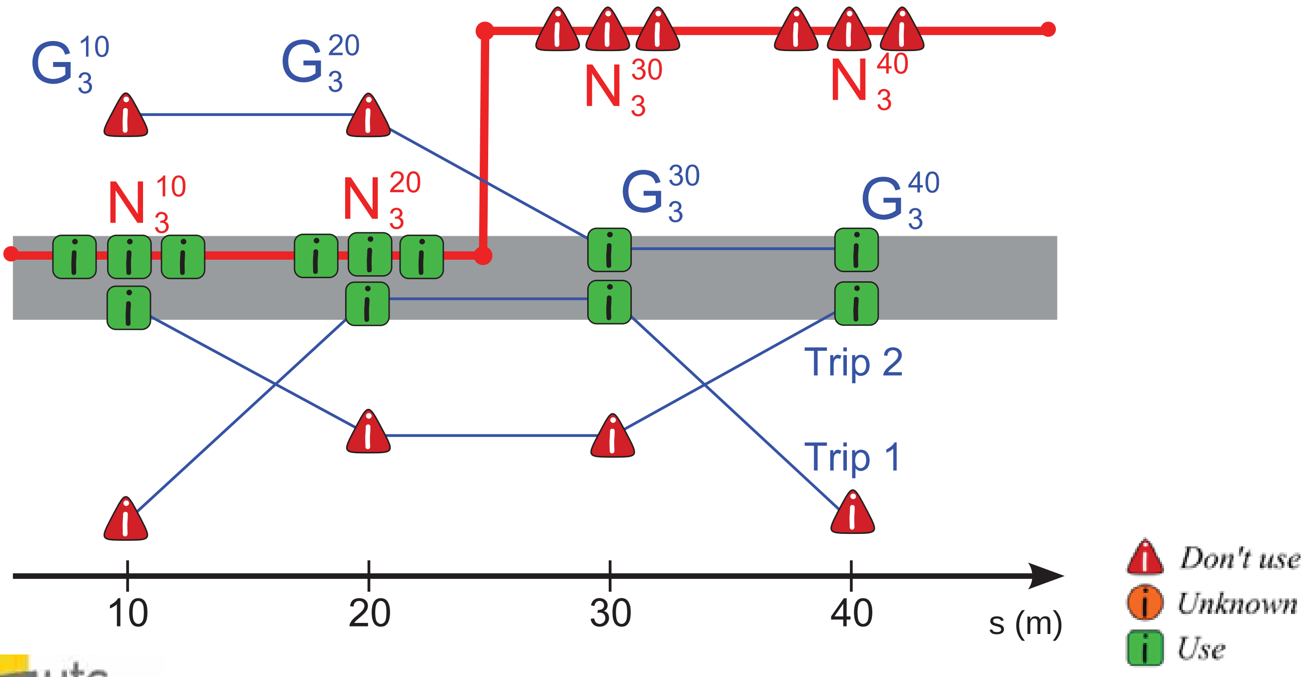
- Adaptation

There is at least one non faulty estimate in isolable sets of faults

- Conservation of adaptation

When isolation is performed, there will be at least one non faulty estimate at the next trip. This will make Adaptation possible

Illustrative example: Third trip



FDIA Algorithm for on board implementation

Before hand computation of the truth tables
In real-time:

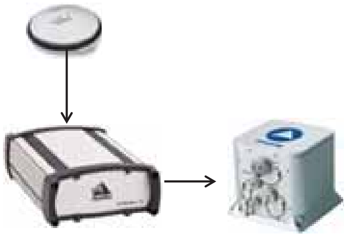
```
Algorithm
  Get the current $G_k$ and $N_k$ estimates
  Store $G_k$ and $N_k$ in the internal memory
  If $r_{G_k N_k} = 0$, Return Use    // The current navigation estimate is not faulty
  Else   // A fault is detected
    If $G_{k-1}$ and $N_{k-1}$ exist   // Try an isolation with previous estimates
      Compute residual vector for two trips
      Find this residual in the truth table
      If this residual is unique
        If $f_{N_k} = 1$, Return Don't Use, provide a fault free estimate for adaptation
        Else, Return Use
      Else, Use trip $k-2$ if available and so on, else return Unknown
    Else, Return Unknown
```

Experimental validation

Probe Vehicle

u-blox GPS receivers

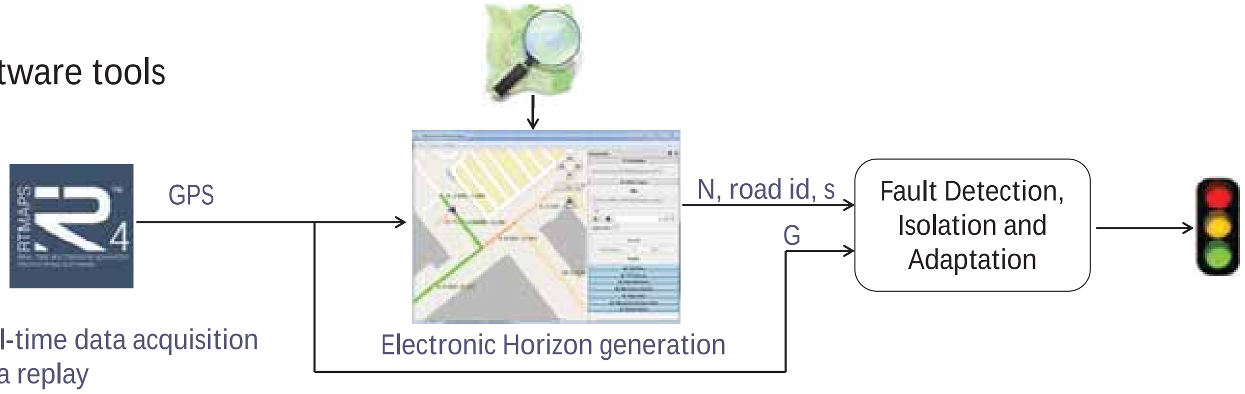
CAN-Bus
- Vehicle speed
- Wheel Speed
- Yaw rate
- Odometer...



GPS + IMU as ground truth for localization

OSM Navigation map

Software tools



Real-time data acquisition
Data replay

Electronic Horizon generation

Metrics

True Validations (TV):

Correct navigation point identified as not-faulty

True Isolations (TI):

Faulty navigation point isolated by the method

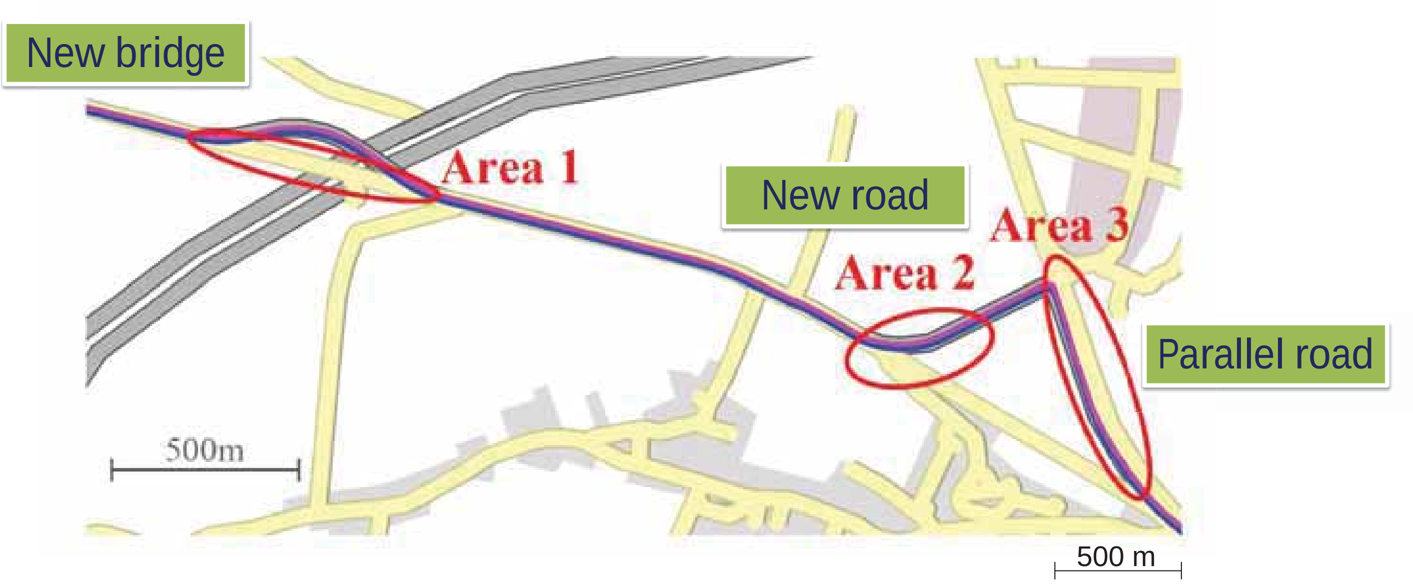
Overall efficiency rate (OER):

$$OER = \frac{TV + TI}{\Omega - \Omega_{unknown}}$$

Information availability rate (IAR):

$$IAR = \frac{\Omega - \Omega_{unknown}}{\Omega}$$

Rural results



- used navigation map in yellow
- correct map is in grey in background
- vehicle goes from left to right
- first trip is in blue; second trip in purple

Rural results

First Trip

Overall Efficiency Rate = 100%

Information Availability Rate = 77%

Second Trip

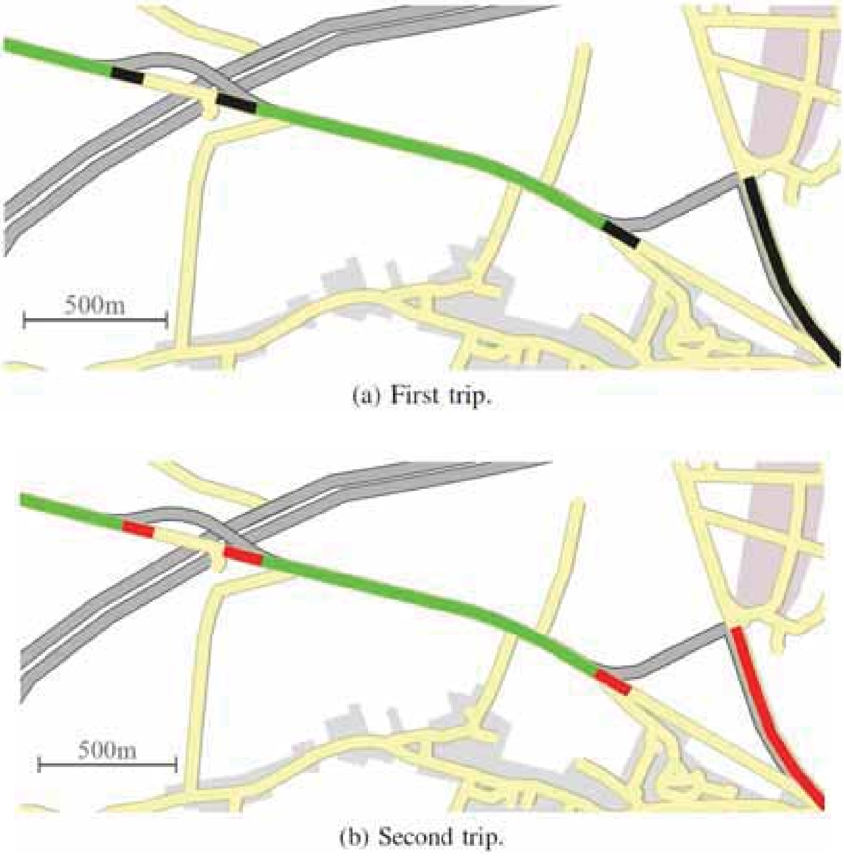
Overall Efficiency Rate = 100%

Information Availability Rate = 100%

Good performance

Real map geometric faults

Simple GPS conditions



(a) First trip.

(b) Second trip.

Outline

Context and Problem Statement

Fault Detection Isolation and Adaptation

Extension to Handle Uncertainties

Page’s trend test

Integration into the FDIA method

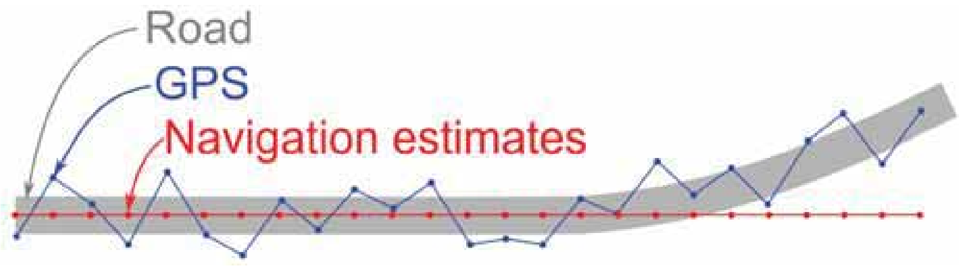
Experimental results

Conclusions and Perspectives

Noisy Position Estimates

Deterministic FDIA method

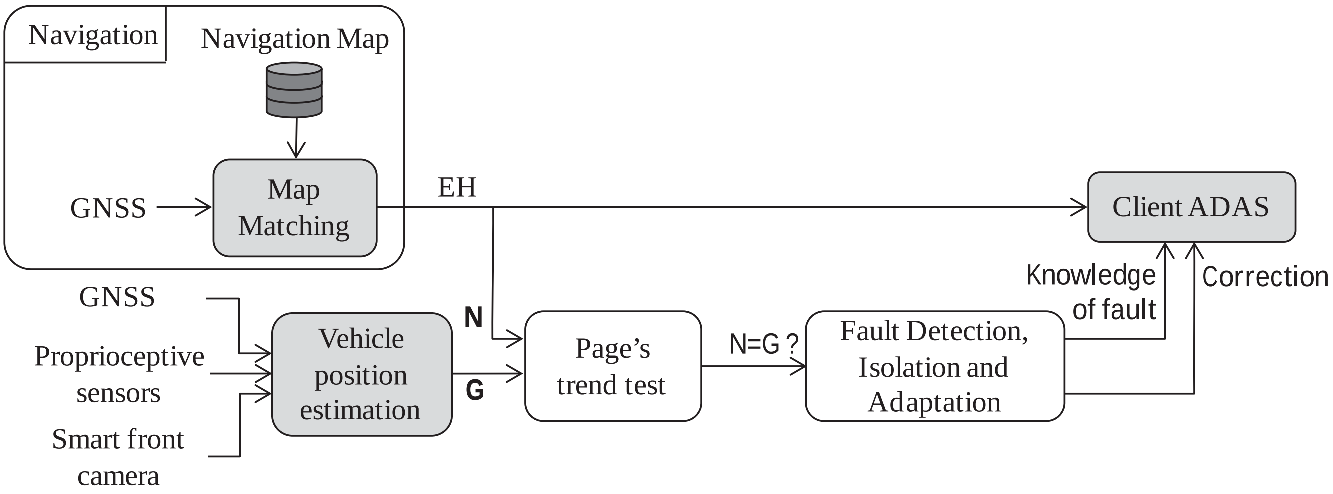
Noise on position estimate particularly in urban environment



Proposed solution

Statistical analysis of GPS and Navigation estimates

System Architecture



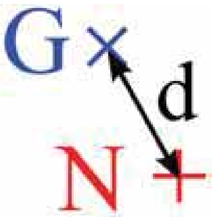
Page’s trend test

Detection of a change in the mean of a random variable

Hypotheses

$$
H_0 \; : \; d_i = \mu_0 + b_i \, , \; i = 1, \ldots, N
$$

$$
H_1 \; : \; \begin{cases} d_i = \mu_0 + b_i \, , \; i = 1, \ldots, r-1 \\ d_i = \mu_1 + b_i \, , \; i = r, \ldots, N \end{cases}
$$



d_i: distance between estimates

μ_0 and μ_1: mean of d before and after the change in the mean

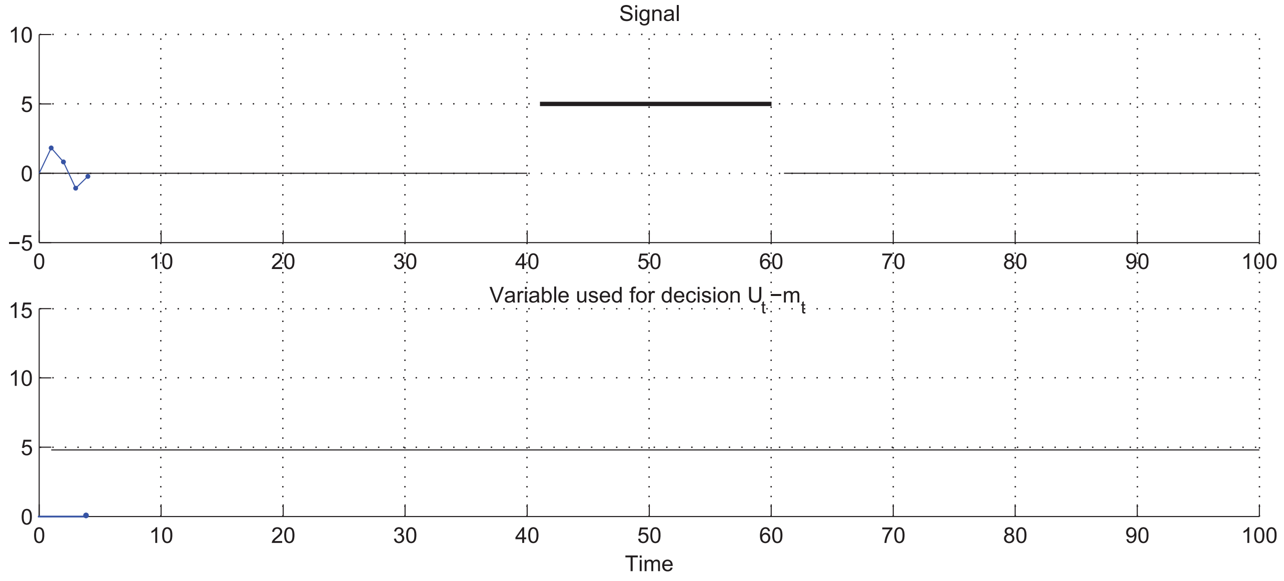
r: time of the change in the mean

b: noise.

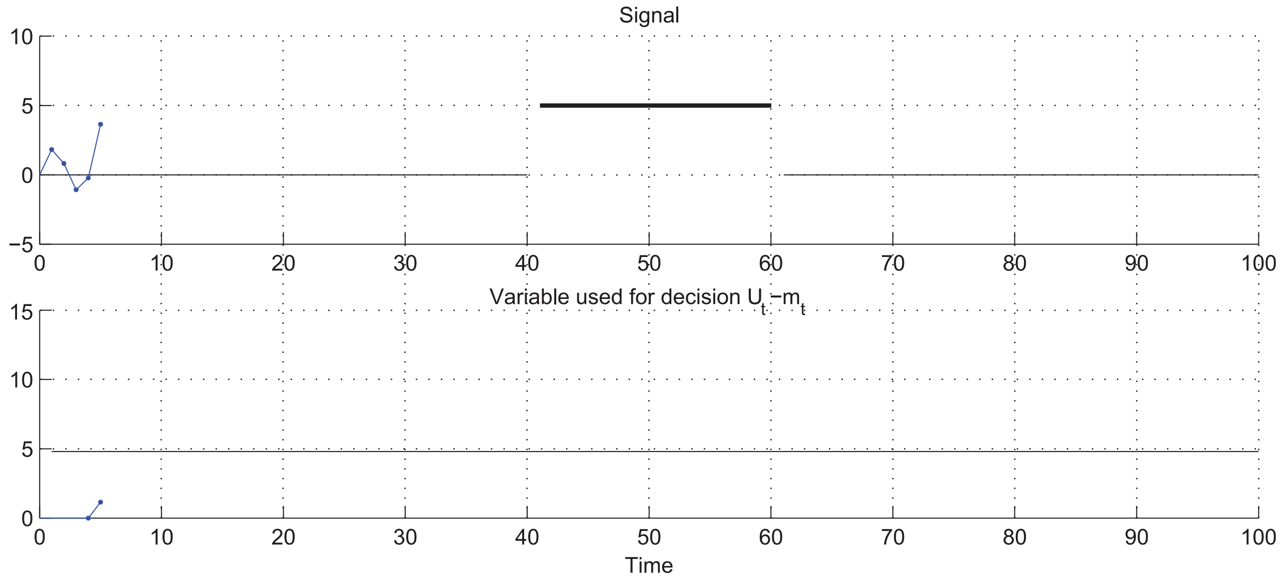
Sequential likelihood ratio testing

Page’s trend test localizes the mean change with a minimized delay

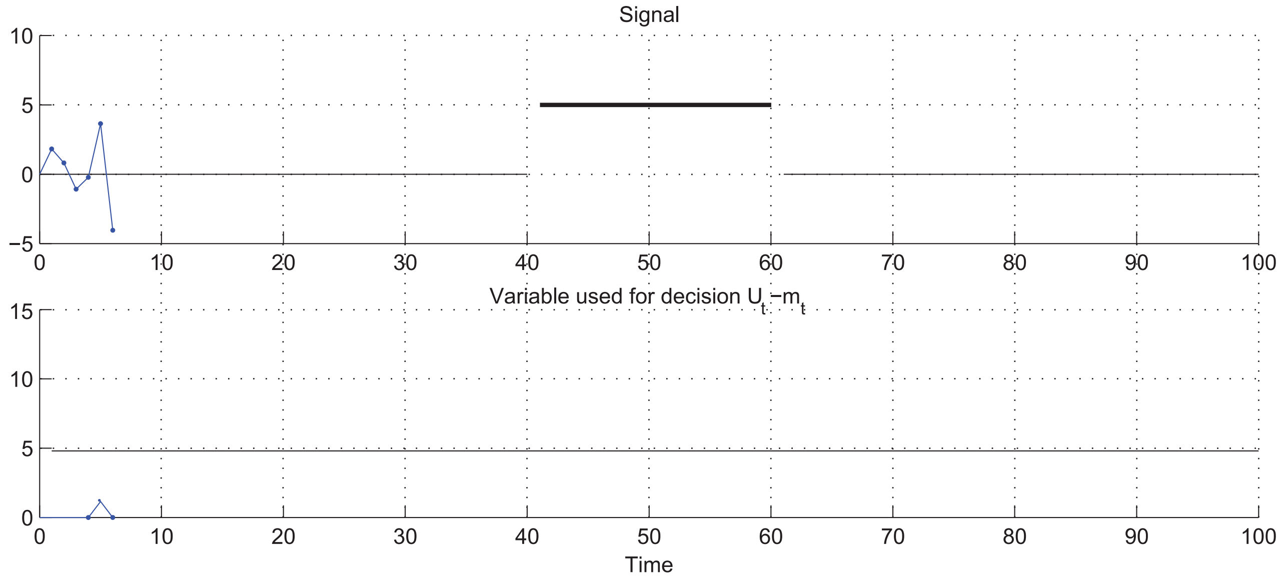
Example



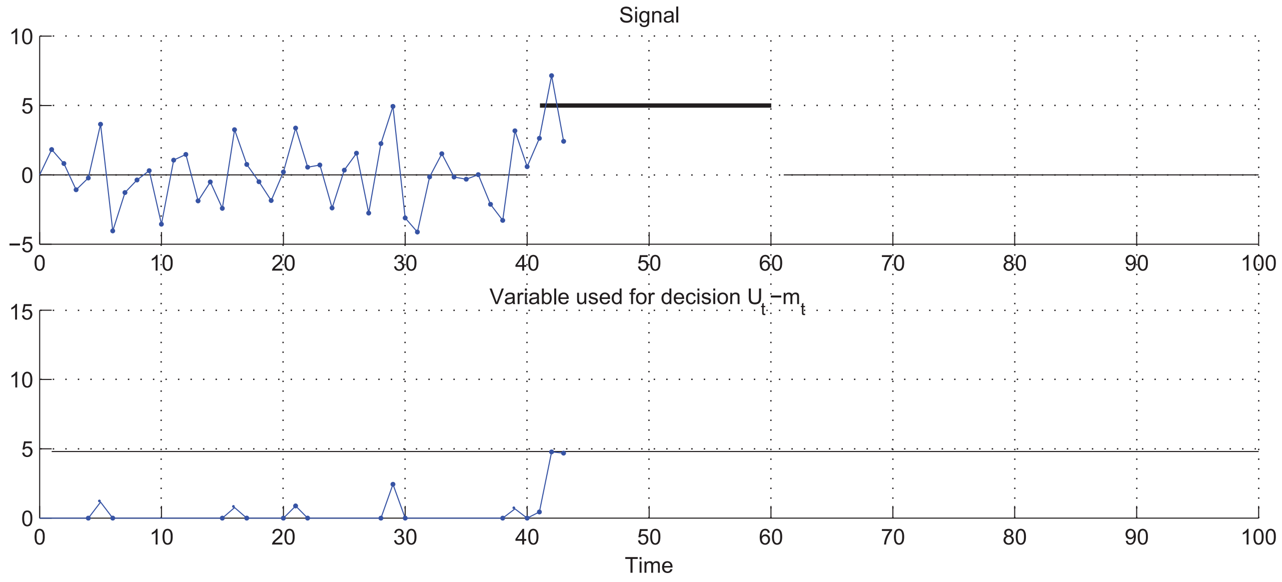
Example



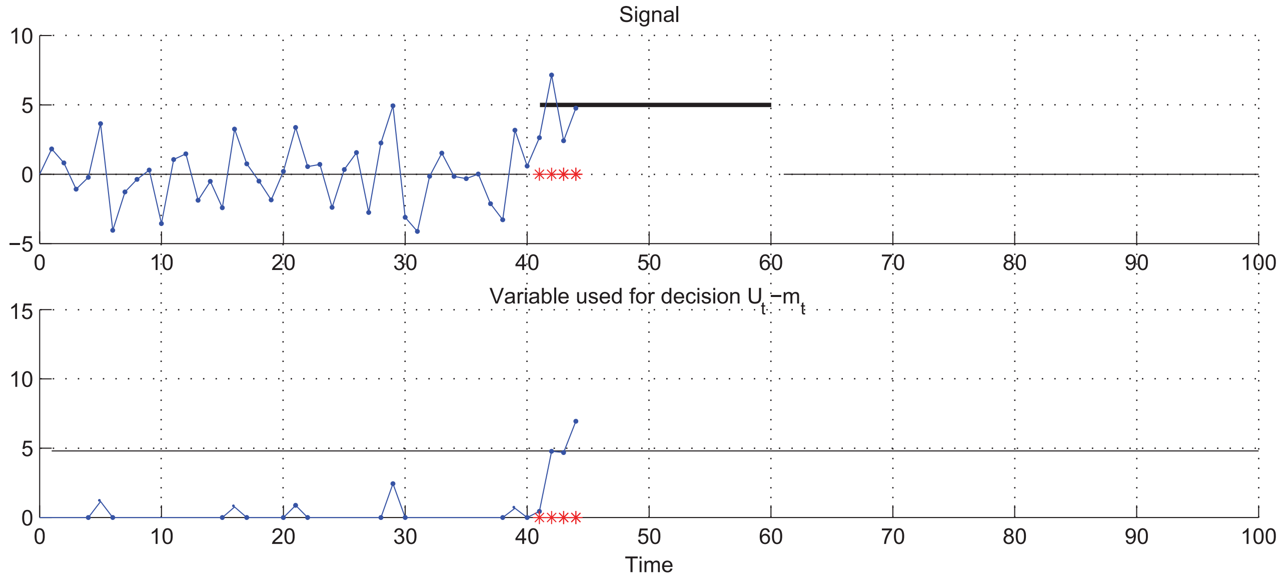
Example



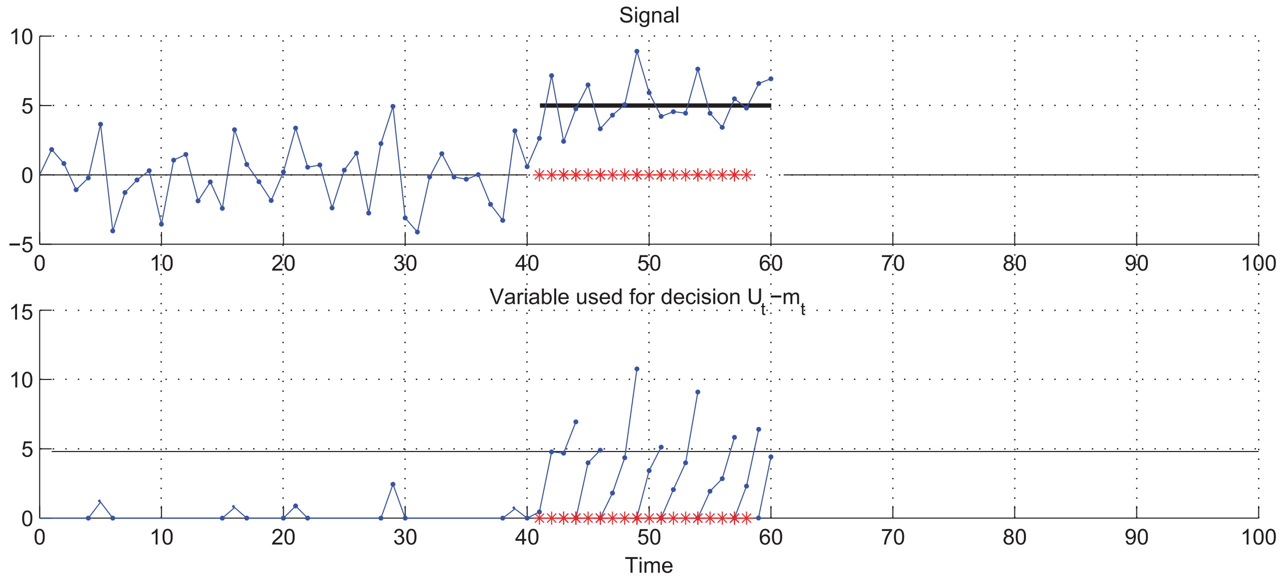
Example



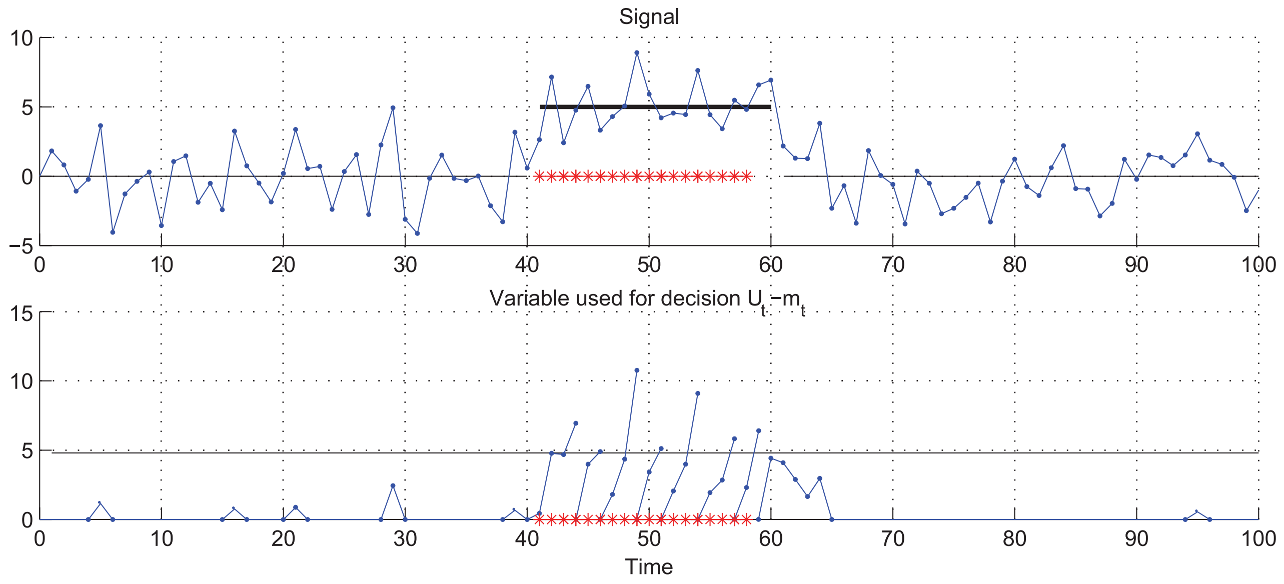
Example



Example

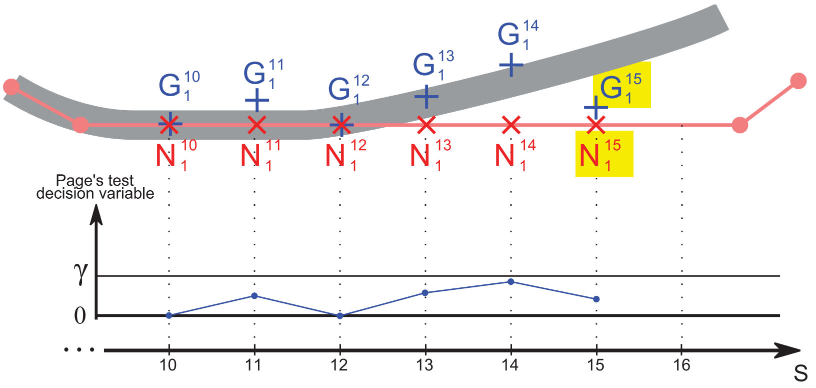


Example



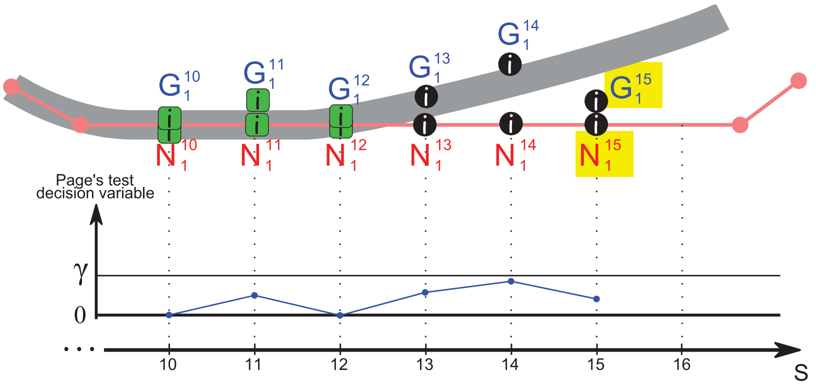
Implementation in the FDIA framework

Page's test provides the value of r_{GN}



G_1^{10} G_1^{11} G_1^{12} G_1^{13} G_1^{14} G_1^{15}
N_1^{10} N_1^{11} N_1^{12} N_1^{13} N_1^{14} N_1^{15}
Page's test decision variable
γ
0
. . .
10
11
12
13
14
15
16
s

When the decision variable is greater than 0 and lower than the threshold, the FDIA is delayed



G_1^{10} G_1^{11} G_1^{12} G_1^{13} G_1^{14} G_1^{15}
N_1^{10} N_1^{11} N_1^{12} N_1^{13} N_1^{14} N_1^{15}
Page's test decision variable
γ
0
. . .
10
11
12
13
14
15
16
s

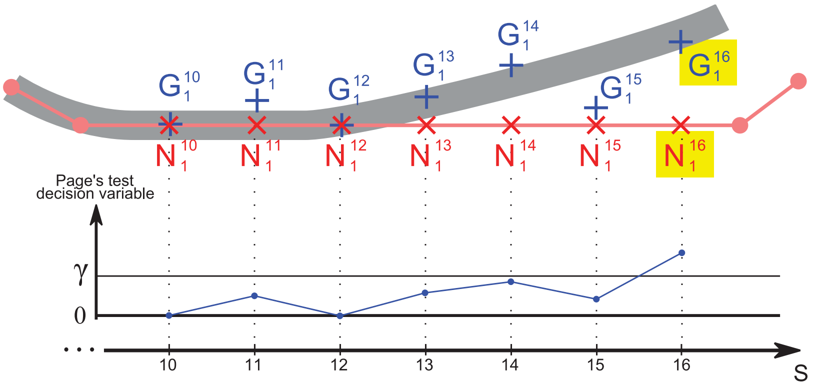
● *FDIA delayed*

🟠 *Unknown*

🟩 *Use*

Implementation in the FDIA framework

When Page's trend test finally settles, FDIA is run at every buffered abscissa.

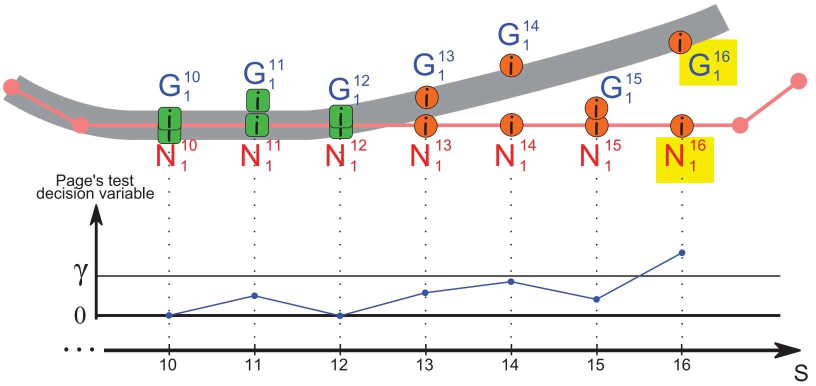


G_1^{10} G_1^{11} G_1^{12} G_1^{13} G_1^{14} G_1^{15} G_1^{16}
N_1^{10} N_1^{11} N_1^{12} N_1^{13} N_1^{14} N_1^{15} N_1^{16}
Page's test decision variable
γ
0
. . .
10
11
12
13
14
15
16
s

Benefits of the test in this example:

False detection avoided at s = 11.

False validation avoided at s = 15.



G_1^{10} G_1^{11} G_1^{12} G_1^{13} G_1^{14} G_1^{15} G_1^{16}
N_1^{10} N_1^{11} N_1^{12} N_1^{13} N_1^{14} N_1^{15} N_1^{16}
Page's test decision variable
γ
0
. . .
10
11
12
13
14
15
16
s

● *FDIA delayed*

🟠 *Unknown*

🟩 *Use*

Random fault injection in maps

Purpose

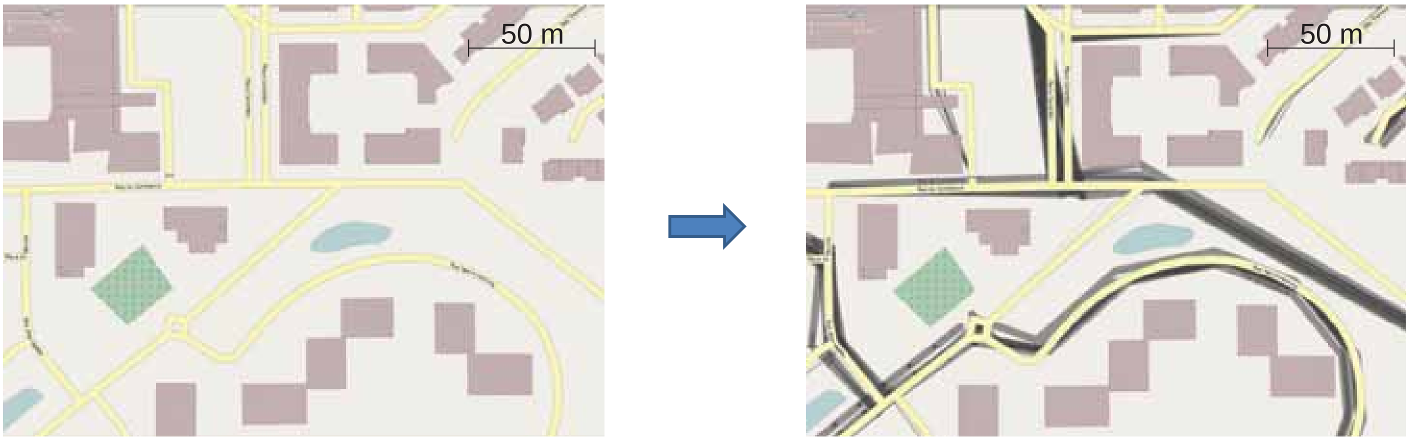
Generates a variety of map faults

Provides quantitative results

Principles

Add noise on position of road shape nodes

Deletes some of the shape nodes



Urban Results

Five maps with random faults

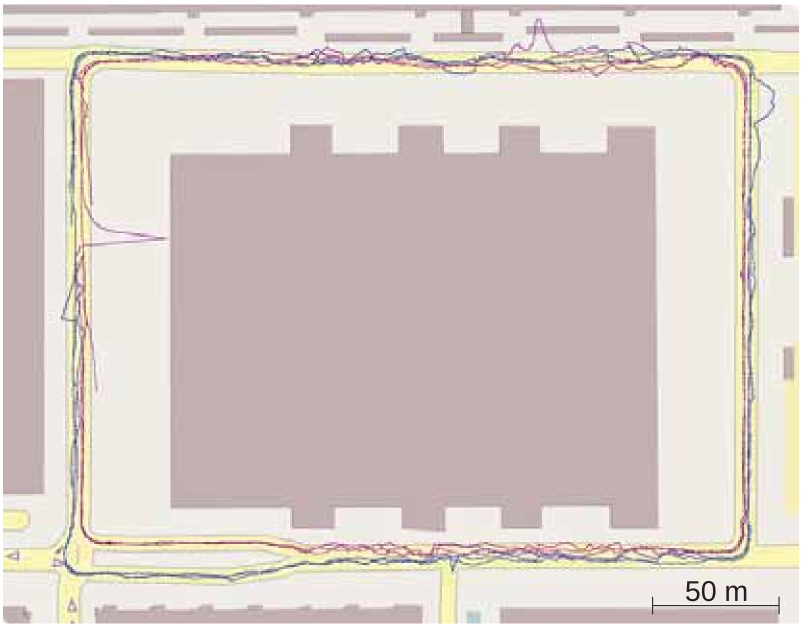
Generated based on a high quality lane level map

Three trips clockwise

Purple lines

Three trips anticlockwise

Blue lines



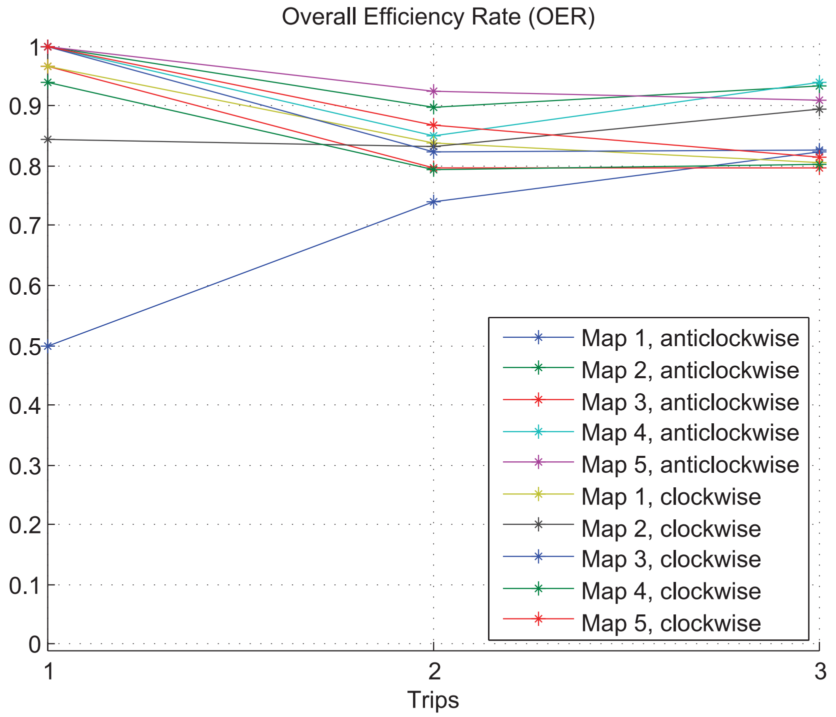
Urban Results

Overall Efficiency

$$OER = \frac{TV + TI}{\Omega - \Omega_{unknown}}$$

Evenly due to True Validations and True Isolations

Gating and temporal data re-sampling effects



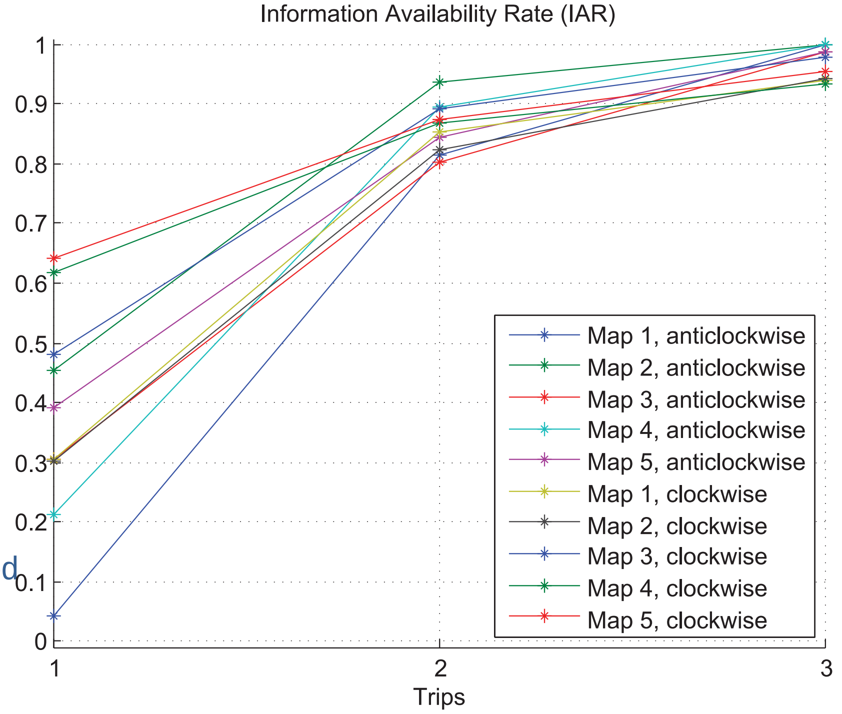
Urban Results

Information Availability

$$IAR = \frac{\Omega - \Omega_{unknown}}{\Omega}$$

Low IAR at trip 1 due to large proportion of faulty map areas and faulty GPS faults

Isolation convergence property verified



Urban Results

With Page’s test:

False Validation rate decreases.

Less faulty map points are identified as correct.

➔ The output “use” is more reliable

False Isolation rate increases.

More correct map points are identified as faulty.

➔ More unjustified “don’t use”.

The FDIA method is **more cautious** but client systems are deactivated more frequently.

Conclusion

Context

Map-aided ADAS

Constant evolution of the road network

Black box systems in passenger vehicles

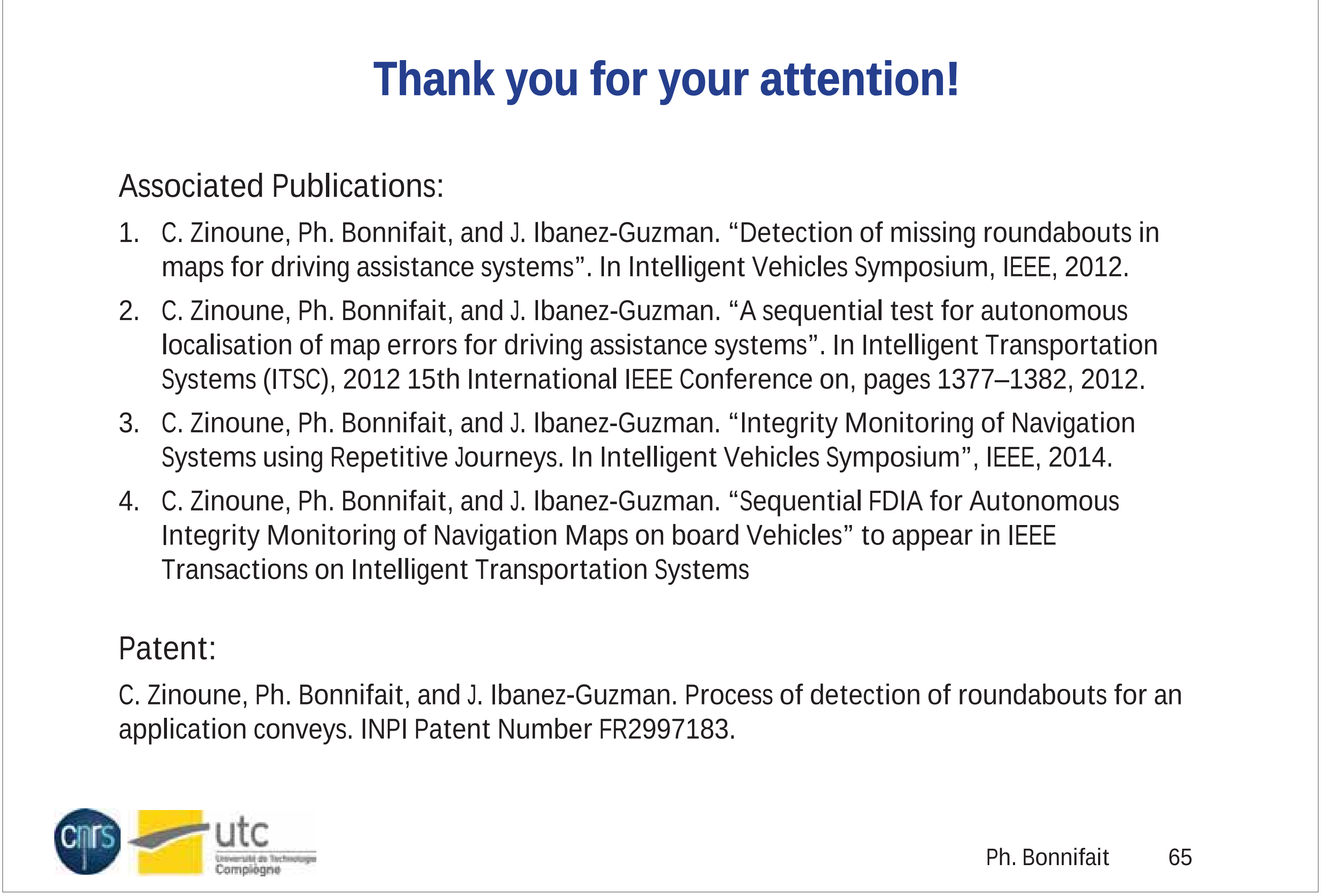
Contributions

An integrity monitoring architecture that uses repetitive trips

A framework for geometric fault detection, isolation and adaptation

An extension of the framework with Page’s test

Tests and evaluations on real vehicle



Thank you for your attention!

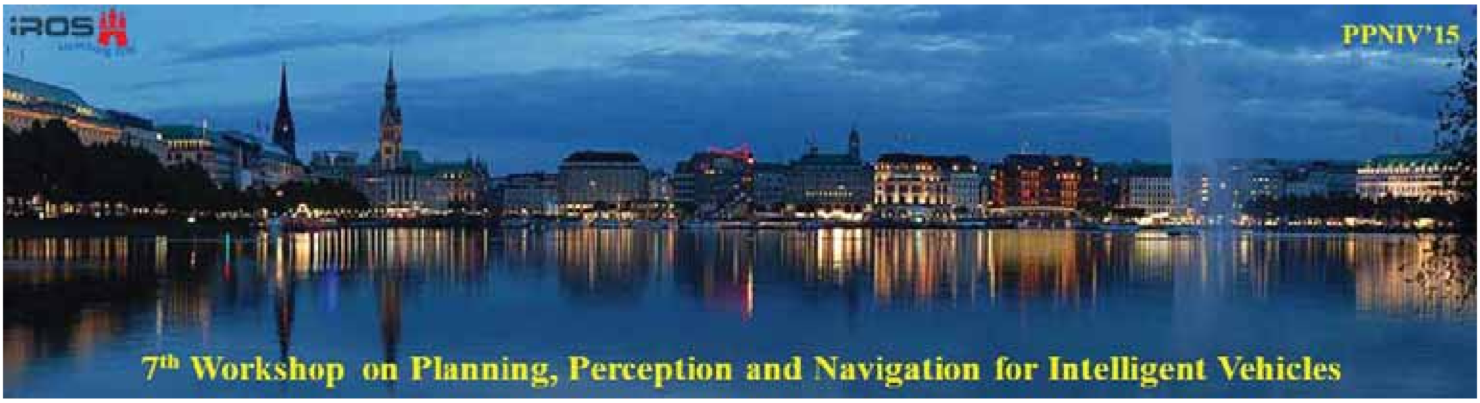
Associated Publications:

1. C. Zinoune, Ph. Bonnifait, and J. Ibanez-Guzman. “Detection of missing roundabouts in maps for driving assistance systems”. In Intelligent Vehicles Symposium, IEEE, 2012.
2. C. Zinoune, Ph. Bonnifait, and J. Ibanez-Guzman. “A sequential test for autonomous localisation of map errors for driving assistance systems”. In Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on, pages 1377–1382, 2012.
3. C. Zinoune, Ph. Bonnifait, and J. Ibanez-Guzman. “Integrity Monitoring of Navigation Systems using Repetitive Journeys. In Intelligent Vehicles Symposium”, IEEE, 2014.
4. C. Zinoune, Ph. Bonnifait, and J. Ibanez-Guzman. “Sequential FDIA for Autonomous Integrity Monitoring of Navigation Maps on board Vehicles” to appear in IEEE Transactions on Intelligent Transportation Systems

Patent:

C. Zinoune, Ph. Bonnifait, and J. Ibanez-Guzman. Process of detection of roundabouts for an application conveys. INPI Patent Number FR2997183.

Ph. Bonnifait 65


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Sequential FDIA for Autonomous Integrity Monitoring of Navigation Maps on board Vehicles

Clément Zinoune[1,2], Philippe Bonnifait[1], Javier Ibañez-Guzmán[2]

***Abstract*—This paper addresses the problem of Fault Detection, Isolation, and Adaptation (FDIA) in navigation systems on board passenger vehicles. The aim is to prevent malfunctions in systems such as advanced driving assistance systems and autonomous driving functions that use data provided by the navigation system. The integrity of the estimation of the vehicle position provided by the navigation system is continuously monitored and assessed. The proposed approach uses an additional estimate of vehicle position that is independent of the navigation system and based on data from standard vehicle sensors. First, fault detection consists in comparing the two estimates using a sequential statistical test to detect discrepancies despite the presence of noise. Second, fault isolation and adaptation is introduced to identify faulty estimates and to provide a correction where necessary. The FDIA framework presented here utilizes repeated trips along the same roads as a source of redundancy. Relevant properties of this formalism are given and verified experimentally using an equipped vehicle in rural and urban conditions and with various map faults. Results show that sequential FDIA performed well, even in difficult GNSS conditions.**

I. Introduction

Among the innovations that are transforming today's passenger vehicles, navigation maps are an important component. Maps were first introduced as part of navigation systems used to provide guidance information to the driver. Now they are used to provide context information to informative Advanced Driving Assistance Systems (ADASs) and their use has been extended to actuating ADASs [1]. Maps are also central components in the autonomous vehicles that are currently under development in the automotive industry [2]. Navigation maps are therefore playing an increasingly significant role in vehicle automation and progressively replacing the human driver as regards inferring the current and future vehicle context.

In recent years maps have sometimes been seen by the intelligent vehicle community as a perfect source of information. This assumption originates from robotics-oriented maps that were made manually with high accuracy, but this assumption is no longer valid when using global maps. The imperfections of a global map may not matter very much when the map is interpreted by a human, but they can have serious consequences as the degree of automation of the vehicle increases. Like any other source of information, navigation maps must be treated with caution.

How well the navigation map represents the geometry of the road has a direct impact on the performance of intelligent vehicle navigation systems. Knowledge of the geometry of the road ahead of the vehicle is currently used to improve sensor tracking (e.g., lane markings for lane-keeping functions, or a leading vehicle for adaptive cruise control applications) and enables hazardous situations to be anticipated, by adapting the vehicle speed (e.g., curve warning systems). Geometric information contained in the navigation map is also essential for some elements of highly automated driving, including path planning, decision making and control functions [3].

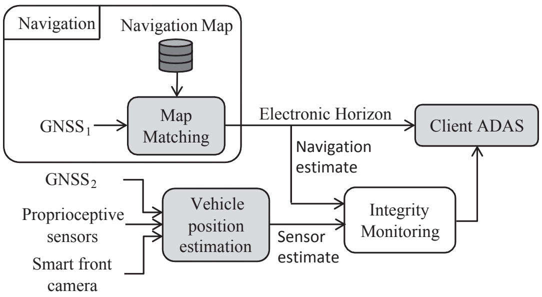


Figure 1. Framework for integrity monitoring in a passenger vehicle.

This paper presents a new method for detecting, isolating, and adapting geometrical errors in maps in order to avoid dysfunctions in client systems. Fig. 1 shows the system architecture used in the proposed approach. In passenger vehicles considered in this work, the navigation system provides context information to client ADASs. Because of industrial constraints, navigation systems cannot be modified for integrity monitoring purposes. A new *Integrity Monitoring* function (Fig. 1) is therefore added to monitor integrity in real time. The estimate of the vehicle position provided by the navigation function is continuously evaluated by comparing it to another position estimate. This sensor estimate is computed independently of the navigation system, using on-board vehicle sensors, and a fault is detected when these two estimates differ. One contribution of this work is to use Page's sequential statistical test to detect discrepancies between these two estimates despite the noise resulting from the use of standard vehicle sensors. When this test detects a discrepancy, ambiguity exists on the estimate affected by the fault (i.e. either the navigation, the independent estimate, or both can be faulty). This paper also develops a complete framework for overcoming this ambiguity by making use of repeated vehicle trips. Using a model of the effects of faults on estimates, fault isolation and adaptation is performed by comparing current and past position estimates. Structural properties of this formalism demonstrate that fault isolation capability improves as the number of trips increases, and that adaptation (i.e. the identification of a fault-free estimate that can serve as a correction) is possible when faults are isolated. Finally, the proposed method is tested using real data provided by a test vehicle in different driving conditions (rural and urban areas).

[1] Sorbonne Universités, Université de Technologie de Compiègne, CNRS Heudiasyc UMR 7253, France. [2] Renault S.A.S, France.

This article is organized as follows. Section II provides the theoretical context and background of this work. Section III states the problem that is addressed. Section IV introduces our fault detection method based on Page's test. Section V presents the proposed framework for fault isolation and adaptation, and details important inherent properties for intelligent vehicle applications. Experimental evaluation of the proposed method is performed with a test vehicle and is presented in Section VI. Conclusions of this work are discussed in Section VII.

II. Background

This section provides definitions of the terminology used, before describing work related to the evaluation of the integrity of each component of the navigation function.

A. Definitions

The terms *fault*, *error*, and *failure* may have different meanings according to the application domain. The following definitions are used in the context of this research and are based on those given in [4]:

- Fault: Error generative process. The presence of a fault may not lead to an error. An incorrect representation of the road network in the digital map of the navigation system is a fault that leads to an error only when the vehicle travels on the road where the fault is present.
- Error: A discrepancy between a computed, observed or measured value and the true, specified, or theoretically correct value.
- Failure: Instant in time when a required function exceeds the acceptable limits or is terminated.

Integrity is an important feature when navigation functions rely on Information technologies. Integrity can be defined as the ability of a system to provide user with accurate timely, complete and unambiguous information and warnings when the system should not be used.

B. Navigation System Integrity Monitoring

In intelligent vehicles, the navigation function provides relevant contextual information to client systems (ADASs or autonomous driving functions) in real time. This might be the distance to the next intersection, the curvature of the road ahead, or the current speed limit. This function can be schematized as having three parts, namely the localization system, the map-matching process and the navigation map. Map-matching consists in finding, within the navigation map, the road on which the vehicle is travelling, according to the position calculated by the localization system.

Localization in passenger vehicles is mainly based on Global Navigation Satellite Systems (GNSS). A GNSS receiver uses the time-of-flight measurements of electromagnetic signals emitted by satellites whose positions can be reconstructed using ephemeris data. The signals will sometimes be perturbed or reflected (i.e. multipath), which induces errors in the computation of position. The integrity risk arises from the use of faulty pseudo-ranges in this process. Classical integrity evaluation involves evaluating the coherency of the satellite measurements (fault detection) and then computing a protection level. This is Receiver Autonomous Integrity Monitoring (RAIM) [5]. It is, however, assumed that there is at most one faulty measurement at any one time, which is an unrealistically optimistic assumption in complex environments such as urban areas. Other approaches extend RAIM principles to a larger number of faulty measurements, using interval-based methods and relaxed intersections of constraints [6], [7] with fast implementations for fault detections [8], or an isotropy-based approach [9]. Terrain elevation models or building heights provided by a three-dimensional navigation map can also be used to determine the Non-Line-Of-Sight (NLOS) satellites, i.e. satellites that must be ignored in the position calculation [10], [11], [12]. The vehicle proprioceptive sensors (e.g., odometer, speedometer, gyroscope) are finally used to estimate the vehicle motion. However, since positional drift increases with time and distance, this technique is combined with GNSS using, for example, an extended Kalman filter [13], [14].

Integrity evaluation of a navigation map is a rather different problem which, unlike RAIM approaches, is not metric. A reference (i.e. ground truth) navigation map can be used to evaluate the vehicle map (subject map). In [15], fuzzy logic is used to compute an outlier index that expresses how a geographical object belongs to its spatio-temporal neighbourhood. This approach aims at detecting faults as well as temporal changes in maps. Studies were done on large-scale databases, in particular by crowdsourcing geographical information, like in the OpenStreetMap initiative [16], [17], [18]. Methods inspired from the SLAM (Simultaneous Localization And Mapping) domain can also be employed. The position information given by the navigation map is treated as an observation analogous to observations from other sensors [19]. To be considered as a ground truth, the reference navigation map must be created by an accurate, complete survey. In the literature some works have used alternative sources of information such as aerial imagery [20], [21], [22] or the mining of a large number of GNSS tracks [23] to create the reference map. These approaches assume that any disparities between the reference and subject maps are due to faults in the subject map, and do not address the possibility of faults in the reference map (due to an offset in aerial imagery or recent changes in the road network) or in both maps.

Integrity evaluation of the map matching process is highly dependent on the method used for the choice of road candidates. Monte Carlo-based approaches such as particle filters can be used when available computational resources allow. A set of particles (each representing a possible vehicle position) are spread over the whole road network. The population changes over time according to available measurements (e.g., GNSS measurement, DR estimation) and finally yields a solution [24], [25]. In [26], [27], the road candidates were represented by a set of hypotheses. A Bayesian framework was used to choose the most likely road. Evidence theory can also be used, since it is a convenient way to handle conflict in data fusion [28]. Fuzzy logic may also be considered, to address the complexity of the map-matching problem and the large number of criteria involved in choosing the road candidate. In [29], [30], the authors used a Sugeno fuzzy inference system to choose the road in the navigation map based on position uncertainty, the distance between the road and the vehicle position estimate, and the angular difference between the road direction and the vehicle heading. The vehicle navigation system usually provides a confidence index associated with the map-matched vehicle position. This corresponds to the final score of the optimization process employed in the map-matching, according to a given GNSS estimate and a given navigation map. However, this index should not be taken as a measure of the quality of the navigation map. In case of a sparse road network, the map-matching function is likely to provide a high-confidence index despite an offset of the road in the navigation map due to the low number of road candidates.

The concept of user-level integrity was introduced in [31] to emphasize the necessity of taking into account every step of the positioning process (GNSS, navigation map and map-matching) in the vehicle position integrity monitoring problem. The authors presented a strategy based on successive evaluation of GNSS integrity, map complexity and map-matching solution integrity. However, this requires having access to the internal data of every sub-function in the navigation system. In the approach presented here, functions are treated as black boxes due to industrial constraints. It is not possible for us to have access to low-level data such as the time-of-flight measurements, the complete navigation map data or internal variables of the map-matching algorithm. Only high-level data is available, such as the calculated vehicle position before and after map-matching, and the contextual information related to the current vehicle position. Consequently, system monitoring approaches can be appropriate.

Observer-based system monitoring consists in comparing outputs with estimations of the outputs based on the inputs. The residuals are signals that result from the difference between estimates and actual outputs [32]. These are null when the system is not affected by any faults. If a fault is activated, the residuals are non-null. When faults are detected, the consequences they have on the system are observed. A look-up table linking the different faults to their corresponding effects on the system would enable them to be identified unambiguously and therefore to be isolated and excluded and/or corrected from the system to keep it operating correctly or at a different level of performance. This kind of process is known as Fault Detection, Isolation, and Adaptation (FDIA). Based on the system model and the available measurements, a logical link between faults and residual values can be established and summarized in a signature matrix. A complete framework to detect multiple faults in a system was presented in [33]. The sensitivity of a set of residuals is determined using a system model, and diagnoses to be applied are established, based on the observed residuals. In this paper we develop a similar approach for an FDIA navigation system. Some kind of processing of the values of the residuals is essential when real signals are used. Because of the noise affecting them, different change detection strategies must be applied. An extensive description of the mathematical tools available for signal change detection can be found in [34].

The approach presented in [35] uses an architecture similar to the navigation systems studied in the present work. It showed that detecting unexpected large discrepancies between estimated and measured positions is not sufficient, since the noise associated with poor quality sensors creates an excessive sensitivity to outliers. A Cumulative Sum (CUSUM) test is therefore implemented to reduce the number of false alarms.

III. Problem Statement

A. Monitoring System

A systemic diagram of the proposed integrity monitoring system in a vehicle is shown in Fig. 2. Relevant information about the vehicle's current and future road environment is sent to the client systems. This information represents a set of context events encountered by the vehicle as it travels, and is consequently known as an Electronic Horizon (EH) [36].

The black box assumption that is made regarding the navigation system means that the only available observation of the road geometry is the map-matched position estimate denoted as N. The purpose of the method presented in this paper is to provide an indication of the integrity of the navigation system (in particular where road geometry faults are present in the map) to the systems that use this information. If a loss of integrity is detected, a correction can be provided to the client systems. To do so, an estimate of the vehicle position independent of the navigation system is required. This estimated position is denoted as G in the figure and computed using an additional GNSS receiver $GNSS_2$ based on a different technology than $GNSS_1$. Vehicle proprioceptive sensors (e.g. odometer and a yaw rate gyroscope) can be employed to improve its accuracy and availability. This estimation might also be affected by a fault. If the two estimates differ, there is an ambiguity in the faulty estimate. This ambiguity cannot be resolved, owing to the low level of redundancy (the degree of freedom being only one). The main idea behind this framework is to make use of repeated vehicle trips to resolve this ambiguity. The output *Knowledge of fault* (Fig. 2) has three possible values:

- *Use*. The estimate provided by the navigation function to client systems is not affected by any fault.
- *Unknown*. A fault has been detected but has not been isolated. The position estimate from the navigation system is possibly affected by a fault.
- *Don't use*. A fault affects the current estimate from the navigation system and the method provides a fault-free estimate to client systems through the output *Correction*.

Let us recall that the fault detection step is merely declaring that at least one of the estimates is affected by a fault. The isolation step is determining which estimate(s) is (are) affected by a fault.

B. Spatial Sampling

In our proposed approach, the integrity of the vehicle position estimate from the navigation system is spatially evaluated. Each location on the road network is considered as an operating point of the system to be monitored (i.e. the navigation system). For a given location of the vehicle, the presence of a fault is investigated using all the estimates recorded at this location during the course of vehicle trips.

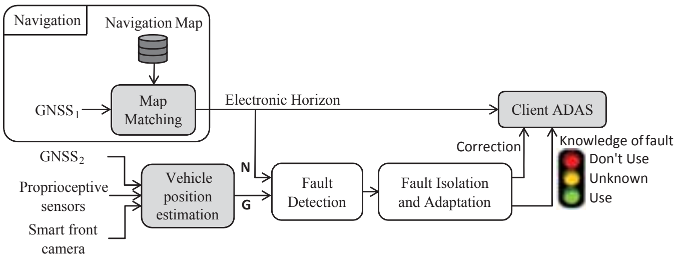


Figure 2. Structure of fault detection isolation and adaptation in a standard passenger vehicle.

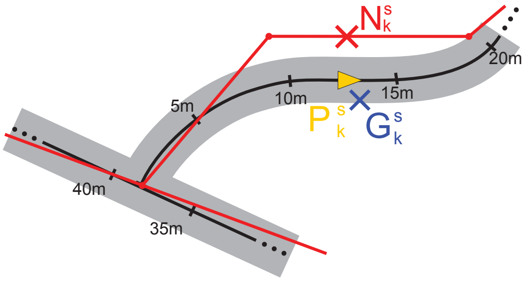


Figure 3. Illustration of the notational convention. The true (i.e. real) road is in grey and its centreline in black. Here it is spatially sampled with a 5-metre interval. The yellow arrow represents the true vehicle pose. The map is in red. The vehicle position as estimated by the navigation (resp. by the vehicle sensors) is the red (resp. blue) cross.

The method is spatially sampled with respect to the curvilinear abscissa of the road. The vehicle curvilinear abscissa on a given road is the distance along the carriageway with respect to its origin and is written $s \in \mathbb{R}^+$, as shown in Fig. 3.

Let $K \in \mathbb{N}$ denote the total number of trips made by the vehicle on a given road. The true vehicle position at abscissa s of a given road and at the k^{th} trip is written P_k^s. This can be encoded as a vector that contains the vehicle's geographic coordinates, that is to say longitude, latitude and ellipsoidal height.

Using the same notational convention, G_k^s and N_k^s are estimates of the vehicle position P_k^s provided by the sensors and the navigation respectively. Whenever the vehicle is at abscissa s of a given road for the k^{th} time, these two estimates are recorded.

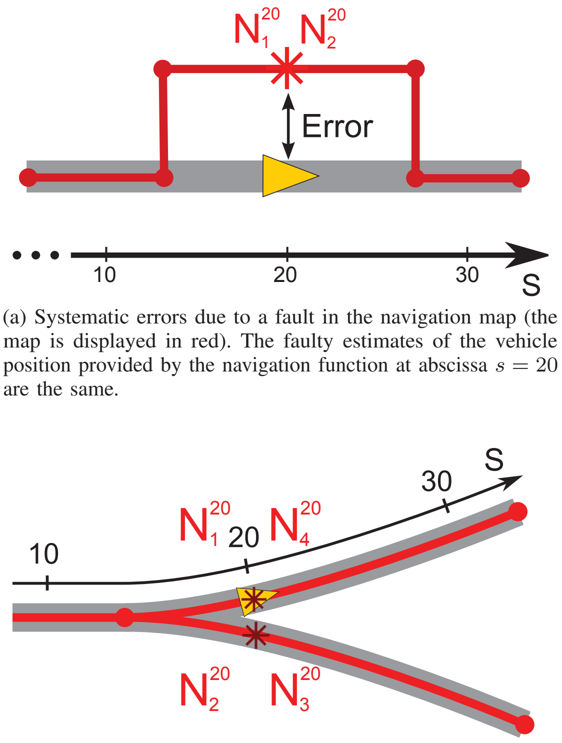
Faults may affect the navigation as well as the position estimate from sensors, and cause their value to be significantly different from the ground truth (if a multipath affects a GNSS receiver for example). In this case, the estimates are said to be faulty.

Let us define the faults $f_{N_k^s}$ and $f_{G_k^s}$ with:

$$f_{N_k^s} \overset{\text{def}}{=} \begin{cases} 1 & if\ N_k^s \neq P_k^s \\ 0 & otherwise \end{cases} \tag{1}$$

$$f_{G_k^s} \overset{\text{def}}{=} \begin{cases} 1 & if\ G_k^s \neq P_k^s \\ 0 & otherwise \end{cases} \tag{2}$$

Fig. 3 illustrates the notational convention. Since physical quantities cannot be strictly equal, a threshold on the distance between the estimates is employed for implementation. Since the true vehicle position is not measurable directly, the fault detection and isolation are based on a pairwise comparison of estimates.



(a) Systematic errors due to a fault in the navigation map (the map is displayed in red). The faulty estimates of the vehicle position provided by the navigation function at abscissa $s = 20$ are the same.

(b) Non-systematic faults in the navigation function. The map-matching chooses either the right or the wrong road from one trip to another. The navigation estimate may or may not be faulty. Errors on the faulty estimates are nevertheless equal (here N_2^{20} and N_3^{20}).

Figure 4. Navigation system fault and error characterization

C. Assumptions

A fault is an error generative process according to the definition stated in Section II-A. An error is therefore the discrepancy caused by the fault, and is measurable by an appropriate external observer. A faulty navigation system (from the client ADAS point of view) can result from a navigation map fault. Map faults cause systematic errors. Every time the vehicle traverses the area shown in Fig. 4a, the navigation system will provide the same faulty position estimate.

A faulty state of the navigation system may not only be a consequence of map faults. Map-matching may choose a wrong road candidate because of a difficult road configuration such as a junction, as shown in Fig. 4b. In this situation the map-matching may or may not choose the right road segment from one trip to another; faults are not systematic. Nevertheless, it will be remarked that where there is a fault, the resulting error is always the same, since the output of the navigation estimate is constrained by the map. Errors by the navigation system, when they occur, are therefore systematic.

In this work we do not address the problem of determining the reason for an estimation error made by the navigation, because this would require access to the internal variables of the system.

The estimate of the vehicle position from the on-board vehicle sensors G depends mainly on the GNSS estimation. Given a location on the road network, faults on a G estimate will have two principal causes: first, multipath (i.e. satellite signal reflection on buildings, for instance) and, second, a poor satellite configuration. The magnitude of the day-to-day multipath correlation of a static receiver is typically around 85% [37]. We have no knowledge of correlation values for a moving receiver, since the receiver motion mitigates the multipath effects. Moreover, a GNSS multipath error can repeat itself only at the same place with the same satellite configuration (the ground track repeat is 23hr 56min for GPS and 10 days for Galileo). Therefore, we believe that a repetition of the same multipath errors and faults is very unlikely from one trip to another. Errors and faults due to poor satellite geometry (multipath aside) result from the propagation of pseudo-range random measurement errors (dilution of precision), and therefore also have weak correlation between two vehicle trips. Errors on G are assumed to be different from one trip to another.

The assumptions underlying the FDIA framework can be summarized as follows:

- When travelling several times on a road, the vehicle follows the same path with small deviations (which can be compensated if necessary by lane marking measurements from a front camera).
- Any fault affecting position estimates from sensors can cause errors of any values. Errors on faulty vehicle position estimates from sensors are different from one trip to another at a given abscissa.
- The navigation map does not change from one trip to another. Errors on the vehicle position estimates from the navigation (when they occur due to a fault) are therefore always the same at a given abscissa.
- Errors on the vehicle position estimates from sensors and from the navigation are independent of each other at a given abscissa.

Given these assumptions, we have:

$$P_i^s = P_{i+1}^s, \, \forall i \in \{1, \ldots, K-1\} \tag{3}$$

Where P is the true vehicle position, s is the curvilinear abscissa, i is the trip index and K is the total number of trips made on the road. In the sections below, the formalism is developed from the system monitoring point of view, putting temporarily aside the application to intelligent vehicles. The estimates from sensors and from the navigation (G and N respectively) are seen as estimates of the same physical quantity P, which is in accordance with the assumptions above. The curvilinear abscissa is understood as an operating point of the system to be monitored and the vehicle trips are iterations of this system.

IV. Sequential Fault Detection

The first step in FDIA consists in detecting faults by comparing the two position estimates N and G. According to the assumptions stated previously, a significant discrepancy between the estimates indicates that a fault affects at least one of them. However, noise on estimates may cause non-faulty estimates to be different from each other and induce false alarms in the detection process. For this reason, this section details the mathematical formulation of a probabilistic sequential test (called Page's test) and its application to the detection of discrepancies between position estimates.

Statistical tests are an appropriate tool for evaluating the parameters of a probability law based on set of outcomes. In our application, we seek to detect a change in the mean of the probability density function (PDF) of a set of observed data, while the standard deviation of this PDF is of the same order of magnitude as the expected mean gap. Page's test works sequentially and is especially efficient for stream data. The problem is therefore formulated as the detection of a change in the mean of a random variable that represents the distance between the estimates from sensors and from navigation.

Page's test (also known as Page's trend test) consists in statistically detecting a change in the mean of a random variable based on a likelihood ratio of hypotheses [34]. It also identifies the sample at which the change in the mean occurred. Formulation of this test in the context of map fault detection is detailed in [38].

The random variable tested here is the distance between estimates G and N. Let us see how the distance signal is generated and described in terms of mean and standard deviation. Let us consider the estimate N from the navigation as the result of a random process based on the true vehicle position P in a frame $\mathcal{R}_1$ aligned with the road:

$$N = P + \alpha \tag{4}$$

$$\Sigma_\alpha = \begin{bmatrix} \sigma_a^2 & 0 \\ 0 & 0 \end{bmatrix}_{\mathcal{R}_1} \tag{5}$$

where α is a noise assumed to be zero-mean, with a diagonal covariance matrix Σ_α. Since roads are represented in the navigation map by zero-width poly-lines, the variance of the navigation map-matched error normal to the road segment is by definition null. However, a map-matched position error along the road segment exists, and σ_a denotes the longitudinal standard deviation of the navigation estimate.

The estimate of the vehicle position from sensors G can be encoded as a two-dimensional point $G = (x, y)^T$ in the East-North plane $\mathcal{R}_0$ locally tangent to Earth with the covariance matrix Σ_β of the estimation error β provided by the localization system:

$$G = P + \beta \tag{6}$$

$$\Sigma_\beta = \begin{bmatrix} \sigma_x^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_y^2 \end{bmatrix}_{\mathcal{R}_0} \tag{7}$$

In order to make the distance signal independent of the road direction, an isotropic approach is used, and this consists in using the outer circle of the ellipsoid. Its radius is $\eta = \max(\eta_i)$, η_i being the eigenvalues of Σ_β. So, the covariance matrix expressed in $\mathcal{R}_1$ is $\eta \cdot I$ (with I being the identity matrix).

The vector L is defined to be the difference between the map-matched and estimated positions as stated by the following equation. L has two independent components, namely lateral error d and longitudinal error e.

$$L = \left[\begin{array}{c} d \\ e \end{array} \right] = N - G = \alpha - \beta \tag{8}$$

Under the hypothesis of independent errors, the signals d and e have the following variances:

$$\begin{array}{c} \sigma_d^2 = \eta \\ \sigma_e^2 = \eta + \sigma_a^2 \end{array} \tag{9}$$

The most relevant information is the lateral position of the roads in the navigation map. The fault detection is therefore done by detecting mean changes in the signal d using η.

V. Fault Isolation and Adaptation Method

Once a fault has been detected, the problem is now to isolate the faulty estimates and perform adaptation according to the assumptions made previously, using the repetition of vehicle trips as a source of redundancy. The adaptation process consists in providing a non-faulty estimate to a client system so that it can continue to operate normally, even if the current estimate is affected by a fault. Non-faulty estimates therefore need to be identified unambiguously. The concepts of Sets of Faults and Residuals are defined first. The mathematical relationship between these two concepts is then demonstrated. Finally, we show how a set of faults can be isolated, based on the observation of residuals.

A. Definitions

1) Sets of Faults: Let the set of faults e_K^s be composed of all $f_{G_k^s}$ and $f_{N_k^s}$ for the considered iterations K at a given abscissa s:

$$e_K^s \overset{\text{def}}{=} \left\{ f_{G_i^s} \, , \, f_{N_j^s} \right\} \, , \, \forall i,j \in \{1,\ldots,K\} \tag{10}$$

The cardinality of e_K^s is $2K$. Each element of e_K^s is a boolean value so there are 2^{2K} possible sets written $e_{K,n}^s$:

$$\left\{ \begin{array}{c} e_K^s \in \left\{ e_{K,n}^s \right\} \\ e_{K,n}^s \in \mathbb{B}^{2K} \, , \, \forall n \in \left\{ 1,\ldots,2^{2K} \right\} \end{array} \right. \tag{11}$$

Let us take an example with $K = 2$. There are $2^{2 \cdot 2} = 16$ different sets. The cardinality of each one is $2 \cdot 2 = 4$. For instance, $e_{2,5}^s = \left\{ \begin{array}{cccc} 0 & 0 & 1 & 0 \end{array} \right\}$ means $\left\{ f_{G_2^s} = 0 \right.$ and $f_{N_2^s} = 0$ and $f_{G_1^s} = 1$ and $\left. f_{N_1^s} = 0 \right\}$.

2) Residual Processing: At a given abscissa s and at system iteration K, every available estimate at the current iteration is compared to all the others and the result is stored in a residual vector R_K^s. The elements of R_K^s are defined as:

$$r_{G_i^s G_j^s} \overset{\text{def}}{=} \left\{ \begin{array}{cc} 1 & if \; G_i^s \neq G_j^s \\ 0 & otherwise \end{array} \right. \; \forall i,j \in \{1,\ldots,K\}, i > j \tag{12}$$

$$r_{G_i^s N_j^s} \overset{\text{def}}{=} \left\{ \begin{array}{cc} 1 & if \; G_i^s \neq N_j^s \\ 0 & otherwise \end{array} \right. \; \forall i,j \in \{1,\ldots,K\} \tag{13}$$

$$r_{N_i^s N_j^s} \overset{\text{def}}{=} \left\{ \begin{array}{cc} 1 & if \; N_i^s \neq N_j^s \\ 0 & otherwise \end{array} \right. \; \forall i,j \in \{1,\ldots,K\}, i > j \tag{14}$$

Equations (12) and (14) are restricted to $i > j$ to avoid useless redundant residuals.

R_K^s is therefore composed of $C\,(2K, 2)$ boolean elements, where $C\,(2K, 2)$ stands for the number of 2-combinations from a given set of $2K$ elements. We know that $C\,(2K, 2) = K\,(2K-1)$ so the residual vector therefore contains $K\,(2K-1)$ elements.

For example, at the second iteration, the size of R_2^s is 6:

$$R_2^s = \left[\begin{array}{cccccc} r_{N_2^s G_2^s} & r_{G_2^s G_1^s} & r_{N_1^s G_2^s} & r_{N_2^s G_1^s} & r_{N_1^s N_2^s} & r_{N_1^s G_1^s} \end{array} \right] \tag{15}$$

If, for example, the estimates are such that $G_1^s \neq N_1^s = G_2^s = N_2^s$ then the residual vector is $R_2^s = \left[\begin{array}{cccccc} 0 & 1 & 0 & 1 & 0 & 1 \end{array} \right]$.

3) Relationships Between Faults and Residuals: Let $\vee$ and $\oplus$ denote boolean *or* and *exclusive or* operators respectively.

Proposition 1. *The elements of the residual vector are the result of boolean operations between the faulty states of the estimates, according to the following equations:*

$$r_{G_i^s G_j^s} = f_{G_i^s} \vee f_{G_j^s} \, , \, \forall i,j \in \{1,\ldots,K\} \, , \, i > j \tag{16}$$

$$r_{G_i^s N_j^s} = f_{G_i^s} \vee f_{N_j^s} \, , \, \forall i,j \in \{1,\ldots,K\} \tag{17}$$

$$r_{N_i^s N_j^s} = f_{N_i^s} \oplus f_{N_j^s} \, , \, \forall i,j \in \{1,\ldots,K\} \, , \, i > j \tag{18}$$

The demonstration of this proposition is developed in [39].

Eq. (16), (17) and (18) of Proposition 1 establish a link between the available estimates (i.e. G and N) and the faults which affected them (i.e. f_G and f_M). The first two equations tell us that if there is at least one fault on the considered estimates, then the residual is affected. In (18), the residual is equal to one if there is a single fault among the two estimates. It is now possible to deduce the presence of faults based on observation and comparison of the estimates.

B. Fault Isolation Principles

The fault detection and isolation strategy involves listing all the possible sets of faults for a given iteration K, and calculating the corresponding theoretical residual vectors with (16), (17) and (18). This forms the truth table for K. In parallel, available estimates are used to compute the observed residual vector based on (12), (13) and (14). This vector, present in the truth table, allows the corresponding set of faults to be determined. Faults affecting each estimate can finally be deduced from this set. It will be remarked that the truth tables are valid for every operating point, so the superscript s is omitted in the tables.

Let us take the example given in Section V-A2. At the first system iteration at operating point s, two estimates are available: G_1^s and N_1^s. The truth table for one iteration is shown in Table I. It is assumed in this example that $G_1^s \neq N_1^s$ is observed, therefore $r_{G_1^s N_1^s} = 1$, according to (12). Table I shows that this residual can be due to three sets of faults: $e_{1,2}^s$, $e_{1,3}^s$ and $e_{1,4}^s$. After one system iteration, it can be concluded that there is at least one faulty estimate among G_1^s and N_1^s, but it is not possible to determine which one. The fault is detected, but not isolated.

Table I
TRUTH TABLE FOR ONE ITERATION ($K = 1$). THE FIRST RESIDUAL $r_{G_1N_1} = 0$ APPEARS ONLY ONCE IN THE TABLE, AND SINCE THIS MAKES ISOLATION POSSIBLE, IT IS SHOWN IN GREEN. CONVERSELY, $r_{G_1N_1} = 1$ IS DUE TO MORE THAN ONE SET OF FAULTS AND IS SHOWN IN RED. THE RESIDUAL USED AS EXPLANATION EXAMPLE IS IN BOLD.

Sets of faults $e_{K,n}$			Residuals
	f_{G_1}	f_{N_1}	$r_{G_1N_1} = f_{G_1} \vee f_{N_1}$
$e_{1,1}$	0	0	0
$e_{1,2}$	1	0	***1***
$e_{1,3}$	0	1	***1***
$e_{1,4}$	1	1	***1***

At the second system iteration at the operating point s, a new pair of estimates is available: G_2^s and N_2^s. The truth table for two system iterations is calculated with (16), (17) and (18) and is shown in Table II. In this example and similarly to Section V-A2, it is assumed that $G_1^s \neq N_1^s = G_2^s = N_2^s$ is observed. This leads to the residual $R_2^s = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$. Table II shows that this residual (in bold) is exclusively due to the set of faults $e_{2,5}^s$. After the second system iteration, fault isolation is performed by concluding that $\left\{ f_{G_2^s} = 0 \text{ and } f_{N_2^s} = 0 \text{ and } f_{G_1^s} = 1 \text{ and } f_{N_1^s} = 0 \right\}$.

C. Conditions of Isolability

By definition, the truth table is exhaustive; the observed residual vector is necessarily included within it. However, some sets of faults induce the same residual vector, as shown by the red colour in Tables I and II. In this case, isolation is not possible. These are called *Adverse* sets. At least one more system iteration is required to perform isolation.

Being adverse depends on the number of faults affecting the estimates, as stated in Proposition 2.:

Proposition 2. *A set of faulty states is adverse if and only if it corresponds to one of the following conditions:*

$f_{N_i} = 1\,,\ \forall i \in \{1, \dots, K\}$ *and* $\exists! j \in \{1, \dots, K\}$ *such that* $f_{G_j} = 0$

$f_{G_i} = 1\,,\ \forall i \in \{1, \dots, K\}$

In other words, it is not possible to isolate faults if:

1) Every estimate N is faulty and there is a single fault-free G.
2) Every G is faulty.

The proof of this Proposition can be found in [40].

It will be remarked in the example developed previously that after the first system iteration (i.e. $K = 1$), the situation corresponded to the second condition of this proposition because $f_{N_1^s} = 0$ and $f_{G_1^s} = 1$. This is why fault isolation was impossible. However, after the second iteration, the set chosen for the example $\left\{ f_{G_2^s} = 0 \text{ and } f_{N_2^s} = 0 \text{ and } f_{G_1^s} = 1 \text{ and } f_{N_1^s} = 0 \right\}$ no longer corresponded to either of these conditions. Fault isolation had therefore become possible.

Proposition 2 is fundamental for demonstrating the internal formalism properties. These are detailed and demonstrated as follows.

D. Formalism Properties

Once the bases of the formalism are established, we have the properties shown below in the listed propositions. Demonstrations of these properties can be found in [39].

Proposition 3. *Guaranteed fault detection: The formalism always detects the presence of faulty estimates. In other words, each time there is a faulty estimate, the formalism detects it (but may not be able to isolate the faulty estimate).*

Proposition 4. *Isolation convergence: The ratio of the number of adverse sets of faulty states to the total number of sets tends to zero as the number of iterations increases. In other words, increasing K improves fault isolation capabilities.*

Proposition 5. *Conservation of isolability: Once fault isolation is performed, fault isolation will be performed at any new iteration.*

Proposition 6. *Adaptation: If fault detection and isolation are performed, then adaptation is possible.*

It should be recalled that adaptation consists in identifying a fault-free estimate once detection and isolation have been performed.

Proposition 7. *Conservation of adaptation: If fault isolation is achieved at the K^{th} iteration, adaptation is possible at iteration $K + 1$ whatever the faults affecting the new estimates.*

These propositions have important consequences for the application of the method in intelligent vehicles. First, Proposition 3 shows that the presence of a fault among the available estimates is always detected by the method. This means that where there is no fault, the method is able to declare this fact with certainty even at the first system iteration, allowing client systems to function. Integrity monitoring is therefore possible with this method. Second, Proposition 4 shows that a new iteration will always contribute information for fault isolation, which justifies multiple system iterations. Third, according to Propositions 5 to 7, once a fault has been isolated, a fault-free estimated can be provided to client systems at any future iteration, allowing client systems to anticipate being able to operate properly at any future iteration.

E. Illustrative Example

We now take the FDIA formalism proposed above and apply it to monitoring the integrity of the navigation vehicle position estimate as introduced in Fig. 2. Using a simple example, each step is described in detail. The map contains an error and we show how the method performs fault detection, isolation and adaptation. In addition to detailing each step of our proposed method, we illustrate the properties introduced in Section V-D.

In this example (depicted in Fig. 5), the real road is straight, while the map's representation of the road includes a bend. Let us detail the proposed formalism at abscissa 25 m in the first trip shown in Fig. 5a.

The first time the vehicle is at abscissa $s = 25$, position estimates are provided by the vehicle state (G_1^{25}) and by the navigation (N_1^{25}) functions. The observed residual can be computed using (13):

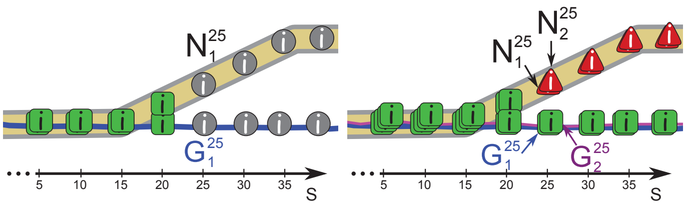
$$G_1^{25} \neq N_1^{25} \Rightarrow r_{G_1^{25}N_1^{25}} = 1$$

This residual is found three times in the truth table for one FDIA trip (Table I): the sets of faults $e_{1,2}^{25}$, $e_{1,3}^{25}$ and $e_{1,4}^{25}$ give

Table II

TRUTH TABLE FOR TWO ITERATIONS ($K = 2$). RESIDUALS OCCURRING ONLY ONCE ARE IN GREEN, SINCE THEY MAKE ISOLATION POSSIBLE. CONVERSELY, RESIDUALS THAT ARE DUE TO MORE THAN ONE SET ON FAULTY STATES ARE IN RED. THE RESIDUAL USED AS EXPLANATION EXAMPLES IS IN BOLD.

	Sets of faults $e_{K,n}$				Residuals					
	f_{G_2}	f_{N_2}	f_{G_1}	f_{N_1}	$r_{N_2G_2}$	$r_{G_2G_1}$	$r_{N_1G_2}$	$r_{N_2G_1}$	$r_{N_1N_2}$	$r_{N_1G_1}$
$e_{2,1}$	0	0	0	0	0	0	0	0	0	0
$e_{2,2}$	1	0	0	0	1	1	1	0	0	0
$e_{2,3}$	0	1	0	0	1	0	0	1	1	0
$e_{2,4}$	1	1	0	0	1	1	1	1	1	0
$e_{2,5}$	0	0	1	0	***0***	***1***	***0***	***1***	***0***	***1***
$e_{2,6}$	1	0	1	0	1	1	1	1	0	1
$e_{2,7}$	0	1	1	0	1	1	0	1	1	1
$e_{2,8}$	1	1	1	0	1	1	1	1	1	1
$e_{2,9}$	0	0	0	1	0	0	1	0	1	1
$e_{2,10}$	1	0	0	1	1	1	1	0	1	1
$e_{2,11}$	0	1	0	1	1	0	1	1	0	1
$e_{2,12}$	1	1	0	1	1	1	1	1	0	1
$e_{2,13}$	0	0	1	1	0	1	1	1	1	1
$e_{2,14}$	1	0	1	1	1	1	1	1	1	1
$e_{2,15}$	0	1	1	1	1	1	1	1	0	1
$e_{2,16}$	1	1	1	1	1	1	1	1	0	1



(a) First trip (blue line)

(b) Second trip (purple line)

Figure 5. A faulty map area. Circular grey marks are for estimates where the method has detected but not isolated a fault. Green squares are for true estimates and red triangles are the faulty estimates.

$r_{G_1N_1} = 1$. The proposed method then detects a faulty estimate among G_1^{25} and N_1^{25} but is not able to isolate it. The integrity monitoring system cannot specify the faultiness of N_1^{25}, but simply sends *Knowledge of fault: Unknown* to client systems, as shown by circular grey marks in Fig. 5a.

The second time the vehicle traverses abscissa $s = 25$ of the same road (Fig. 5b), a new pair of position estimates becomes available: G_2^{25} and N_2^{25}. The dimension of the residual vector increases to 6. The elements are calculated using (12), (13) and (14):

$$N_2^{25} \neq G_2^{25} \Rightarrow r_{N_2^{25}G_2^{25}} = 1$$

$$G_2^{25} = G_1^{25} \Rightarrow r_{G_2^{25}G_1^{25}} = 0$$

$$N_1^{25} \neq G_2^{25} \Rightarrow r_{N_1^{25}G_2^{25}} = 1$$

$$N_2^{25} \neq G_1^{25} \Rightarrow r_{N_2^{25}G_1^{25}} = 1$$

$$N_1^{25} = N_2^{25} \Rightarrow r_{N_1^{25}N_2^{25}} = 0$$

$$G_1^{25} \neq N_1^{25} \Rightarrow r_{G_1^{25}N_1^{25}} = 1$$

Then $R_2^{25} = \left[\begin{array}{cccccc} 1 & 0 & 1 & 1 & 0 & 1 \end{array}\right]$.

Table II is the truth table for two trips. Following the first trip observation it is known that $f_{G_1^{25}}$ and $f_{N_1^{25}}$ are not both null, and the first four rows of Table II may therefore be ignored. The observed residual is found only once in this table (caused by the set of faults $e_{2,11}^{25}$). Consequently, it can be concluded that $f_{G_2^{25}} = 0$, $f_{N_2^{25}} = 1$, $f_{G_1^{25}} = 0$ and $f_{N_1^{25}} = 1$.

The integrity monitoring system returns the instruction *Knowledge of fault: don't use the navigation position estimate* (N_2^{25}) and provides a fault-free estimate instead in the output *Correction* (either G_2^{25} or G_1^{25}). On Fig. 5b, faulty (resp. true) estimates are represented by red triangles (resp. green squares). From Proposition 7 we know that the integrity monitoring system will be able to perform adaptation, i.e. provide an error-free position estimate for all future trips along this road, whatever the faults affecting the future estimates.

F. Complete Fault Detection, Isolation and Adaptation Method

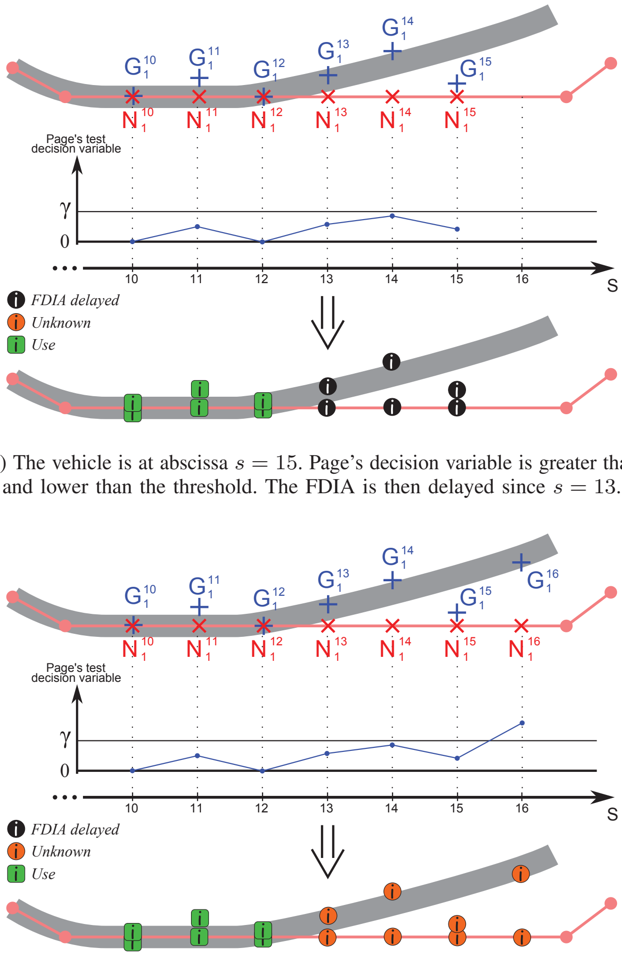
The FDIA framework introduced previously is based on the calculation of a residual vector R_K^s (s is the vehicle curvilinear abscissa on the road and K is the number of trips on this road). In practice, the elements of R_K^s are defined on the basis of comparisons of the distance between each pair of available estimates N and G with a threshold λ_d, and denoted by $r_{G_i^sG_j^s}$, $r_{G_i^sN_j^s}$ and $r_{N_i^sN_j^s}$, $\forall i,j \in \{1,\ldots,K\}$ which we recall below:

$$r_{G_i^sG_j^s} = \left\{ \begin{array}{ll} 1 & if\ dist\left(G_i^s, G_j^s\right) > \lambda_d \\ 0 & otherwise \end{array} \right. \quad \forall i,j \in \{1,\ldots,K\}, i > j \tag{19}$$

$$r_{G_i^sN_j^s} = \left\{ \begin{array}{ll} 1 & if\ dist\left(G_i^s, N_j^s\right) > \lambda_d \\ 0 & otherwise \end{array} \right. \quad \forall i,j \in \{1,\ldots,K\} \tag{20}$$

$$r_{N_i^sN_j^s} = \left\{ \begin{array}{ll} 1 & if\ dist\left(N_i^s, N_j^s\right) > \lambda_d \\ 0 & otherwise \end{array} \right. \quad \forall i,j \in \{1,\ldots,K\}, i > j \tag{21}$$

Page's test is used here instead of the distance measure for comparing G and N. According to this new formulation, the residual vector element $r_{G_K^sN_K^s}$ is zero if Page's test gives the mean of the signal d as zero. Reciprocally, $r_{G_K^sN_K^s}$ is one if the test detects a mean change in d. The manner in which the other residual elements ($r_{G_i^sG_j^s}$ and $r_{N_i^sN_j^s}$) are calculated remains unchanged.



(a) The vehicle is at abscissa $s = 15$. Page's decision variable is greater than 0 and lower than the threshold. The FDIA is then delayed since $s = 13$.

(b) Page's decision variable exceeds the threshold at $s = 16$. FDIA is run for abscissas 13 to 16 with $r_{G_1 N_1} = 1$. This results in *Knowledge of fault: Unknown*.

Figure 6. Example of the use of Page's test with the FDIA framework. The road as recorded in the navigation map is the red poly-line. The estimates from navigation N_1^s are the red crosses and the estimates from sensors G_1^s are the blue crosses. The vehicle is travelling from left to right, so its curvilinear abscissa is denoted by the axis S. The decision variable used in Page's test is plotted on the middle graph. The result of FDIA with Page's test is shown in the lower parts of the figures.

As shown in Section IV, Page's test may require a few samples before it is able to give definitive results. This is highlighted by the distance-to-alert and distance-to-recovery metrics. In such a situation, the estimates are buffered until Page's test provides a definitive output. The FDIA framework is then run on each estimate, taking the result of the test as an input.

As an example, Fig. 6 shows the progression of Page's test as the vehicle advances, and illustrates the strategy employed here for the FDIA. We are looking at the first trip along this road. The method is detailed step by step, as shown in this figure. When the vehicle is at abscissa $s = 10$, the estimates from sensors G_1^{10} and N_1^{10} are very close to each other. The Page's test decision variable equals zero, and therefore the estimates are considered to be equal. The FDIA framework is run with $r_{G_1^{10} N_1^{10}} = 0$: this residual is found only once in the truth table so it is not necessary to use previous trips, and the FDIA concludes that the estimates are fault-free. At abscissa $s = 11$, the Page's test decision variable is not null, but has not yet reached the threshold γ. A discrepancy between the estimates G and N is likely, but not yet detected. The estimates that correspond to abscissa $s = 11$ (i.e. G_1^{11} and N_1^{11}) are stored in the memory buffer until the Page's decision variable either exceeds the threshold or returns to zero. When the vehicle reaches abscissa $s = 12$, the estimates G_1^{12} and N_1^{12} make the decision variable return to zero, which indicates that Page's test gives no discrepancy for the two last abscissas. The FDIA is run at every abscissa in the memory buffer: at $s = 11$ with $r_{G_K^{11} N_K^{11}} = 0$, the FDIA concludes that there is no fault affecting G_1^{11} and N_1^{11}; at $s = 12$ with $r_{G_K^{12} N_K^{12}} = 0$, the FDIA concludes that there is no fault affecting G_1^{12} and N_1^{12}.

The memory buffer is cleared. At $s = 13$ to $s = 15$, the estimates are such that the decision variable is not null so these are buffered as shown by black marks on Fig. 6a. At abscissa $s = 16$ the decision variable finally exceeds γ, and so Page's test now declares that a discrepancy, starting at $s = 13$, is detected. The FDIA is then successively run at $s = 13$, $s = 14$, $s = 15$ and $s = 16$ with $r_{G_1^{13} N_1^{13}} = 1$, $r_{G_1^{14} N_1^{14}} = 1$, $r_{G_1^{15} N_1^{15}} = 1$ and $r_{G_1^{16} N_1^{16}} = 1$ respectively. Since these residuals are adverse and there are no previous trips available, the FDIA outputs *Unknown* for the estimates, as shown by orange marks on Fig. 6b. The memory buffer is emptied and the decision variable is set to zero for the following evaluation points.

Figure 7. Test vehicle

VI. Experimental Evaluation

A. Test Vehicle

Experiments were done in real conditions using the Renault Espace passenger vehicle shown in Fig. 7. The navigation system used in the vehicle is fed by a standard single frequency Ublox 6T GPS receiver (corresponding to $GNSS_1$ in Fig. 2). The GNSS receiver denoted by $GNSS_2$ in Fig. 2 is a Ublox 4T GPS receiver. The vehicle odometer, speed, rear wheel speed difference and yaw rate are production-standard sensors and are available on the vehicle CAN-bus. An extended Kalman filter is used to compute the position estimate from sensor G, based on the vehicle sensors and the Ublox 4T GPS receiver [38].

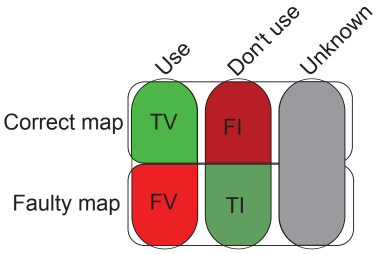


Figure 8. Metrics employed for method evaluation. The columns are the output of the method and the lines are the actual state of the navigation map.

An Ixsea LandIns Inertial Navigation System (INS) tightly coupled to a Novatel GPS receiver provides position estimates with an error of less than 1 m and is considered as position ground truth for the experiments.

B. Metrics

We saw above that our proposed method has three possible output states that refer to the current navigation estimate integrity, namely *Use, Unknown, Don't use.* The navigation map can be *correct* or *faulty*. A set of metrics is introduced as follows and illustrated in Fig. 8. These are evaluated with respect to the number of vehicle trips so that the performance of our method can be evaluated precisely.

The overall efficiency corresponds to the number of relevant diagnoses made by the method, equal to the sum of True Validations (TV) and True Isolations (TI). A TV occurs when a correct point of the navigation map has been declared with no fault. A False Validation (FV) occurs when the method trusts a faulty navigation estimate. A False Isolation (FI) occurs when a correct navigation estimate is classified as faulty by the method. The Overall Efficiency Rate (OER) is:

$$OER \overset{\text{def}}{=} \frac{TV + TI}{\Omega - \Omega_{unknown}} \tag{22}$$

where Ω is the number of navigation points evaluated by the method and $\Omega_{unknown}$ is the number of navigation estimates for which the method outputs *Unknown*. An OER close to one would indicate that whenever the method provides an output different from *Unknown*, this diagnosis is reliable.

The output *Unknown* does not provide information on the integrity of the navigation estimate from the point of view of client systems. From the applicative point of view, this output should occur as little as possible. The performance of the method in terms of information availability is measured by the Information Availability Rate (IAR):

$$IAR \overset{\text{def}}{=} \frac{\Omega - \Omega_{unknown}}{\Omega} \tag{23}$$

This is expected to converge to one as the number of trips increases.

C. Urban Test Track

In this experiment, the vehicle was driven close to large buildings. The GPS receiver was perturbed by multipath effects caused by signals reflecting off buildings. These measurements are expected to be isolated by the method. As shown in Fig. 9, over parts of the circuit conditions are good, and the deviation of the GPS measurements is less than the width of the road. It will be remarked that for testing the method, these experimental conditions are challenging. The length of each trip is 1100 m and the spatial sampling has been done along the map with a 10 m period, and the tolerance on the vehicle curvilinear abscissa is $\lambda_s = 2$ m. Hence, $\Omega = 110$ points on the navigation map need to be evaluated at each trip. This value varies by a few points from one trip to another because data recordings were not started and stopped rigorously at the same positions. The threshold on the distance between the estimates must be chosen according to two criteria. First, it must be as small as possible to comply with assumptions made as bases for the method. Second, it must be greater than the tolerance on the vehicle abscissa λ_s, so that two estimates from navigation that correspond to the same abscissa are considered as equal by the method. Page's test is therefore set to detect a discrepancy of $\delta_m = \lambda_s = 2$ m between the estimates with the detection threshold $\gamma = 4.\sigma/\delta_m$.

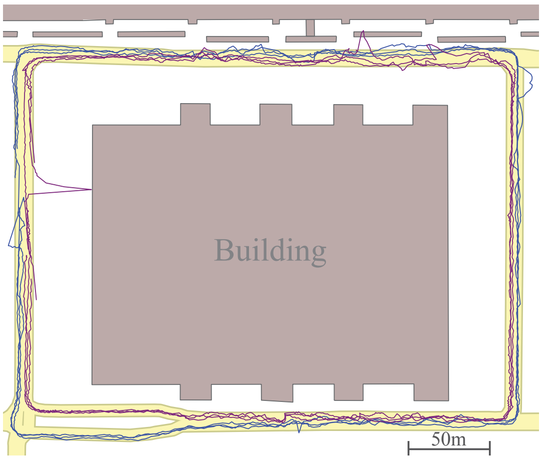


Figure 9. GNSS tracks on the correct navigation map. Clockwise (blue) and anticlockwise (purple)

Faults were generated randomly in five different maps using dedicated software. The performance of the complete FDIA method is evaluated using the metrics introduced previously and detailed in Fig. 10 and 11.

Fig. 10 shows the ratio of correctly identified points to the number of isolated or validated points. At the first vehicle trip the method cannot perform isolation. The OE is then only composed of TV. The OER at the first trip is therefore favoured by the absence of false validation; the OER of five of the ten tests therefore equal one. It will be noted that the OER of map 1 anticlockwise is especially low at the first trip (50%), but this is not significant since it is calculated using only four points. The OER tends to remain constant from the second to the third trip with medians equal to 84% and 83% respectively.

Fig. 11 summarizes the ratio of the number of validated or isolated points to the number of points considered *Unknown*. The IAR increases with the number of trips for all the tests and exceeds 90% at the third trip. The FDIA method is therefore seen to converge as stated by Proposition 4.

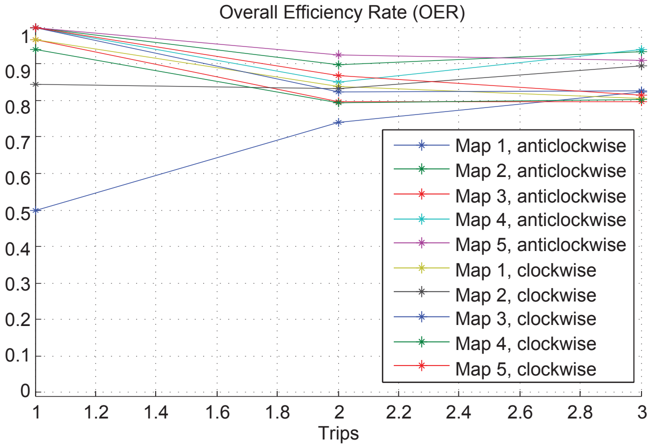


Figure 10. Overall Efficiency Rate.

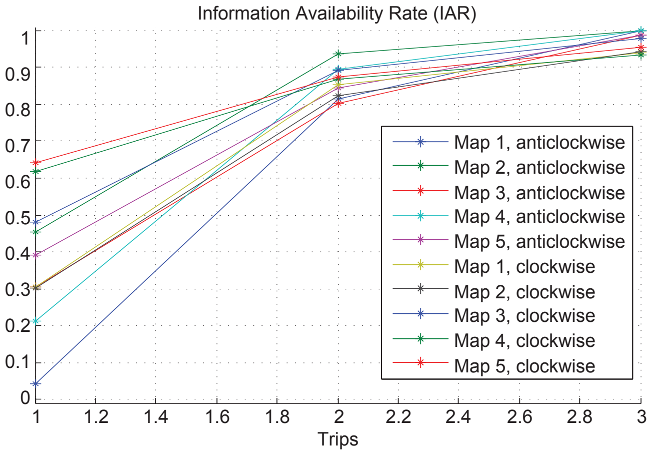


Figure 11. Information Availability Rate.

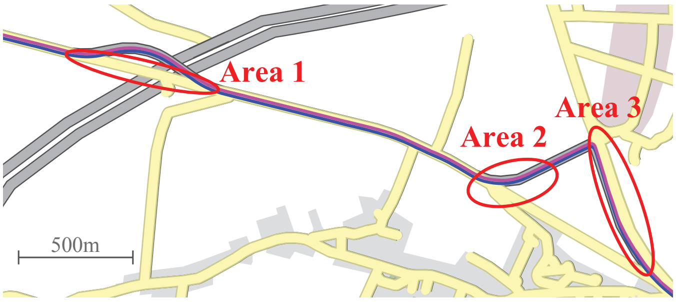


Figure 12. Rural test track. The navigation map used in the experiment is in yellow, the correct map is in grey in background. The vehicle goes from left to right. The estimates from sensors of the first (resp. second) trip is in blue (resp. purple).

D. Rural Test Track

Here we look at how the method performed in an area where real map errors were present. The road had been modified when a new motorway was built. A 2008 *Navteq* navigation map was used to run the FDIA method. Fig. 12 shows that this map contains three major faults, described below from left to right.

The first fault is where the road now deviates as it passes over the motorway. The second is where a completely new stretch of road has been created, deviating significantly from the old one. For these two cases the confidence accorded to the estimates both from the sensors and from navigation are high. In a rural environment, many satellites are in the receiver line-of-sight, which increases the level of confidence and reduces the position standard deviations and Dilution of Precision. Moreover, the road network is quite simple, so the map-matching algorithm provides a high level of confidence even if the GNSS measurement is a few metres away from the road. The challenge is therefore to determine precisely the reason for any disparity between estimates from sensors and from navigation, that is to determine which estimate is affected by a fault. When the real road is too far from the map road, the map-matching confidence index suddenly decreases and the navigation function switches to *off-road* mode and stops providing navigation estimates. The FDIA method consequently stops until a new estimate is provided by the navigation system.

The third fault is where a new road now exists parallel to the old one. Even if the estimate from navigation is relatively close to the true vehicle position in this area, the method should identify the fault. Fig. 12 also shows the estimates from sensors for the two trips used in this experiment.

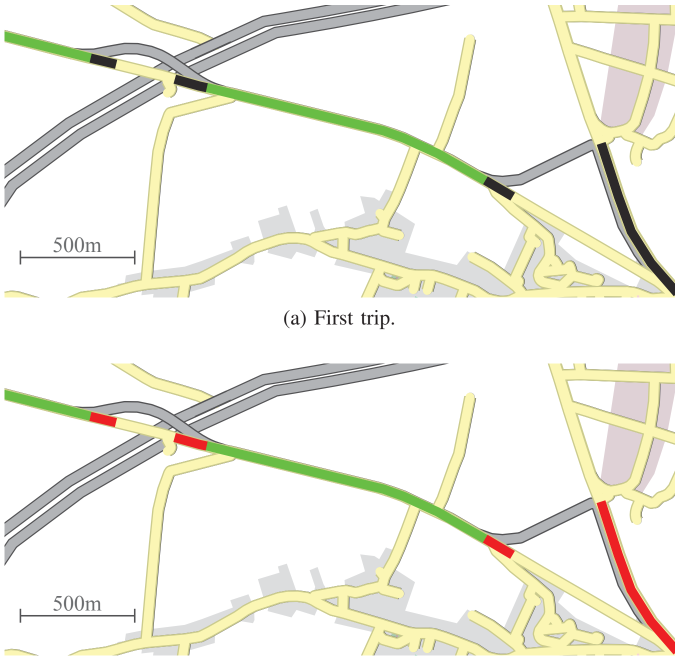
Fig. 13a shows the result of the FDIA applied to this dataset after the first trip. The green stretches are where the method returned *Use* and the black stretches are where the output was *Unknown.* There is no FI, since the method cannot isolate a fault at the first trip, as described above. It will nevertheless be noted that there is no FV of 0% and the OER is 100%. This means that the method correctly identified situations where estimates were not affected by faults and consequently provided the output *Use,* and also that it detected situations where at least one fault affected the estimates and consequently provided the output *Unknown* to client systems. The IAR of this first trip is 77% which corresponds to the proportion of erroneous roads in the navigation map.

The results obtained after the second vehicle trip in this area are shown in Fig. 13b. Here again, $OER = 100$ % which means that every estimate not declared *Unknown* at the second trip was correctly identified. Moreover, every point traversed during the course of two trips was declared either *Use* or *Don't use,* and so the Information Availability Rate equals 100%.

This experiment shows that the method performed well when using real vehicle data and a real navigation map with faults. The absence of False Isolations and particularly False Validations, and the high Information Availability in these conditions indicate that the FDIA framework is a realistic option for navigation integrity monitoring.

E. Discussion

These results, obtained using map faults that were either injected or real, show that the isolation convergence property is verified, since the number of points for which the method cannot perform isolation decreases and can reach zero. The



(a) First trip.

(b) Second trip.

Figure 13. Results of the FDIA for the rural test track. The road sections for which the method outputs *Use* are in green, those for which the method outputs *Don't use* are in red, and those for which the method outputs *Unknown* are in black. The true navigation map is in the background in grey.

information availability rate increases and can reach one as the number of trips increases.

Faults that are not correctly isolated by the method (i.e. False Validations and False Isolations) result mainly from the trade-off between spatial re-sampling tolerance of sensor data and the comparison threshold used for the computation of residuals.

The design of the method is based on several assumptions (random faults in the observer estimates, systematic faults in the map, and independence between them). The results confirm experimentally the validity of these assumptions. Faults in the observer are essentially due to the additional GPS receiver and can arise from multipath. If at a given abscissa, the same multipath induces the same error on the receiver computation fix at two different trips used by the FDIA method, the first assumption is violated. In this case, the method fails to isolate faults. Nevertheless, this requires two conditions to be fulfilled: first, the same satellite geometry at the same abscissa during two different trips, and, second, the same position fix error after filtering. For these reasons, we believe that the violation of the first assumption is very unlikely. This situation was never encountered during the course of our experiments.

VII. Conclusions

This paper introduces a framework for monitoring the integrity of navigation map geometry by detecting and isolating faults on the estimate of the vehicle position from the navigation system. We showed that the context of intelligent vehicles in which this work takes place limits the quality of the sensors and the redundancy of the sources of information. The FDIA framework detailed in this work fills this gap by making use of repeated vehicle trips.

The framework is based on a pairwise comparison of spatially-sampled vehicle position estimates between the current and past vehicle trips that gives rise to residual vectors. We demonstrate that under the assumptions made the proposed FDIA framework is theoretically always able to perform fault detection. However, depending on the number of faults that affect the estimates and on the number of vehicle trips, it may not be possible to perform isolation, that is, to determine without ambiguity which estimate(s) is (are) affected by a fault. By defining such sets of faults mathematically, we demonstrate that the fault isolation and adaptation capabilities of the method improve as the number of vehicle trips increases. The proposed framework was tested using real sensor data and navigation map faults. Performance was excellent in open sky areas and promising in urban conditions. This highlights the interest of using this FDIA approach in intelligent vehicles.

References

[1] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, C. Keller, E. Kaus, R. Herrtwich, C. Rabe, D. Pfeiffer, F. Lindner, F. Stein, F. Erbs, M. Enzweiler, C. Knoppel, J. Hipp, M. Haueis, M. Trepte, C. Brenk, A. Tamke, M. Ghanaat, M. Braun, A. Joos, H. Fritz, H. Mock, M. Hein, and E. Zeeb, "Making bertha drive, an autonomous journey on a historic route," *Intelligent Transportation Systems Magazine, IEEE*, vol. 6, no. 2, pp. 8–20, Summer 2014.

[2] A. Eskandarian, *Handbook of Intelligent Vehicles*, ser. Handbook of Intelligent Vehicles. Springer, 2012, no. vol. 2.

[3] S. Durekovic and N. Smith, "Architectures of map-supported ADAS," in *IEEE Intelligent Vehicles Symposium (IV)*, june 2011, pp. 207 –211.

[4] V. Popovic and B. Vasic, "Review of hazard analysis methods and their basic characteristics," *FME Trans.*, vol. 36, no. 4, pp. 181–187, 2008.

[5] A. K. Brown, "Receiver autonomous integrity monitoring using a 24-satellite GPS constellation," in *Institute of Navigation, Technical Meeting*, vol. 1, 1987, pp. 256–262.

[6] L. Jaulin, M. Kieffer, I. Braems, and E. Walter, "Guaranteed nonlinear estimation using constraint propagation on sets," *International Journal of Control*, vol. 74, no. 18, pp. 1772–1782, 2001.

[7] V. Drevelle and P. Bonnifait, "Localization confidence domains via set inversion on short-term trajectory," *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1244–1256, 2013.

[8] ——, "Interval-based fast fault detection and identification applied to radio-navigation multipath," *International Journal of Adaptive Control and Signal Processing*, 2015. [Online]. Available: http://dx.doi.org/10.1002/acs.2535

[9] J. Cosmen-Schortmann, M. Azaola-Saenz, M. A. Martinez-Olague, and M. Toledo-Lopez, "Integrity in urban and road environments and its use in liability critical applications," in *IEEE/ION Position, Location and Navigation Symposium*, 2008, pp. 972–983.

[10] C. Pinana-Diaz, R. Toledo-Moreo, D. Betaille, and A. Gomez-Skarmeta, "GPS multipath detection and exclusion with elevation-enhanced maps," in *Intelligent Transportation Systems, IEEE 14th Int. Conf. on*, 2011.

[11] S. Peyraud, D. Bétaille, S. Renault, M. Ortiz, F. Mougel, D. Meizel, and F. Peyret, "About non-line-of-sight satellite detection and exclusion in a 3D map-aided localization algorithm," *Sensors*, vol. 13, pp. 829–847, 2013.

[12] M. Obst, S. Bauer, P. Reisdorf, and G. Wanielik, "Multipath detection with 3D digital maps for robust multi-constellation GNSS/INS vehicle localization in urban areas," in *IEEE Intelligent Veh. Symp. (IV)*, 2012, pp. 184–190.

[13] P. Bonnifait, P. Bouron, P. Crubille, and D. Meizel, "Data fusion of four ABS sensors and GPS for an enhanced localization of car-like vehicles," in *Robotics and Automation, IEEE Int. Conf. on*, 2001.

[14] O. Le Marchand, P. Bonnifait, J. Ibanez-Guzman, and D. Betaille, "Vehicle localization integrity based on trajectory monitoring," in *Intelligent Robots and Systems, IEEE/RSJ Int. Conf. on*, 2009.

[15] G. Grekousis and Y. Fotis, "A fuzzy index for detecting spatiotemporal outliers," *GeoInformatica*, vol. 16, pp. 597–619, 2012.

[16] J.-F. Girres and G. Touya, “Quality assessment of the french openstreetmap dataset,” *Transactions in GIS*, vol. 14, no. 4, pp. 435–459, 2010.

[17] V. Noronha and M. F. Goodchild, “Map accuracy and location expression in transportation, reality and prospects,” *Transportation Research Part C: Emerging Technologies*, vol. 8, no. 1 - 6, pp. 53 – 69, 2000.

[18] M. Haklay, “How good is volunteered geographical information? a comparative study of openstreetmap and ordnance survey datasets,” *Environment and planning. B, Planning & design*, vol. 37, no. 4, p. 682, 2010.

[19] C. Boucher and J.-C. Noyer, “Automatic detection of topological changes for digital road map updating,” *Instrumentation and Measurement, IEEE Transactions on*, vol. 61, no. 11, pp. 3094–3102, Nov 2012.

[20] L. Deren, S. Haigang, and X. Ping, “Automatic change detection of geo-spatial data from imagery,” *Geo-Spatial Information Science*, vol. 6, pp. 1–7, 2003.

[21] N. de Freitas, R. Dearden, F. Hutter, R. Morales-Menendez, J. Mutch, and D. Poole, “Diagnosis by a waiter and a Mars explorer,” *Proceedings of the IEEE*, vol. 92, no. 3, pp. 455 – 468, mar 2004.

[22] O. Pink and C. Stiller, “Automated map generation from aerial images for precise vehicle localization,” in *Intelligent Transportation Systems (ITSC), IEEE 13th Int. Conf. on*, sept. 2010, pp. 1517 –1522.

[23] G. Agamennoni, J. Nieto, and E. Nebot, “Robust inference of principal road paths for intelligent transportation systems,” *Intelligent Transportation Systems, IEEE Transactions on*, vol. 12, pp. 298–308, 2011.

[24] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, “Particle filters for positioning, navigation, and tracking,” *Signal Processing, IEEE Transactions on*, vol. 50, no. 2, pp. 425–437, 2002.

[25] I. Szottka and M. Butenuth, “Advanced particle filtering for airborne vehicle tracking in urban areas,” *Geoscience and Remote Sensing Letters, IEEE*, vol. 11, pp. 686–690, 2014.

[26] C. Fouque and P. Bonnifait, “Matching raw GPS measurements on a navigable map without computing a global position,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, pp. 887–898, 2012.

[27] O. Mazhelis, “Using recursive bayesian estimation for matching GPS measurements to imperfect road network data,” in *Intelligent Transportation Systems (ITSC), IEEE 13th Int. Conf. on*, 2010, pp. 1492–1497.

[28] G. Nassreddine, F. Abdallah, and T. Denoeux, “Map matching algorithm using interval analysis and dempster-shafer theory,” in *Intelligent Vehicles Symposium, IEEE*, June 2009, pp. 494–499.

[29] M. A. Quddus, W. Y. Ochieng, and R. B. Noland, “Integrity of map-matching algorithms,” *Transportation Research Part C: Emerging Technologies*, vol. 14, no. 4, pp. 283–302, 2006.

[30] W. Y. Ochieng, M. Quddus, and R. B. Noland, “Map-matching in complex urban road networks,” *Revista Brasileira de Cartografia*, vol. 2, 2009.

[31] N. R. Velaga, M. A. Quddus, A. L. Bristow, and Y. Zheng, “Map-aided integrity monitoring of a land vehicle navigation system,” *Intelligent Transportation Systems, IEEE Trans. on*, vol. 13, pp. 848–858, 2012.

[32] S. X. Ding, *Model-based Fault Diagnosis Techniques: Design Schemes, Algorithms, and Tools*. Springer, 2008.

[33] M. Krysander, F. Heintz, J. Roll, and E. Frisk, “Flexdx: A reconfigurable diagnosis framework,” *Engineering Applications of Artificial Intelligence*, vol. 23, no. 8, pp. 1303 – 1313, 2010.

[34] M. Basseville and I. V. Nikiforov, *Detection of Abrupt Changes: Theory and Application*, E. Cliffs, Ed. Prentice-Hall, Inc, 1993.

[35] P. Sundvall and P. Jensfelt, “Fault detection for mobile robots using redundant positioning systems,” in *Robotics and Automation (ICRA). IEEE Int. Conf. on*, may 2006, pp. 3781 –3786.

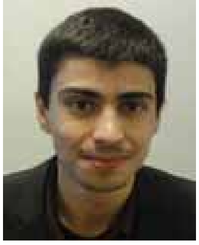
[36] C. Ress, D. Balzer, A. Bracht, S. Durekovic, and J. Löwenau, “Adasis protocol for advanced in-vehicle applications,” in *15th World Congress on Intelligent Transport Systems*, 2008.

[37] R. S. Radovanovic, “High accuracy deformation monitoring via multipath mitigation by day-to-day correlation analysis,” in *13th International Technical Meeting of the SAT Division of the ION, September*, 2000, pp. 19–22.

[38] C. Zinoune, P. Bonnifait, and J. Ibanez-Guzman, “A sequential test for autonomous localisation of map errors for driving assistance systems,” in *Intelligent Transportation Systems (ITSC), IEEE 15th Int. Conf. on*, 2012, pp. 1377–1382.

[39] ——, “Integrity monitoring of navigation systems using repetitive journeys,” in *Intelligent Vehicles Symposium Proceedings, IEEE*, June 2014, pp. 274–280.

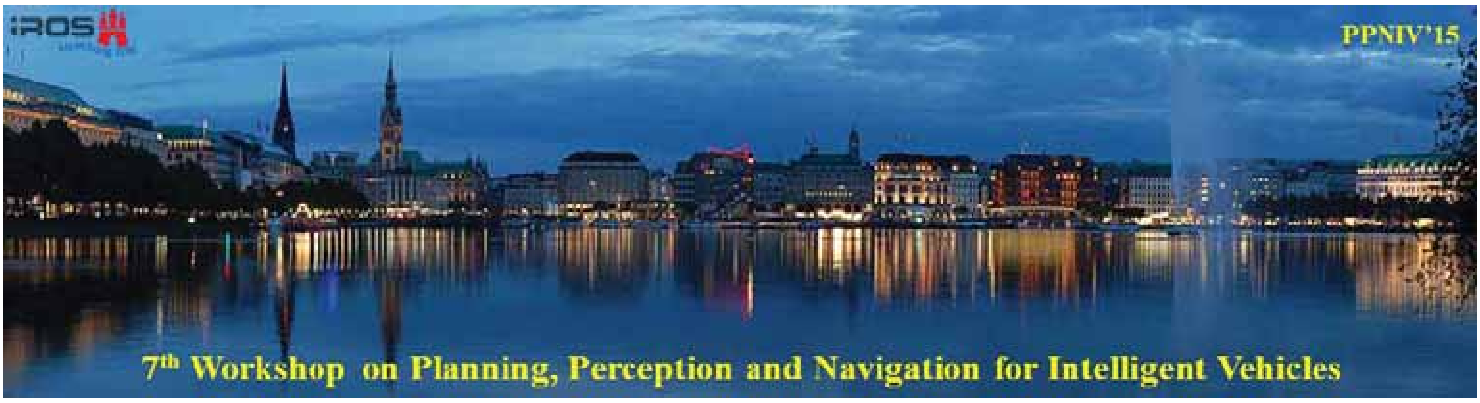
[40] A. Monteil and C. Zinoune, “Demonstration of the rules non isolability of sets of faulty states,” University of Technology of Compiegne, http://hal.archives-ouvertes.fr/hal-01072412, Tech. Rep., Jan 2014.



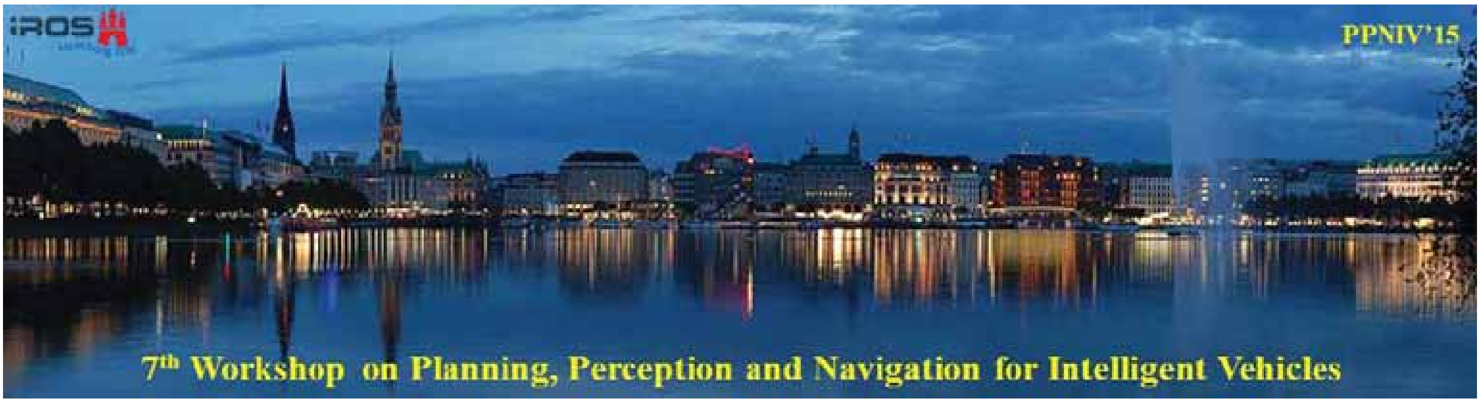
Clément Zinoune obtained his Ph.D. in Computer Science and Engineering at the Université de Technologie de Compiègne (UTC) in France in 2014. He graduated as a mechanical engineer in 2010 at UTC. In 2010 he also obtained his master of science in autonomous vehicle dynamics and control at Cranfield University in the UK. In 2014 he joined the team charged with the development of autonomous vehicle at Renault S.A. in France.

Philippe Bonnifait is a professor in the Computer Science and Engineering department of the Université de Technologie de Compiègne (UTC) in France. He obtained his Ph.D. in automatic control and computer science at the École Centrale de Nantes in 1997. Since 1998, he has been with Heudiasyc UMR 7253, a joint research unit between UTC and CNRS. His research interests are Intelligent Vehicles, high integrity positioning and map-matching for mobile robot navigation in structured outdoor environments.

Javier Ibañez-Guzmán (IEEE Member) obtained his Ph.D. at the University of Reading on a SERC-UK fellowship, and his MSEE at the University of Pennsylvania (USA) as a Fulbright scholar. In 2011, he was visiting scholar at the University of California, Berkeley (CITRIS), working on connected vehicle applications. He is currently a member of the technical staff at Renault S.A., carrying out work on autonomous vehicle navigation technologies and driving assistance systems. Formerly he was senior scientist at a national research institute in Singapore, where he spearheaded work on autonomous ground vehicles operating in unstructured environments. Dr. Ibañez-Guzmán has several publications and patents in the robotics and automotive domains. He has successfully supervised a number of Ph.D. students. He is a C.Eng. (UK) and Fellow of the Institute of Engineering Technology (UK).


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

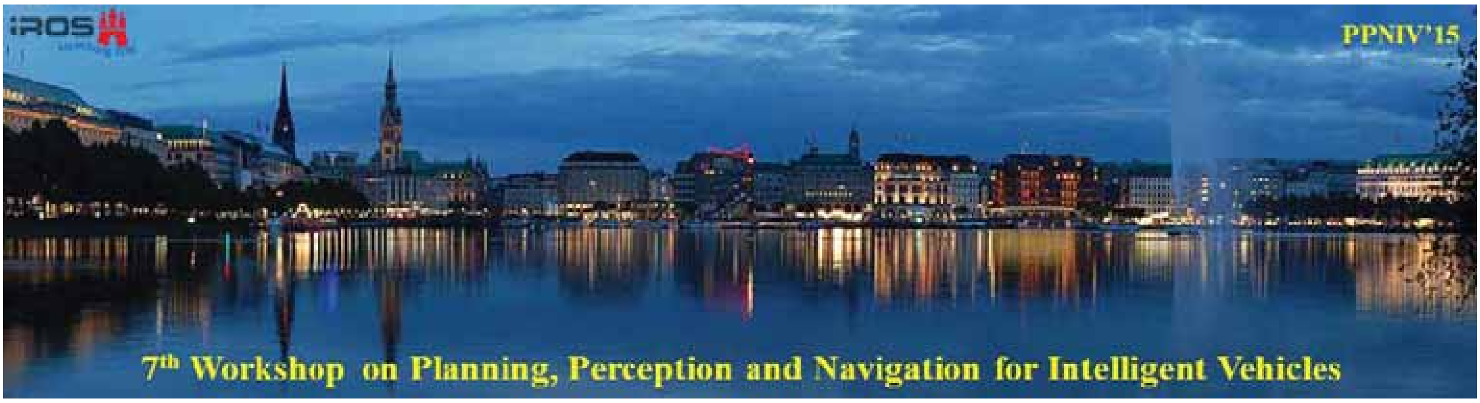
Session I

Localization & Mapping

- **Title: Collaborative Visual SLAM Framework for a Multi-Robot System**
  Authors: Nived Chebrolu, David Marquez-Gamez and Philippe Martinet

- **Title: Improving Vision-based Topological Localization by Combining Local and Global Image Features**
  Authors: Shuai Yang and Han Wang

- **Title: PML-SLAM: a solution for localization in large-scale urban environments**
  Authors: S. Alsayed, G. Bresson, F. Nashashibi, A. Verroust Blondet


PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Collaborative Visual SLAM Framework for a Multi-Robot System

Nived Chebrolu[1], David Marquez-Gamez[2] and Philippe Martinet[1]

***Abstract*— This paper presents a framework for collaborative visual SLAM using monocular cameras for a team of mobile robots. The robots perform SLAM individually using their on-board processors thereby estimating the seven degrees of freedom (including scale) for the motion of the camera and creating a map of the environment as a pose-graph of keyframes. Each robot communicates to a central server by sending local keyframe information. The central server merges them when a visual overlap is detected in the scene and creates a global map. In the background, the global map is continuously optimized using bundle adjustment techniques and the updated pose information is communicated back as feedback to the individual robots. We present some preliminary experimental results towards testing the framework with two mobile robots in an indoor environment.**

I. INTRODUCTION

Autonomous robots are increasingly being used for more and more complex problems each day such as exploration of large unstructured environments etc. In order to deal with these complex scenarios, a multi-robot system consisting of a team of robots (such as mobile robots, aerial vehicles etc) which are equipped with perception sensors is necessary. A multi-robot system extends the capability of a single robot by merging measurements from several team members and providing each robot with information beyond the range of their individual sensors. This facilitates more efficient usage of resources and achieves tasks which are not feasible for a single robot system.

Moreover the use of a multi-robot system allows parallel execution of tasks and also some degree of redundancy increasing both the efficiency and the robustness of the system. Consider a scenario of employing a multi-robot team for the purpose of mapping of a large unknown environment. The task can be divided among all the team members which can collaboratively build a global map reducing the overall execution time. This collective information including the relative positions of the robots can be used for making exploration strategies, path-planning and other higher level decision. However, in general the advantages of a collaborative system come at the cost of increased computations and communication load among robots.

In this work, we deal with a team of mobile robots each equipped with a monocular camera to perform SLAM. Using a monocular camera gives the advantage for the system to be used both for indoor/outdoor applications and also for scenes with large variations in depth. Typically these conditions impose severe restrictions on other vision sensors such as RGB-D cameras and stereo pairs. However, this makes running the visual SLAM process more challenging as the scale needs to be continuously estimated since no depth information is directly available.

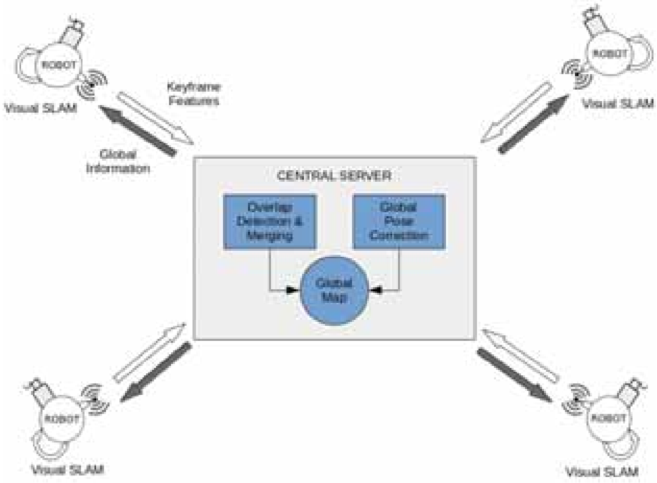


Fig. 1. System Architecture: Collaborative Visual SLAM

In this paper, we propose a framework for collaborative visual SLAM (as shown in Fig. 1) where:

- Each individual robot performs monocular visual SLAM and sends local keyframe information to a central server.
- The central server merges this information to create a global map and performs a pose correction using bundle adjustment.
- The updated pose is communicated back to individual robots as feedback thereby improving the local map and the localization estimate of the individual robots.

The organization of the paper is as following. The next section presents the related works. Then a general overview on our system is given in section III where every function is shortly described. Section IV presents the methodology where we detail each function. Finally experimental results are presented and analyzed in the section V.

II. RELATED WORK

Traditionally, SLAM has been performed using range sensors like laser scanners, sonars or using stereo vision [1]. More recently, monocular cameras (bearing only sensors) are also being used as the primary vision sensor like in [2],[3],[4]. In the multi-robot context, this problem has been studied under the banner of multi-camera structure from motion (SfM) [5] or multi-camera SLAM [6].

In [7], the authors analyse the improvement in localization quality of cooperative multi-robot localization over single robot localization. [8] demonstrates that by incorporating relative bearing information of the cameras, the overall accuracy of the localization is strongly improved. In this

[1] Nived Chebrolu and Philippe Martinet are with Institut de Recherche en Communications et Cybernétique de Nantes (IRCCyN), Ecole Centrale de Nantes, France.nived.chebrolu@eleves.ec-nantes.fr, philippe.martinet@irccyn.ec-nantes.fr

[2] David Marquez-Gamez is with Robotics Division, IRT Jules Verne, Nantes, France. david.marquez@irt-jules-verne.fr

approach, an Extended Kalman Filter (EKF) is adopted to maintain the state containing configurations of all robots. In [9], the authors propose an interesting idea where two UAVs with monocular cameras act as a flexible stereo rig. With additional input from IMUs the relative poses are recoverd with absolute scale starting from an unknown initial configuration.

[10] deals with large-scale collaborative SLAM in an outdoor environment involving heterogeneous robots such as UAVs and ground mobile robots equipped with stereo cameras. It employs a global graph which maintains the relative relationships between a series of submaps built by each robot. The links between each submaps are created by events like robot rendezvous, scene feature matches or absolute localization information provided by GPS etc. These constraints allow the correction of the position estimates of submaps with respect to each other. [11] uses a multi-camera system to estimate the trajectory of moving objects in the scene along with building a 3D map of static objects. However, the system requires the image streams from all the cameras to be synchronized making it impractical to be used for real-time applications.

Several decentralized solutions have been proposed where data fusion is performed using only robots which are in direct communication range of each other. [12] proposes a method to efficiently distribute map information across a team of robots which is robust to node failures and changes in network topology. The proposed scheme consists of a local optimization module which executes single robot SLAM, a communication module which propagates the local graphs to other robots and a neighbourhood graph optimization module which combines all the local graphs into maps describing the neighbourhood of a robot. On the other hand, recently many centralized cloud based architectures for collaborative SLAM have been designed where the data intensive tasks can be mitigated to a powerful back-end cluster system [13], [14], [15]. This allows the use of small and energy efficient on-board processor to be placed on the robots while offloading major computations to the cloud.

In [16], the authors propose a centralized framework for a group of MAVs equipped with monocular cameras. Each MAV performs visual odometry on its on-board processor and sends keyframe information to a ground server where it is merged to realize a global map. In this paper, we present a similar framework however each robot is capable of performing complete SLAM individually using full image information instead of using only features. In addition, a feedback mechanism is put in place which corrects the local estimates continuously. Also the framework allows the robots to join asynchronously and no prior relative information is required.

III. SYSTEM OVERVIEW

Figure 2 illustrates the overall scheme of our collaborative SLAM system. Each mobile robot equipped with a monocular camera performs visual SLAM using its on-board computer. This provides each robot with an estimate of its pose and a 3D map of the environment in their respective co-ordinate frames. In our approach, we use a monocular SLAM algorithm based on direct image alignment which is able to estimate the seven DoF's including the scale of the scene. The map of the environment is stored as a pose-graph of keyframes each consisting of a semi-dense depth map of the corresponding view. This function is based on LSD SLAM [3].

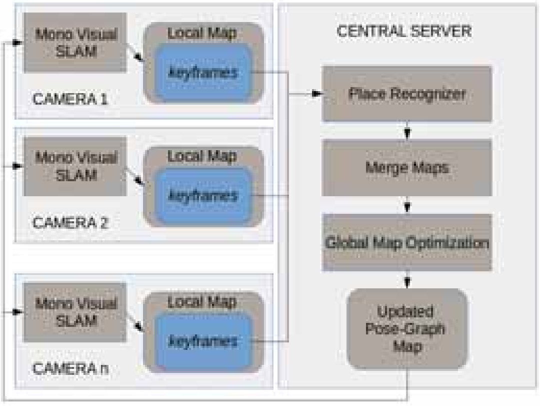


Fig. 2. Overall scheme of our collaborative SLAM system

Each robot sends its keyframe information including its pose in the local co-ordinate frame to the central server. Here the *place recognizer* function constantly monitors all the keyframes to detect overlapping scenes from different robots. The overlap detection is performed in the appearance space by extracting visual features from each keyframe and comparing them in a fast manner using Bag of Words (BoW) technique [17].

Once an overlap is detected between two cameras, the map merging sequence is initiated. It involves computing an initial transformation estimate between the matched keyframes by using a RANSAC version of the traditional Horn's algorithm [18]. This estimate is used as a starting point to run an optimization algorithm which estimates the similarity transformation between the two matched keyframes. Finally, this estimate is refined by performing an iterative closest point algorithm as described in [19].

After computation of this transformation, the two corresponding maps are merged into a global map and a new constraint is added between the two matched keyframes.In parallel, a bundle adjustment procedure is run over the global graph and the updated poses of the keyframe graph are communicated back to the individual robots as feedback. This information is in turn used by each robot to improve its localization estimate and the local map.

IV. METHODOLOGY

In this section we detail each individual function in Fig. 2.

A. Visual SLAM

Each mobile robot performs an on-board monocular SLAM process which is able in real time to estimate its pose and create the map of the environment as a pose-graph of keyframes . The problem of scale drift is addressed by implicitly including it as a parameter in the overall optimization procedure. This function is based on LSD-SLAM [3]. The overall method consists of the following main components:

1) Tracking: The camera pose $\xi \in se(3)$ is estimated with respect to the current keyframe K_i which consists of the image (I_i), the depth map (D_i) and the depth map variance V_i. For each new image I_j, the relative pose $\xi_{ji} \in se(3)$ is computed by minimizing the photometric error:

$$\xi_{ji} = \underset{\xi}{\operatorname{argmin}} \sum_p \left\| r_p^2(p, \xi_{ji}) \right\| \tag{1}$$

where the photometric residual $r_p(p, \xi_{ji}) = I_i(p) - I_j(\omega(p, D_i(p), \xi_{ji}))$ and ω is a warping function which computes the location of a pixel from the first image in the second image given the relative transformation ξ_{ji}. Note that in the actual implementation, a variance normalized residual is minimized thereby implicitly including depth accuracy in the computation of ξ_{ji}. The optimization problem can be posed as a weighted least squares problem [20] which can be solved using the Gauss-Newton minimization method [21].

2) Depth Map Estimation: A semi-dense inverse depth map is continuously estimated for each new frame. The depth map is computed by making several stereo comparisons of varying baseline over consecutive frames of the input video. The variable base line allows for accurate estimation of both near and far regions of the image. The method maintains a probabilistic depth hypothesis for each pixel modelled by a gaussian distribution which is continuously refined using an filtering approach described in [22]. Finally when the camera moves far from the current keyframe, a new keyframe is created and its depth map is initialized by projecting points from the previous keyframe on it.

3) Map Management and Optimization: The frame to frame alignment method previously introduced in IV-A.1 inherently accumulates drift over time due to small errors in each estimate arising from sensor noise and other model inaccuracies.

To deal with this problem, the SLAM system maintains the map as a graph where each vertex is the pose of the keyframe and each edge represents the relative transformation between the corresponding keyframes. Each time a new keyframe is added to the map, new edges are created and finally when previously visited regions of the scene are encountered, additional edges (loop closures) are added which help in reducing the accumulated drift.

However in the case of monocular SLAM, the scale of the scene cannot be observed directly which over a long trajectory leads to a drift causing major errors in tracking. To take care of the scale parameter, the overall pose graph is constructed in a manner such that the mean inverse of each keyframe is one and instead the edges are represented as transformation $\xi_{ji} \in sim(3)$. This allows the integration of the scale parameter directly in the optimization problem. So, the scaled transformation between the keyframes is estimated by minimizing the error function:

$$E(\xi_{ji}) = \sum_p \left\| r_p^2(p, \xi_{ji}) + r_d^2(p, \xi_{ji}) \right\| \tag{2}$$

where the depth residual is $r_d(p, \xi_{ji}) = [p']_3 - D_j([p']_{(1,2)})$ and $p' = \omega(p, D_i(p), \xi_{ji})$

Finally, the overall map consisting of keyframe poses as vertices and $sim(3)$ constraints as edges, is continuously optimized in parallel using a general graph optimization framework like g2o [21]. This optimization over the graph reduces the drift both in scale and pose estimates.

B. Place Recognizer

This module runs continuously on the central server and is responsible to find scene overlap between different robots. Since the relative position of each robot is not known in the global coordinate frame at the beginning, the overlap is detected using the appearance space information only.

For every new keyframe image, visual features (e.g. SURF [23]) are computed which are view-point invariant. These features are then quantized with respect to a vocabulary and the resulting *visual words* description is stored. This *bag of words* (BoW) technique allows the scene to be represented as a collection of words which facilitates fast comparisons of feature descriptors.

We use the FAB-MAP method [24] to detect a scene overlap. This algorithm takes as input the BoW description of each image, compares it against all previously seen images. It gives as output the probability with which the current image matches any of the previously seen images. Moreover, it also computes the probability of the current image being a new one. These probabilities are calculated by solving a recursive Bayes estimation problem:

$$p(L_i|Z_k) = \frac{p(Z_k|L_i, Z_{k-1})p(L_i|Z_{k-1})}{p(Z_k|Z_{k-1})} \tag{3}$$

where L_i is a scene (location) in the world, Z_k is an observation (visual words) at time k. In equation (3), $p(L_i|Z_{k-1})$ is the prior belief of our location, $p(Z_k|L_i, Z_{k-1})$ is the observation likelihood and $p(Z_k|Z_{k-1})$ is a normalizing term. The exact evaluation of these terms can be found in [24].

Traditionally, the FAB-MAP technique has been used to find loop closures over long trajectories. Instead in our application, we use it to find if a place has been visited by other robots in the team and in-effect creating a *virtual loop closure*. It is termed *virtual* since the loop closure is obtained as a result of the same place being visited by two different robots (as opposed to the same place being visited by the same robot). Finally, in order to avoid spurious matches, we only proceed for map merging if FAB-MAP reports a match over three consecutive images.

C. Map Merge

When the place recognizer module detects a scene overlap between robots and indicates a match point, the map merging procedure is initiated. The transformation between matching frames is done in three steps followed by an update to the global map.

1) Initial Transformation Estimate Using Horn's Method: For each keyframe image arriving at the central server, SURF features are computed and stored. Note that the same features were also used to compute a BoW representation required as an input to FAB-MAP.

The Horn's method describes a closed form solution using unit quaternions to compute the scaled transformation given three 3D point correspondences between two point clouds [18]. From the matching keyframe candidates proposed by FAB-MAP, 2D feature correspondences are extracted. Later the depth of each 2D feature is computed by taking an average over the keyframe depth weighted by their variances in the descriptor neighbourhood. It should be noted that descriptors often end up in regions of discontinuities (such as corners or edges) and the averaging step may result in bad depth estimates. Finally, in order to deal with bad matches between features, we implement a robust RANSAC based version of the Horn's algorithm.

2) Refining Estimate Using $Sim3$ Tracker: As a second step, we use the tracking method based on minimizing the cost function as described in equation 2 to find an improved estimate for the scaled transformation. The estimate provided by Horn's method is used as a starting point for the tracker.

3) Correction using ICP: A final correction is made using the iterative closest point (ICP) algorithm, a technique from point cloud registration literature, which tries to find a transformation that minimizes the distance between a set of corresponding points in two clouds. We use an augmented version of ICP which also includes surface normal and tangent information to improve the estimate as described in [19]. Note that both the $Sim3$ tracker and the ICP procedure use a gradient approach to find the solution. In theory both these methods could be used in any order. However, the ICP procedure was found to be most accurate starting from a better estimation of the scale factor. Therefore, it was decided to perform the ICP step after the tracking step.

4) Global Map Update: Once the transformation is computed, new similarity constraints are added between the matching keyframes. The corresponding local maps are transformed into the global coordinate frame considering one of the two as reference (if its the first map merge) or using the existing reference otherwise. After the new constraints have been added a bundle adjustment step is performed over the merged graph.

D. Overall Feedback System

Each time different robots visit the same place in the environment, new constraints are created in the global graph. While the mobile robots move in the environment, they may cross each others path multiple times resulting in *virtual loop closures*. These loop closure constraints help in reducing the overall drift.

Finally, the central server communicates the updated pose graph to individual robots which can then use this information to update their localization estimate and the local maps.

This overall feedback mechanism facilitates the extension of sensing capability of an individual robot beyond the direct reach of their respective on-board sensors. In a sense, each robot in the team is able to “look” beyond what they can directly see and thus taking advantage of the collaborative system.

V. EXPERIMENTAL RESULTS

In this section we present preliminary results with the aim of validating the concepts presented before. The experiments presented are not intended to be particularly challenging examples, they are simply used to take the reader through the functionality of the system.

The experiments were performed using two *Turtlebots*, each equipped with a *uEye* monocular camera attached with a wide-angle lens ($\sim$130°Field of View) and a Core 2 Duo laptop. The images are captured at 30 Hz with a resolution of 640 x 480 pixels. The experiments were conducted in an industrial-like indoor environment approximately 20m x 20m. The two robots start exploring the environment asynchronously. Moreover, their starting positions are not known to the central server. The robots traverse through regions with large variations in scene scales. The depth of theses scenes range from 1m to 15m. Finally, the communication between the robots and the central server is carried through the standard Wi-Fi protocol.

Figures 3 and 4 shows results from the monocular SLAM process running on the local computers of the two robots $R1$ and $R2$ respectively. The three columns in these figures show the images captured by the camera, the trajectory of the robot and the corresponding map built at three different instants (corresponding to the three rows). The trajectory of the robot $R1$ is illustrated in red and that of robot $R2$ in blue. Each pyramid in the map represents a keyframe location and the lines joining these keyframes represent the constraints. The green pyramid depicts the current keyframe for both the cameras.

At instant 1, we see that robot $R1$ has completed a small square loop. During this trajectory, it makes some loop closures as well. Later robot $R2$ starts exploring some other part of the environment. At instant 2, the robot $R2$ completes a loop closure. As a result, we see that the overall trajectory of robot $R2$ has been optimized and the scale factor is corrected as well. At instant 3, robot $R1$ enters a region previously visited by robot $R2$. At this time, the visual place recognition system triggers a merge between the two maps. The relative transformation between these two views is computed in three stages as described in section IV.

In this example, robot $R1$'s origin is considered as the reference coordinate frame. All the keyframes of robot $R2$ are transformed with respect to this origin. The global trajectory of the two robots and the joint map is shown in figure 5 . Finally, after searching for additional constraints and optimizing the global map using bundle adjustment, the updated keyframe poses are sent back to the two robots. This updated keyframe pose is then used by robots $R1$ and $R2$ to correct the localization estimate and the local map.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a framework for collaborative visual SLAM for a team of mobile robots using a centralized approach. Each robot is able to individually perform monocular SLAM using its camera and on-board computer. The central server continuously receives local keyframe information from individual robots over Wi-Fi. Keyframes from all the robots are merged at this server and an optimization procedure is followed which minimizes the overall pose and mapping error. The updated pose information is sent back to the individual robots incorporating a feedback mechanism. No prior information regarding the relative position of the robots or the initial configuration is required. The system allows the robots to join and leave the team asynchronously.

The framework can be extended to work with different type of cameras by including the corresponding projection functions. In addition to the centralized framework, it would be interesting to add robot-to-robot communication. In this case, the robots can also exchange pose and map information with each other when they are in direct communication range.

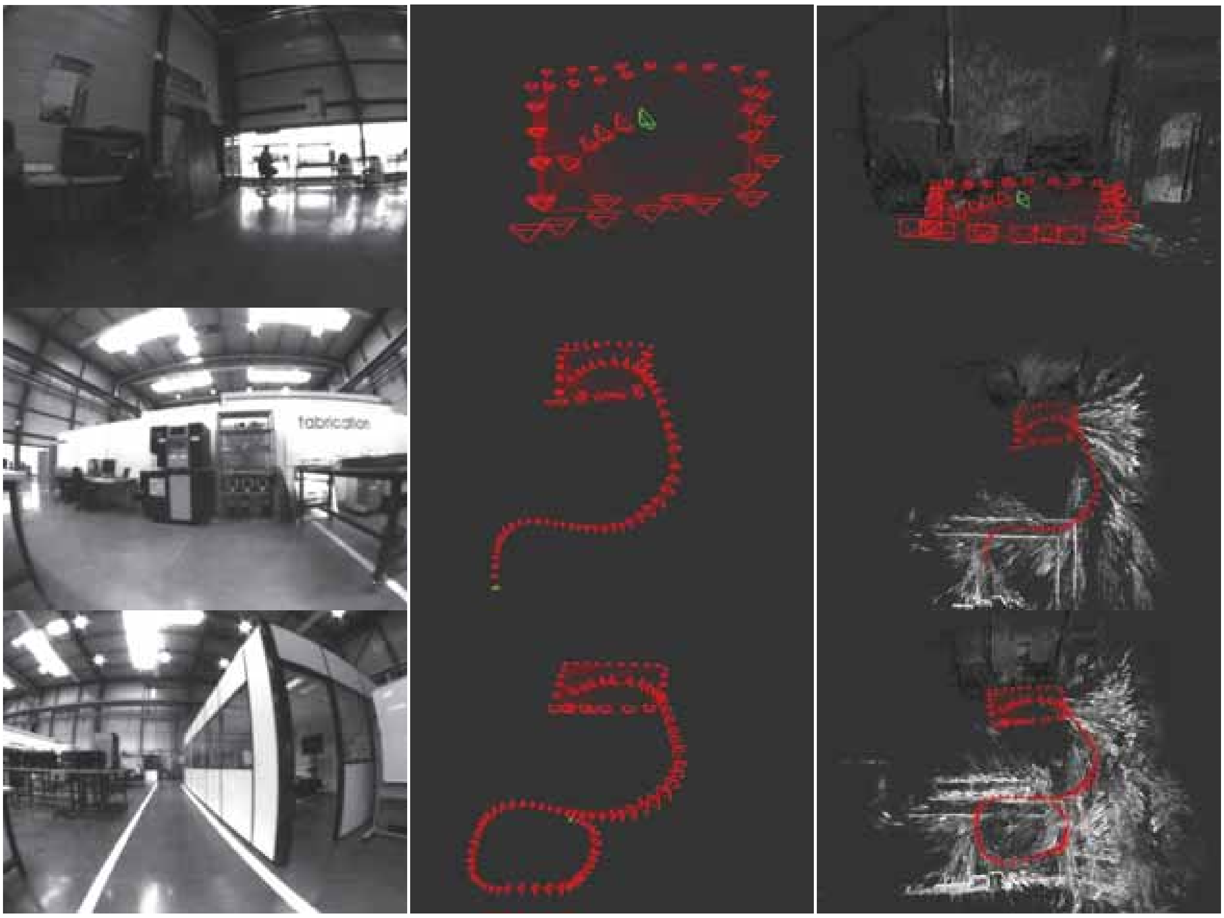


Fig. 3. Monocular SLAM Process on robot $R1$ at three different instants. Left: Image captured by the camera. Middle:Trajectory built by the robot. Right: Map built by the robot.

REFERENCES

[1] S. Thrun *et al.*, "Robotic mapping: A survey," *Exploring artificial intelligence in the new millennium*, pp. 1–35.

[2] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *CoRR*, vol. abs/1502.00956, 2015.

[3] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *European Conference on Computer Vision (ECCV)*, September 2014.

[4] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, June 2007.

[5] M. Kaess and F. Dellaert, "Probabilistic structure matching for visual slam with a multi-camera rig," *Comput. Vis. Image Underst.*, vol. 114, no. 2, pp. 286–296.

[6] J. Sola, A. Monin, and M. Devy, "Bicamslam : Two times mono is more than stereo," in *IEEE Int. Conf. on Robotics Automation*, 2007.

[7] D. Fox, W. Burgard, H. Kruppa, and S. Thrun, "A probabilistic approach to collaborative multi-robot localization." *Auton. Robots*, vol. 8, no. 3, pp. 325–344, 2000.

[8] A. Martinelli, F. Pont, and R. Siegwart, "Multi-robot localization using relative observations," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, April 2005, pp. 2797–2802.

[9] M. W. Achtelik, S. Weiss, M. Chli, F. Dellaert, and R. Siegwart, "Collaborative stereo," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, Sept 2011, pp. 2242–2248.

[10] T. A. Vidal-Calleja, C. Berger, J. Sol, and S. Lacroix, "Large scale multiple robot visual mapping with heterogeneous landmarks in semi-structured terrain," *Robotics and Autonomous Systems*, vol. 59, no. 9, pp. 654 – 674, 2011.

[11] D. Zou and P. Tan, "Coslam: Collaborative visual slam in dynamic environments," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 2, pp. 354–366, Feb 2013.

[12] A. Cunningham, B. Paluri, and F. Dellaert, "Ddf-sam: Fully distributed slam using constrained factor graphs." in *IROS*. IEEE, 2010, pp. 3025–3030.

[13] R. Arumugam, V. Enti, L. Bingbing, W. Xiaojun, K. Baskaran, F. F. Kong, A. Kumar, K. D. Meng, and G. W. Kit, "Davinci: A cloud computing framework for service robots," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, May 2010, pp. 3084–3089.

[14] D. Hunziker, M. Gajamohan, M. Waibel, and R. D'Andrea, "Rapyuta: The roboearth cloud engine," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, May 2013, pp. 438–444.

[15] "C2tam: A cloud framework for cooperative tracking and mapping," *Robotics and Autonomous Systems*, vol. 62, no. 4, pp. 401 – 413, 2014.

[16] C. Forster, S. Lynen, L. Kneip, and D. Scaramuzza, "Collaborative monocular slam with multiple micro aerial vehicles," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, Nov 2013, pp. 3962–3970.

[17] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ser. ICCV '03, 2003, pp. 1470–.

[18] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America A*, vol. 4, no. 4, pp. 629–642, 1987.

[19] J. Serafin and G. Grisetti, "Using augmented measurements to improve the convergence of icp," in *Proc. of the 4th International Conference on Simulation, Modeling and Programming for Autonomous Robots. (SIMPAR 2014)*, 2014.

[20] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for RGB-D cameras," in *2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, May 6-10, 2013*, 2013, pp. 3748–3754.

[21] R. Kuemmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *Proceedings*

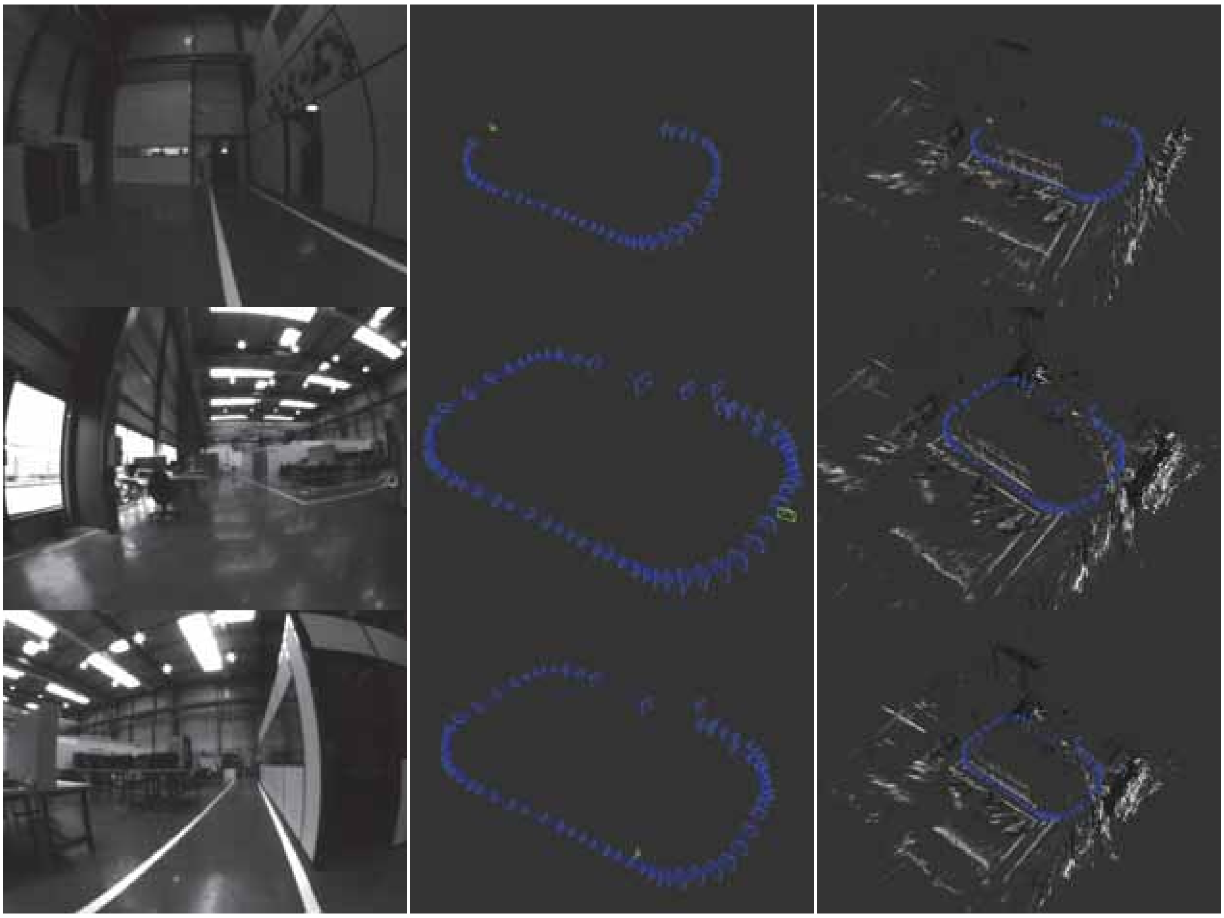


Fig. 4. Monocular SLAM Process on robot $R2$ at three different instants. Left: Image captured by the camera. Middle:Trajectory built by the robot. Right: Map built by the robot.

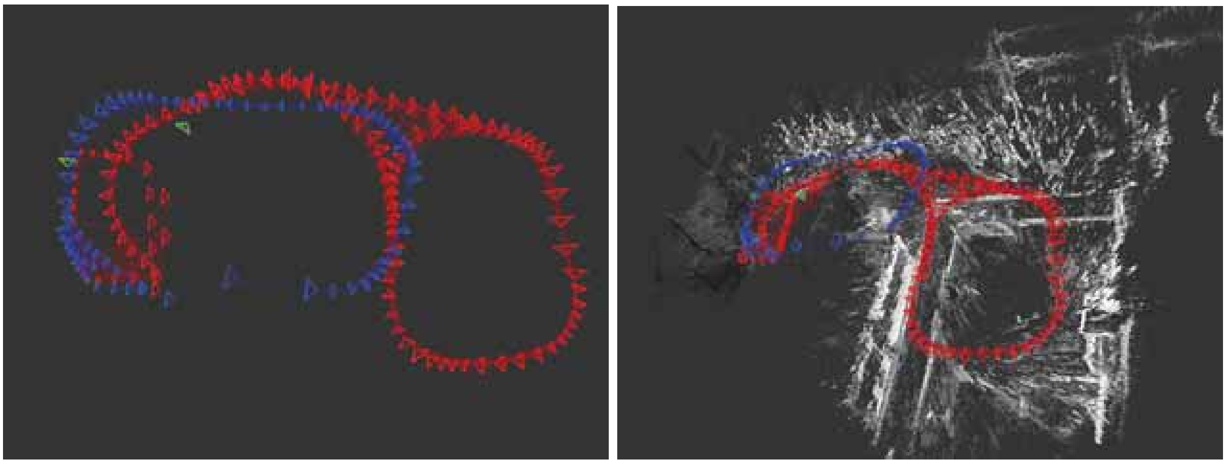


Fig. 5. Global map computed at the central server . *Left:* Trajectory of the two robots in the merged map. *Right:* Depth map associated with the keyframes from both robots.

of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, May 2011, pp. 3607–3613.

[22] J. Engel, J. Sturm, and D. Cremers, “Semi-dense visual odometry for a monocular camera,” in *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, 2013, pp. 1449–1456.

[23] H. Bay, T. Tuytelaars, and L. V. Gool, “Surf: Speeded up robust features,” in *In ECCV*, 2006, pp. 404–417.

[24] M. Cummins and P. Newman, “FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance,” *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665.

Improving Vision-based Topological Localization by Combing Local and Global Image Features

Shuai Yang and Han Wang
School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore 639798
{sayang, hw}@ntu.edu.sg

***Abstract*— Vision-based mobile robot localization method has become popular in recent years. Most of the current approaches use either global or local image feature for appearance representation. However, pure global image feature based method suffers from its low invariance to occlusions and view point changes. Meanwhile, pure local image feature based method ignores the global context of the image, which is very important in place representation. In this paper, we present a vision-based robot localization system that uses both local and global image features to represent locations, and the estimated location of the robot is determined by a Bayesian tracking filter which uses the proposed hybrid image representation. Experiments were conducted using a 1200 m campus route and the results show that the proposed method works reliably even with large appearance changes.**

I. INTRODUCTION

Self-localization ability is one of the most basic requirements for mobile robot. Whatever a self-driving car, a bomb disposal robot, or even a self-guided vacuum cleaner, the position should firstly be obtained before the robot can work automatically. Most commonly used methods for self-localization are sensor-based, which use sensory information to locate the robot in its environment. Many different sensors have been used, among which GPS is the most popular for outdoor environment. GPS can provide a good accuracy under ideal condition; however, it can be inaccurate or unavailable due to obstacles such as skyscrapers and trees. GPS signals can also be affected by multi-path issues [1], which tend to cause significant errors on robot localization. Another optional sensor is IMU; it can allow a GPS receiver to work when GPS signals are unavailable, but it is not appropriate for practical usage because of hight prices.

Recently, vision-based place recognition has attracted a lot of attention due to the widespread use of cameras on mobile devices. The basic idea is to match a query image with a database of geo-tagged images with known poses, which is also called image retrieval problem. Once this retrieval task is accomplished, it is possible to recover the position of the query image. Based on place recognition technique, many robot localization systems were proposed. One of the most well-known approach is Fast Appearance-Based Mapping (FAB-MAP) [2], [3], [4], which can perform very large trajectory estimation based on bag of visual words model. Zamir et al. [5] proposed a method, which utilizes Google Street View images as the reference dataset. They extract local features (e.g., SIFT) from the query image and retrieve a number of nearest neighbors for each query feature from the reference dataset, then a feature pruning method which incorportates geo-spatial information is employed to discover incorrectly matched features. Valgren and Lilienthal [6] evaluated outdoor appearance-based topological localization for a mobile robot over seasons using SIFT and SURF features. Badino et al. [7] proposed a system, dubbed topometric localization method: at the first step, images are recorded and image features (standard SURF) describing each location of the mapping route are extracted from the images and stored as a map. During localization stage, current image features are matched with the stored dataset and position estimates are smoothed by fusing velocity information.

The aforementioned methods choose local image measurements as image representations. While local image features offer great robustness to occlusions and illumination effects, as well as great discriminative power, they ignore the global information of the image, such as spatial relationships among the patches, which are very important in image representation. Another group of image representations are based on global image features, which can describe an image with a single vector. Early examples of these were color histogram, histogram of gradient orientation or frequency transforms [8]. But they can only be used in specific cases, such as static indoor environment. More recently, many other global image features have been used for mobile robot localization works. Milford et al. proposed a framework, dubbed SeqSLAM [9], [10], [11], which use the whole image to learn and recognize a route even with extreme perceptual changes, such as changes from day to night, summer to winter, and from clear weather to rain. Murillo et al. [12] used global gist descriptor to represent a panorama, and compared the performance with local image features. These whole-image descriptor based localization systems have the advantages of high efficiency, compact representation, suitable for handling large dataset. Nevertheless, purely global descriptor based systems are more sensitive to occlusions and illumination changes.

Researchers have also proposed hybrid approaches which use both global and local image features. Zamir et al. [13] suggested a method for image localization which finds one

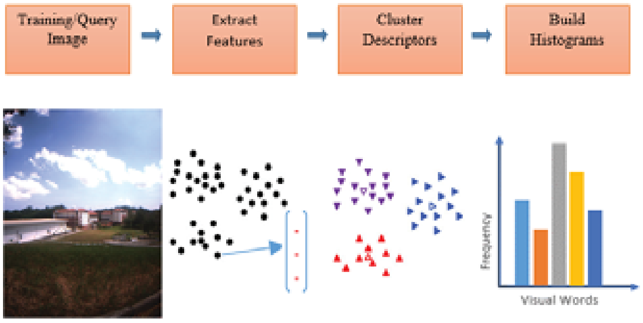


Fig. 1: Schematic for representing an image with Bow model.

or a few strongly matching reference images to a query by robustly discovering local feature correspondence at first, and then adopt some global contexts, like GIST, color histogram, to select the correct match. The assumption of this method is that the local feature matching step should be correct enough so that the candidate nearest neighbors contain the correct ones. Weiss et al. [14] also came up with an approach that use local features only for difficult images, and global features for all other images. The main advantage of this method is that it does not have to extract the local features for each test image, but the algorithm must decide which kind of feature to use for each image.

In this paper, we propose an approach that localizes the mobile robot who follows a previously taken route or almost the same route. The main difference between our work and previous approaches is that we use both global and local image features for each test image. We do not simply combine the global feature vector with the local feature vector, but consider them independently. This makes local feature and global feature contribute to the localization goal separately. A Bayesian framework is used to estimate the probability of the vehicle position as the vehicle moves and new observations are acquired. In order to show the performance of our proposed method, experiments are conducted using a 1.2 km route under different weather conditions.

The rest of the paper is organized as follows: Section II describes the proposed hybrid features we used. Section III explains the Bayesian tracking filter and its formulation. Experimental architecture and results are demonstrated in Section IV. Section V concludes this work and discusses the future work for this study.

II. LOCAL AND GLOBAL IMAGE FEATURES

This section describes the local image feature and global image feature we use in the proposed approach. Considering computational efficiency, the quantized local feature–visual bag of word model instead of exact local image feature is employed. As for global image feature, we choose GIST descriptor, which is a low dimensional representation of the scene.

A. Bag–of–Word Model

The idea of traditional exact local image feature based method is that, the most similar reference image to a query image is the one which has the highest ratio :

$$r = \frac{d_{matching}}{d_{total}} \tag{1}$$

where $d_{matching}$ is the number of local features in the reference image that can be matched to the features of the query image, d_{total} is the total number of features in the reference image. The shortcoming of this method is very time and memory consuming, because each query image feature must be compared to each training image feature. This issue become more prominent when large dataset are used.

Recently, researchers embedded approximate nearest neighbor methods and quantization schemes in the local feature based methods, which make it more suitable for large dataset with thousands of images. Based on these observations, we use typical bag of word model as our local image feature.

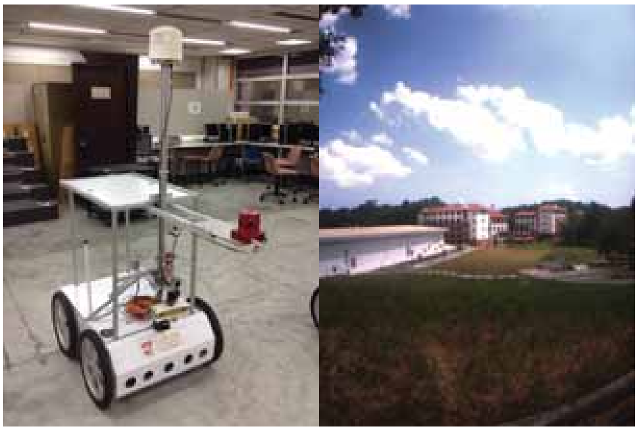
Representing an image with BoW model typically consists of three steps (as in Fig. 1): feature extraction, codebook generation and visual-word histogram generation.

In this study, we use U-SURF [15] descriptor, which is faster than SURF, but still invariant to scale and rotations of the vertical axis. For each detected feature, the descriptor vector is 64 dimensional which is half size of SIFT descriptor. We get a large number of U-SURF features extracted from the training images of our own dataset, and then we use k-means clustering method [16] to obtain cluster centres which can also be called visual-words. The number of the clusters is the codebook size, which can decide the performance of the bag of words method. As we increase the codebook size, the performance will improve, but the requirements for memory storage and computation will also increase. So a trade-off between performance and efficiency must be made during the experiment. This clustering process may take some time because of the large amount of data, but luckily this can be done off-line. Fast approximate nearest neighbor search library (FLANN [17]) is used to assign each local image feature to the nearest visual word. An image is then represented by a weighted histogram of visual words, in which the weights for each individual bin are determined by the occurrence frequency of the visual word.

B. GIST

GIST descriptor was originally proposed to model the shape of a scene [18]. Recently, it has been used by many researchers and shown good results for outdoor localization [19], [20]. The “gist” here means dominant spatial structure (e.g., buildings, trees, streets) that can get noticed when the scene is observed.

Given an input image, the gist feature is computed by convolving the image with the an oriented filter (Gabor filter) at several different orientations and scales. Consequently, the high and low-frequency repetitive gradient directions of an image can be extracted. The scores for the filter convolution at each orientation and scale are stored in an array, which is the GIST descriptor for that image.



(a) Our Robot (b) Captured Images

Fig. 2: Mobile robot and captured images. (b) shows the obtained image by the left sideways-looking camera. The robot is navigated by joystick.

In our experiment, the image is divided into 4x4 grid, and a Gabor filter at 8 orientations at 4 scales is used. To get the final 1x512 feature vector for an image, the histograms of the 16 grid are concatenated.

III. BAYESIAN FRAMEWORK

To estimate current position of the vehicle, we use a Bayesian tracking filter. The Bayesian tracking filter updates the probability of the robot location based on functions of image similarities which are computed using the hybrid image features described in the previous section. In this section, we will give the mathematical derivation of the filtering scheme.

A. Bayes Filter

We now describe how Bayes filter is used to estimate the location of the mobile robot. Our approach is motivated by [21]. Let l_k be the random variable representing the vehicle location (latitude and longitude) at time k, z_k be an observation of the scene appearance at time k, which is represented by the hybrid image features. Z_k denotes the history of the observations $z_1, z_2, ..., z_k$. The objective is to obtain the distribution $p(l_k|Z_K)$, which indicates the probability of the location L_k given all observations up to time k. Rewrite this probability, and follow Bayes rule, we have:

$$p(l_k|Z_k) = p(l_k|z_k, Z_{k-1}) \tag{2}$$

$$p(l_k|Z_k) = \frac{p(z_k|l_k, Z_{k-1})p(l_k|Z_{k-1})}{p(z_k|Z_{k-1})} \tag{3}$$

The denominator:

$$c = p(z_k|Z_{k-1}) = \int p(z_k|l_k)p(l_k|Z_{k-1})dl_k \tag{4}$$

is just a normalization constant, which is independent of variable l_k, and observation z_k at time k is also independent of all previous observations. So rewrite the probability equation, we have:

$$p(l_k|Z_k) = \frac{p(z_k|l_k)p(l_k|Z_{k-1})}{c} \tag{5}$$

here, $p(z_k|Z_k)$ is the likelihood or measurement probability, $p(l_k|Z_{k-1})$ is the prior of the state l_k without knowing the incoming measurement z_k and knowing only the previous measurement Z_{k-1}. Assume this dynamical model to be a Markov model, which indicates that the current true state is conditionally independent of all previous states given the last state, then the prior probability becomes:

$$p(l_k|Z_{k-1}) = \int p(l_k|l_{k-1})p(l_{k-1}|Z_{k-1})dl_{k-1} \tag{6}$$

The probability $p(l_k|l_{k-1})$ specifies how the camera moves from previous time to the current time, and it can be derived from the motion model of the mobile vehicle. The probability $p(l_{k-1}|Z_{k-1})$ is the posterior at previous time $k-1$, which we assume is available at time k. Replacing equation (6) into equation (5), a closed-form expression can be written like this:

$$p(l_k|Z_k) = \frac{1}{c}p(z_k|l_k)\int p(l_k|l_{k-1})p(l_{k-1}|Z_{k-1})dl_{k-1} \tag{7}$$

In this study, we define the route map as a dense grid, so all the formulas derived above become discrete form, which makes a discrete Bayes filter. The discrete Bayes filter also requires the definition of the prediction equation and measurement equation. The derivation of these two equations will be detailed in the following sections for our localization problem using hybrid features.

B. Prediction Equation

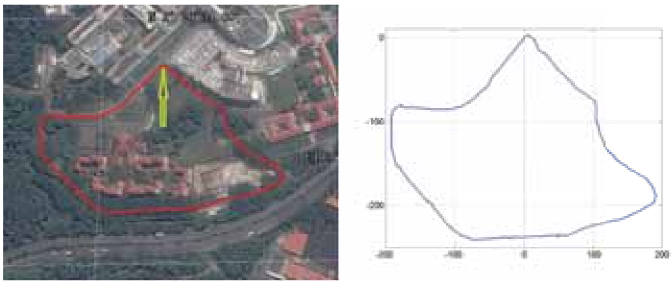
The probability $p(l_k|l_{k-1})$ is used to model transitions between states at time $k-1$ and time k. In the proposed method, no odometry sensor or other instruments are available to read the position of the robot. Alternatively, a simple constant velocity motion model is used to update the state variable l_k. Assume that the velocity of our camera has not changed since the last time step and approximates the robot trajectory with segments of constant linear and angular velocities. To cope with the additional uncertainty due to velocity changes, some randomness are added. Thus the mathematical expression characterizing the state prediction given previous state can be formulated as:

$$\begin{aligned} p(l_k|l_{k-1}) = & \frac{1}{\sqrt{2\pi^2\Sigma}} \cdot exp(-\frac{1}{2}(l_k - (l_{k-1} + v_k))^T \\ & \Sigma^{-1}(l_k - (l_{k-1} + v_k))) \end{aligned} \tag{8}$$

where v_k is the velocity at time k and Σ is the covariance matrix of velocity.

C. Measurement Equation

The conditional probability $p(z_k|l_k)$ is the likelihood of the current observation z_k (represented by hybrid image features) given position l_k, and it can be expressed by functions of the similarity between the extracted features



(a) Robot Path

(b) Ground Truth Data

Fig. 3: Experimental path. The arrow in (a) shows start and end point. The red curve is the trajectory of our robot. The units are shown in meters.

and those in the database. In our experiment, we model the likelihood function like this:

$$
\begin{aligned}
p(z_k|l_k = l_i) = & exp(-w_g \cdot \frac{L_2(g^k - g^i)}{{\delta_g}^2}) \cdot \\
& exp(-w_{bow} \cdot \frac{L_2(bow^k - bow^i)}{{\delta_{bow}}^2})
\end{aligned} \quad (9)
$$

here g^k is the gist descriptor of the image at current time k, g^i is the gist descriptor of the image stored at location l_i; similarly bow^k is the bag-of-word descriptor of the image at current time k, bow^i is the bag-of-word descriptor of the image stored at location l^i. These descriptor vectors are normalized. L_2 is the Euclidean norm distance between the image descriptor vectors; δ_g and δ_{bow} are user-defined variance; w_g and w_{bow} are weighting parameters for local and global descriptors with the following rules:

(1) If $w_g = 1$, $w_{bow} = 0$, we only use global features for image representation

(2) If $w_g = 0$, $w_{bow} = 1$, we only use local features for image representation.

(3) Otherwise, we use hybrid features for image representation.

D. Final State Estimation

The estimated location (latitude and longitude) of the vehicle at every time step will be given by the expectation:

$$
E[l_k|Z_k] = \sum_i {l_k}^i p^i(l_k|Z_k) \quad (10)
$$

IV. EXPERIMENTS

To verify the effectiveness of our system, experiments are conducted using our mobile robot (Fig. 2(a)). The robot is equipped with a Ladybug2 panoramic camera, which consists of six small CCD cameras, five radially configured on horizontal ring and one pointing vertically. In the experiment, we only use the left sideways-looking camera because this orientation contains maximum amount of high-information regions (e.g., a building) [22]. Image obtained by this camera is shown in Fig.2(b), with a resolution of 1024x768. The robot is also equipped with a DGPS, which is used for ground truth generation. The positioning accuracy is 0.5 m under ideal condition.

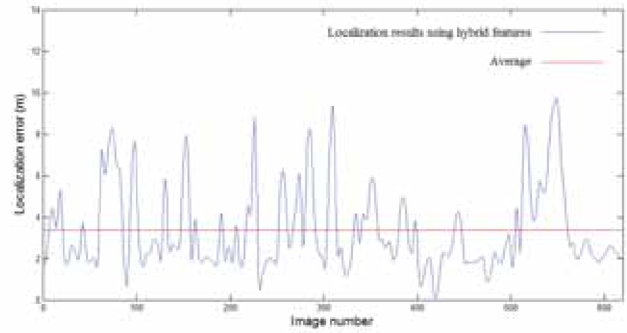


Fig. 4: Localization results using parameter set $w_g = 0.5$, $w_{bow} = 0.5$.

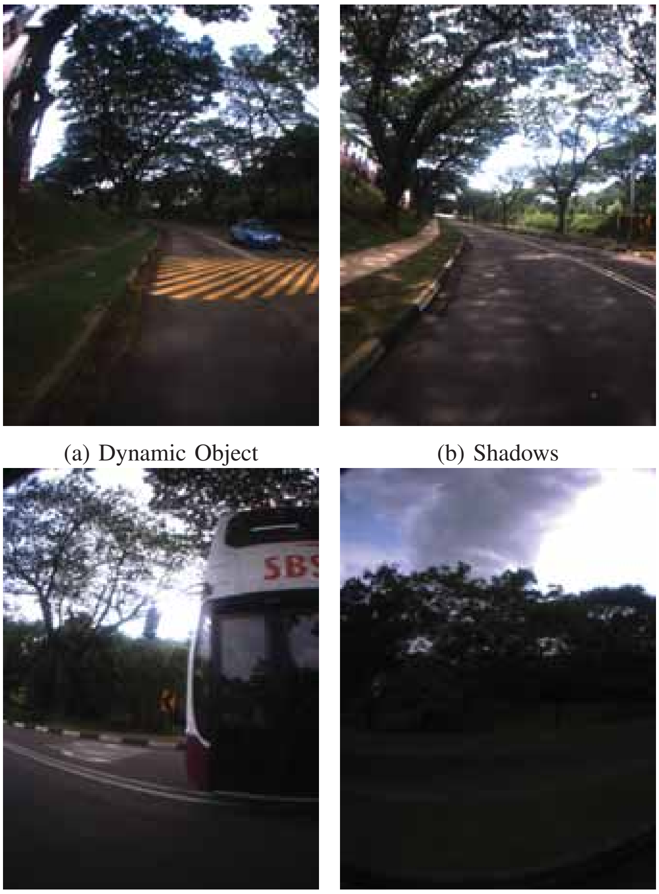
TABLE I: Comparison of the localization results for different weight sets using our test dataset

Different Weights	Avg Error	Std Dev	Div Times
$w_g = 1$, $w_{bow} = 0$	8.06 m	2.42 m	3
$w_g = 0$, $w_{bow} = 1$	8.34 m	2.26 m	4
$w_g = 0.7$, $w_{bow} = 0.3$	3.46 m	2.10 m	0
$w_g = 0.5$, $w_{bow} = 0.5$	3.37 m	2.06 m	0
$w_g = 0.3$, $w_{bow} = 0.7$	3.39 m	2.19 m	0

We chose a 1200m campus path as our experimental route (Fig. 3a.). The route is a loop around hall seven in Nanyang Technological University, Singapore; even though it is not a long route, it includes urban area scenery (e.g. buildings) as well as suburban area scenery (e.g. trees, grass). We collected the datasets at 18:10 PM on April 22, 2014 (a cloudy day after raining) and 14:00 PM on July 26, 2014 (a sunny day). The robot was navigated by joystick and moved at human walking speed. Images were captured at every 1.5 s which approximately makes a two-meter interval between consecutive images. There are around 600 images for both dataset.

Since Singapore does not have very obvious seasonal variations, the appearance of the two dataset should be similar. But the April dataset was collected at rush hours of a weekday, when there were many dynamic objects (Fig. 5a), such as vehicles and people passing by. In contrast, the July dataset was collected during weekend, when the campus was relatively quiet. The different lighting conditions of the two datasets also make the appearance vary greatly. We select the April dataset as the reference, and localization experiment is conducted using the July dataset. Each entry in the datasets contains a bag-of-word descriptor bow^i, a gist descriptor g^i, and the location l_i of the robot when the image was captured.

To evaluate the performance of the hybrid feature in relation to the single local feature and single global feature approaches, different weight sets are tried on our dataset and the results are shown in Table I. We first set $w_g = 1, w_{bow} = 0$, meaning only global image feature is used for image representation. Divergences (A divergence is detected



(a) Dynamic Object

(b) Shadows

(c) Occlusions

(d) Non-discriminative

Fig. 5: Challenging images from our April dataset.

if the localization error is more than 10 m) occur because of occlusions (Fig. 5c) and lateral shift which change the spatial structure of the appearance greatly. The mean error for this weight set is 8.06 m. We then set $w_g = 0, w_{bow} = 1$ so that this time we only use local image feature for image representation. Divergences occur again when robot enters into the areas full of trees and grass which have a lot of repetitive features (Fig. 5d). This weight set has a mean error value of 8.34 m. Lastly, we sweep w_g and w_{bow} from 0 to 1, and several typical results are shown in the last three rows of Table I. As can be seen, the localization accuracies are improved by the fusion of local and global features, and no divergence happens again since local and global image features can make up each other's shortcomings. Among these parameter sets, the optimal set is $w_g = 0.5, w_{bow} = 0.5$, in which the global and local image feature occupy the same proportion. Fig.4 shows the localization results using this set of parameter. The localization error is calculated from the distance between the estimated location using equation (10) and the ground truth location. It shows that the errors do not increase as the camera moves, and the average localization error is 3.37 m (red line of Fig. 4) with a standard deviation of 2.06 m.

V. CONCLUSION

We proposed a mobile robot localization system using both local and global image features. Experiments conducted using a 1.2 km route under different weather conditions showed the robustness of the proposed method.

Bigger dataset as well as other local and global image features will be tested in the future research.

REFERENCES

[1] T. Kos, I. Markezic, and J. Pokrajcic, “Effects of multipath reception on gps positioning performance,” in *ELMAR, 2010 PROCEEDINGS*, pp. 399–402, Sept 2010.

[2] M. Cummins and P. Newman, “FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance,” *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[3] R. Paul and P. Newman, “Fab-map 3d: Topological mapping with spatial and visual appearance,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 2649–2656, May 2010.

[4] M. Cummins and P. Newman, “Highly scalable appearance-only SLAM - FAB-MAP 2.0,” in *Proceedings of Robotics: Science and Systems*, (Seattle, USA), June 2009.

[5] A. R. Zamir and M. Shah, “Accurate image localization based on google maps street view,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2010.

[6] C. Valgren and A. J. Lilienthal, “Sift, SURF & seasons: Appearance-based long-term localization in outdoor environments,” *Robotics and Autonomous Systems*, vol. 58, no. 2, pp. 149–156, 2010.

[7] H. Badino, D. F. Huber, and T. Kanade, “Real-time topometric localization,” in *IEEE International Conference on Robotics and Automation, ICRA 2012, 14-18 May, 2012, St. Paul, Minnesota, USA*, pp. 1635–1642, 2012.

[8] I. Ulrich and I. Nourbakhsh, “Appearance-based place recognition for topological localization,” in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 2, pp. 1023–1029 vol.2, 2000.

[9] M. Milford and G. F. Wyeth, “Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights,” in *IEEE International Conference on Robotics and Automation, ICRA 2012, 14-18 May, 2012, St. Paul, Minnesota, USA*, pp. 1643–1649, 2012.

[10] M. Milford, “Vision-based place recognition: how low can you go?,” *I. J. Robotic Res.*, vol. 32, no. 7, pp. 766–789, 2013.

[11] W. P. Maddern, M. Milford, and G. Wyeth, “CAT-SLAM: probabilistic localisation and mapping using a continuous appearance-based trajectory,” *I. J. Robotic Res.*, vol. 31, no. 4, pp. 429–451, 2012.

[12] A. C. Murillo, G. Singh, J. Kosecká, and J. J. Guerrero, “Localization in urban environments using a panoramic gist descriptor,” *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 146–160, 2013.

[13] A. R. Zamir and M. Shah, “Image geo-localization based on multiplenearest neighbor feature matching usinggeneralized graphs,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 8, pp. 1546–1558, 2014.

[14] C. Weiss, H. Tamimi, A. Masselli, and A. Zell, “A hybrid approach for vision-based outdoor robot localization using global and local image features,” in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 29 - November 2, 2007, Sheraton Hotel and Marina, San Diego, California, USA*, pp. 1047–1052, 2007.

[15] H. Bay, T. Tuytelaars, and L. J. V. Gool, “SURF: speeded up robust features,” in *Computer Vision - ECCV 2006, 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Proceedings, Part I*, pp. 404–417, 2006.

[16] K. Alsabti, S. Ranka, and V. Singh, “An efficient space-partitioning based algorithm for the k-means clustering,” in *Methodologies for Knowledge Discovery and Data Mining, Third Pacific-Asia Conference, PAKDD-99, Beijing, China, April 26-28, 1999, Proceedings*, pp. 355–359, 1999.

[17] M. Muja and D. G. Lowe, “Fast approximate nearest neighbors with automatic algorithm configuration,” in *VISAPP 2009 - Proceedings of the Fourth International Conference on Computer Vision Theory and Applications, Lisboa, Portugal, February 5-8, 2009 - Volume 1*, pp. 331–340, 2009.

[18] A. Oliva and A. Torralba, “Modeling the shape of the scene: A holistic representation of the spatial envelope,” *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

[19] G. Singh, “Visual loop closing using gist descriptors in manhattan world,” in *in Omnidirectional Robot Vision workshop, held with IEEE ICRA*, 2010.

[20] N. Sunderhauf and P. Protzel, “Brief-gist - closing the loop by simple means,” in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp. 1234–1241, Sept 2011.

[21] G. Vaca-Castano, A. Zamir, and M. Shah, “City scale geo-spatial trajectory estimation of a moving camera,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1186–1193, June 2012.

[22] A. Bansal, H. Badino, and D. Huber, “Understanding how camera configuration and environmental conditions affect appearance-based localization,” in *IEEE Intelligent Vehicles Symposium*, June 2014.

PML-SLAM: a solution for localization in large-scale urban environments

Zayed Alsayed*†, Guillaume Bresson*, Fawzi Nashashibi† and Anne Verroust-Blondet†

*Institut VEDECOM
Versailles, France
firstname.name@vedecom.fr

†Inria Paris-Rocquencourt
Le Chesnay, France
firstname.name@inria.fr

***Abstract*—Localization is considered a key factor for autonomous cars. In this paper, we present a Simultaneous Localization And Mapping (SLAM) solution. This algorithm is based on probabilistic maximum likelihood framework using grid maps (the map is simply presented as a grid of occupancy probabilities). The solution mainly solve three renowned localization problems (1. localization in unknown environment, 2. localization in a pre-mapped environment and 3. recovering the localization of the vehicle). Memory issues caused by the open size of outdoor environment are solved using an optimized management strategy that we propose. This strategy allows us to navigate smoothly while saving and loading probabilities-grid submaps into/from a hard-disc in a transparent way. We present the results of our solution using our own experimental dataset as well as the KITTI dataset.**

I. Introduction

One of the key aspects induced by a self-driving car is the ability to localize itself in its environment. This perception task, essential to every navigation system, is crucial for autonomous driving as it is the basis on which trajectory planning and command laws will rely. This topic has therefore received a great amount of attention from the scientific community. However, it is a difficult goal to achieve as such a solution must be able to operate in large environments, provide accurate results in real time and be able to find the vehicle's location in a previously explored environment when necessary.

Simultaneous Localization And Mapping (SLAM) techniques play a central role in making cars truly autonomous. The idea behind SLAM is, for a vehicle, to be able to incrementally build a map of its surroundings while estimating its pose (position and orientation) inside this map. An interesting, but often neglected aspect is that, once a map has been built, it can then be enriched or/and used for re-localization purposes if a vehicle re-enters a previously mapped area.

In order to work in a fully autonomous car (embedding other detection and safety algorithms), the requirements of each algorithm should be optimized. This includes localization system, as well as the architecture surrounding it. They must be designed with this constraint in mind and should not limit the size of the environment in which the vehicle is moving.

We propose here a complete SLAM system, called PML-SLAM for Probabilistic Maximum Likelihood - SLAM, based on horizontal laser sensors. PML-SLAM is well-suited to real-life applications, it's designed to perform in low resource requirements, both in terms of computational time and memory consumption management. PML-SLAM has the particularity to be designed around a solid map management strategy in order to allow navigation to be smooth while save/load/update/enrich map of any size environment. Based on this map management strategy three of the most popular problems related to localization are tackled. 1. localization in unknown environment, where our solution offers the possibility to build a consistent map. 2. localization in known environment, using a pre-built map which offers the possibility to define vehicle's trajectory. 3. the kidnapped robot problem, in this case the vehicle can retrieve precisely its location and orientation after kidnapping without any priori knowledge about its new position. In these three cases the environment map can be updated in real-time even when the vehicle route from a previously explored environment or after kidnapping.

The paper is organized as follows: Section II presents the state of the art regarding SLAM algorithms. Then, Section III introduces the PML-SLAM algorithm with it's different modules that have been developed. The architecture as well as the underlying mathematical framework are presented. Finally, Section IV deals with the experiments and the results obtained.

II. State of the art

Early work on the SLAM problem led to the foundation of a probabilistic framework [1][2][3]. Even though the SLAM problem can now be considered as being theoretically solved [1], many issues have emerged in practical implementations.

The map representation has a great impact on which environments could be tackled and the performance of the algorithm. Three main map representations can be found in the literature: landmark-based maps, grid maps and raw-measurement maps. The first one relies on the environment's specific characteristics to extract significant landmarks (points, edges, etc.) [4]. One strong constraint is to be able to have a sufficiently high number of landmarks in the scene to compute a proper localization. Grid maps divide the environment into cells. Each cell is associated to an occupancy probability [5]. Consequently, these algorithms depend on the geometric structure of the environment. Last, approaches based on raw-measurement maps directly integrate raw sensor data in order to have the most accurate map possible. The direct consequence is that these maps can be difficult to store and can require a high computational cost for processing [6].

The sensors used (information type), the environment's characteristics (information extraction) as well as the estimation process (information processing) should guide the choice of a map representation. Regarding estimation processes, the first SLAM approaches were based on Extended Kalman Filters [1] and Particle Filters [3][7]. Optimization methods, such as Bundle Adjustment [8], are now computationally viable and provide interesting results. These approaches are better suited to a landmark-based representation. Conversely, other solutions based on Likelihood Maximization [9][6] are suited to grid maps. Some implementations also consider a raw-measurement map in addition. Landmark maps are usually built with vision and laser sensors whereas grid maps rely mostly on lasers.

The computational performance of SLAM algorithms is also a major criterion when real-life applications are targeted. This aspect has been considered in many recent SLAM algorithms such as [10] and [11][12]. Still in the objective of making vehicles autonomous, being able to solve the kidnapped robot problem (localize the vehicle in a map given a set of observations) is important [13]. Indeed, the vehicle should be able to use a reference map when the environment has already been mapped, as it increases the relative localization accuracy and allows the map to be enriched [8].

In this paper, we propose a solution to deal with large-scale environments using a standard computer. Our system is based on laser sensors data only, which require less processing for data extraction than cameras. This choice led us to a grid-based representation due to its lightness. Around these choices, we built an estimation process based on likelihood maximization.

III. PML-SLAM

A. Probabilistic SLAM

The general formulation of the SLAM problem is the estimation of the joint probability posterior of the robot pose and the environment map simultaneously over all previous sensor observations and command inputs (cf.[1] for more details) :

$$P(\mathbf{x}_t, \mathbf{M}_t | \mathbf{z}_{0:t}, \mathbf{u}_{0:t}, \mathbf{x}_0) \tag{1}$$

where $\mathbf{x}_t$ is the position and orientation of the robot (x, y, α) at time t, $\mathbf{M}_t$ is the environment map at time t, $\mathbf{z}_{0:t} = \{\mathbf{z}_0, \mathbf{z}_1, ..., \mathbf{z}_t\}$ is the set of all sensor observations up to time t, $\mathbf{u}_{0:t} = \{\mathbf{u}_0, \mathbf{u}_1, ..., \mathbf{u}_t\}$ is the set of robot motion measurements up to time t and $\mathbf{x}_0$ is the initial position and orientation of the vehicle.

SLAM algorithms mainly consist of a two-step recursive process:

- *time update (prediction):* which is a prediction of the robot state and the joint map, based on previous observations as well as command inputs.

$$\begin{aligned} P(\mathbf{x}_t, \mathbf{M}_t | \mathbf{z}_{0:t-1}, \mathbf{u}_{0:t}, \mathbf{x}_0) &= \int P(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t) \\ \times P(\mathbf{x}_{t-1}, \mathbf{M}_{t-1} | \mathbf{z}_{0:t-1}, &\mathbf{u}_{0:t-1}, \mathbf{x}_0) dx_{t-1} \end{aligned} \tag{2}$$

- *measurement update (correction):* which is a correction of the state (vehicle and map) based on the current observation.

$$\begin{aligned} &P(\mathbf{x}_t, \mathbf{M}_t | \mathbf{z}_{0:t}, \mathbf{u}_{0:t}, \mathbf{x}_0) = \\ &\frac{P(\mathbf{z}_t | \mathbf{x}_t, \mathbf{M}_t) \times P(\mathbf{x}_t, \mathbf{M}_t | \mathbf{z}_{0:t-1}, \mathbf{u}_{0:t}, \mathbf{x}_0)}{P(\mathbf{z}_t | \mathbf{z}_{0:t-1}, \mathbf{u}_{0:t})} \end{aligned} \tag{3}$$

B. General Overview

PML-SLAM for Probabilistic Maximum Likelihood - SLAM, is a complete SLAM framework for autonomous cars based on information provided by laser sensors. PML-SLAM offers two different operating modes (localization with or without an a prior map of the environment). It is based on a complete framework to manage large-scale maps (there is no limitation on the environment size and the map can be gradually expanded and/or updated in real-time).

The general flowchart of the PML-SLAM approach is given in Figure 1. It shows the architecture of the implementation which is basically built around three main blocks (dark green boxes).

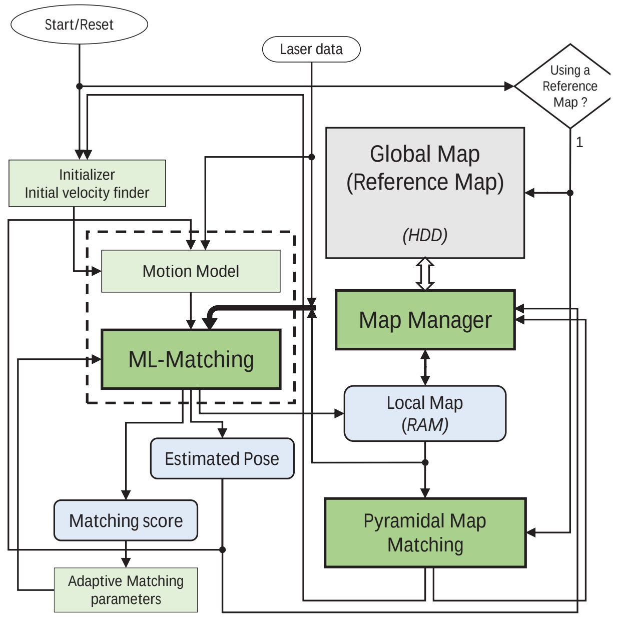


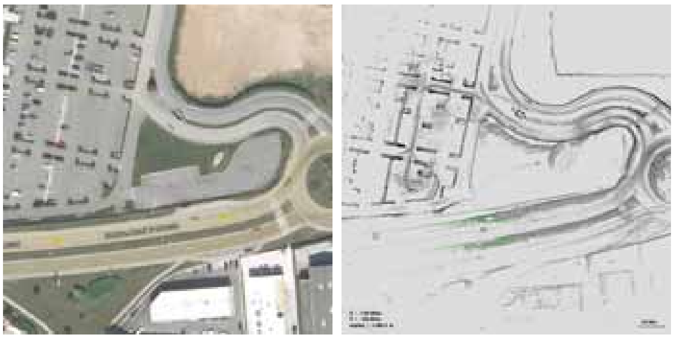
Fig. 1. General flowchart of the PML-SLAM algorithm

Different module illustrated on this flowchart will be detailed in the rest of this section.

C. Overview on Data Representation

The map is expressed through a 2D grid of probability cells with a parameterizable resolution, which allows us to change the discretization level of the environment. We also take advantage of this discretization to maintain maps at different resolutions in order to perform localization recovery faster (cf.

III-H). Although in order to reach the best possible accuracy, the highest resolution of the grid is determined by the laser sensor characteristics. An example of such a map is shown in Figure 2.



(a) Satellite map (b) Map built by PML-SLAM

Fig. 2. Map from Plaisir-France

The map, at resolution level r and at time t, is denoted as $\mathbf{M}_{t,L=r}$. It is the fusion of all submaps $\{\mathbf{M}^1, \mathbf{M}^2, \cdots \mathbf{M}^t\}_{L=r}$, where the submap $\mathbf{M}^t_{L=r}$ is the environment seen at time t. The different resolution level are used to solve kidnapped robot problems (cf. Section III-H).

$\mathbf{M}^t_{L=r}$ is the collection of cells $\{m^1, m^2, \cdots m^N\}$ with N being the maximum number of cells in the submap. Each cell corresponds to an occupancy probability $[0, 1]$ where a cell occupancy probability close to 0 means that it is likely to be free. Conversely, a probability close to 1 means that a cell is likely to be occupied.

Figure 2(b) shows the probability grid map of the real environment presented in Figure 2(a). The dark cells are those with a high occupancy probability and the light cells represent the ones that are likely to be free. The green points represent the current observation data.

D. Motion Model

The Motion Model computes the pose prediction based on a vehicle model. It takes into account the vehicle's previous pose $\mathbf{x}_{t-1}$ and the vehicle's velocity $\mathbf{v}_{t-1}$ to provide a list of candidate poses $\mathbf{X}_t = \{\mathbf{x}_{1,t}, \mathbf{x}_{2,t}, \cdots \mathbf{x}_{n,t}\}$ where the vehicle can be located.

Here, we are using a *Constant Velocity* model to describe the behavior of the vehicle's motion. Indeed, as the time interval δt between two consecutive laser scans is small ($\approx$ 60 milliseconds), we assume that the variation in the vehicle's speed $\delta\mathbf{v}_t$ is also small. Our prediction model is expressed as:

$$\mathbf{\Delta x}_t = \mathbf{\Delta x}_{t-1} + \delta t \times \delta \mathbf{v}_t \tag{4}$$

$$\mathbf{x}_{t|t-1} = \mathbf{x}_{t-1|t-1} + \mathbf{\Delta x}_t \tag{5}$$

The set of candidate poses $\mathbf{X}_t$ is the discretization of the area defined by the velocity variation $\delta t \times \delta \mathbf{v}_t$ (cf. III-A).

E. Initial velocity finder

The role of the Initial velocity finder module is to deal with scenarios which begin when the vehicle is already moving. With the Constant Velocity model presented previously, these cases can lead to wrong estimates at the beginning of such a trajectory.

The module computes a rough estimation of the movement (transformation) between the two first laser scans $\mathbf{z}_0 = \{\mathbf{z}^0_0, \mathbf{z}^1_0 \cdots \mathbf{z}^{N-1}_0\}$ and $\mathbf{z}_1 = \{\mathbf{z}^0_1, \mathbf{z}^1_1 \cdots \mathbf{z}^{N-1}_1\}$ where $\mathbf{z}^i_j$ is the i^{th} 2D point in the j^{th} laser scan composed of N points.

The search space is limited by the maximum distance the vehicle can travel during the time between the first two observations $(\mathbf{z}_0, \mathbf{z}_1)$. Figure 3 shows the distribution of candidate positions all around to cover the possible area. Each square represents one candidate on a low resolution map. These candidate positions are then combined with candidate orientations (depending on the vehicle mechanics) to form a list of candidate poses $\tilde{\mathbf{X}}_1$.

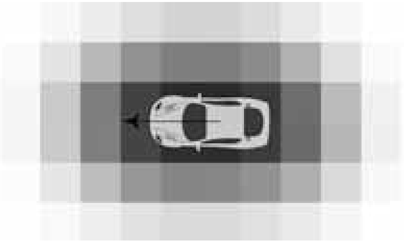


Fig. 3. Search area around the vehicle to compute its movement.

Then the first rough estimate is calculated as follows, where $dist$ is the function computing the Euclidean point-to-point distance between two observations:

$$\tilde{\mathbf{x}}_1 = \underset{\tilde{\mathbf{x}}_1 \in \tilde{\mathbf{X}}_1}{\arg\min} \{ \sum_{i=0}^{N-1} dist(\mathbf{z}^i_0, \mathbf{z}^i_1) \} \tag{6}$$

Finally the initial rough velocity $\mathbf{v}_0$ can then be calculated and transmitted to the Motion Module presented above.

$$\mathbf{v}_0 = (\tilde{\mathbf{x}}_1 - \mathbf{x}_0)/\delta t_{(0,1)} \tag{7}$$

F. Maximum Likelihood Matching

The Maximum Likelihood Matching module corresponds to the correction process in SLAM algorithms (see Equation 3)). It takes into account the new measurement of the environment $\mathbf{z}_t$ in order to refine the state estimation $\mathbf{x}_{t|t-1}$ proposed in the output of the Motion Model.

This module performs a maximization of similarity between the current laser scan $\mathbf{z}_t$ and the map $\mathbf{M}_{t-1}$ from the previous time using the list of candidate poses $\mathbf{X}_t$ provided by the motion model:

$$\begin{aligned} \mathbf{x}_{t|t} = \underset{\mathbf{x}_{t|t-1}}{\arg\max} \{ & P(\mathbf{z}_t | \mathbf{x}_{t|t-1}, \mathbf{M}_{t-1}) \\ & \times P(\mathbf{x}_{t|t-1} | \mathbf{x}_{t-1|t-1}, \mathbf{u}_t) \} \end{aligned} \tag{8}$$

$$P(\mathbf{z}_t|\mathbf{x}_{t|t-1}, \mathbf{M}_{t-1}) \propto \sum_{cell\ i=1}^{N} P(m_{t-1}^i) \tag{9}$$

where m_{t-1}^i is occupied.

In other words, this step consists in finding the best position and orientation $\mathbf{x}_{t|t}$ of the vehicle by fitting the observation $\mathbf{z}_t$ on the map $\mathbf{M}_{t-1}$.

After performing the correction step, we update the system joint state $(\mathbf{x}_t, \mathbf{M}_t)$ by merging the new observation in the map:

$$\mathbf{M}_t = \mathbf{M}_{t-1} \cup \mathbf{M}^t \tag{10}$$

This module is completed by a feedback process to adapt the matching parameters depending on the matching score. The threshold for cell selection (which is based on their probability) is adjusted to maximize the number of matched cells in the next SLAM update. If the matching score is too low, the feedback triggers a reset of the system in order to re-localize the vehicle.

G. Map Manager

The map manager is the key feature of our work. The map is represented by a high resolution occupancy grid and we aim at performing SLAM in large-scale environments. This is not compatible as maps grow in size and we have limited memory resources. The Map Manager module provides a robust management of our limited resources. The goal is to be able to have high resolution maps for accurate matching without limiting the environment size.

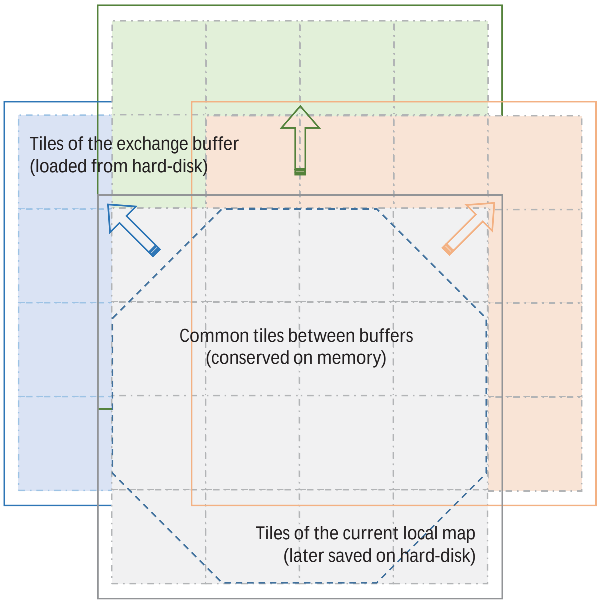

Tiles of the exchange buffer
(loaded from hard-disk)
Common tiles between buffers
(conserved on memory)
Tiles of the current local map
(later saved on hard-disk)

Fig. 4. Map buffers to deal with large-scale environments

Our technique consists in managing the map between the limited Random Access Memory (RAM) and the hard-disk-drive (HDD) where space is not an issue. First, the map is divided into small areas that we call tiles. Then, we only keep in the memory a current local part of the global map which is still big enough to navigate seamlessly from one local map to another. The other tiles are saved on the hard-disk. Tiles are then loaded and unloaded depending on the pose of the vehicle in order to be able to match current observations. An illustration of this map management strategy is given in Figure 4. The grey square is the current local map buffer, containing a grid of 16 tiles, which is actually loaded in the RAM. The other squares (blue, green and red) are candidate configurations to which the local buffer may next move, depending on the vehicle's direction. The arrows of same color shows the next configuration to be adopted.

Furthermore, the key ingredient behind exchanging buffers seamlessly between RAM and HDD in real-time is by compressing probability grid submaps. The buffer (grid of probability) is seen as an image buffer. This allows us to use image compression techniques. Here we use PNG, which keep our data precision, accelerates files saving and loading operations as well as reducing the space required on the hard-drive.

H. Pyramidal Map Matching

The Pyramidal Map Matching module is able to solve the kidnapped robot and relocalization in pre-mapped environments (with what is called a reference map) problems. The idea is to perform an extensive search of the current observation over the reference map to find the robot's pose with relation to this map. As this process can be unreliable and time consuming, the reference map is expressed at different resolution levels. At low resolutions, finding a correspondence is fast but not accurate. By going through the different levels, we are able to refine the pose quickly, by testing only a few pose candidates.

This module uses a similar strategy than proposed by Xie et al. [9]. The main difference comes from the map representation. Instead of a raw-measurement map, we take advantage of our Map Manager module to directly use a grid representation. It has the advantage to allow a faster and constant loading time compared to [9] where the time will vary according to the map size.

As the search can be long without prior knowledge (GPS position for instance), the vehicle continues to perform SLAM while the Pyramid Map Matching module works as a background task. Once the pose on the reference map is found, the current state is brought back into the frame of the reference map.

IV. Experiments

The first experiments were carried out using the KITTI odometry database [14] which provides datasets (with Velodyne, cameras and a ground truth) in various environments for performance evaluation. In order to adapt the data to our implementation inputs, we cut the original Velodyne data to simulate an ordinary single-layer LiDAR scanner configuration (360° field of view with a 0.25° resolution).

In order to test the relocalization performance of PML-SLAM, we also built our own datasets using a vehicle equipped with 5 laser scanners covering a 360° field of view with a 0.25° resolution. Only one layer was used to obtain the results presented in this article. An IMU was fused with a RTK-GPS to provide a ground truth.

All the experiments took place in urban and peri-urban environments in real conditions (moving obstacles were present). All tests were performed on a computer equipped with a Core-i7 running at 2.9 GHz. For all the results presented below, PML-SLAM took on average 2 ms to process a scan layer.

A. SLAM in unknown environments

In this experiment, we used the sequence ”05” from the KITTI odometry dataset, collected in a residential environment. We performed SLAM without any prior knowledge on the environment.

At the beginning of the trajectory, a map of 16 tiles (resolution 10 $pixels/meter$) were initialized. During the whole trajectory, a total of 32 tiles were created and saved on the hard disk, covering a $720,000\ m^2$ surface. The map produced takes up 981 KB on the hard disk. The computed trajectory is shown in Figure 5.

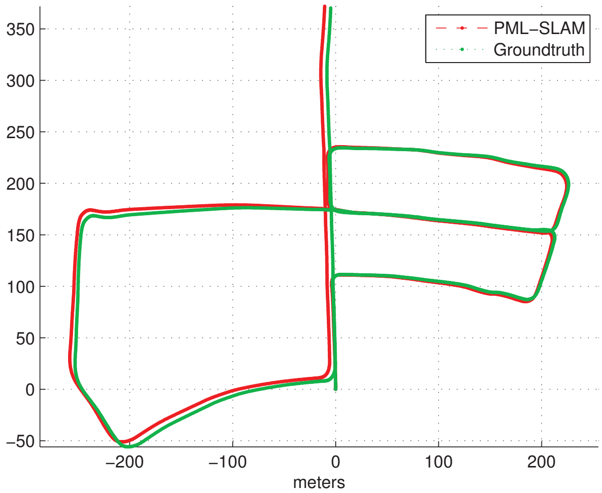


Fig. 5. Vehicle paths from the Kitti odometry dataset, the path estimated by PML-SLAM and the ground truth path.

The errors in displacement and heading at each iteration are illustrated in Figure 6. As we can see, the errors in displacement are bounded by ± 5 cm, while the errors in heading are limited to ± 0.3°.

Path estimated by SLAM is close to the ground truth. The deviation throughout the trajectory in distance and in heading are shown in Figure 7. We can see that after $2,204$ meters the deviation in distance is about 6 meters with 0.1°of deviation in heading.

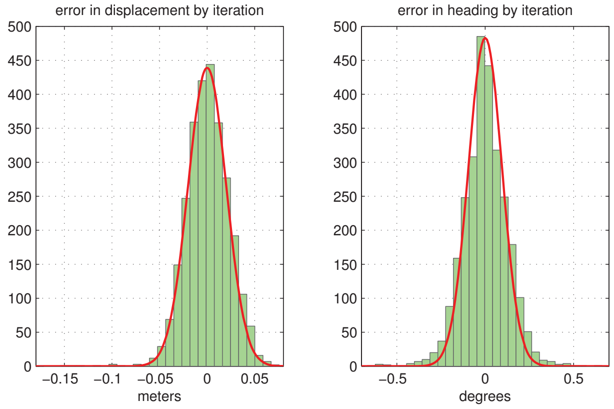


Fig. 6. Errors in heading and displacement by iteration during the test.

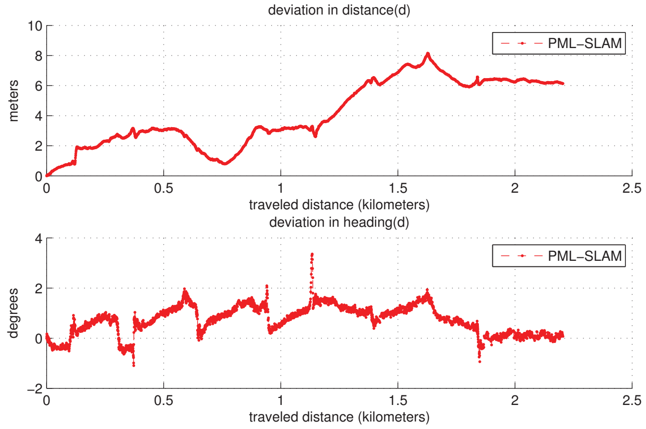


Fig. 7. Deviation from real path in distance and heading along journey.

B. Recovering from kidnapping

In this experiment we used a dataset which we collected in a city using our platforms. The aim of this test is to illustrate the kidnapped robot problem and how we deal with it. The trajectory performed and the recovery from kidnapping are shown in Figure 8.

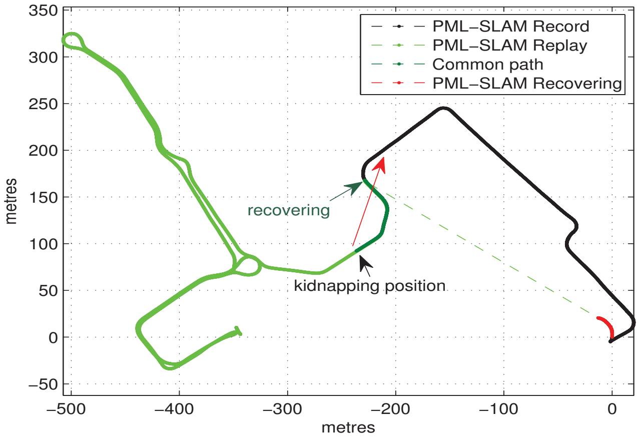


Fig. 8. Vehicle path estimated by PML-SLAM during experiment in mapping with kidnapping and recovering from kidnapping.

The vehicle starts to build a map (black path), then the vehicle is kidnapped and put in another location. As a consequence, the vehicle starts a new local SLAM process (red path) whose computed trajectory takes a similar shape to the previous route. During this time, the position is provided in a local reference. The matching process then finds the vehicle new position on the pre-built map (recover from kidnapping). The recovery moment is represented by the dashed green lines. The vehicle's positions provided from now on are given on the global map. The map is further enriched during the common path (in dark green). Once the vehicle enters areas not previously mapped (light green), the vehicle continues to map and localize itself.

The map built at the beginning of the experiment served as a recovering-from-kidnapping map, and covered $360,000\ m^2$. The localization precision after recovering was ± $5\ cm$.

The recovering operation took about 27 seconds in this example. However, this time depends on the random selection of matching candidates and is thus variable. Nevertheless, while searching for its position, the vehicle is still localizing itself with the SLAM process. The final map built takes up 508 KB on the hard disk and it should cover $517,500\ m^2$ of surface.

C. Memory consumption

The limitation in resources is an important issue. Figure 9 shows a comparison in terms of memory consumption regarding the maximal size of the area covered by the SLAM system with and without the Map Manager module presented in this article (see Section III-G). Here we consider a scenario where the vehicle is going straightforward.

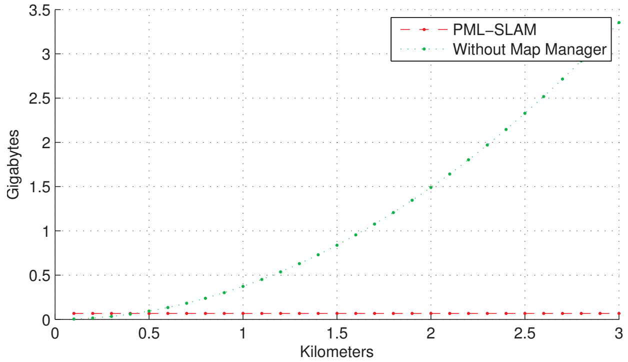


Fig. 9. Memory consumption regarding the area covered by SLAM

We can see on the graphic that memory required by a SLAM system without Map Management may increases quadratically in function of the environment size. In contrast the memory required by a SLAM with a Map Manager is constant regardless the environment size. In realistic scenarios of autonomous driving we can't predefine the map size, a vehicle needs to have it's freedom to navigate.

V. Conclusion

We have presented a complete SLAM algorithm, called PML-SLAM. The proposed system is based on the probabilistic maximum likelihood framework coupled with a grid representation. Its integration inside a complete architecture allows to use any previously built map if available. A map matching algorithm, based on a pyramidal search with multi-resolution map, is able to re-localize the vehicle inside a map given an observation, thus solving the kidnapped robot case. Our approach has been designed with real-life applications in mind and is consequently inexpensive in terms of memory requirements and processing time. The map is smartly managed to be loaded and unloaded on the hard drive when needed thus allowing to work in large-scale environments.

We have performed several experiments over different scenarios: SLAM localization, relocalization and navigation inside an existing map. The results show that our approach is viable for real-time urban localization without restrictions on the size of the environment.

In future work, we plan to fuse several localization algorithms within a supervision layer in order to increase robustness and localization accuracy.

References

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous Localization and Mapping: Part I," *IEEE Robotics and Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[2] T. Bailey and H. Durrant-Whyte, "Simultaneous Localization and Mapping (SLAM): Part II," *IEEE Robotics and Automation Magazine*, vol. 13, no. 3, pp. 108–117, 2006.

[3] S. Thrun, W. Burgard, and D. Fox, "A Real-Time Algorithm for Mobile Robot Mapping With Applications to Multi-Robot and 3D Mapping," in *IEEE International Conference on Robotics and Automation*, vol. 1, 2000, pp. 321–328.

[4] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time Single Camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[5] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[6] T.-D. Vu, "Vehicle Perception: Localization, Mapping with Detection, Classification and Tracking of Moving Objects," Ph.D. dissertation, Inria Grenoble Rhône-Alpes - LIG Laboratoire d'informatique de Grenoble, 2009.

[7] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A Factored Solution to the Simultaneous Localization And Mapping Problem," in *AAAI/IAAI*, 2002, pp. 593–598.

[8] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Real Time Localization and 3D Reconstruction," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006, pp. 363–370.

[9] J. Xie, F. Nashashibi, M. Parent, and O. G. Favrot, "A real-time robust global localization for autonomous mobile robots in large environments," in *Control Automation Robotics & Vision (ICARCV), 2010 11th International Conference on*. IEEE, 2010, pp. 1397–1402.

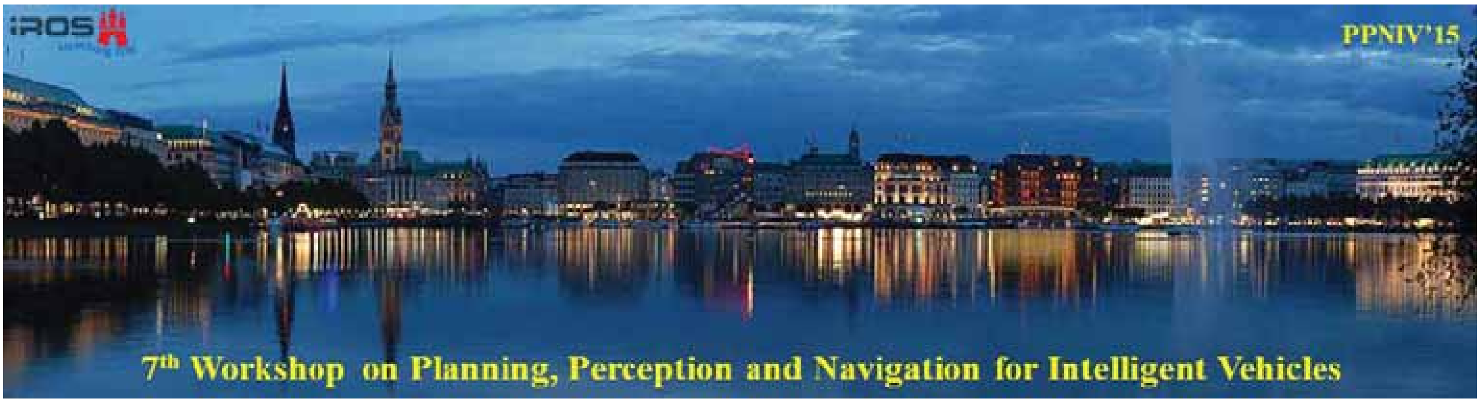
[10] O. E. Hamzaoui, "A fast scan matching for grid-based laser SLAM using streaming SIMD extensions," in *Control Automation Robotics & Vision (ICARCV), 2010 11th International Conference on*, 2010, pp. 1986–1990.

[11] ——, "tinySLAM: A SLAM algorithm in less than 200 lines C-language program," in *Control Automation Robotics & Vision (ICARCV), 2010 11th International Conference on*, 2010, pp. 1975–1979.

[12] J. Levinson, M. Montemerlo, and S. Thrun, "Map-Based Precision Vehicle Localization in Urban Environments," in *Proc.˜of Robot. Sci. Syst.*, vol. 4. Citeseer, 2007, p. 1.

[13] S. Se, D. Low, and J. Little, "Global Localization using distinctive visual features," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002, pp. 226–231.

[14] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets Robotics: The KITTI Dataset," *International Journal of Robotics Research (IJRR)*, 2013.


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session II

Perception & Situation Awareness

- **Keynote speaker: Dr.-Ing. Michael Darms (Volkswagen Aktiengesellschaft, Group Research, Germany)**

  Title: Perception for automated and assisted driving

- **Title: Free-space Detection using Online Disparity-supervised Color Modeling**

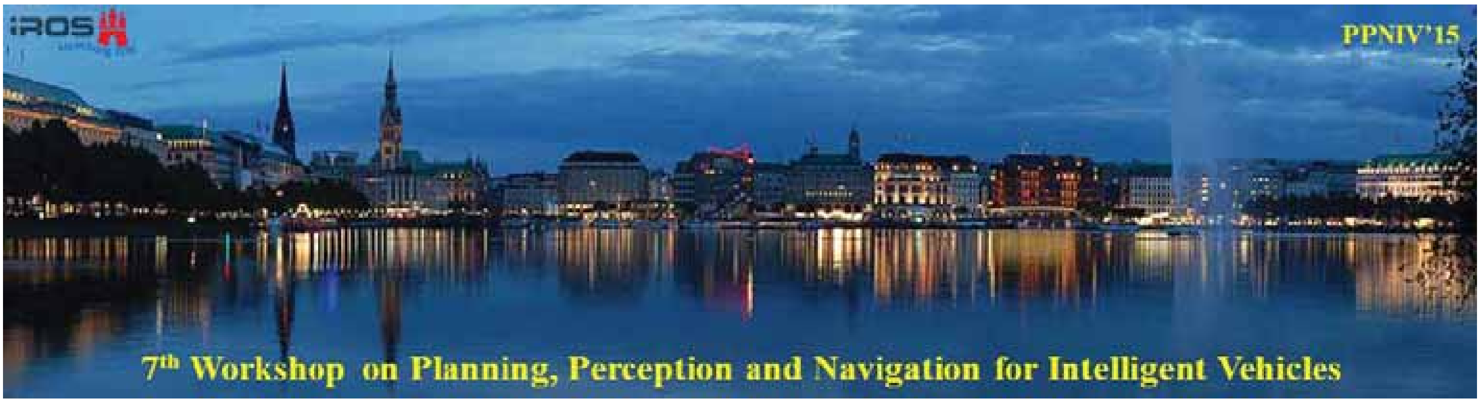
  Authors: Willem P. Sanberg, Gijs Dubbelman and Peter H.N. de With

- **Title: Vision-Based Road Detection using Contextual Blocks**

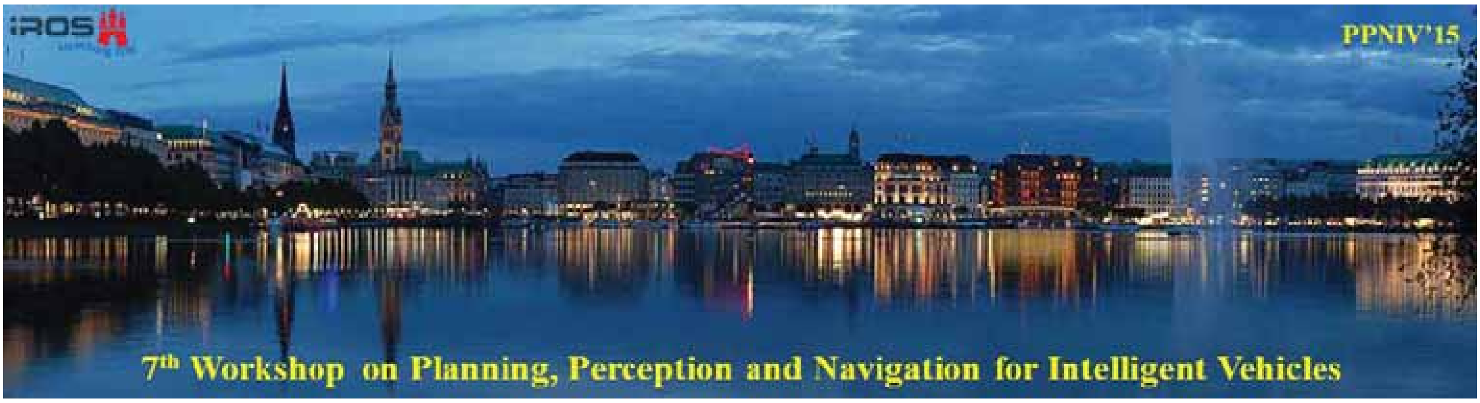
  Authors: Caio Cesar Teodoro Mendes, Vincent Frémont and Denis Fernando Wolf

- **Title: Following Dirt Roads at Night-Time: Sensors and Features for Lane Recognition and Tracking**

  Authors: Sebastian F. X. Bayerl, Thorsten Luettel and Hans-Joachim Wuensche


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems


IROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

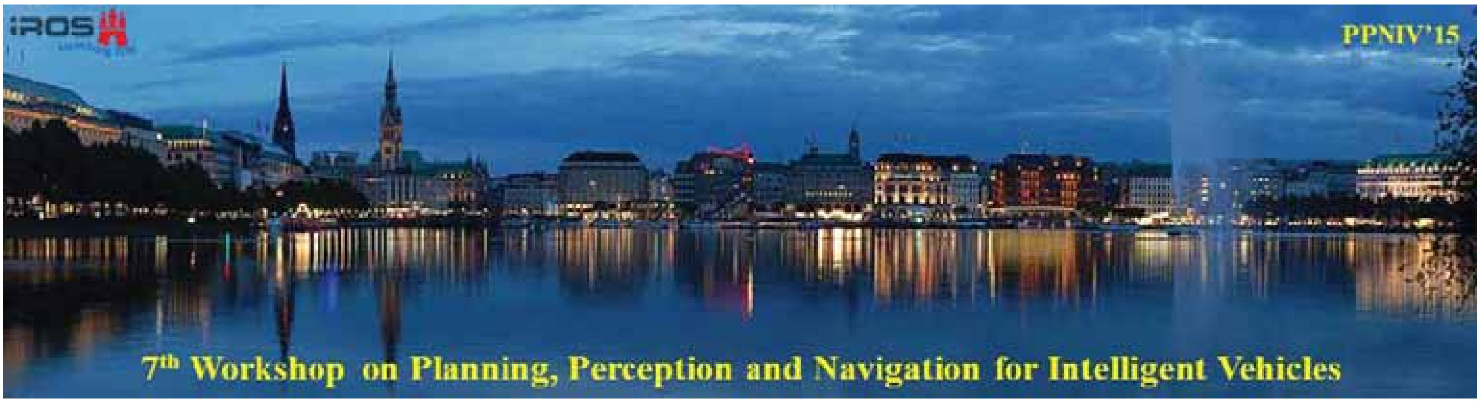
Session II

Keynote speaker: **Dr.-Ing. Michael Darms**
(Volkswagen Aktiengesellschaft, Group Research, Germany)

Perception for automated and assisted driving

Abstract: The talk gives an overview on the design of perception systems for automated and assisted driving. It compares the requirements of the two different domains and discusses the challenge of having an application dependent situation awareness layer with an application independent perception layer. One focus of the talk is on the task of deriving information about the location of the road and lanes from sensor data which is still a key challenge for automated and assisted driving. It is discussed how methods stemming from the field of neural networks can be applied and how a priori information stemming from maps can be used in the data fusion process. Finally the talk will give an overview of the perception system implemented in Jack, an automated vehicle with which Audi and VW Group Research demonstrated the maturity of the current stage of development of automatic driving. The vehicle did a piloted drive over two days and 550 miles under real conditions on a highway from the San Francisco Bay Area to Las Vegas. An outlook is given on how such a perception system can be integrated in a modular and scalable architecture and which approaches are thinkable for testing such a system.

Biography: Dr.-Ing. Michael Darms received his doctoral degree on Sensor Data Fusion in 2007 at Darmstadt University of Technology, Institute of Automotive Engineering. From 2002 to 2006 he worked on the perception system of project PRORETA at this institute. Within the project a new driver assistance system for emergency evasion maneuvers was developed. From 2006 to 2007 he was visiting researcher at the Robotics Institute of Carnegie Mellon University. As part of Team Tartan Racing he was working on the perception system of the autonomous vehicle BOSS which took first place in the DARPA Urban Challenge. From 2008 to 2010 he was employed at the Advanced Engineering Department of Continental AG leading the perception group. Continental's first demonstrators for automated driving where set up during this time. From 2010 to 2014 he was technical project manager for camera systems at Continental. As one of the key projects he led the technical development of Continental's first series stereo camera system. In 2014 he joined VW Group Research and is now leading the department Sensors and Fusion. Focuses of his work are perception systems for future automated and assisted vehicles.


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

VOLKSWAGEN
AKTIENGESELLSCHAFT

PERCEPTION FOR AUTOMATED AND ASSISTED DRIVING

DR.-ING. MICHAEL DARMS
GROUP RESEARCH / HEAD OF SENSORS AND FUSION

IROS – 28.09.2015

VOLKSWAGEN
AKTIENGESELLSCHAFT

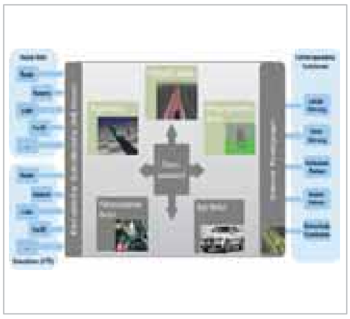
ENVIRONMENT PERCEPTION

Motivation

- Highly automated driving functions impose increased requirements on the performance of the perception module
- Perception module has to cope with contradictory requirements (comport vs. safety systems)

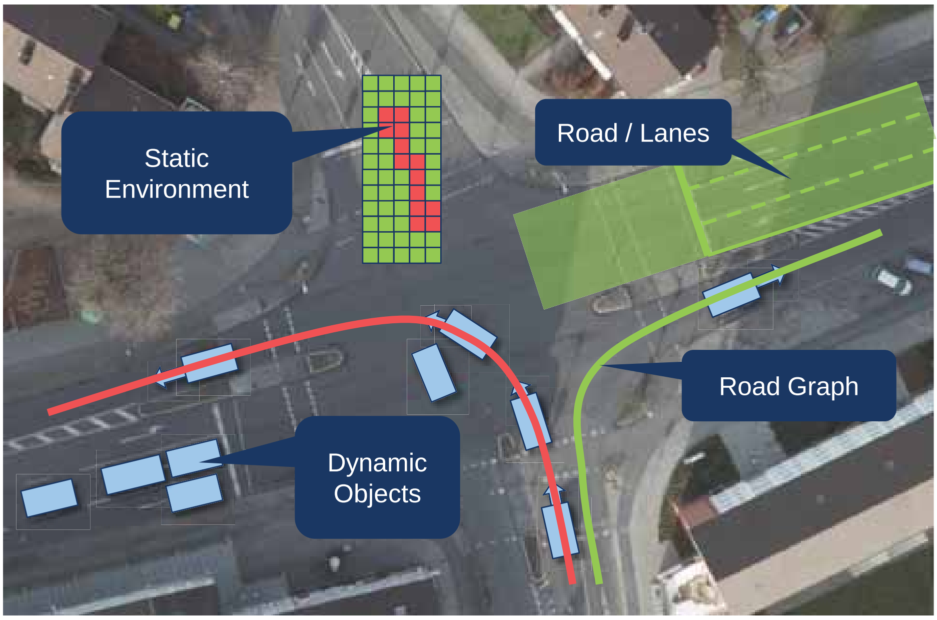
Goals

- Development of a modular and extensible perception architecture
- Environment model has to be independent of specific ADAS function
- Implementation of unified interface to function

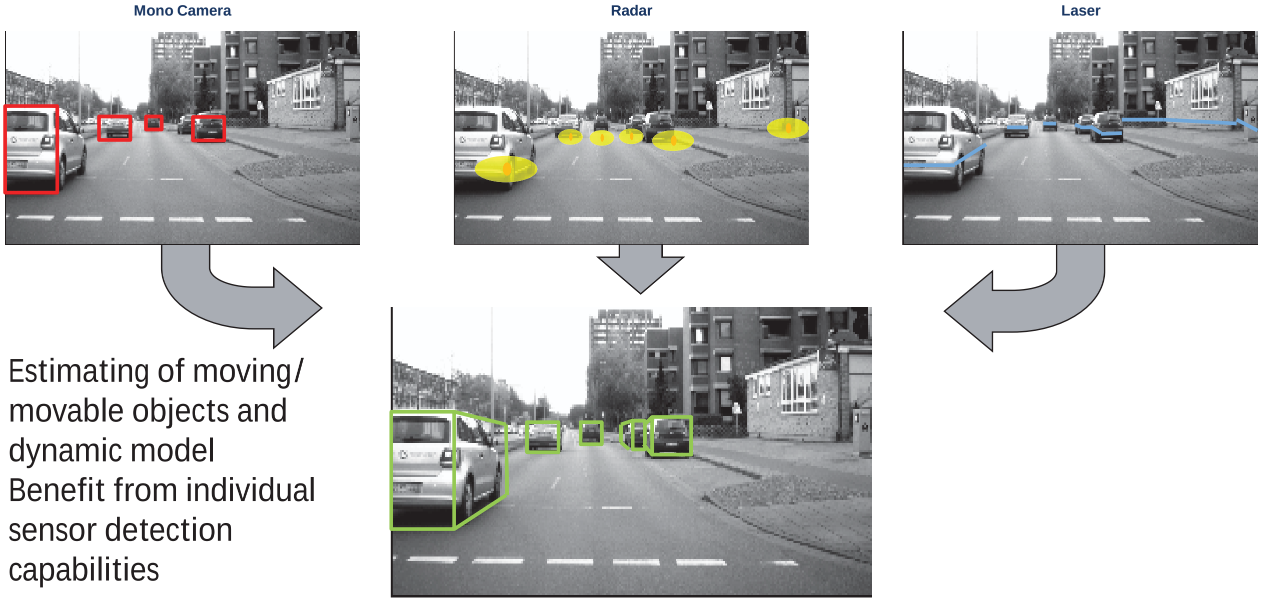


WORLD MODEL

Representation of the world around the vehicle

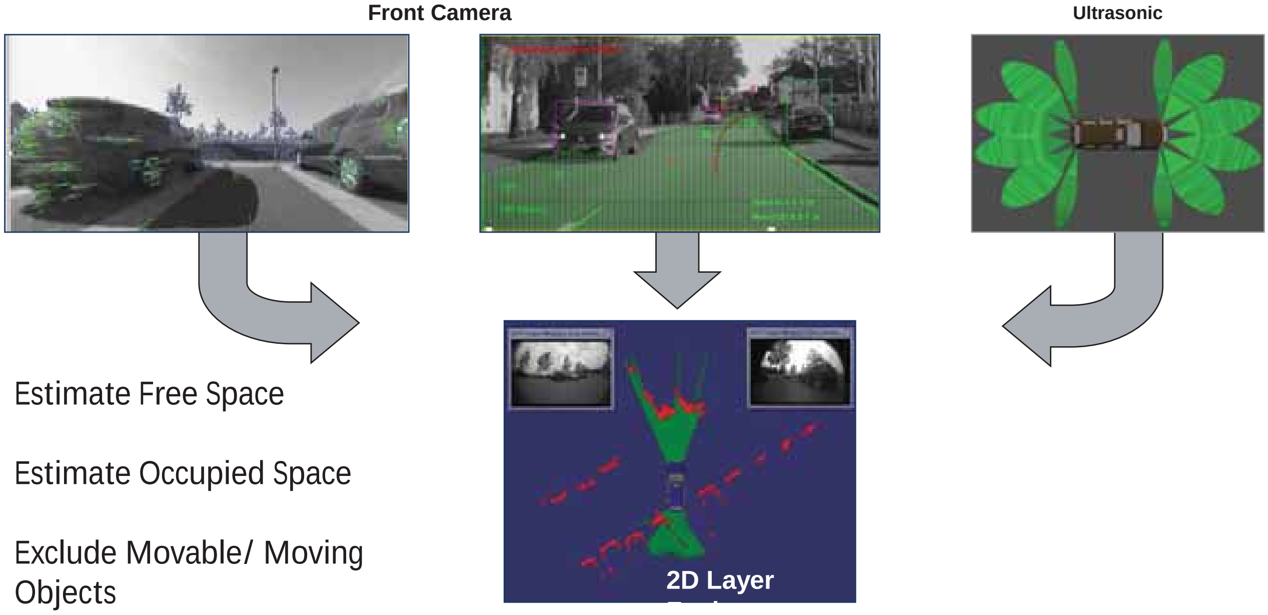


OBJECT ESTIMATION



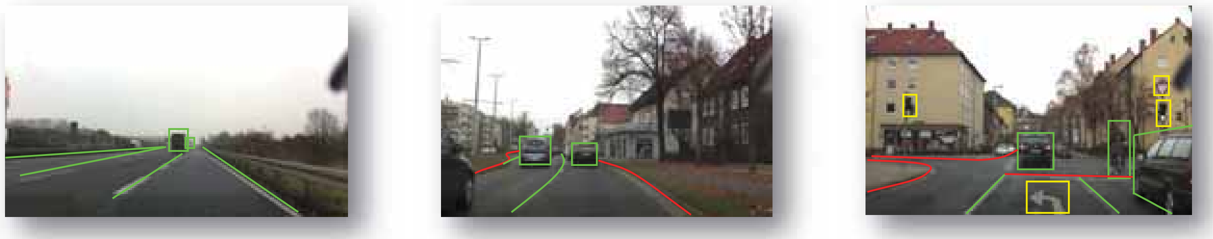
- Estimating of moving / movable objects and dynamic model
- Benefit from individual sensor detection capabilities

GRID ESTIMATION



- Estimate Free Space
- Estimate Occupied Space
- Exclude Movable/ Moving Objects

ROAD ESTIMATION



- Interpret where in the world vehicles should drive using feature cues from the environment

Highway

- Lane Markings
- Other Vehicles

Rural Roads

- Lane Markings
- Other Vehicles
- Curbs, sward

Inner City

- Lane Markings
- Other Vehicles
- Curbs
- Traffic Lights, Trafic Signs
- Digital Maps

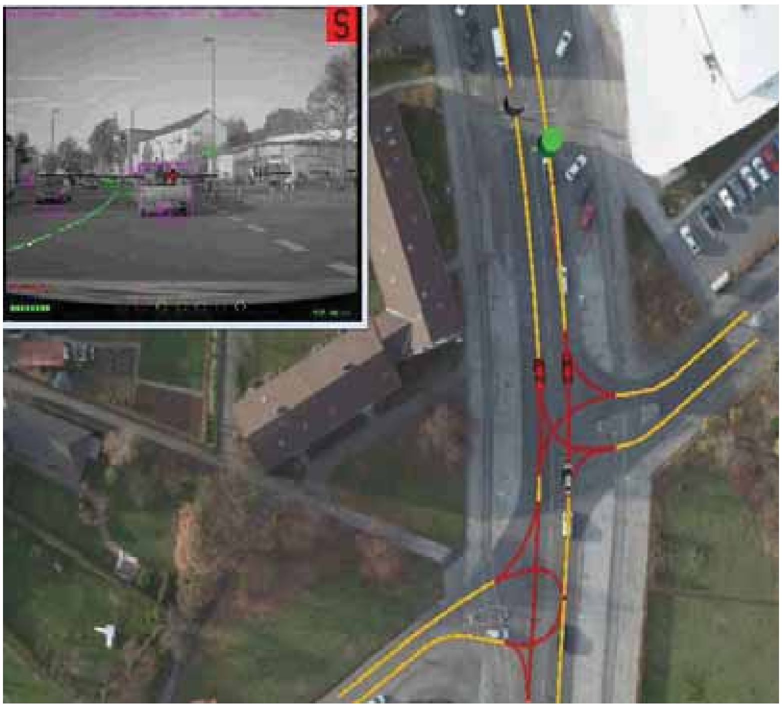
SCENE ESTIMATION

RoadGraph

- Output of the perception modules is integrated into one model
- Roadgraph is main interface to driving functions

Challenges

- Fusion of road estimation and context knowledge into unified, consistent and comprehensive model
- Scene Estimation for urban areas, especially in complex intersection scenarios
- Handling of traffic participants that do not behave as expected



V-CHARGE:
AUTONOMOUS VALET PARKING AND CHARGING FOR E-MOBILITY

Automated valet parking and charging

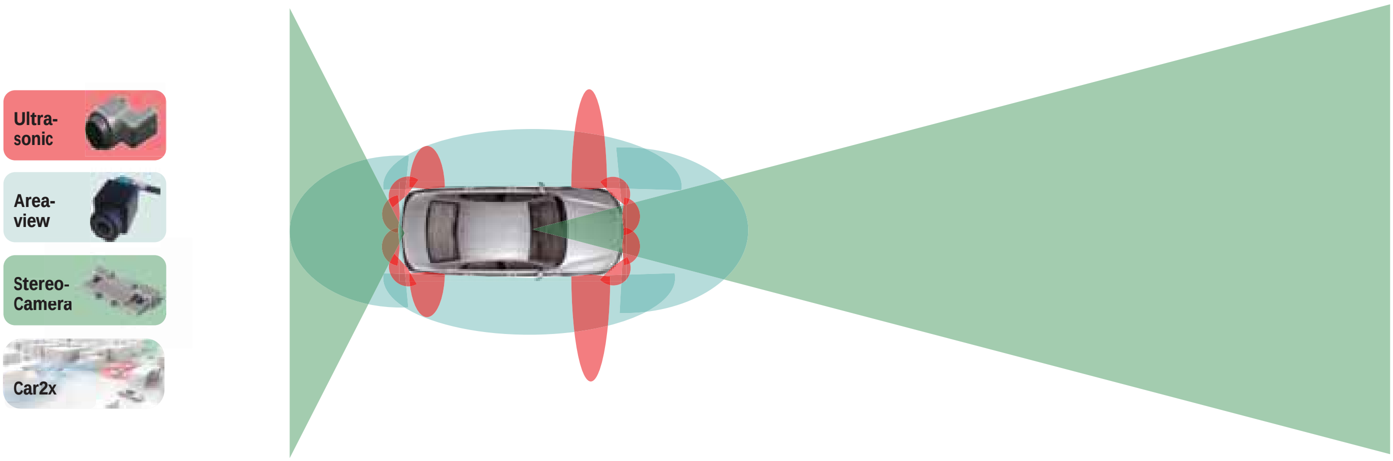
- no time-consuming search for parking spots any more
- driverless valet service
- no human intervention

Fully automated driving

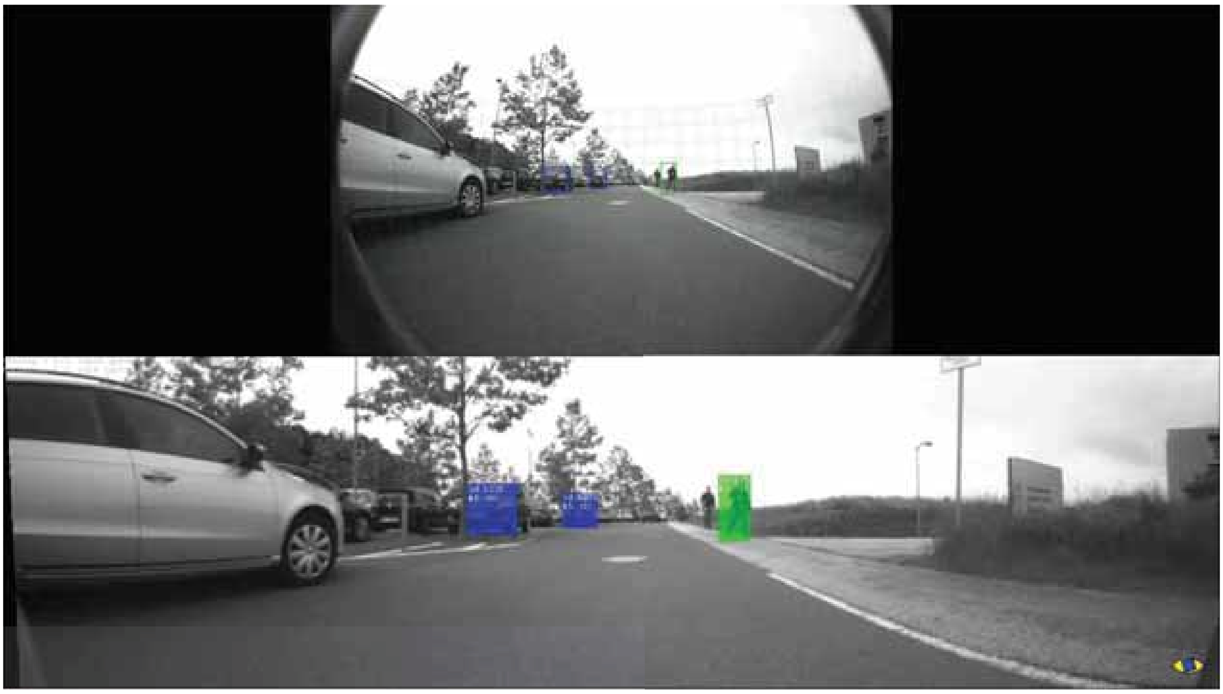
- in mixed-traffic scenarios
- in indoor and outdoor parking lots and parking garages without GPS

V-CHARGE: AUTONOMOUS VALET PARKING AND CHARGING FOR E-MOBILITY

Close-to-series sensors

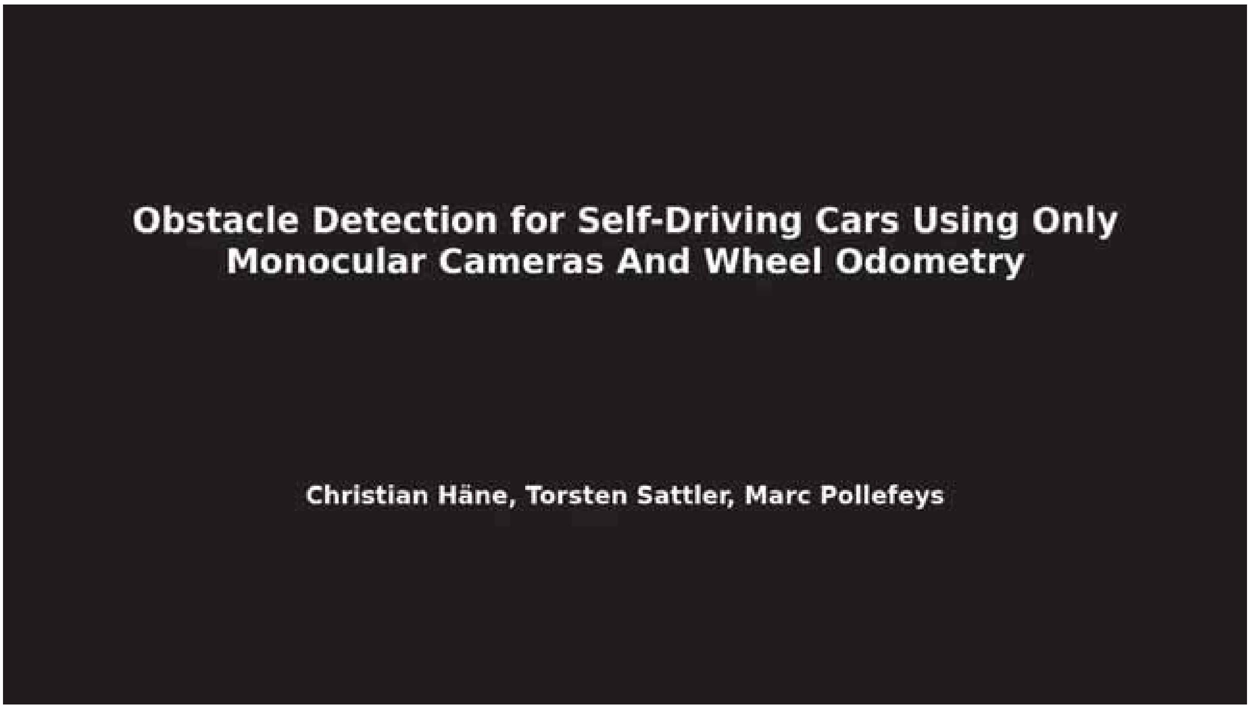


OBJECT ESTIMATION USING FISH-EYE CAMERAS



University of Parma, VisLab

GRID ESTIMATION USING FISH-EYE CAMERAS

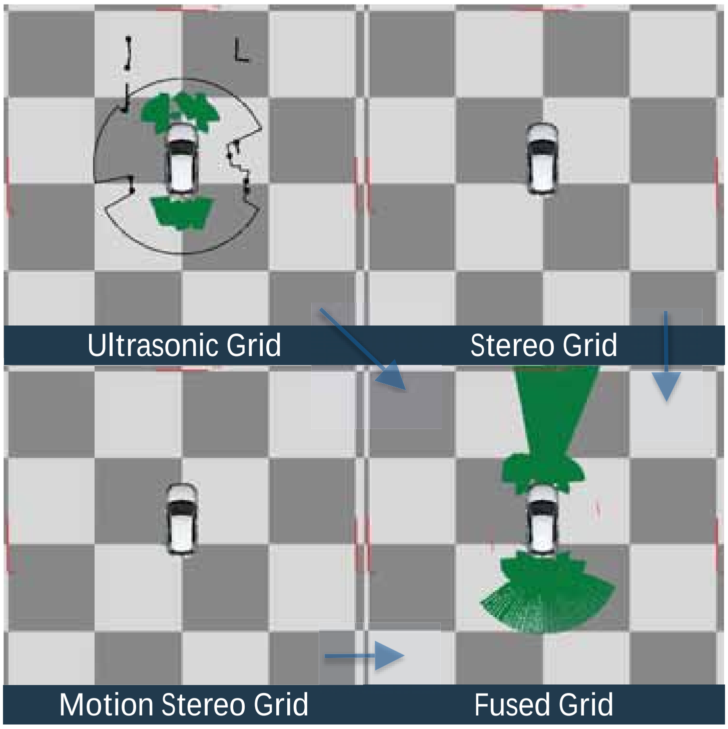


Obstacle Detection for Self-Driving Cars Using Only Monocular Cameras And Wheel Odometry

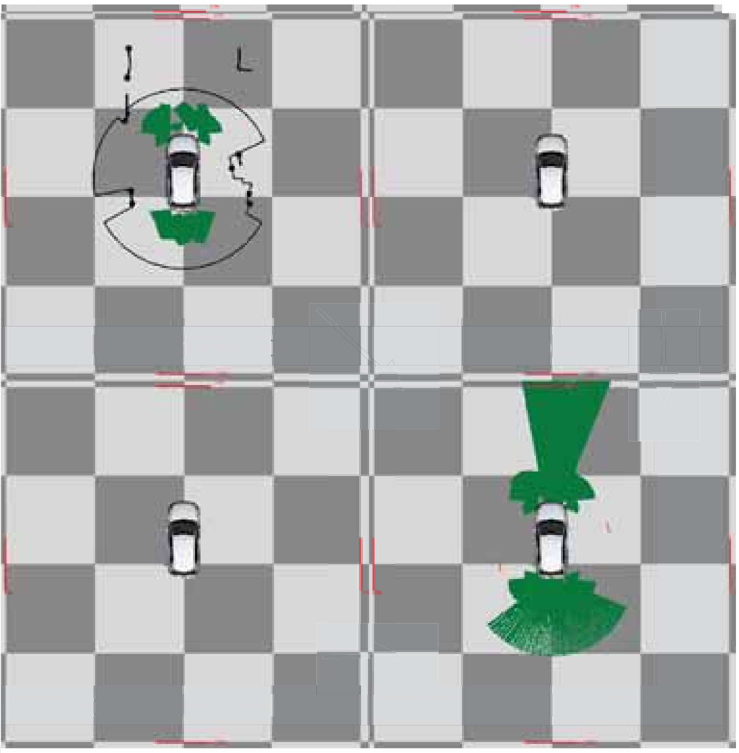
Christian Häne, Torsten Sattler, Marc Pollefeys

C. Häne, T. Sattler, M. Pollefeys, Obstacle Detection for Self-Driving Cars Using Only Monocular Cameras and Wheel Odometry

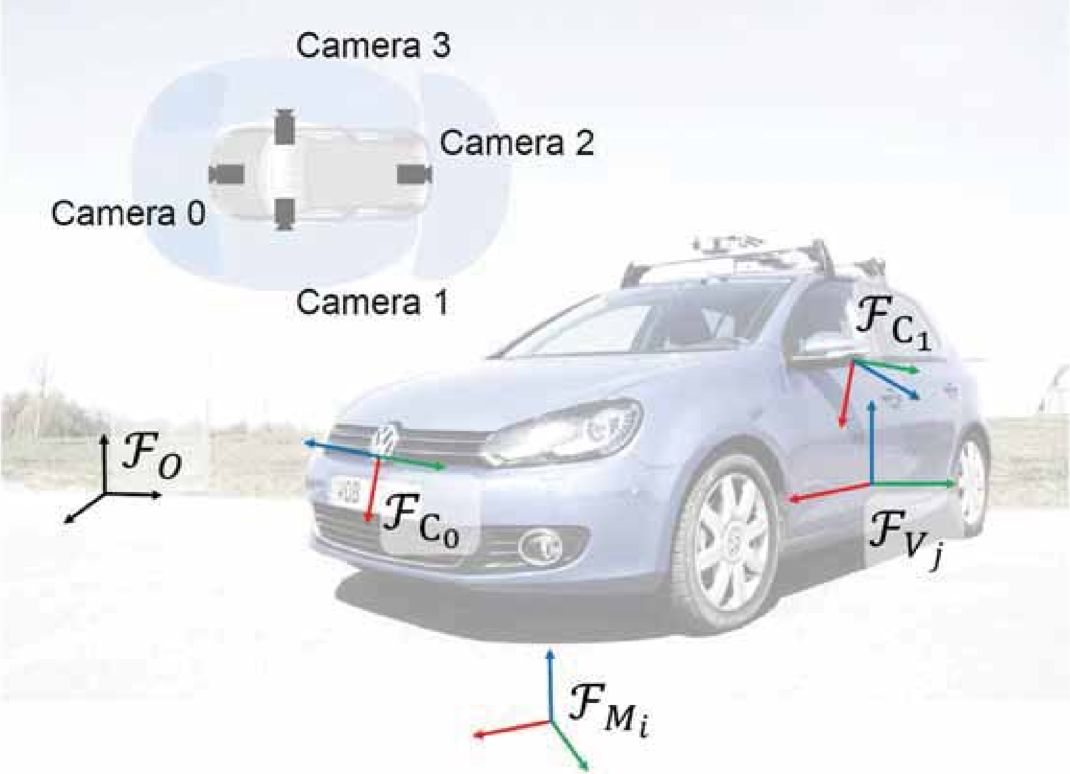
GRID ESTIMATION WITH ALL SENSORS



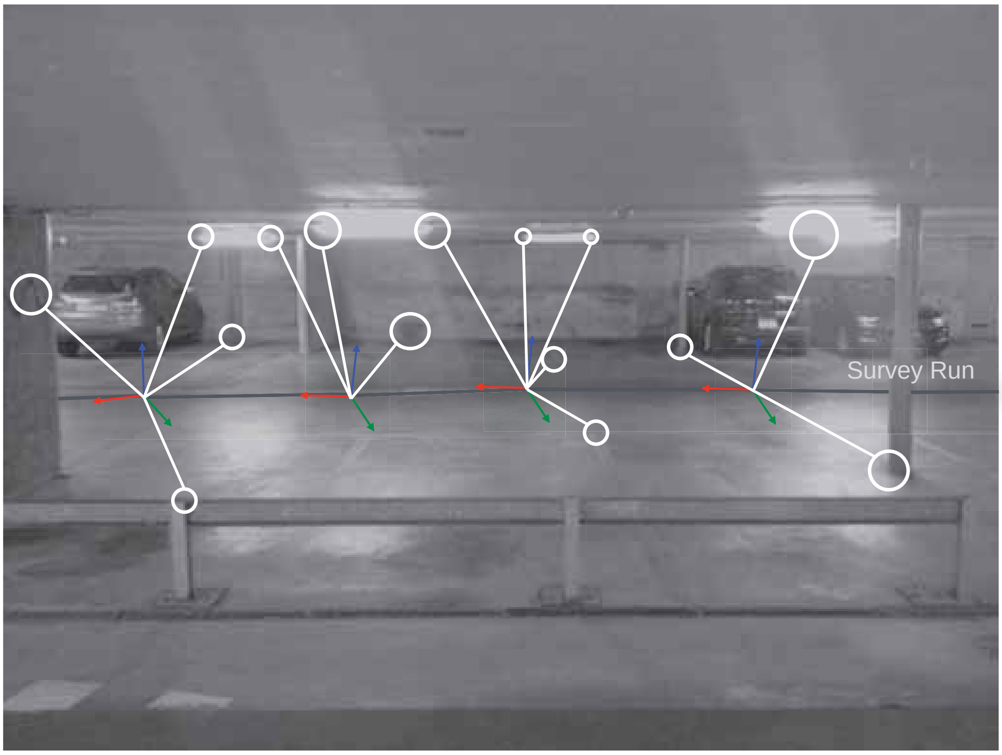
GRID ESTIMATION WITH ALL SENSORS



ROAD ESTIMATION: KNOWING WHERE THE ROAD IS VIA LOCALIZATION

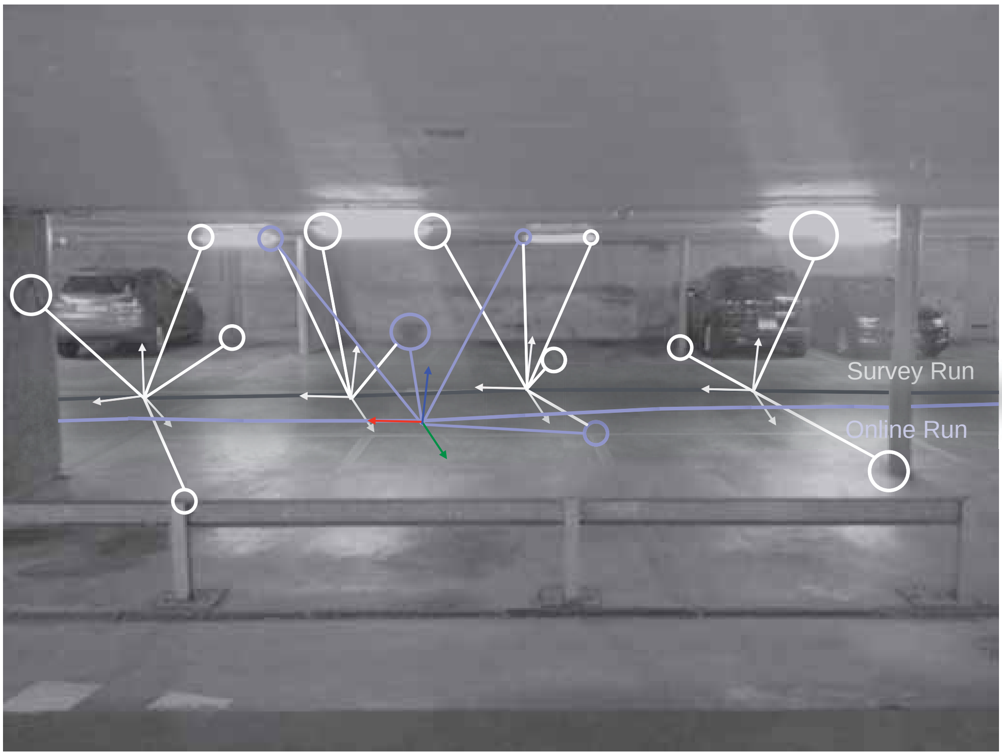


APPROACH: KNOWING WHERE THE ROAD IS - LOCALIZATION

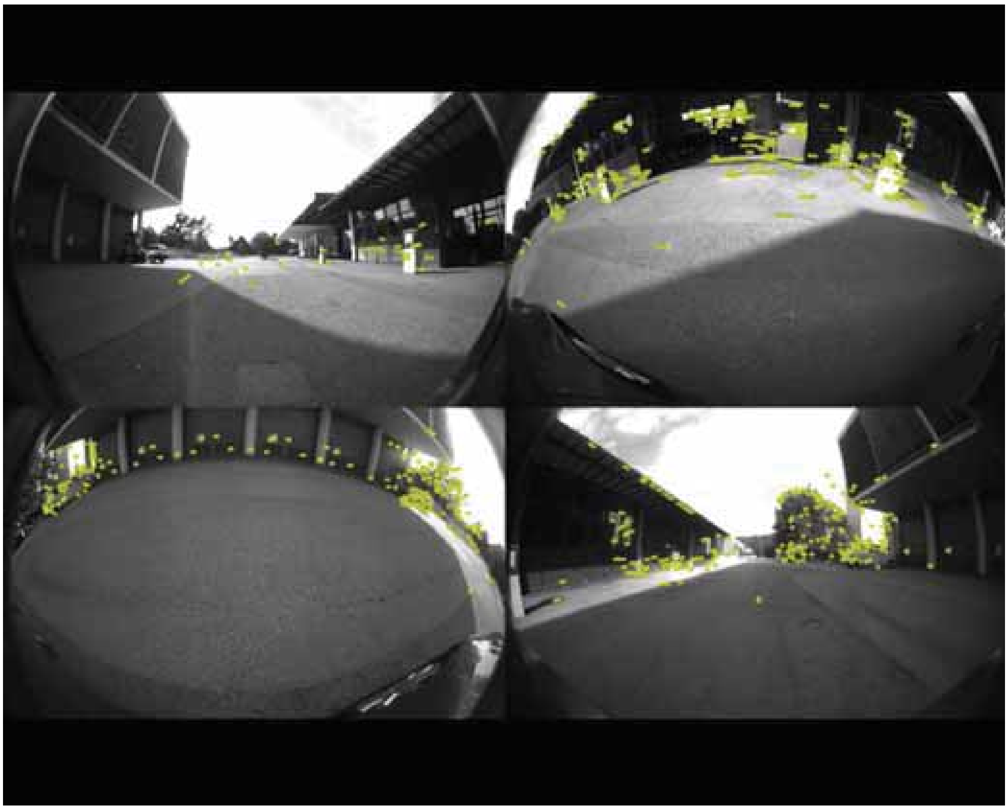


Furgale et al., ICRA2014

APPROACH: KNOWING WHERE THE ROAD IS - LOCALIZATION



Furgale et al., ICRA2014

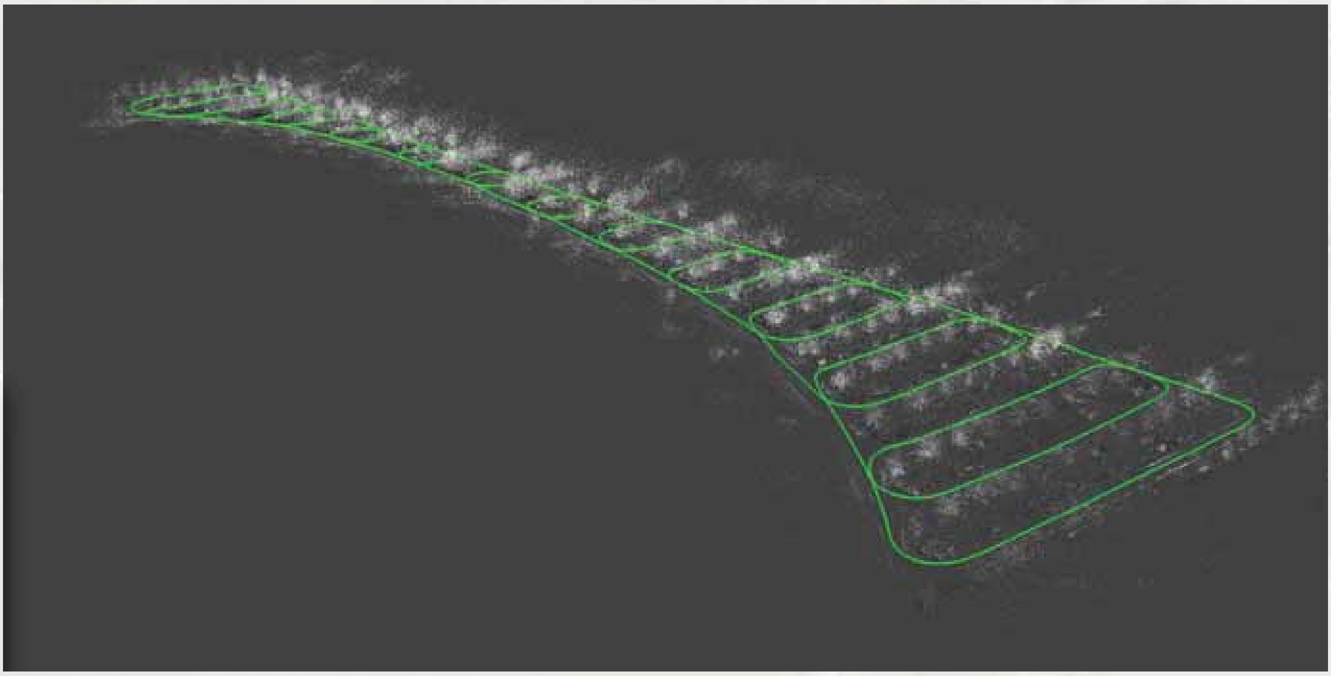


Furgale et al., ICRA2014

APPROACH: KNOWING WHERE THE ROAD IS - LOCALIZATION

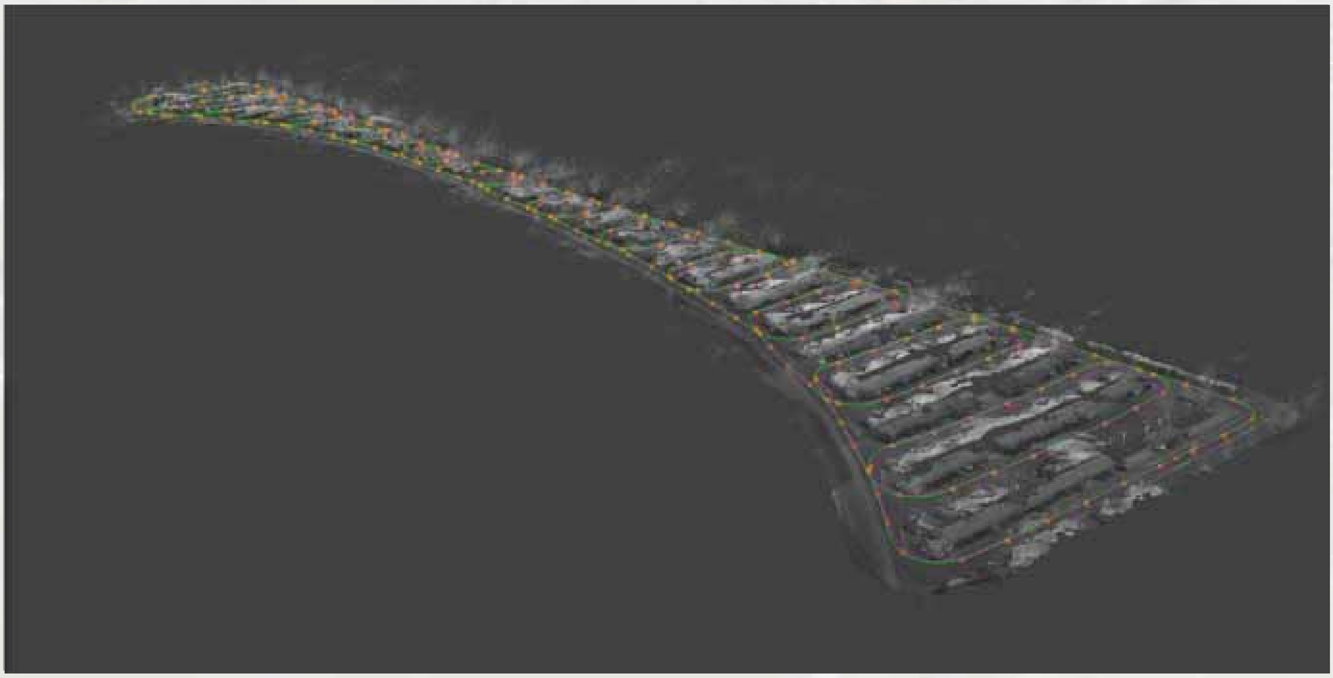
Furgale et al., ICRA2014

APPROACH: KNOWING WHERE THE ROAD IS - LOCALIZATION



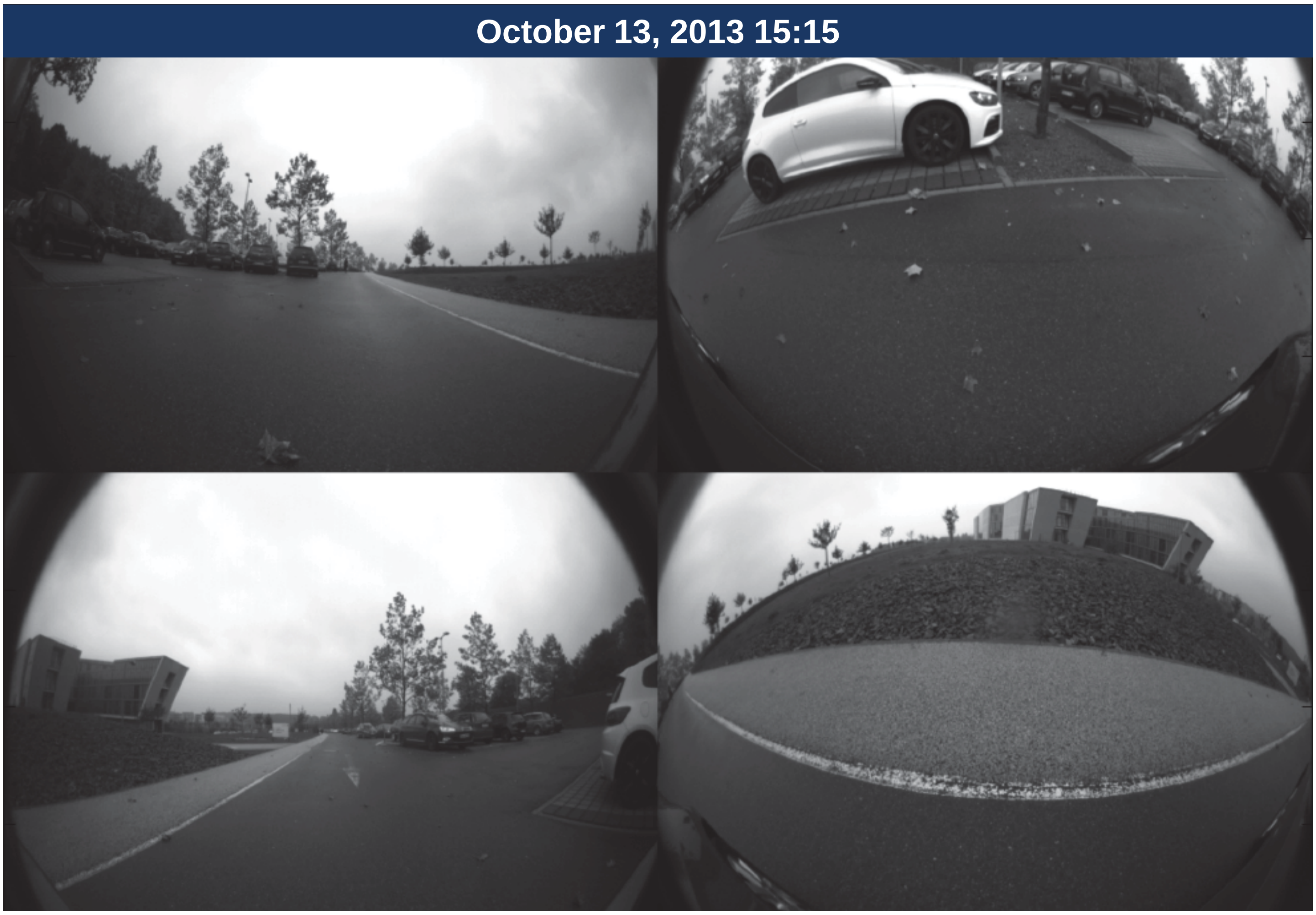
Furgale et al., ICRA2014

APPROACH: KNOWING WHERE THE ROAD IS - LOCALIZATION

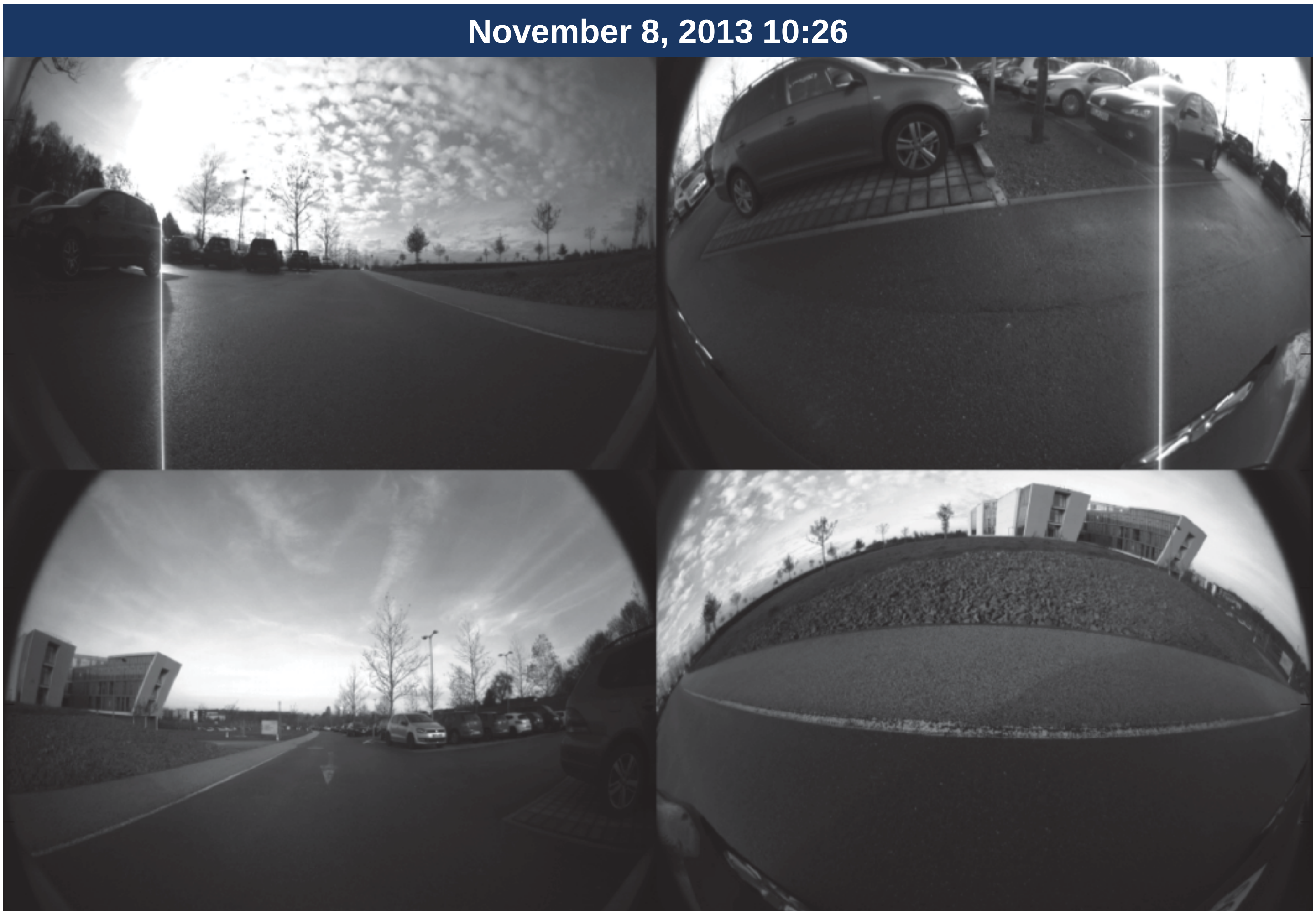


Furgale et al., ICRA2014

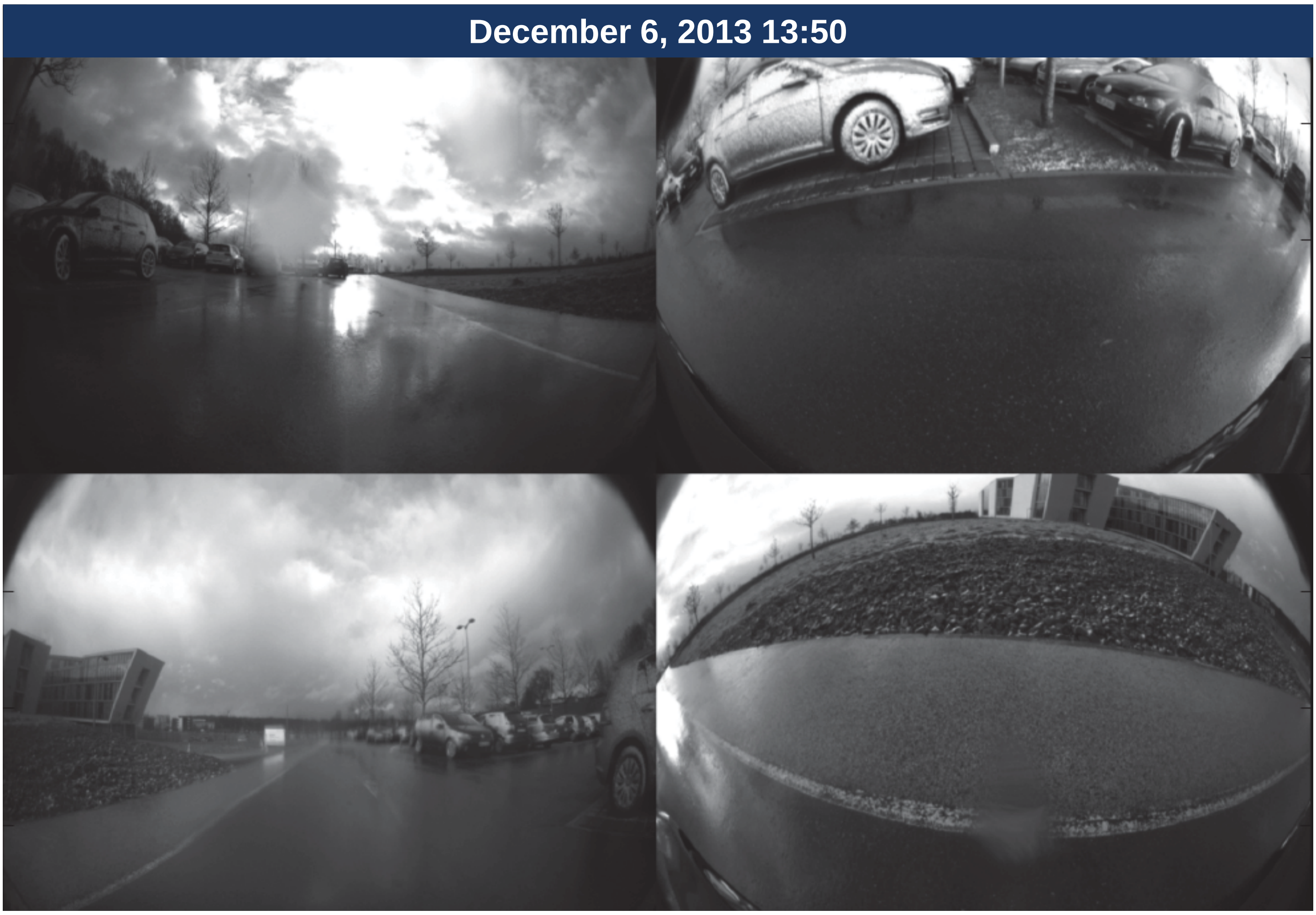
October 13, 2013 15:15



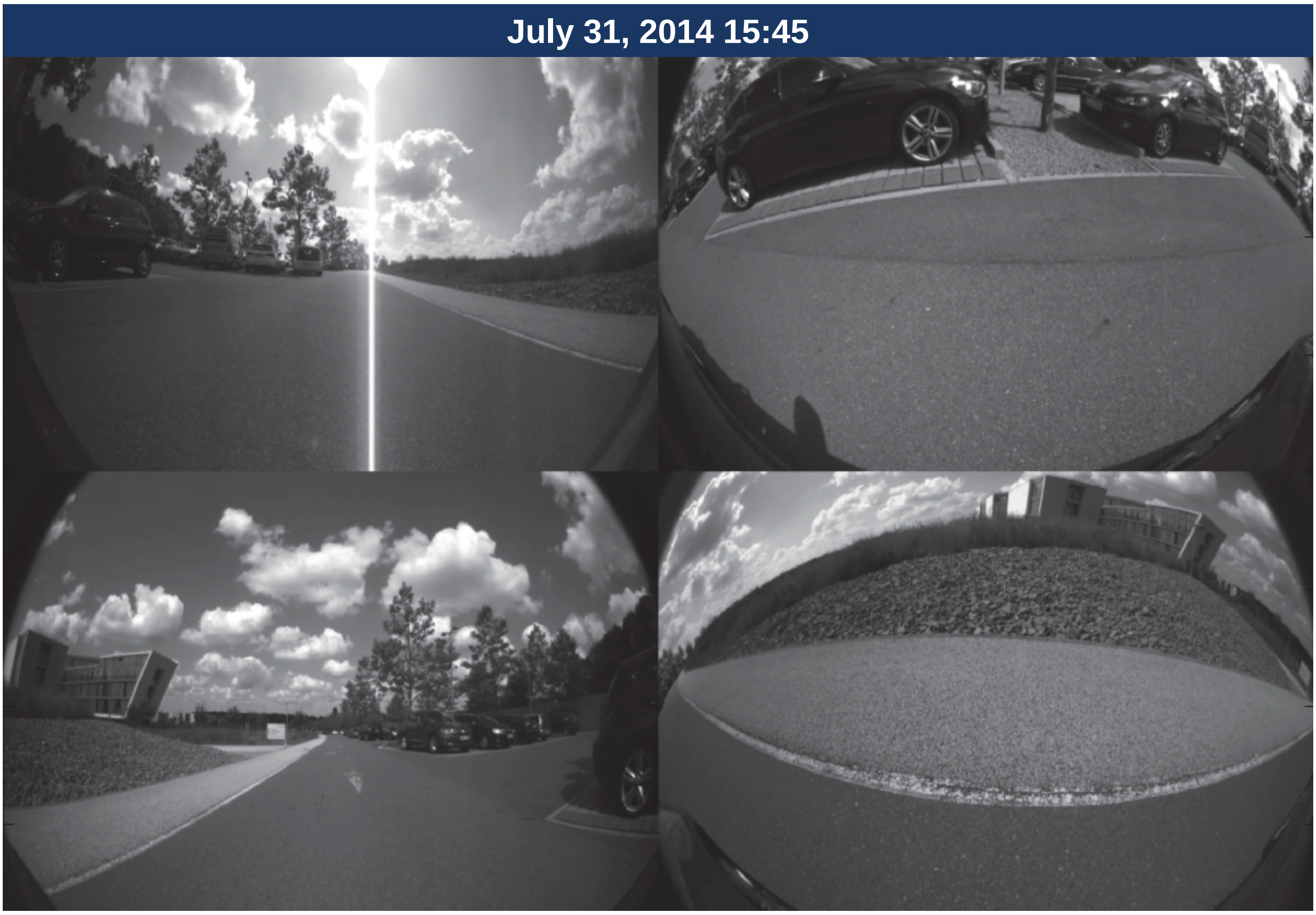
November 8, 2013 10:26



December 6, 2013 13:50

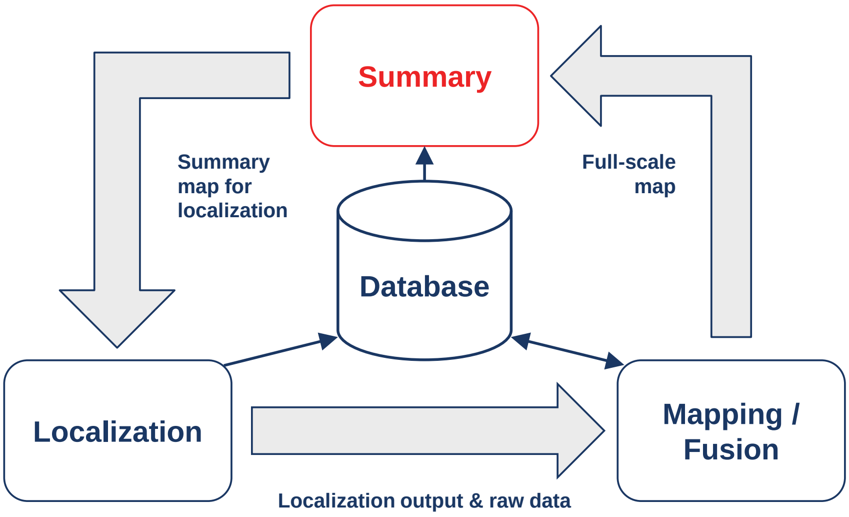


July 31, 2014 15:45



VOLKSWAGEN

APPROACH: KNOWING WHERE THE ROAD IS - LOCALIZATION



Mühlfellner, Peter : Lifelong Visual Localization for Automated Vehicles, Doctoral thesis

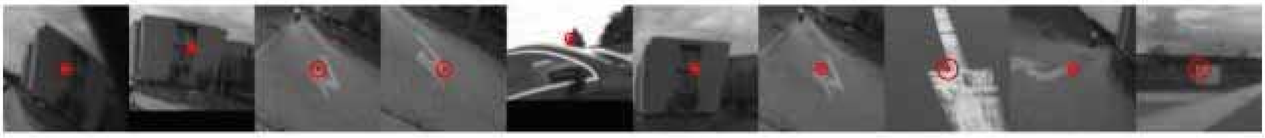
VOLKSWAGEN

APPROACH: KNOWING WHERE THE ROAD IS - LOCALIZATION

Find the most useful Landmarks:

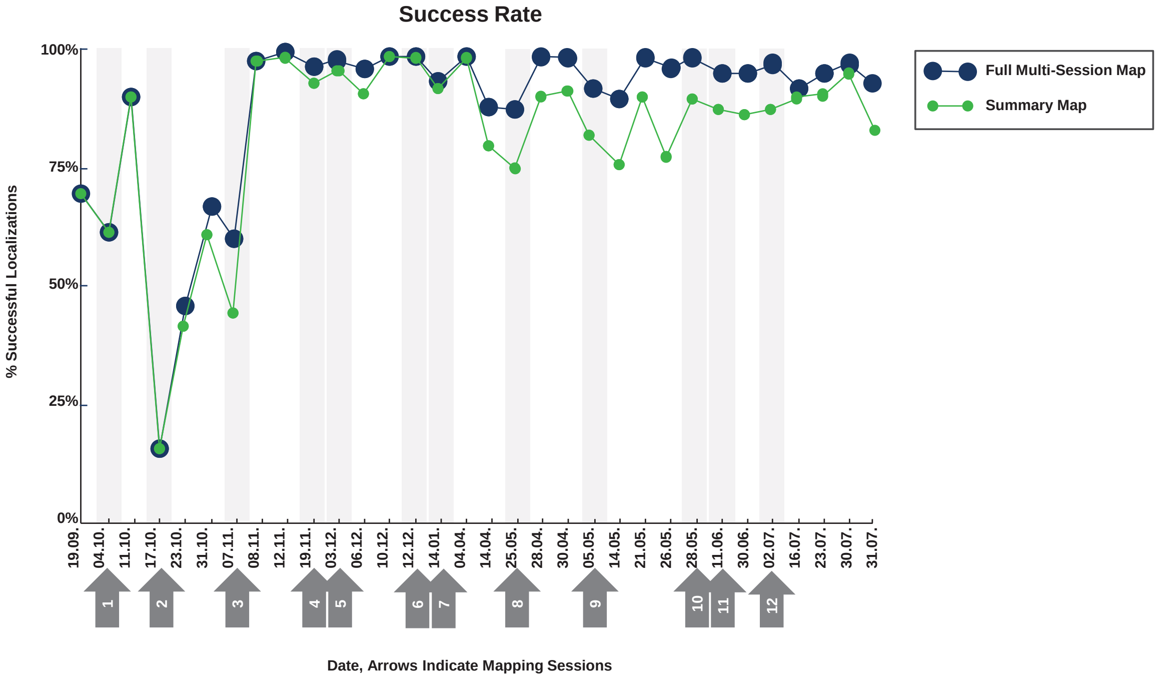
Most observed landmarks from a single session:

Landmarks observed over the most different sessions:



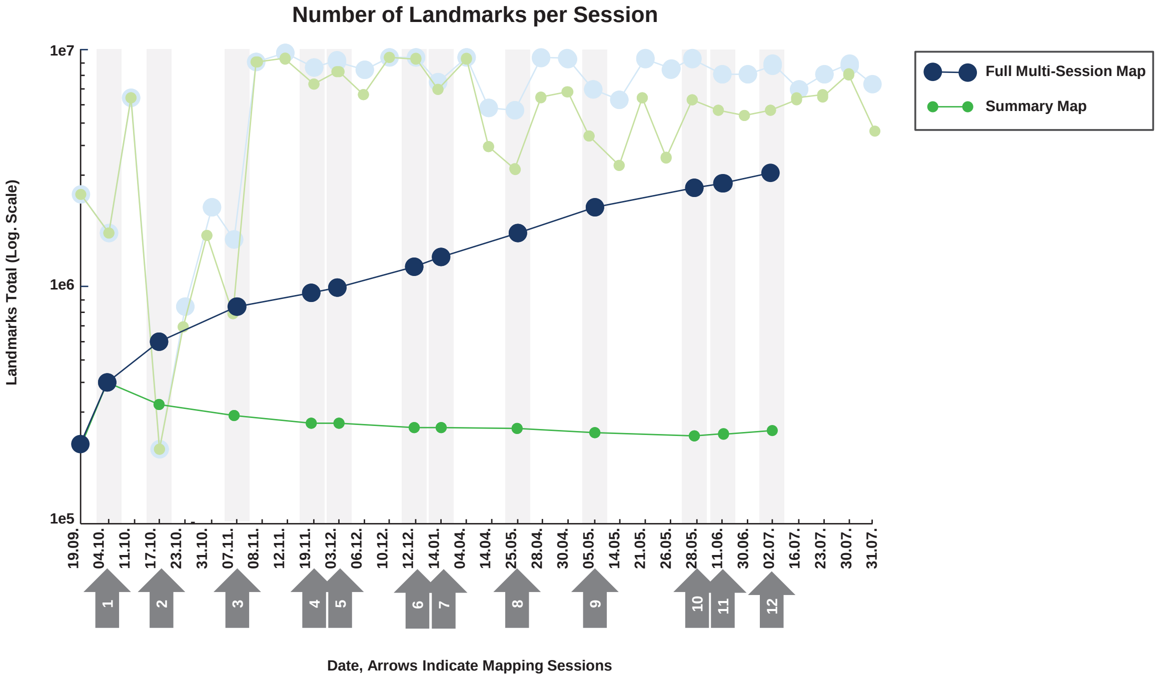
Mühlfellner, Peter : Lifelong Visual Localization for Automated Vehicles, Doctoral thesis

APPROACH: INTERPRETING WHERE THE ROAD IS – ROAD ESTIMATION



Mühlfellner, Peter : Lifelong Visual Localization for Automated Vehicles, Doctoral thesis
Mühlfellner et al.: Summary Maps for Lifelong Visual Localization, JFR, 2015

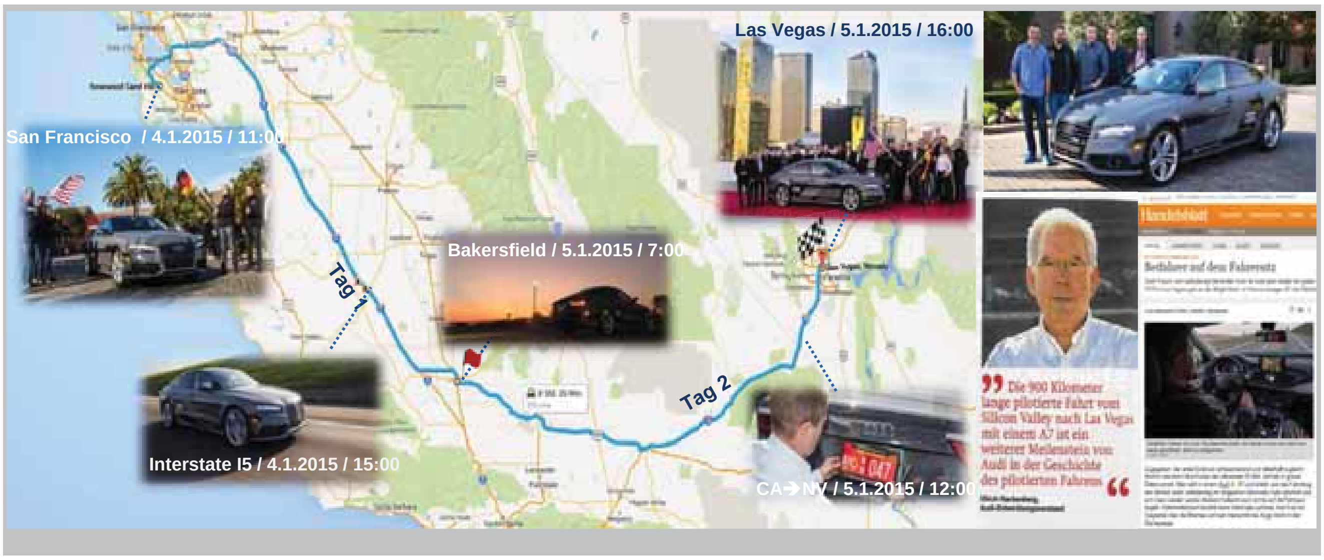
APPROACH: INTERPRETING WHERE THE ROAD IS – ROAD ESTIMATION



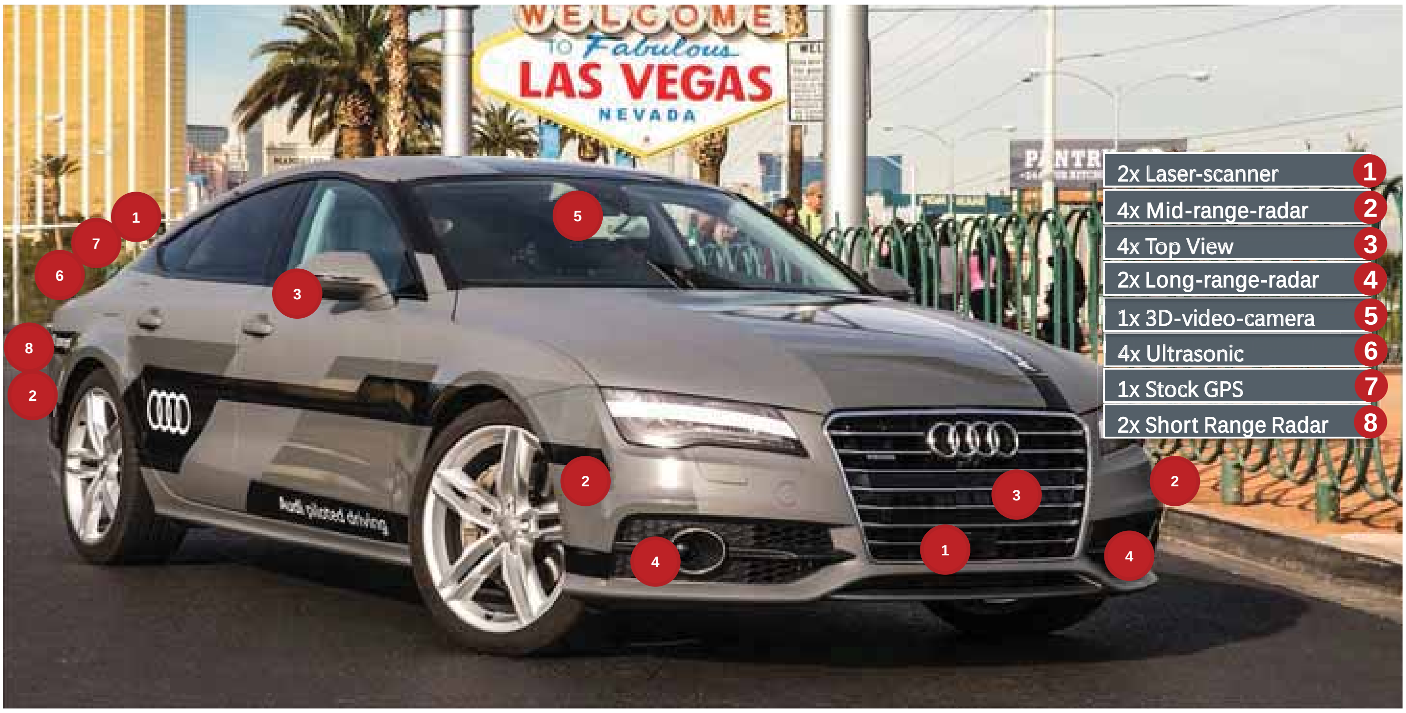
Mühlfellner, Peter : Lifelong Visual Localization for Automated Vehicles, Doctoral thesis
Mühlfellner et al.: Summary Maps for Lifelong Visual Localization, JFR, 2015

CES DEMO DRIVE 2015

- About 900 km on Highways
- 5 Journalists as drivers using the automated vehicle


Las Vegas / 5.1.2015 / 16:00
San Francisco / 4.1.2015 / 11:00
Tag 1
Bakersfield / 5.1.2015 / 7:00
Tag 2
Interstate I5 / 4.1.2015 / 15:00
CA→NV / 5.1.2015 / 12:00
Handelsblatt
Beifahrer auf dem Fahrersitz
„Die 900 Kilometer lange pilotierte Fahrt vom Silicon Valley nach Las Vegas mit einem A7 ist ein weiterer Meilenstein von Audi in der Geschichte des pilotierten Fahrens"

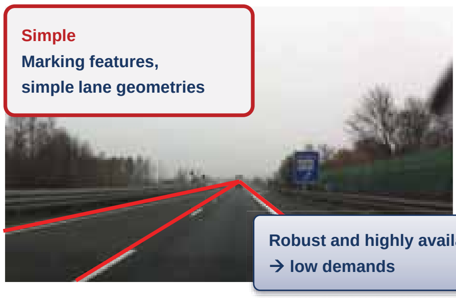
TECHNOLOGY – TODAY



- 2x Laser-scanner 1
- 4x Mid-range-radar 2
- 4x Top View 3
- 2x Long-range-radar 4
- 1x 3D-video-camera 5
- 4x Ultrasonic 6
- 1x Stock GPS 7
- 2x Short Range Radar 8

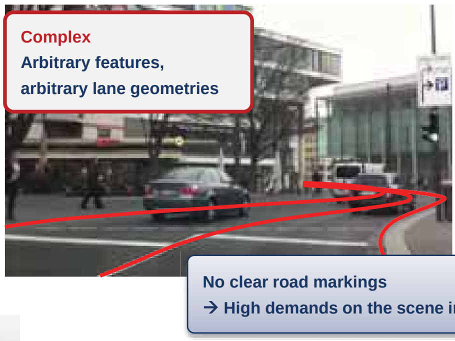
VOLKSWAGEN

KEY CHALLENGE: WHERE IS THE ROAD?



Simple
Marking features,
simple lane geometries

Robust and highly available marking features
→ low demands



Complex
Arbitrary features,
arbitrary lane geometries

No clear road markings
→ High demands on the scene interpretation

Medium
Marking features, more complex geometries
(smaller radius of curvature,
not parallel geometries)

Marking features with more complex geometries
→ medium-level demands

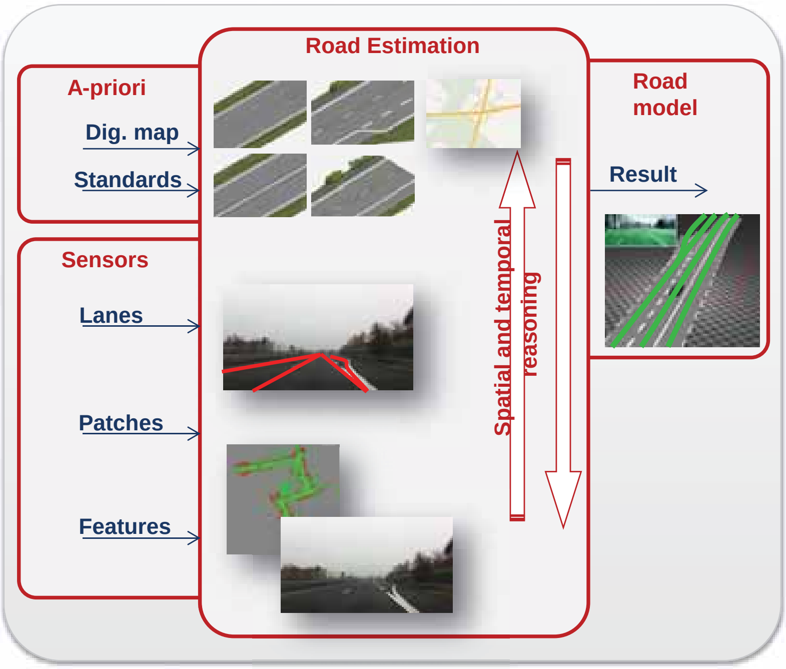
Töpfer, Spehr et al: Efficient Scene Understanding for Intelligent Vehicles Using a Part-Based Road Representation

VOLKSWAGEN

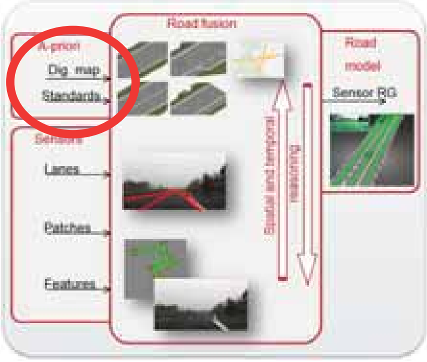
Credits Spehr et al

INPUT AND OUTPUT OF THE ROAD ESTIMATION

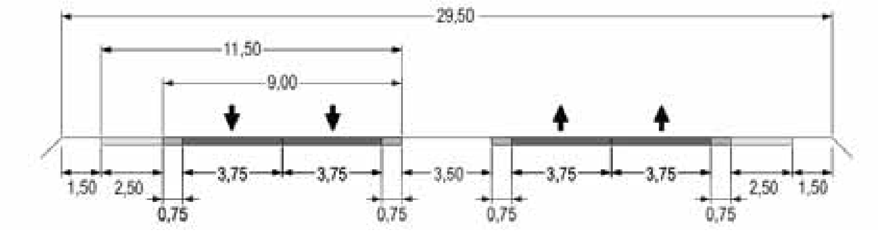
- Spatial and temporal reasoning



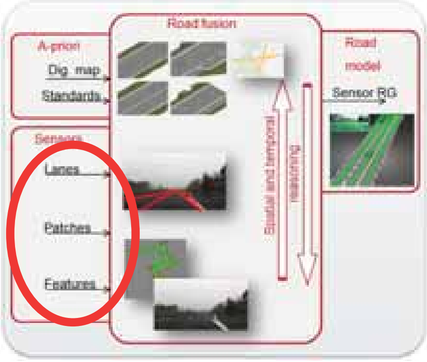
INTEGRATION OF A-PRIORI KNOWLEDGE



- Standards:
  - Guidelines for building roads, freeways and street
  - Use of country-dependent policies (e.g. in Germany „Straßenbaunormen DIN EN 1423 und DIN EN 1424“)
  - Deviations are modeled with appropriate probability distributions
- Digital maps:
  - Information beyond the detection range of the sensors
  - Used for integrating the expected road geometry and topology during spatial reasoning.

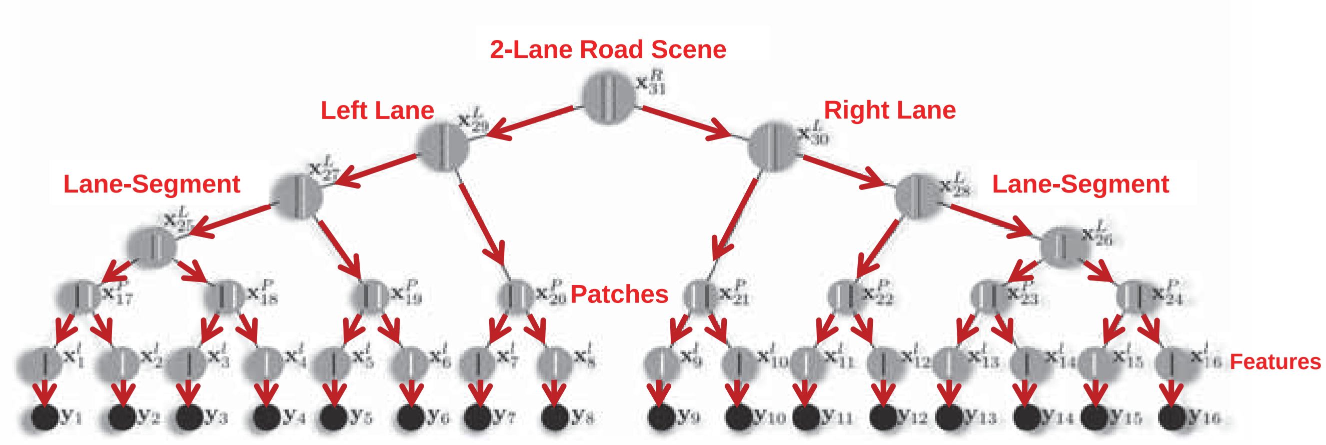


SENSOR SETUPS



- High-level sensor information (Lanes)
  - Covers a wide range in the vehicle’s environment
  - Trajectories of other vehicles
  - Lane output of external preprocessing units
- Low-level sensor information (Patches/ Features)
  - Spatially restricted features in the vehicle’s environment such as boundary features
  - But also features like grid cells (occupied yes/no), color values of a camera image

HIERARCHICAL MODEL OF A SINGLE SCENE



The graphical model comprises hidden random variables $\boldsymbol{x}_i$ and observations $\boldsymbol{y}_i$

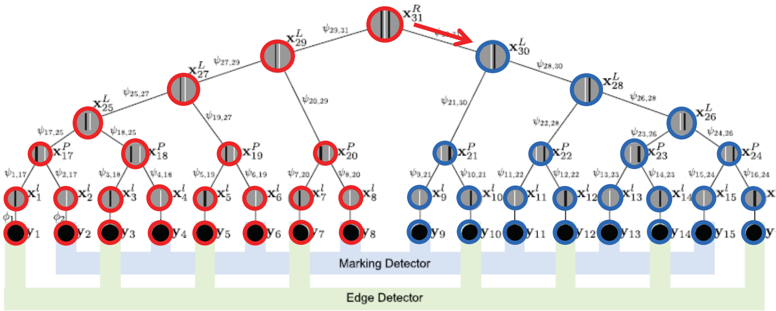
- Hidden variables represent parts and sub-parts of a scene encoded by the root node
- Variables are **continuous**, **multi-dimensional, and multi-modal**

Edges encode probabilistic dependencies between pairs of variables

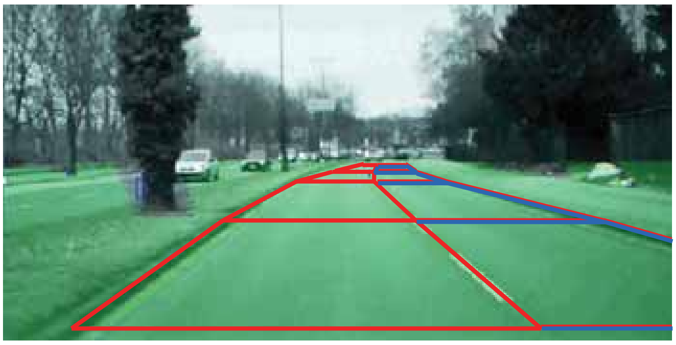
INFERENCE
DEPTH-FIRST MESSAGE PASSING

bottom-up = generating a hypothesis

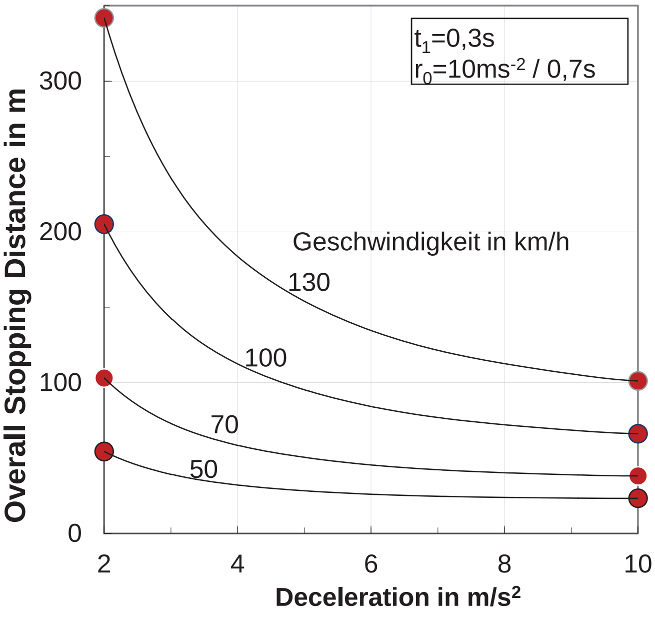
top-down = verifying the hypothesis



Example:
11. Verifying the road hypothesis (top-down)

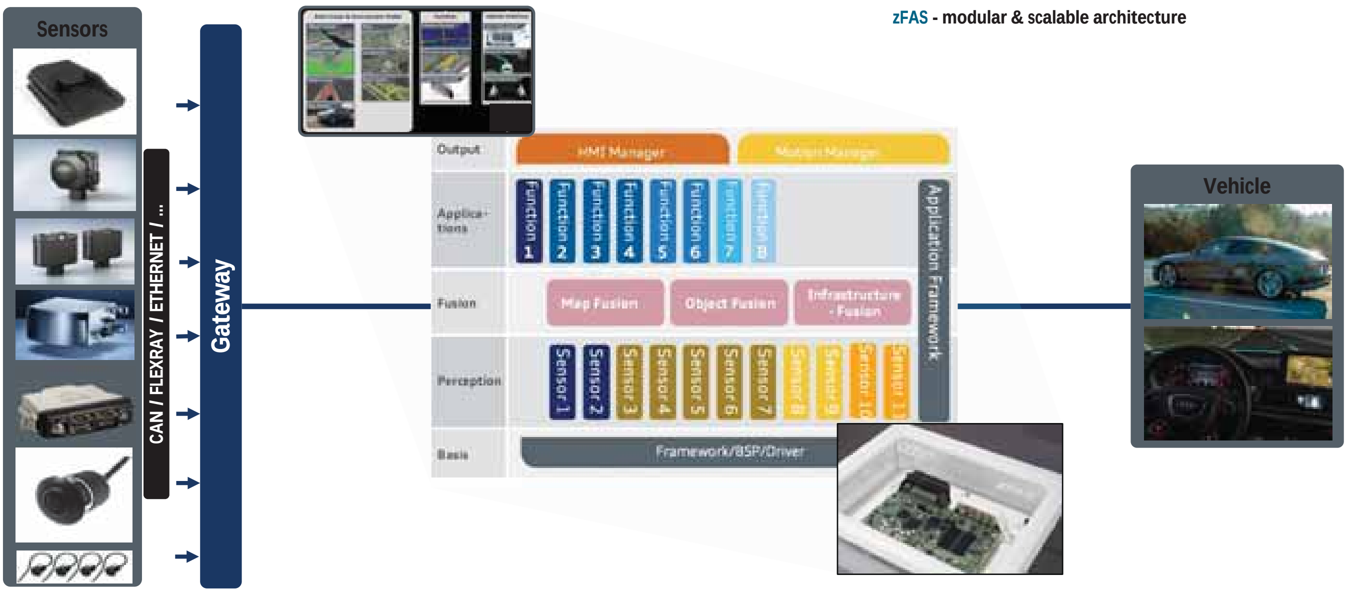


IS THE ROAD DRIVEABLE – DISTANCE TO STOP



		Velocity			
		50 km/h	70 km/h	100 km/h	130 km/h
Deceleration	2 m/s^2	54 m	103 m	205 m	342 m
	10 m/s^2	23 m	38 m	66 m	101 m

APPLICATION FOR SERIES PRODUCTS


zFAS - modular & scalable architecture

VOLKSWAGEN

Ensuring reliability – challenges

Today:

- Release testing of todays ADAS with up to 2 million test km and 1.000 test drivers

System	Activation frequency per 10.000 km	km till activation
Distance Warning	40 - 60	170 – 250
Breaking Assistance (BAS) Plus	0,5 - 1	10.000 – 20.000
PRE-SAFE breaking, level 1	0,1 – 0,2	50.000 – 100.000
PRE-SAFE breaking, level 2	0	-

data basis: 2.000.000 km, > 1.000 drivers; source: Dr. Markus Fach et al., Daimler AG, VDI/VW Gemeinschafts-Tagung, 2010)

Tomorrow:

- Increasing system complexity of future ADAS will increase diversity of relevant test scenarios
- Forecast for high automation: 100 Million km = 0,67 x average distance sun earth = 5,6 light minutes
- Costs for such release testing: several 100 Million EUR

source: Prof. Winner et al., Darmstädter Kolloquium „Mensch und Fahrzeug", 2011

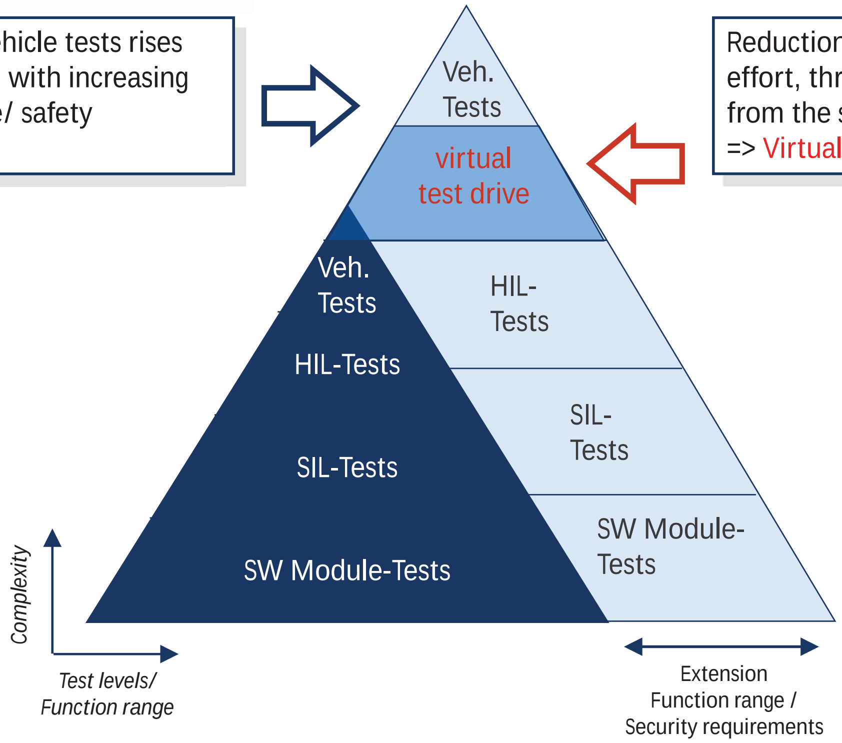
Objective:

- Sustainable and affordable concept for test and release of future ADAS

VOLKSWAGEN

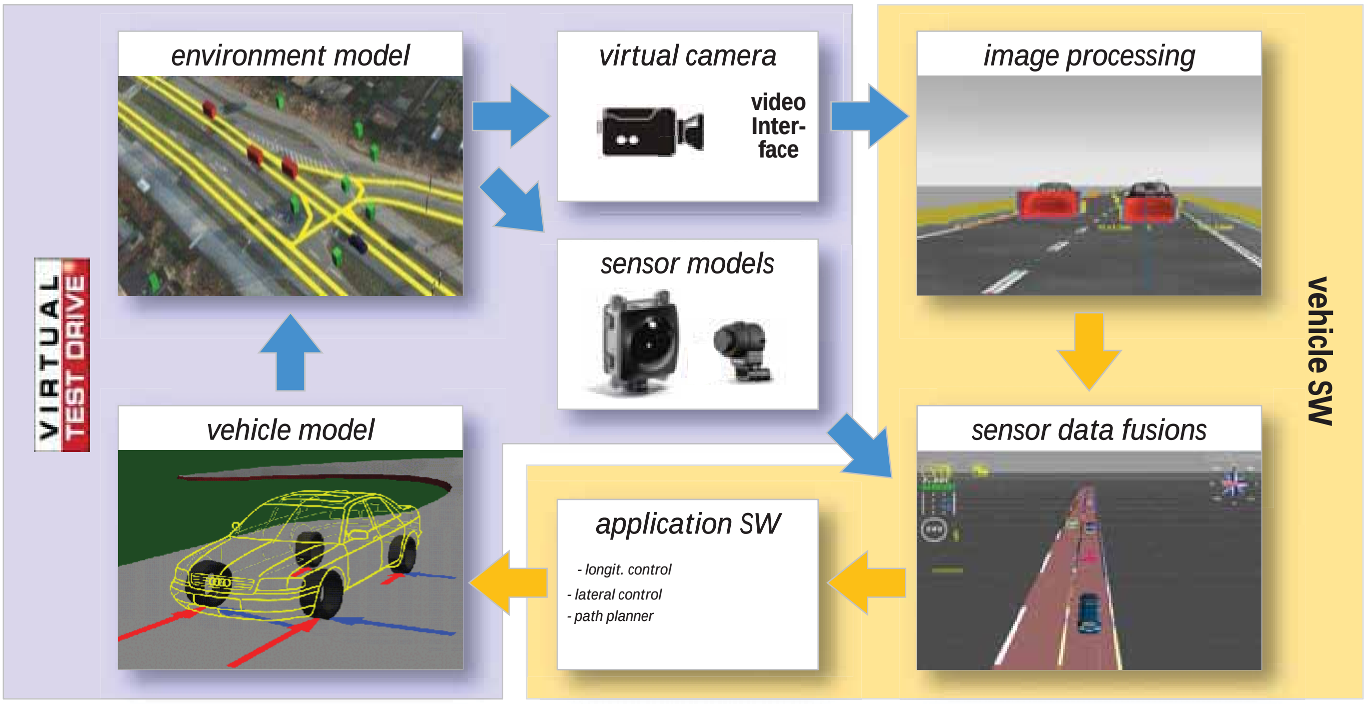
Test levels and complexity

The effort of vehicle tests rises disproportional with increasing functions range/ safety requirements



Reduction of the vehicle tests effort, through shifting the test from the street to the simulation. => Virtual Test Drive

Simulation via Virtual Test Drive



CONCLUSION

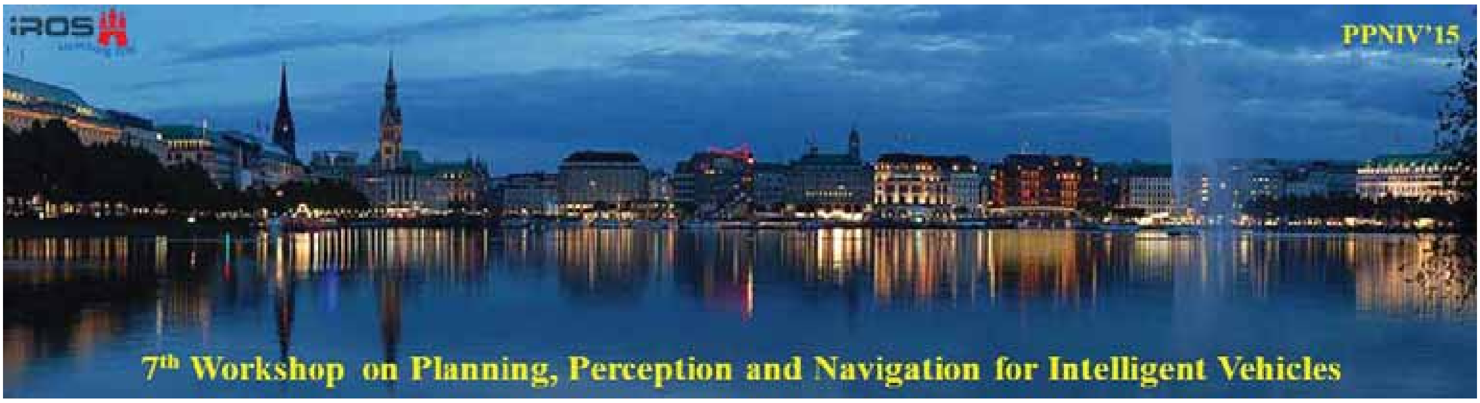
- Knowledge about the location of the road is a key factor for automated driving and future driver assistance systems
- Interpretation based approaches using environment sensors work well in easy to medium challenging scenarios
- Using additional map information leads to more robust results
- Localization techniques are currently used to solve the most complex scenarios

- Drivability estimation at long ranges for high speed driving is still challenging
- New sensor principles and machine learning approaches are one way for solving this topic

- Testing environment perception is one of the key challenges
- Shifting tests from street to simulation reduces vehicle test efforts significantly
- Centralized ECUs like the zFAS help facilitating testing procedures

VOLKSWAGEN
AKTIENGESELLSCHAFT

THANK YOU FOR YOUR ATTENTION


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

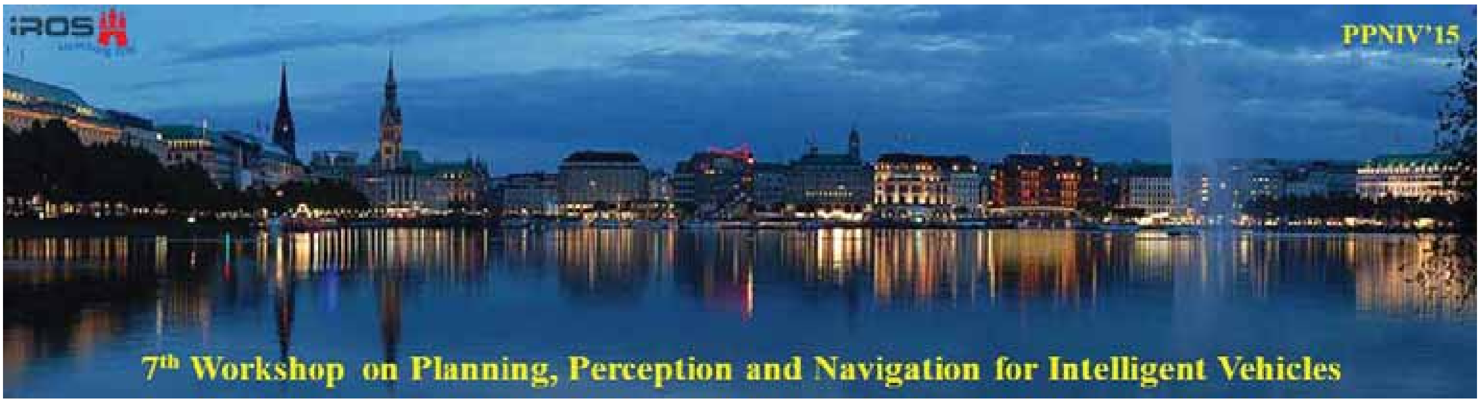
Session II

Perception & Situation Awareness

- **Title: Free-space Detection using Online Disparity-supervised Color Modeling**
  Authors: Willem P. Sanberg, Gijs Dubbelman and Peter H.N. de With

- **Title: Vision-Based Road Detection using Contextual Blocks**
  Authors: Caio Cesar Teodoro Mendes, Vincent Frémont and Denis Fernando Wolf

- **Title: Following Dirt Roads at Night-Time: Sensors and Features for Lane Recognition and Tracking**
  Authors: Sebastian F. X. Bayerl, Thorsten Luettel and Hans-Joachim Wuensche


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Free-space Detection using Online Disparity-supervised Color Modeling

Willem P. Sanberg, Gijs Dubbelman and Peter H.N. de With

***Abstract*— This work contributes to vision processing for intelligent vehicle applications with an emphasis on Advanced Driver Assistance Systems (ADAS). A key issue for ADAS is the robust and efficient detection of free drivable space in front of the vehicle. To this end, we propose a stixel-based probabilistic color-segmentation algorithm to distinguish the ground surface from obstacles in traffic scenes. Our system learns color appearance models for free-space and obstacle classes in an online and self-supervised fashion. To this end, it applies a disparity-based segmentation, which can run in the background of the critical system path and at a lower frame rate than the color-based algorithm. This strategy enables an algorithm without a real-time disparity estimate. As a consequence, the current road scene can be analyzed without the extra latency of disparity estimation. This feature translates into a reduced response time from data acquisition to data analysis, which is a critical property for high-speed ADAS. Our evaluation over different color modeling strategies on publicly available data shows that the color-based analysis can achieve similar (77.6% vs. 77.3% correct) or even better results (4.3% less missed obstacle-area) in difficult imaging conditions, compared to a state-of-the-art disparity-only method.**

I. INTRODUCTION

In recent years, vehicles are becoming increasingly intelligent with so-called Advanced Driver Assistance Systems (ADAS). This development is expected to significantly reduce traffic accidents, traffic congestion and fuel consumption simultaneously. To ensure traffic safety, ADAS can e.g. indicate the location of potentially hazardous obstacles to the driver and the area position of safely drivable road. On the longer term, ADAS and related technologies will allow the development of fully autonomous vehicles. In this work, we improve a state-of-the-art vision-based free-space detection system by efficiently exploiting multiple image modalities.

To robustly facilitate situational awareness at a moving platform, several complementary sensor modalities should be employed. These modalities can include RADAR, LIDAR, ultrasound, and (thermal) imaging. The benefit of using vision-based systems is that they provide dense scene information in a cost-effective way. Image data is also a rich source of information, since it comprises of several informative properties. For stereo-based video imaging, these informative aspects include not only the usual *texture, color* and *shape* features, but also *optical flow* motion analysis and *disparity* estimation. All these elements can contribute to a robust situational analysis, such as e.g. the detection of partially occluded pedestrians who are about to cross the street. Although LIDAR, RADAR or ultrasound provide valuable complementary information, in this paper we solely focus on vision-based detection systems.

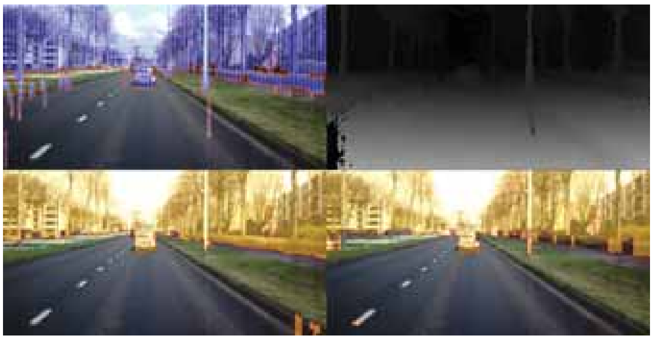


Fig. 1. Stixel segmentation results comparing a disparity-only result (top left), a RGB-only result using 10 frames for online learning (bottom left), and a new gray-only result for which just 3 frames are used for the online learning (bottom right). The orange overlay with a dark border depicts the space occupied by obstacles according to the detection algorithm. At the top right, the disparity signal is shown, which has several artifacts, due to low texture in the road region (left) and a pole reflection in the windshield (middle). These artifacts cause false obstacle detections in the original disparity-based algorithm [1]. Here, we show that it is possible to obtain similar or even better segmentation results with our color modeling, while requiring less data in the process.

Multi-view image processing, in particular stereo vision, has the potential to provide 3-D scene information at a more affordable price point than that of high-end laser-based systems, which are often accompanied by RTK-GPS, e.g. [2][3]. In stereo vision, the disparity, which is analogous to depth, can be estimated densely and in real-time [4]. This gives a direct description of the geometry of the scene and it facilitates, for example, a separation of flat, drivable surfaces from erect obstacles [5][6]. The Stixel World method [1] is a state-of-the-art approach to analyze such a geometry description of the scene. It is a fully probabilistic framework to distinguish free space from obstacles in the disparity signal, which can be implemented efficiently provided that several assumptions are made. This framework is generally more flexible and more robust than its predecessors.

A pitfall of the original Stixel World framework is that it requires a disparity signal of a certain quality. However, the quality of disparity estimation often degrades in cases of occlusions, reflections or image regions with too little texture information. Unfortunately, such degradations are common in traffic scenery. As a result of this degraded signal, the original Stixel World framework detects many false obstacles, rendering the results useless for a practical system under adverse conditions. An example of this is shown at the top-left image of Fig. 1. In our recent work [7], we show that

Willem Sanberg, Gijs Dubbelman and Peter de With are with the Department of Electrical Engineering, Video Coding and Architectures Research Group, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands w.p.sanberg@tue.nl

the performance of such a disparity-based framework can be improved by fusing color into the algorithm. This strategy resolves many erroneous results of the disparity analysis at a low additional computational cost, in contrast to alternative solutions such as high-quality cameras or more advanced disparity estimation techniques.

A key property of any ADAS is the response time, i.e. the time delay between data acquisition and the response to the result of the analysis. Since ADAS preferably need to function at high vehicle speeds, the response time of such systems should be as fast as possible. Hence, any delay that can be removed from the critical path of the analysis is beneficial to the value and applicability of the system, provided that it does not degrade the reliability of the results. Therefore, we will explore the possibility of removing the disparity analysis from the critical system path. Although fast disparity estimation methods exist [4][8], they typically either rely on sub-optimal algorithms processing at a low resolution, or they are based on customized hardware that is not commonly available. To illustrate this, even in the state-of-the-art system presented in [1], the dedicated FPGA disparity estimation takes 40 ms per frame, whereas the stixel analysis of the data takes 30 ms, when executed on a general, high-quality multi-core CPU.

For these reasons, we will not rely on a *strong fusion* of disparity and color in this work, even though the result presented in [7] clearly shows the qualitative benefits of that. In contrast, we propose here to process the most recent camera frame using an efficient color-only stixel segmentation. The disparity estimation and analysis, which is only required for our online color modeling, can be processed in parallel and at a lower frame rate. Two examples of our novel color-only stixel segmentation are shown at the bottom-left and bottom-right image of Fig. 1, illustrating that we can achieve better results than the state-of-the-art disparity approaches, even with color modeling of a low complexity.

An alternative to online color modeling is offline color modeling [9], which would completely remove the need for online disparity estimation. However, we have a strong preference for an online learning approach, given the challenging nature of traffic environments, which is full of varying weather conditions, complex scenery, varying geographical settings and highly dependent on the time of the day. For instance, in low-light situations, urban traffic scenes tend to contain predominantly gray-tones. We consider it more feasible to build a robust, yet discriminating color model that is tuned to that specific time and place, rather than building a generic model that holds for every environment and weather condition.

The remainder of this paper is structured as follows. First, we will describe the probabilistic Stixel World framework in Section II and explain briefly how it can be used with disparity, color or both data signals. Then, in Section III, we present the aspects of the system that will be evaluated in this paper. We then describe our validation method and the corresponding results in Sections IV and V, respectively. Lastly, conclusions are provided in Section VI.

II. THE STIXEL WORLD

Let us now give a short overview of the general Stixel World framework from [1], which we have used as a basis of our work. The main goal of stixel segmentation is to find the optimal labeling L^* of vertically stacked, piecewise planar ground or obstacle segments for input data $\mathbb{D}$, which can be any signal modality. Finding L^* can be formulated as a MAP estimation problem, as in

$$L^* = \arg\max_{L \in \mathbb{L}} P(L|\mathbb{D}), \tag{1}$$

which can be solved efficiently using Dynamic Programming. Using Bayes' theorem and assuming, among others, independence between columns and between data measurements at individual pixels, the posterior probability can be written as a chain of conditional probabilities by

$$P(L|\mathbb{D}) \sim \prod_{u=0}^{w-1} P(D_u|L_u) \cdot P(L_u). \tag{2}$$

Here, u is the column index and w the image width. The probability $P(L_u)$ models *a-priori* world knowledge constraining the labeling, to avoid dispensable segments and physically unlikely situations. This world model offers a way to regularize the results for image-column optimality, whereas the methods of [5] and [6] potentially lead to sub-optimal results, since they mostly analyze data locally. The details concerning $P(L)$ are presented in [1]. Finally, the likelihood of the data given a certain labeling, can be written as

$$P(D_u|L_u) \sim \prod_{n=1}^{N_u} \prod_{v=v_n^b}^{v_n^t} P(d_v|s_n, v), \tag{3}$$

where n is the segment index, N_u the number of segments in L_u, and v_n^b and v_n^t the bottom and top row-index of segment s_n, respectively. This segment has a label $l_n \in \{g, o\}$, representing the ground and obstacle classes, respectively.

The distribution $P(d_v|s_n, v)$ in Eq. (3) represents the probability of a single valid data measurement d_v at a certain row v, assuming that it would belong to a potential segment s_n. The model for $P(d_v|s_n, v)$ should reflect the nature of the employed signal modalities. There are several relevant approaches in literature. The authors of [1] employed a dense stereo-disparity signal as the sole data modality. They proposed to model $P(d_v|s_n, v)$ as a mixture model, containing a uniform distribution that models outliers and a Gaussian distribution that models inliers, to assess how well the measurement fits the potential segment for each class. For ground segments, the expected disparity is a linear planar surface and for obstacle segments a fronto-parallel surface.

A different approach is presented in [7], where the Stixel framework is extended, such that it exploits both the disparity signal and the color data. This strategy increases the robustness of the system against adverse conditions such as low light, bad weather, or a low-quality sensing system. To this end, the authors redefine the likelihood term of Eq. (3) to be $P(D_u, C_u|L_u)$ with the additional term $P(c_v|l_n)$ in the right-hand side of the equation, thereby treating color

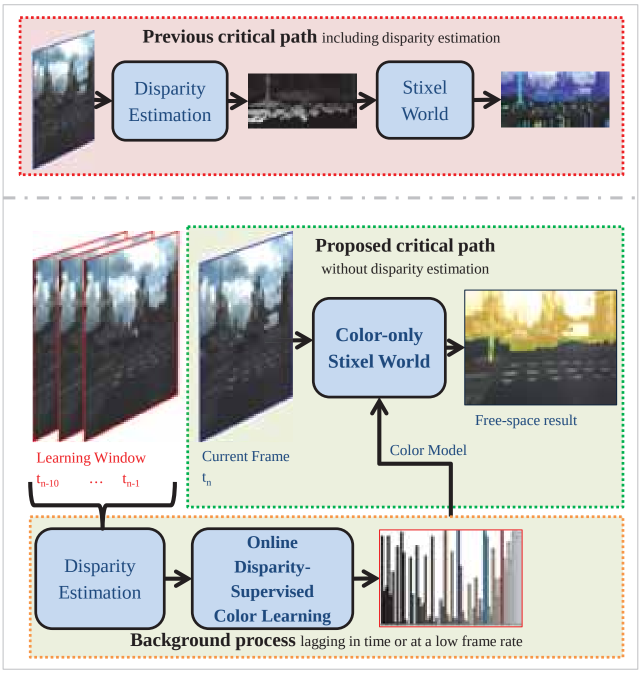

Previous critical path including disparity estimation
Disparity Estimation
Stixel World
Proposed critical path without disparity estimation
Color-only Stixel World
Free-space result
Color Model
Learning Window
t_{n-10} … t_{n-1}
Current Frame
t_n
Disparity Estimation
Online Disparity-Supervised Color Learning
Background process lagging in time or at a low frame rate

Fig. 2. A comparison between the existing and our proposed framework. Previous systems [1][7] require disparity estimation in their critical path, in contrast to our proposed system. Note that our disparity-supervised color modeling in the lower part of the scheme can be lagging or run at a lower frame rate than the critical path, by varying the range of the learning window.

and disparity as independent signals. Additionally, note that this color posterior is assumed to be independent of the segment location, since it only considers the class label l_n in contrast with disparity modeling. The color posterior $P(c_v|l_n)$ is learned in an online fashion using the labeling of several preceding frames as training masks for both the ground and obstacle classes. With these masks, normalized color histograms are calculated, which are then transformed to posteriors using Bayes' rule.

In our subsequent research, we explore the feasibility of segmenting the traffic scene images using a color-only version of the Stixel World algorithm [10]. The benefit of this approach is that the disparity estimation can be removed from the critical system path, as illustrated in Fig. 2. In [10], we present color modeling that is more suited for stand-alone processing compared to [7], by making it distance-aware. To this end, we have specified the color-based likelihood as

$$P(C_u|L_u) \sim \prod_{n=1}^{N_u} \prod_{v=v_n^b}^{v_n^t} P(c_v|s_n, v). \tag{4}$$

The distance-aware color processing consists of two aspects: (a) weighing each pixel with its corresponding real-world surface during the process of calculating the color histograms, and (b) leveraging the regular and the distance-weighted color posteriors based on v while evaluating Eq. (4). This approach leads to a more balanced color analysis of far-away and close-by image regions to cope with the inherent geometric distortion of cameras in a robust way [10].

III. Online Color Modeling Experiments

The key contribution here is to perform an elaborate analysis on the critical design choices of the online, distance-aware, self-supervised learning framework, as presented in [10]. The framework processes preceding stereo frames and generates a free-space vs. obstacle labeling based on disparity. Consecutively, this labeling is exploited as self-supervised training masks for the color representation of these two classes. The relevant design choices concern the color representation, consisting of preprocessing and color space selection, and the selection of the frames in the learning window.

A. *Color Representation*

Several aspects of the color representation are kept constant throughout this paper. First, we employ the median-cut algorithm on the frames in the learning window [11]. This ensures that we have an adaptive color representation that has both a sufficiently low complexity for fast processing and is still suitable for the current traffic scene, as the color reduction is performed online. Second, we perform a further reduction of the data by employing stixels that span 11 image columns. This increases the robustness and decreases the computational load at the cost of horizontal resolution in the labeling. To condense the image data into a single stixel-data vector, we calculate the first and the second mode of an 11×11 pixel window in the color data, horizontally centered at the central image column of the corresponding stixel. These aspects are recommended approaches as presented in [7] and [10]. Let us now briefly describe the evaluated color settings.

1) HEQ: We test the added value of performing Histogram Equalization on the raw RGB images (separately on each color plane) prior to converting it to a different color space.

2) RGB: We employ RGB as the full-color reference color space.

3) HS: To increase the robustness against varying lighting conditions, we test the strength of the Hue and Saturation dimensions of the HSV color space in our proposed framework.

4) IllumInv: The Illuminant Invariant color space presented in [12] is a more elaborate method for robust handling of changing lighting conditions and even shadows. It requires an automated offline camera-calibration method to find a parameter θ, which can then be used to transform each new image into an illuminant-invariant gray-scale image. We have adopted the proposed robust entropy-based calibration method and found that $\theta = 90 \pm 0.5^\circ$ for our camera, but refer explicitly to [12] for more details on this color space and calibration method.

5) Gray: We also execute our segmentation on a gray-scale representation as a baseline for extreme cases of monochrome lighting conditions. Moreover, it would significantly reduce the constraints on the camera hardware and the corresponding data bandwidth when the gray-scale analysis is successful.

B. Learning Window

As described earlier, our method exploits preceding frames (which are analyzed based on their disparity signal) as Learning Window (LW) to construct color models of the obstacle and ground classes. The settings of a LW are the oldest frame, counting backwards from the current one, the step size and the final frame to be considered, annotated 'LW start:step:end'. The recommended approach in [7] relies on the 10 most recent preceding frames (LW10:1:1), which we will use as a reference setting. Since the aim of this work is to reduce the computational complexity and, most importantly, the system latency, we experiment with two alternative frame selections. First, we test a learning window without the two most recent frames (LW10:1:3). This way, the color model lags by two frames but it allows a longer data investigation interval before it is required for the analysis. Second, we test the more extreme case that has a lower frame rate and a lag, by only considering frames $t-9$, $t-6$ and $t-3$ (LW9:3:3).

Analogous to [10], we generate class-posterior color distributions from the labeled pixels in the LW frames and apply distance-aware weighting to correct the geometric distortion in the imaging process.

IV. EXPERIMENTS

To evaluate the different design settings, we employ two publicly available stereo RGB data sets with 188 annotated test frames in total [7][10]. The data consists of a large variety of relevant traffic situations under both good and adverse imaging conditions, such as dark roads with cyclists and cars, road repair sites, highway scenes etc. Both images with bright weather and under dim, clouded or even rainy conditions are present, leading to many low-contrast regions that are especially difficult for disparity-based methods. All frames are captured with a BumbleBee2 stereo camera (baseline: 12 cm; resolution: 1024×768 pixels; frame rate: 20 Hz), which is a relatively basic, low-cost camera when compared to several high-end or custom models used in other set-ups [1][13]. The details of our employed SGBM disparity estimation [14] and several improvements that we made for the baseline system are provided in [7] and [10]. Note that we cannot execute our algorithm on benchmarks such as the KITTI dataset [15], since those, unfortunately, do not contain the required preceding frames of annotated road images.

V. RESULTS

We have tested all combinations of the selected color and learning window settings, resulting in 24 different executions (runs). The effect of the individual color and learning window settings is shown in Fig. 3, by means of a box plot. Using a paired t-test, applying *HEQ* provides a significant improvement over not using equalization ($p = 3.04 \times 10^{-8}$). Likewise, the *RGB* color space outperforms the *HS*, *IllumInv* and *Gray* representations ($p = 1.12 \times 10^{-18}$, $p = 1.93 \times 10^{-4}$ and $p = 1.85 \times 10^{-29}$, respectively), and a full learning window (LW10:1:1) is better than a shorter, lagging one (LW10:1:3) with $p = 3.25 \times 10^{-3}$. No significant difference was found between the results of using the full or low frame-rate learning window (LW9:3:3) ($p = 8.33 \times 10^{-1}$).

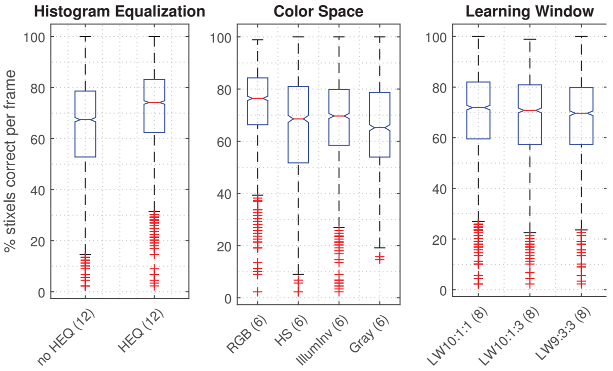
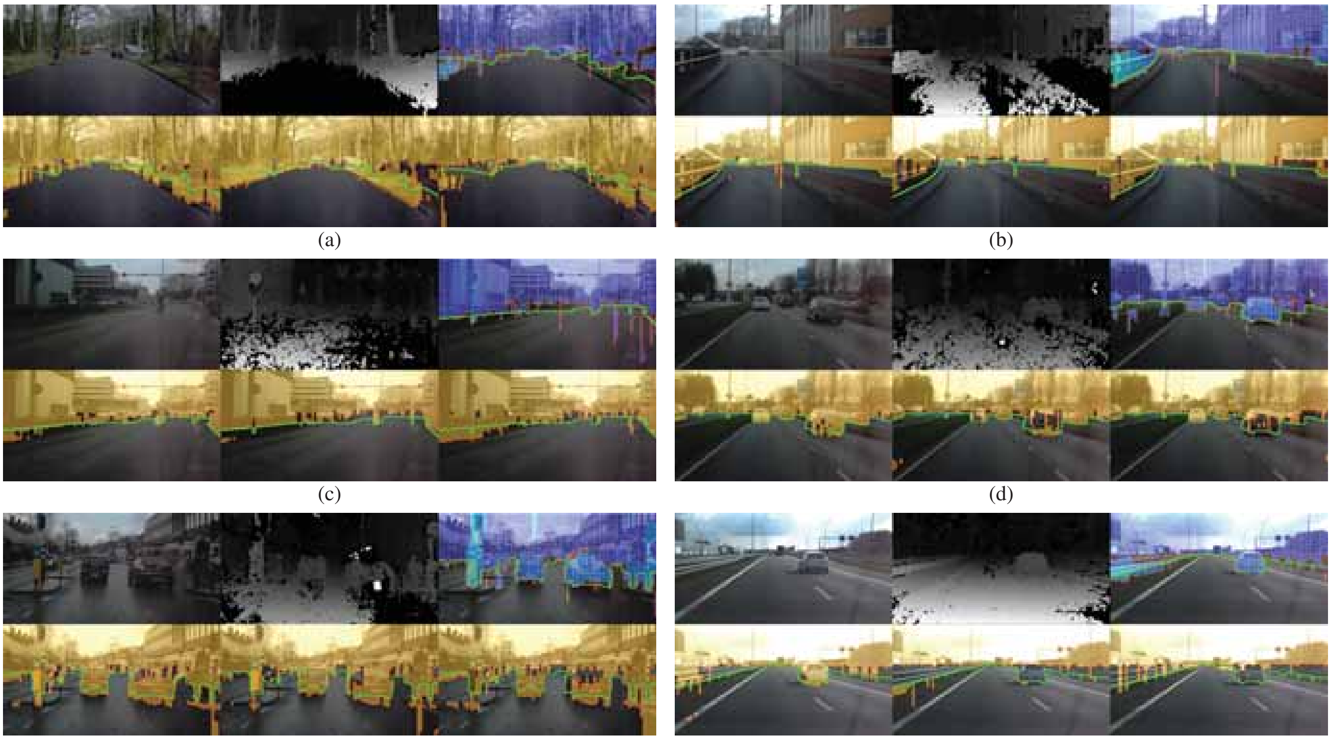


Fig. 3. Box plots comparing the different settings over all runs. For each frame in each run, the percentage of stixels with a correct free-space estimate is calculated, which is visualized as a box plot per setting. Hence, each box contains 188 data points per run. The number of runs per box is denoted in brackets in each label.

Additional quantitative results are provided in Fig. 5. In this figure, all stixels over all frames are evaluated together for each run individually. For each stixel, a free-space evaluation is performed, by comparing the detected free space by the true free space, generated from the ground-truth annotations. We calculate the deviation as a percentage of the true free space. For robustness reasons, free-space detections are counted as correct when they are within the range of 30% too short or 15% too long. This asymmetrical range reflects the fact that missing an obstacle is more dangerous than detecting one too close. For the same reason, we distinguish the incorrect stixels into obstacle misses (free space is too long) and false obstacle detections (free space too short). Although a deviation of 30% may seem a large fraction, it corresponds to only a few pixels after several meters and only some centimeters before that. The results are shown on the combined data as well as on the individual datasets. The rightmost graph in Fig. 5 clearly shows that the added value of color processing is more pronounced for the EHV-road-ITSC15 data. This can be explained by the fact that the EHV-road-ITSC14 contains both frames with bright and dim lighting conditions, whereas EHV-road-ITSC15 is solely focused on dark, clouded, low-light and rainy frames. These situations are specifically difficult for disparity-based methods, rendering color data more advantageous. Of all color settings, Run *f* results in the highest percentage of correctly detected free-space (77.64%, averaged over all data), which is similar to the disparity-only method (77.25%). For the EHV-road-ITSC15 data, the improvement is higher: 78.01% compared to 74.39%. When specifically focusing on reducing the number of missed obstacles in difficult imaging conditions, Run *h* reduces the percentage of erroneous stixels from 17.18% to 12.85%, compared to the disparity-only method. On the combined data, the stixel-error fraction reduces from 13.81% to 11.52%.

We provide additional analysis by means of five theoretical



(a) (b) (c) (d) (e) (f)

Fig. 4. Qualitative results of the disparity method and three of our runs on six stereo frames. Each subfigure contains two input and four result images. The top left and middle images show the rectified and cropped left camera image and the corresponding disparity image. Next to that, the disparity baseline result is shown, where the stixel segments are colored by their distance (red (close) to blue (far)). The three bottom result images illustrate different color settings, from left to right: RGB+HEQ with LW10:1:1, IllumInv+HEQ with LW10:1:1 and Gray+HEQ with LW9:3:3. In the color-only results, a homogeneous overlay of the detected obstacle region is visualized. The bright green line indicates the border of the ground-truth annotation of the drivable surface. Subfigures a, c and e show that our color-only results provide similar or better results in various situations. The right column (b, d and f) shows examples of scenes where not all color settings provide equally acceptable results.

runs at the bottom of Fig. 5. These scores are generated by selecting the optimal setting for each frame out of a (subset of) the available runs, to assess the added value of the processing choices and to provide insights in where the most gain is to be expected in future research. First of all, it is noteworthy that for *every* setting, there are frames in the data set on which it performs best. If the optimal score is selected from all possible runs (including disparity), the highest theoretical score can be achieved (86% correct), as could be expected. However, also with the color data alone there is room for improvement, compared to using the same color space and preprocessing step for every frame. So, even with our adaptive median-cut color indexing, the system can extract more information from different color representations in different situations (fourth bar from below in Fig. 5; 83% correct). Also, note that even with the simplest learning window (LW9:3:3), the color-only Stixel World can outperform the disparity one with a more sophisticated color representation (the bottom bar in Fig. 5; 80% correct), even though using more frames is still better (third bar from below in Fig. 5; 82% correct).

The aforementioned observations are illustrated with the qualitative results in Fig. 4, where the disparity-only results are compared to three of our color-only strategies. We show the setting that performed best (RGB+HEQ, LW10:1:1), one of the runs that relied on the color space that was specifically designed for this context (IllumInv+HEQ, LW10:1:1), and the results with the lowest computational complexity, since it uses gray-scale images and only three LW frames (GRAY+HEQ, LW9:3:3). The left three images show that our methods are all capable of delivering similar or better results than the disparity-only framework. The images in the right column of Fig. 4 illustrate that different settings perform best in different situations, so that the system performance could be increased by adapting the color modeling in even more ways than we currently do. For example, color spaces may be combined or selected online, or the most informative frames within the learning window could be selected adaptively. Metrics and methods guiding this online decision-process will be investigated in future research.

VI. CONCLUSIONS

We have explored a stixel-based probabilistic framework for color-based free-space vs. obstacle segmentation. Our system learns color appearance models for free-space and obstacle classes in an online and self-supervised fashion. To this end, it applies a disparity-based segmentation, which can run in the background of the critical system path and at a lower frame rate than the color-based algorithm. As a bonus, this approach enables operation without a real-time

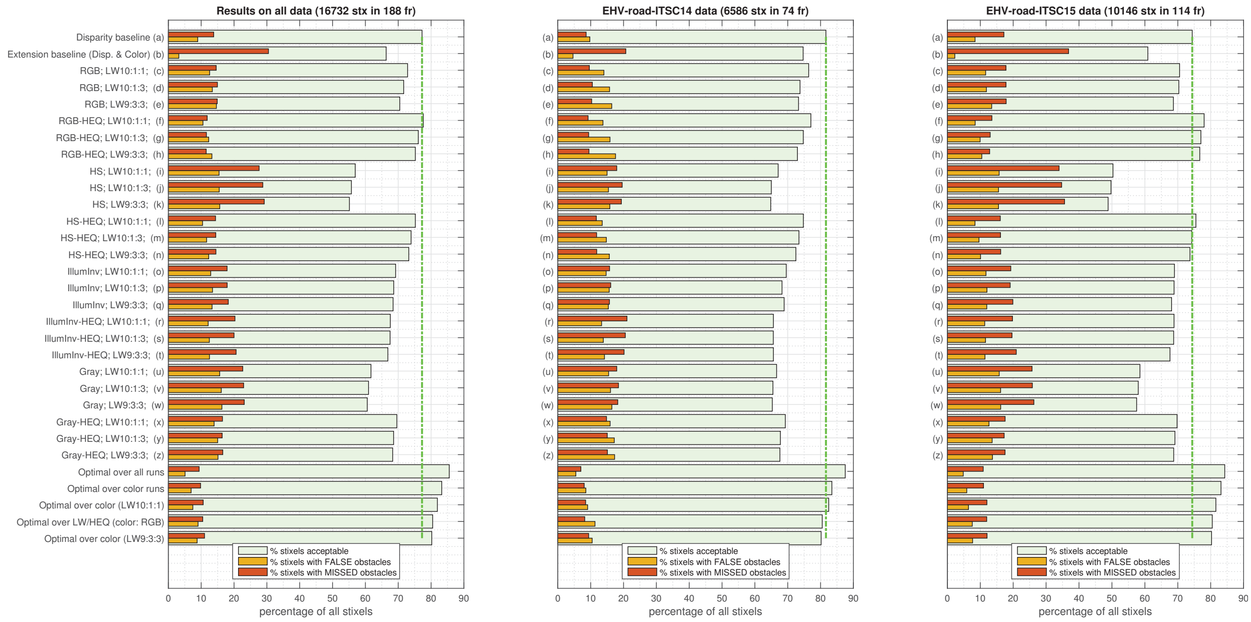


Fig. 5. Quantitative results of free-space segmentation for the baseline that use disparity alone (a) [1] or strongly fused color and disparity (b) [7], and all of our runs. The labels show if histogram equalization (HEQ) is applied, which color space is used and which frames are in the learning window (LW). The learning window parameters are indicated with LW start:step:end as before. The final four entries are theoretical optima, generated by selecting the optimal setting for each frame out of all runs (a-z, including disparity), out of all color-only runs alone (c-z), out of all runs with LW10:1:1 (c, f, ..., x), out of all runs with RGB (c-h) and out of all runs with LW9:3:3 (e, h, ..., z). They give an approximate upper bound for the current processing framework, which could be achieved by, e.g., online selection of color spaces.

disparity estimate. Consequently, the current road scene can be analyzed without the extra latency of disparity estimation. This feature results into a reduced response time from data acquisition to data analysis, which is a critical property for high-speed ADAS.

To achieve reliable color-only free-space detection, we have experimented with several color spaces and different online learning settings. Our evaluation on publicly available data shows that the color-based analysis can achieve similar or even better results in difficult imaging conditions, compared to the state-of-the-art disparity-only method. As an illustrative example, our color-processing detects the correct free-space for 77.6% of all stixels, compared to the disparity-only score of 77.3%. Furthermore, our color-only method results in 4.3% less stixels with missed obstacles on the most challenging data set.

Besides the previous system aspects, the provided meta-analysis of the results shows that our approach of online color modeling is beneficial and can be extended for further improvements, with potential scores of up to 82% within the currently assessed parameter-setting space.

References

[1] D. Pfeiffer, "The Stixel World," Ph.D. dissertation, Humboldt-Universitat Berlin, 2011.

[2] S. Thrun and M. Montemerlo, "Stanley: The robot that won the DARPA Grand Challenge," *J. of Field Robotics*, vol. 23, pp. 661–692, 2006.

[3] C. Urmson, C. Baker, J. Dolan, P. Rybski, B. Salesky, W. Whittaker, D. Ferguson, and M. Darms, "Autonomous driving in urban environments: Boss and the Urban Challenge," *AI magazine*, vol. 30, pp. 17–28, 2008.

[4] W. Van Der Mark and D. M. Gavrila, "Real-time dense stereo for intelligent vehicles," *IEEE Trans. on Intelligent Transportation Systems (TITS)*, vol. 7, no. 1, pp. 38–50, 2006.

[5] G. Dubbelman, W. van der Mark, J. C. van den Heuvel, and F. C. A. Groen, "Obstacle detection during day and night conditions using stereo vision," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2007, pp. 109–116.

[6] R. Labayrade, D. Aubert, and J.-P. Tarel, "Real time obstacle detection in stereovision on non flat road geometry through "v-disparity" representation," *IEEE Intelligent Vehicle Symp. (IV)*, vol. 2, pp. 646–651, 2002.

[7] W. P. Sanberg, G. Dubbelman, and P. H. de With, "Extending the stixel world with online self-supervised color modeling for road-versus-obstacle segmentation," in *IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014.

[8] S. K. Gehrig, F. Eberli, and T. Meyer, "A Real-Time Low-Power Stereo Vision Engine Using Semi-Global Matching," in *Int. Conf. on Computer Vision Systems (ICVS)*, vol. 5815 LNCS, 2009, pp. 134–143.

[9] J. M. Álvarez, M. Salzmann, and N. Barnes, "Learning appearance models for road detection," *IEEE Intelligent Vehicles Symp. (IV)*, pp. 423–429, June 2013.

[10] W. P. Sanberg, G. Dubbelman, and P. H. de With, "Color-based free-space segmentation using online disparity-supervised learning," in *IEEE Con. on Intelligent Transportation Systems (ITSC) [Accepted for Publication]*, 2015.

[11] P. Heckbert, "Color image quantization for frame buffer display," *Computer Graphics*, vol. 16, no. 3, pp. 297–307, 1982.

[12] J. M. Álvarez and A. M. López, "Road detection based on illuminant invariance," *IEEE Trans. on Intelligent Transportation Systems (TITS)*, vol. 12, no. 1, pp. 184–193, 2011.

[13] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets Robotics: The KITTI Dataset," *Int. J. of Robotics Research (IJRR)*, no. October, pp. 1–6, 2013.

[14] H. Hirschmüller, "Stereo processing by semiglobal matching and mutual information," *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 30, no. 2, pp. 328–41, Feb. 2008.

[15] J. Fritsch, K. Tobias, and A. Geiger, "A New Performance Measure and Evaluation Benchmark for Road Detection Algorithms," in *IEEE Conf. on Intelligent Transportation Systems (ITSC)*. IEEE, 2013.

Vision-Based Road Detection using Contextual Blocks

Caio César Teodoro Mendes[1,2], Vincent Frémont[2] and Denis Fernando Wolf[1]

Abstract— **Road detection is a fundamental task in autonomous navigation systems. In this paper, we consider the case of monocular road detection, where images are segmented into road and non-road regions. Our starting point is the well-known machine learning approach, in which a classifier is trained to distinguish road and non-road regions based on hand-labeled images. We proceed by introducing the use of "contextual blocks" as an efficient way of providing contextual information to the classifier. Overall, the proposed methodology, including its image feature selection and classifier, was conceived with computational cost in mind, leaving room for optimized implementations. Regarding experiments, we perform a sensible evaluation of each phase and feature subset that composes our system. The results show a great benefit from using contextual blocks and demonstrate their computational efficiency. Finally, we submit our results to the KITTI road detection benchmark achieving scores comparable with state of the art methods.**

I. INTRODUCTION

Autonomous vehicles, and more concretely Advanced Driver Assistance Systems (ADAS), can potentially reduce accidents, improve traffic flow, save fuel and consequently change the transport landscape. Road detection is a key component of such systems, providing not only free and valid space for maneuvers but also invaluable information for others tasks such as pedestrian and vehicle detection.

In this work, we aim at estimating the road region using a monocular color camera. Visual road detection is a challenging task, where one has to deal with the continuously changing background, illumination issues, and most importantly, the high intra-class variability, i.e. the large variation in road appearance from place to place. Some works estimate the road area by relying on lane markings or sudden changes in appearance near the road boundaries. For instance, [1] uses steerable filters for robustly detecting lane markings. Another popular approach [2], [3] consist of using machine learning techniques, where a classifier is trained to distinguish between road and non-road regions based on images features (e.g. color and texture). In this context, many works focus on proposing new image features for road detection. In [4], the authors propose the use of an illumination invariant color space to deal with shadowed areas.

A common limitation of most machine learning methods is that they independently classify each image region or pixel, ignoring the contextual information and are therefore subject to misclassifying areas of similar appearance. Some efforts have been made to address this issue; [5] uses Conditional Random Fields (CRF) and [2] spatial rays features to incorporate contextual cues. Nevertheless they are limited because first-order CRFs only allow the direct influence of adjacent regions while spatial rays require a pre-segmented image. More powerful ways to exploit contextual information are presented in [6] and [7], the former creates a hierarchical image segmentation, specific classifiers for each level of the hierarchic and uses the classification of one level as features to the next one. The later uses region-specific Convolutional Neural Networks (CNNs) allowing non-linear influence of distant regions. Both approaches, however, are computationally costly and were not able to reach real-time even with parallel implementations.

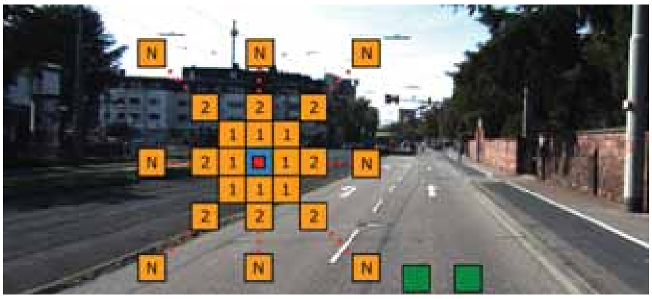


Fig. 1: Illustration of the proposed block scheme. The classification block is show in red, the contextual blocks in orange, the possible support block in blue and the road blocks in green.

We hold that the key for reliable monocular road detection lies in the efficient use of contextual information, and consequently we propose a block scheme to efficiently incorporate contextual cues. Our method classifies small images patches using images features while the so-called "contextual blocks" provide contextual information. The other components, namely the image features and the classifier, were chosen taking into account not only their performance and adequateness to the task but also their computational cost, leaving room for optimized and possible real-time implementations.

The rest of the paper is organized as follows: Section II presents the methodology; Section III shows the method evaluation; The results are discussed in Section IV; finally, Section V draws the conclusions and suggests future works.

II. PROPOSED METHODOLOGY

Our system makes extensive use of rectangular block/patches as shown in Fig. 1. These blocks are divided into three categories: classifications blocks, contextual

Authors[1] are with the Mobile Robotics Lab, Institute of Mathematics and Computer Science (ICMC), University of São Paulo (USP), SP, Brazil, {caiom; denis}@icmc.usp.br

Authors[2] are with Heudiasyc UMR CNRS 7253, Université de Technologie de Compiègne, France, vincent.fremont@hds.utc.fr

Algorithm 1 Feature Concatenation

```
$\mathbf{v}_{final} \leftarrow \varnothing$                                    ▷ Empty vector
$\mathbf{v}_{final} \leftarrow \mathbf{v}_{final} \oplus \mathbf{v}_{class}$                  ▷ $\oplus$: Concatenation
for $i \leftarrow 1$ to $radius \times 8$ do
    $\mathbf{v}_{final} \leftarrow \mathbf{v}_{final} \oplus \mathbf{v}^{i}_{context}$
end for
if $size(class) \neq size(context)$ then
    $\mathbf{v}_{final} \leftarrow \mathbf{v}_{final} \oplus \mathbf{v}_{support}$
end if
$\mathbf{v}_{final} \leftarrow \mathbf{v}_{final} \oplus (\mathbf{v}^{1}_{road} - \mathbf{v}_{class})$
$\mathbf{v}_{final} \leftarrow \mathbf{v}_{final} \oplus (\mathbf{v}^{2}_{road} - \mathbf{v}_{class})$
```

blocks and road blocks. Classification blocks are the ones whose pixels are classified while contextual and roads blocks are auxiliary and delimit regions from which features are extracted. To classify a single image region or classification block, one should extract features from the block itself, its respective contextual blocks and from the road blocks. All these features are pre-processed and concatenated into a single final vector $\mathbf{v}_{final} \in \mathbb{R}^N$ that is fed to a classifier. The output of the classifier, road or non-road, is attributed to every pixel of the classification block. To classify an entire image, this task should be repeated for every image region or classification block, and since we classify every pixel in a classification block, its vertical and horizontal stride is always equal to its vertical and horizontal size.

The features employed in this work do not provide spatial information, i.e. they do not make distinction between pixels positions within a block, hence we use what we call "contextual blocks" to provide information about the surroundings of the classification blocks. The first contextual blocks are positioned in the direct neighborhood of a reference block according to the eight connected scheme and further blocks are aligned in a "star" shape pattern. The reference block is the classification block itself if the classification block and the contextual blocks have the same size. Otherwise it consists of an additional block, called support block, centered on the classification block and with the same size as the contextual blocks. The number of contextual blocks for a classification one is given by their "radius". For instance, a radius of one yields 8 contextual blocks while a radius of 2 yields 16. The feature vector $\mathbf{v}^{i}_{context}$ of each one of the contextual blocks, and the possible support one $\mathbf{v}_{support}$, are concatenated into the final feature vector $\mathbf{v}_{final}$.

Finally, road blocks are positioned in the bottom part of the image and they provide a frame relative notion of the road appearance. The feature vector of each road block $\mathbf{v}^{i}_{road}$ is subtracted from the classification block feature vector $\mathbf{v}_{class}$ and concatenated into $\mathbf{v}_{final}$. The subtraction is made to directly provide the classifier a similarity notion of the block being classified and a supposed road region. We opted for using two small road blocks instead of a larger one, as it is usually done, to minimize the effect of lane markings in the road blocks features. The feature concatenation is summarized in Algorithm 1.

TABLE I: Image features selection

Feature		Dim.
RGB	Mean and std. dev. of each channel	6
Grayscale	Mean and std. dev.	2
Entropy	Mean and std. dev.	2
LBP	Normalized LBP histogram (4-connected)	16
LM Filters 1	Mean and std. dev. of filter responses	30
LM Filters 2	Normalized histogram of the max. responses	15
Total		71

A. Image Features

In this paper, we decided to use some simple color and texture/structure features. We gave preference for fast (low computational cost) and low dimensional features. As small image regions are being classified, there is no need for complex features, such as those employed for object recognition (e.g. Histogram of Oriented Gradients). Furthermore, as we are using a parametric classifier, a low-dimensional feature vector is desirable since it can improve generalization.

Table I presents the selected image features. Entropy, Local Binary Patterns (LBP) and Leung-Malik (LM) [8] filters responses features are generated based on the grayscale image. The entropy is calculated using a circular support region with a radius of 5 pixels. For the LBP descriptor, we chose to use four neighbors instead of the usual eight reducing its histogram dimensionally from 256 to 16. We employed a subset of the original LM filter bank consisting of 6 edge, 6 bar, 1 Gaussian and 2 Laplacian of Gaussian filters, with a 19×19 pixel support, $\sqrt{2}$ scale for oriented and blob filters and 6 orientations.

A spatial prior, in the form of the position of the classification block, is also included in our final feature vector. Preliminary tests suggested that it is preferred to input it encoded as a one-hot bit vector instead of a floating point. Intuitively, this encoding may facilitate the learning of strong priors in parametric models. Concretely we normalized each classification block coordinate, discretize it in 11 parts and represent each discretized coordinate as an 11 bins one-hot bit vector. Therefore the dimensionality of the spatial prior feature is $dim(\mathbf{v}_{spatial}) = 22$, 11 for each coordinate. The exact number of bins should make a small difference in performance as long as it is not too small (e.g. < 5), compromising its discriminative power, or too large (e.g. > 100), significantly increasing the model complexity in parametric models.

If we assume the use of the additional support block, the dimensionality of the final feature vector is given by:

$$\begin{aligned} dim(\mathbf{v}_{final}) = dim(\mathbf{v}_{class}) + dim(\mathbf{v}_{support}) \\ + radius \times 8 \times dim(\mathbf{v}_{context}) \\ + 2 \times dim(\mathbf{v}_{road}) + dim(\mathbf{v}_{spatial}) \end{aligned} \quad (1)$$

where the function dim returns the dimensionality of the input vector. It should be noted that, for this work $dim(\mathbf{v}_{class}) = dim(\mathbf{v}_{context}) = dim(\mathbf{v}_{road})$ and if we consider all features, they are all equal to 71.

B. Classifier

We chose to use a standard Multilayer Perceptron (MLP) neural network, which is a parametric non-linear model. MLPs present a reasonable classification performance in a wide range of tasks and are easily parallelizable to exploit the processing power of Graphics Processing Units (GPUs) and multi-core systems and, as it is a parametric model, its prediction computational cost does not depend on the training procedure (unlike SVMs, for instance).

Our model consists of one hidden layer with Rectified Linear (ReLU) activations functions and an output layer with the sigmoid activation function. We used the cross entropy cost function, therefore only one output neuron is used for the binary classification task. Formally, given that the feature vector $\mathbf{v}_{final}$ is a column vector the prediction is given by:

$$g(\mathbf{v}_{final}) = \sigma(\mathbf{W}_o \cdot \psi(\mathbf{W}_h \cdot \mathbf{v}_{final})), \qquad (2)$$

where $\mathbf{W}_h$ and $\mathbf{W}_o$ are the weight matrices of the hidden and output layer respectively (each row stores the weights of a neuron), ψ is the ReLU function and σ is the sigmoid function. Finally, the output of the model is thresholded according to:

$$L = \begin{cases} \text{Road} & \text{if} \quad g(\mathbf{v}_{final}) > 0.5 \\ \text{Non-road} & \text{if} \quad g(\mathbf{v}_{final}) \leq 0.5 \end{cases}, \qquad (3)$$

where L is the label of the referent classification block.

For regularization, we limit the Euclidean norm of the MLP weights (parameters), the maximum value is chosen per layer and it is applied individually to the weights corresponding to a single neuron (output dimension of the layer). When the norm exceeds the limit, it is scaled down to have exactly the limit value. The training is done using mini-batch stochastic gradient descent with momentum. The training is finished after a number, here called of "patience", of epochs without any improvement in the accuracy of the validation set.

One drawback of MLPs is their large number of hyperparameters. To tackle this issue, before every training, we use a small subset of the training and validation sets to perform a hyperparameter optimization. For this optimization procedure we use the particle swarm optimization (PSO) algorithm and optimize the following parameters: number of neurons, learning rate, hidden layer maximum norm and output layer maximum norm.

III. Experiments

A. Dataset and Setup

To evaluate our approach, we made use the KITTI Vision Benchmark Suite [9]. Specifically, we use the road detection benchmark, which provides 289 annotated images for training and 290 test images. Both sets are divided into three categories: urban unmarked (UU), urban marked (UM) and urban multiple marked lanes (UMM). Methods are ranked according to their pixel-wise maximum F-measure on the Bird's-eye view (BEV) space. The benchmark further provides laser points (Velodyne data), stereo images and GPS data. In our work, only the monocular color images are used and we do not make distinction between the three road categories.

TABLE II: Hyperparameter Search Setup

Parameter	Value
N. Iterations	10
Particles	10
N. Hidden Neurons	$(16, 2000)$
Learning Rate	$(0.001, 0.5)$
Max. Norm Hidden	$(0.5, 5)$
Max. Norm Output	$(0.5, 5)$

To evaluate each component of our system and to select the most adequate parameters/hyperparameters, we divide the 298 annotated images into a set of training/validation containing 260 images and a set for testing containing 29 images. All results reported in this paper, excluding our benchmark submission, are referent to these 29 images. The evaluations are performed in the same way as the benchmark server, i.e. the prediction and the ground truth images are both converted into BEV space and are compared pixel-wise.

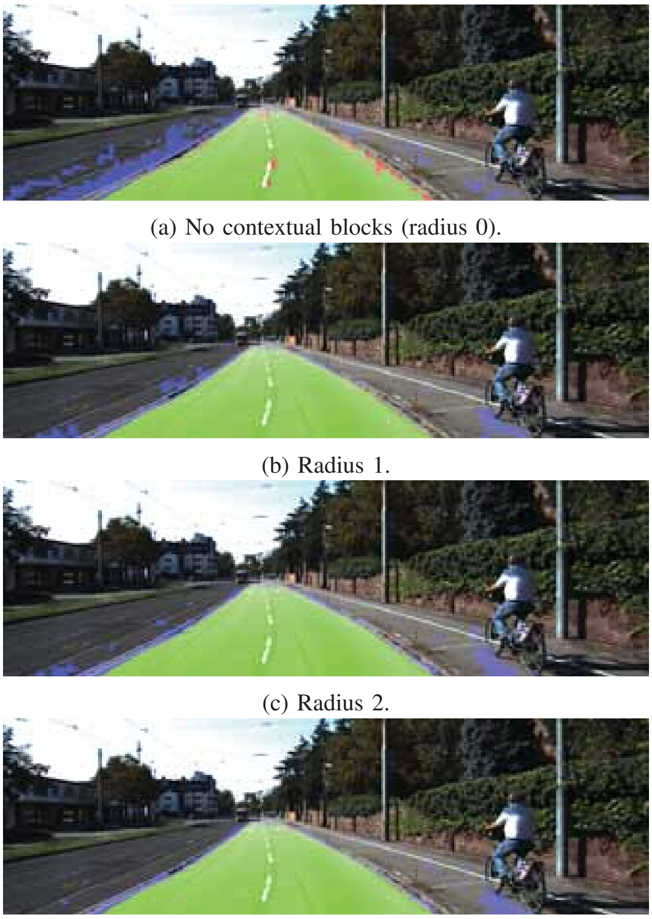
We implemented our system using the Python-based SciPy software ecosystem and scikit-image library for feature extraction. We use the MLP GPU implementation provided by the Pylearn2 [10] library and conduct the PSO hyperparameter optimization using the Optunity [11] library. The tests were conducted on a machine equipped with an Intel Core i7-4930K, 64GB RAM and an NVIDIA Titan X. The GPU was utilized only for model training and testing, the rest of the system runs on a single core.

For every test and the benchmark submission, we fixed the blocks size at 10×10 for the classification blocks and 20×20 for the contextual blocks, hence we always use the additional support block. We believe that those sizes yield a good compromise of computational cost, discriminative power and classification granularity. In this work, we choose to focus on evaluating the blocks scheme itself rather than parameters effects.

B. Training Scheme

To generate the features vectors (samples) for training, we used only classification blocks whose ground truth pixels are all of the same class, excluding, therefore, ambiguous cases. We adequately pad images to accommodate the selected block sizes and contextual blocks in order for the classification blocks to cover the full original image. As the top 150 lines of every image contain only negative examples and are not considered in the BEV space evaluation, we ignore this region when generating the feature vectors for training. This measure reduces the training time and helps to improve the class balance.

The samples extracted from the 260 images that are selected for training/validation are randomly split into 70% training and 30% validation. Each of these datasets are further subsampled at 20% for the hyperparameters search, where the validation set is used for early stopping and for the hyperparameter selection. Once the best hyperparameters



(a) No contextual blocks (radius 0).

(b) Radius 1.

(c) Radius 2.

(d) Radius 3.

Fig. 2: Classification results using different radius parameter values where green represents true positive, red false negative and blue false positive.

TABLE III: Contextual Blocks Radius Evaluation (in %)

Radius	F-measure	Accuracy	Precision	Recall
0	83.7	87.8	87.8	79.9
1	86.3	89.4	87.5	85.0
2	87.3	90.4	89.6	85.2
3	88.2	91.0	90.2	86.2

TABLE IV: Road Block Evaluation (in %)

Blocks	F-measure	Accuracy	Precision	Recall	Diff.
All	88.2	91.0	90.2	86.2	0.0
No Road	87.9	90.8	90.1	85.9	-0.3

are established, the training proceeds by using the initial $70-30$ split, where the validation set is used only for early stopping. All samples are standardized feature-wise based on the training dataset. The hyperparameter search configuration is presented in Table II. We use 10 particles and 10 iterations resulting in 100 training procedures. Further MLP parameters are the 100 mini-batch size, 0.9 momentum and 30 patience.

C. *Evaluation of Contextual and Road Blocks*

We initially evaluate the effects of using contextual blocks and their radius parameter using all image features. Table III shows the results when varying the radius on the 29 testing images. A radius of 0 means that no contextual block is in use. The results show a substantial increase in the F-measure from no contextual block use (radius 0) to radius 1 and further radius increases yield a small but consistent improvement. This effect is also clearly visible in the classification results as shown in Fig. 2. The image classified using no contextual blocks presents a significant amount of false positive and false negative pixels. With a radius of 1, all false negative pixels are removed and the number of false positives is reduced. The number of false negatives continues to decrease until radius 3, when the left side of the resulting image is almost clear of false positives.

These results highlight the validity of our contextual blocks approach and despite the higher dimensionality of the feature vector, the classifier was able to take advantage of the additional information. We did not test radiuses larger than 3 due to hardware constraints (especially the working memory), nevertheless the benefit of larger radiuses is expected to fade and not compensate the additional computational cost.

We also evaluate how the road blocks affect the performance. For that purpose, we removed the road blocks features while maintaining the best radius parameter previous obtained (3) and all image features. Table IV shows the results where the column “Diff.” refers to the difference in F-measure when using all blocks. The removal of the road blocks has a minor effect on the performance, affecting it less than a single decrease in the contextual blocks radius. We can therefore conclude that, for this dataset, our method is robust and does not depend on the usage of road blocks. However in datasets where the change in road appearance between training and test sets is more enunciated, these blocks could play a major role in helping with generalization.

D. *Features Evaluation*

Using the best radius deducted from previous experiments, we evaluated the contribution of each feature subset. To do so, we removed each feature subset and evaluated the performance on the 29 test images. The results are show in Table V. These results show that the LBP and LM 2 texture features provided the most significant contribution despite LBP using the unusual 4 neighbor parameter and the small subset of filter selected for the LM features. The LM 1 features did not provide benefit and, in fact, their removal resulted in a 0.3 F-measure increase. Considering that a non-linear parametric model is employed, we suspect that the unique information content of the LM1 features did not compensate for their relative high dimensionality (750 considering all blocks). The RGB features provided a reasonable contribution while the gray features made little difference, probably due to their redundancy with the RGB ones. The spatial prior showed of little importance for our method, which is expected since we use a large contextual support. Methods with smaller or no contextual support

TABLE V: Features Evaluation (in %)

Feature Subset	F-measure	Accuracy	Precision	Recall	Diff.
All	88.2	91.0	90.2	86.2	0.0
No RGB	87.7	90.7	90.4	85.2	-0.5
No Gray	88.1	90.9	90.4	85.8	-0.1
No Entropy	87.7	90.7	90.3	85.3	-0.5
No LBP	87.3	90.4	90.2	84.6	-0.9
No LM 1	88.5	91.2	90.4	86.8	+0.3
No LM 2	87.0	90.0	88.6	85.5	-1.2
No Spatial	88.0	90.8	90.4	85.7	-0.2

would greatly benefit from using a spatial prior. Overall, the method is robust to the feature selection as no feature subset removal reduced the F-measure to the level of not using contextual blocks.

E. Processing Time

Table VI shows the average processing time to classify an image relating it to each major stage in our system and the contextual blocks radius parameter. These results were produced using all but LM 1 features. The feature extraction phase refers to the generation of grayscale, entropy, LBP and filtered images. The pre-processing and concatenation phase encompasses the mean, standard deviation, histogram calculations and also the feature concatenation of all blocks involved. Finally, the model prediction phase refers to the prediction time of the model for all classification blocks.

Overall, the processing time of our system implementation is far from achieving real-time but we believe that an optimized implementation, e.g. using C language and taking advantage of multi-core systems, may achieve it. This belief is motivated by the fact that there is no stage in our system that is intrinsically costly (e.g. Textons or HoG features) and the most time consuming parts of our system (convolutions, windowed operations) are suitable for parallelization. One important thing to notice is how the processing time scales with the radius, although the number of blocks (and features) greatly increases with larger radiuses, the processing times are less affected. This is due to the efficient implementation of the contextual blocks, where their features are pre-calculated for the whole image and then appropriately concatenated for each classification block. The feature extraction phase takes longer due to the larger padding used.

TABLE VI: Average Processing Time for Classifying an Image (in seconds).

Radius	0	1	2	3
Feature Extraction	0.44	0.49	0.51	0.65
Pre-processing and Concatenation	0.94	1.07	1.17	1.27
Model Prediction	0.02	0.03	0.04	0.05
Total	1.40	1.59	1.72	1.97

TABLE VII: Urban Road KITIT Benchmark Results (in %)

Method	MaxF	Pre.	Rec.	FPR	FNR	Runtime
DNN [6]	93.43	95.09	91.82	2.61	8.18	2s
HIM [7]	90.64	91.62	89.68	4.52	10.32	7s
NNP	89.68	89.67	89.68	5.69	10.32	5s
NED	89.12	85.80	92.71	8.45	7.29	1s
Our method	88.97	89.50	88.44	5.71	11.56	2s
FusedCRF	88.25	83.62	93.44	10.08	6.56	2s
ProbBoost [3]	87.78	86.59	89.01	7.60	10.99	150s
SPRAY [2]	87.09	87.10	87.08	7.10	12.92	0.04s
RES3D-Velo [12]	86.58	82.63	90.92	10.53	9.08	0.36s

F. Benchmark Submission

To compare our method with others, we submitted our method results to the road detection KITTI Benchmark[1] using all but the LM 1 features and a radius of 3 for the contextual blocks (the best configuration according to the performed experiments). Table VII presents the first nine benchmark results in the *Urban Road* category which includes all road images types (UU, UM and UMM). Our method achieved the fifth best score out of 31 participants, including the ones taking advantage of LIDAR (FusedCRF and RES3D-Velo) or stereo vision (NNP and ProbBoost).

The first two methods (DNN and HIM) uses global context (takes the whole image into consideration) which may explain their high scores. The next two methods are yet to be referenced, the only information available tells that the NNP method uses stereo vision (plane fitting) and NED uses some form of CNN. The fastest method in the benchmark is the SPRAY method. As our work, this method focuses on providing the classifier contextual cues in an efficient way. All methods scoring better than ours uses some form of parallel processing and could not achieve real-time.

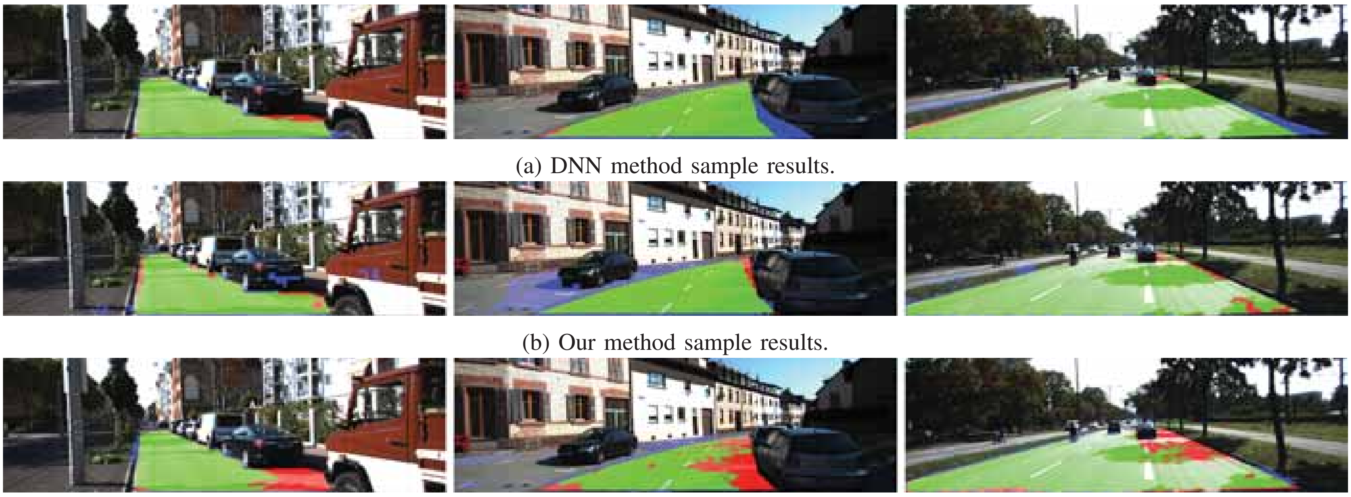
Figure 3 shows a visual comparison[2] of the first method, our method and the fastest one using images provided by the benchmark server. The DNN method tends to obtain smoother boundaries and an overall better result. Our method and the SPRAY one have a tendency to misclassify similar regions, but our does so to a lesser extent. Our method, however, presents a few more false positives predictions than the other two.

IV. Discussion

The proposed approach yields results in line with state of the art methods. The use of contextual blocks provides significant performance improvements that scales adequately with the radius parameter. The method run-time depends mostly on the images features selection, while the block scheme itself have a low computational cost since their features can be pre-calculated and simply concatenated afterwards. One advantage of our method is its simplicity, especially when compared to other road detection works (e.g. [3], [2]). We provide a small image features selection that seems to be adequate for road detection and whose implementation can be highly optimized. We also presented other details such

[1] `http://www.cvlibs.net/datasets/kitti/index.php`

[2] Video demo: `http://youtu.be/QFmOZyqtClU`



(a) DNN method sample results.

(b) Our method sample results.

(c) SPRAY method sample results.

Fig. 3: Sample classification results extracted from the KITTI benchmark server where green represents true positive, red false negative and blue false positive.

as the training scheme and hyperparameters search that may have contributed to the method performance.

Despite encouraging results, our method has some limitations. Global features are unpractical to include due to padding requirements and, even with a large radius, the whole image can not be considered. The presented implementation is not optimized and, although we hold that it could be optimized for real-time purposes, we do not provide evidence that it is the case. Compared to deep learning methods [6], our method has the disadvantage of requiring a selection of hand-crafted features, which is mostly intuitive since it is not possible to evaluate all combinations of image features present in the literature. Finally, the use of road blocks is controversial as it is based on the assumption that the bottom part of the image always refers to a road region. In this work, however, the road blocks could be removed with a minimal performance penalty or, more generally, other sensors (e.g. stereo camera) could be used to support that assumption.

V. Conclusions and Future Works

This work proposed an efficient block scheme to exploit contextual information and also sensible choices for image features and classifier. Each system component has been evaluated, along with image feature subsets and processing times. The results reaffirm the importance of contextual information for road detection and demonstrate the method effectiveness that, despite being simple, could achieve results comparable with state of the art methods. Unfortunately the method still has some limitations that need to be addressed, as its inability to incorporate fully global contexts and the current implementation run time. As future work, we intend to perform an optimized implementation of our method and use convolution neural networks as feature extractors.

Acknowledgment

The authors would like to acknowledge the support granted by FAPESP (process nr. 2011/21483-4) and CNPq (process nr. 202415/2014-7).

References

[1] J. McCall and M. Trivedi, “Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 1, pp. 20–37, March 2006.

[2] M. Passani, J. Yebes, and L. Bergasa, “Crf-based semantic labeling in miniaturized road scenes,” in *IEEE Conference on Intelligent Transportation Systems (ITSC)*, October 2014, pp. 1902–1903.

[3] G. B. Vitor, A. C. Victorino, and J. V. Ferreira, “A probabilistic distribution approach for the classification of urban roads in complex environments,” in *Workshop on IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[4] J. Alvarez and A. Lopez, “Road detection based on illuminant invariance,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 1, pp. 184–193, March 2011.

[5] T. Kuhnl, F. Kummert, and J. Fritsch, “Spatial ray features for real-time ego-lane extraction,” in *IEEE Conference on Intelligent Transportation Systems (ITSC)*, September 2012, pp. 288–293.

[6] R. Mohan, “Deep Deconvolutional Networks for Scene Parsing,” *ArXiv e-prints*, 2014. [Online]. Available: http://arxiv.org/abs/1411.4101

[7] D. Munoz, J. A. Bagnell, and M. Hebert, “Stacked hierarchical labeling,” in *European Conference on Computer Vision (ECCV)*, September 2010, pp. 57–70.

[8] T. Leung and J. Malik, “Representing and recognizing the visual appearance of materials using three-dimensional textons,” *International Journal of Computer Vision*, vol. 43, no. 1, pp. 29–44, 2001.

[9] J. Fritsch, T. Kuhnl, and A. Geiger, “A new performance measure and evaluation benchmark for road detection algorithms,” in *IEEE Conference on Intelligent Transportation Systems (ITSC)*, October 2013, pp. 1693–1700.

[10] I. J. Goodfellow, D. Warde-Farley, P. Lamblin, V. Dumoulin, M. Mirza, R. Pascanu, J. Bergstra, F. Bastien, and Y. Bengio, “Pylearn2: a machine learning research library,” *ArXiv e-prints*, 2013. [Online]. Available: http://arxiv.org/abs/1308.4214

[11] M. Claesen, J. Simm, D. Popovic, Y. Moreau, and B. D. Moor, “Easy hyperparameter search using optunity,” *ArXiv e-prints*, 2014. [Online]. Available: http://arxiv.org/abs/1412.1114

[12] P. Shinzato, D. Wolf, and C. Stiller, “Road terrain detection: Avoiding common obstacle detection assumptions using sensor fusion,” in *IEEE Intelligent Vehicles Symposium (IV)*, June 2014, pp. 687–692.

Following Dirt Roads at Night-Time: Sensors and Features for Lane Recognition and Tracking

Sebastian F. X. Bayerl, Thorsten Luettel and Hans-Joachim Wuensche

***Abstract*— The robust perception of roads is a major prerequisite in many Advanced Driver Assistant Systems such as Lane Departure Warning and Lane Keeping Assistant Systems. While road detection at day-time is a well-known topic in literature, few publications provide a detailed description about handling the lack of day-light.**

In this paper we present multiple sensors and features for perceiving roads at day and night. The presented features are evaluated according to their quality for road detection. We generated a large number of labeled sample data and extracted the quality of the features from their probability distributions. The challenge of tracking an unmarked road under bad lighting conditions is demonstrated by comparing receiver operating characteristics (ROC) of the features at day and night-time.

Based on these results we present a road tracking system capable of tracking unmarked roads of lower order regardless of illumination conditions. Practical tests prove the robustness up to unmarked dirt roads under different weather conditions.

I. Introduction

In the last decades autonomous robots became more and more a focus of interest. Both scientific research facilities and car companies are investing considerable man-power into this field. On the way from Advanced Driver Assistant Systems (ADAS) to a completely self-driving vehicle a lot of challenging tasks have to be solved robustly to enable the robot to participate in traffic. One of those issues is environment perception. In order to keep an autonomous vehicle driving on a lane or to give a lane departure warning to the driver, one has to robustly detect the lane's geometry and position. The most common approach are vision systems, but due to the varying appearance of roads and its strong dependency to illumination conditions this is still a challenging task ([1], [2], [3]).

Moreover, little work has been done on tracking roads at night. In this paper we present characteristics of rural roads and sensors in order to perform a robust tracking at day and night. We are especially interested in tracking roads without any kind of boundary markings. To this end, we utilize and extend our road tracking approach shown in [4].

This paper is structured as follows: In the following section some related work on road tracking methods is described focusing on their performance at night. In Section III we show our autonomous vehicle, the sensors it is equipped with and how the sensor measurements are fused into a multilayer terrain map. The features extracted from the sensor data are shown in Section IV, followed by an evaluation in Section V. A tracking system utilizing these features is presented in chapter VI. Finally, we conclude with a short summary and an outlook to future work.

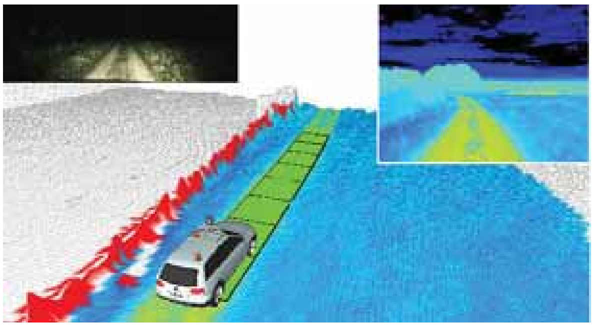


Fig. 1. Tracking a rural road at night using a multi layer terrain map (thermal layer is shown). Red cells indicate obstacle cells. Upper left: Color Night Vision camera, Upper right: thermal camera

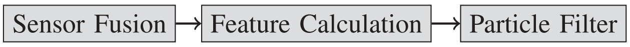


Fig. 2. Global Architecture of the Tracking System

II. Related Work

In the past, a lot of different lane recognition systems have been developed. Today perception of high- and freeways work quite well and is already implemented in modern cars for lane departure warning systems. Most of them are based on vision systems and are developed for daytime application. A commonly used recursive filtering technique allows estimating the parameters of road models ([5], [6], [7]) such as clothoids or splines.

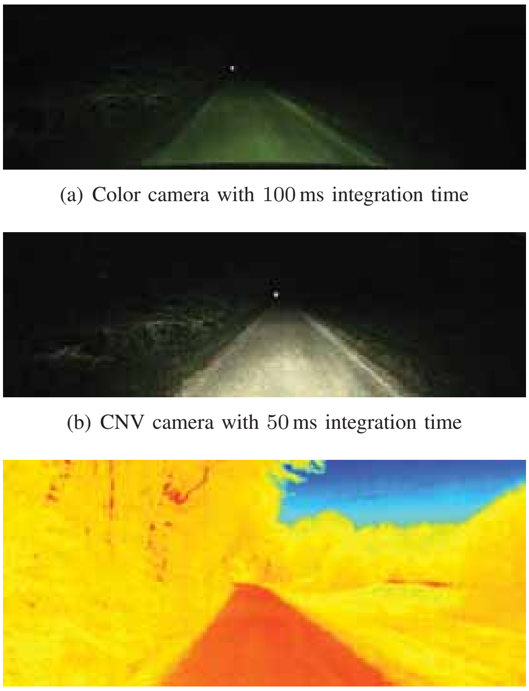
In vision-based approaches features like extracted lane-markings, color and/or texture information is used. For example a particle filter approach investigated by *Franke et al.* [8] treats the road recognition as a maximum-a-posteriori estimation task that optimizes the parameters of a road model given an image sequence. For weighting the large number of particles they calculate a joint probability of each hypothesis by considering features such as color saturation, texture and edges. One benefit of this kind of tracking is the easy way to add or remove features as they are treated independently. Expanding on this idea *Manz et al.* [9] developed a hybrid estimation approach that was able to even follow dirt roads. Parameterless systems have also been presented. In [3] the problem of detecting the road was interpreted as finding a minimum-cost path from the lower to the upper image part.

All authors are with the Institute for Autonomous Systems Technology (TAS), University of the Bundeswehr Munich, 85577 Neubiberg , Germany, Contact author email: `sebastian.bayerl@unibw.de`

The costs are defined by extracted lane markers, the grass verge (a color based classification) and the free space that was detected by stereo vision.

Although all system proved their strength in difficult scenarios, they all have weakness, too. The vision-only approaches strongly suffer from the disadvantage of their single sensor. Even the best camera does not reach the quality of a human eye, and thus, challenging lighting conditions or less colored environments are limiting the power of this approach to road recognition. Multi-sensor systems combine the benefits of different sensors to achieve a higher level of robustness. In [4] sensor data from color-camera and a LiDAR is fused, accumulated and used for tracking road networks.

One aim of this paper is to show features that indicate the existence and the position of the road at night. An example application of road tracking at night is introduced in [10]. *Serfling et. al* developed a particle filter which is able to estimate position errors inside a digital map by comparing the map-based road information with the sensor data. Each particle represents a potential road course with a different width, robot position and orientation. They suggest measuring road's boundary gradient and its orientation by edge operations on a night-vision image. Due to the different reflectivities of the road and the non-road surface, the road boundary is also visible without lane-markings. Additionally, a lighting independent imaging radar sensor is used for weighting the particles with information about road area and gradient. Another night-vision approach for map-matching is shown in [11]. *Schüle et. al* propose a sensor fusion system that employs digital map information in combination with radar and camera sensors to estimate the 3D road course. This information is fused with the results of an optical lane recognition system. The result is an accurate road course. Road curvature estimation is performed in [12] by using a far infrared camera, a near infrared camera and an imaging radar sensor. *Harmann et. al* are training a Convolutional Neural Network in order to distinguish between three different road curvatures. This kind of detection system is not able to keep a robot on a road, since only a few discrete statements about the lane are made. For our application we need a detailed description of the lane that has to be followed.


(a) Color camera with 100 ms integration time

(b) CNV camera with 50 ms integration time

(c) Thermal image scaled from 9°C (blue) to 13°C (red)

Fig. 3. Camera images showing a dirt road. Headlights are enabled.

III. Sensors

A. Robot and Hardware

The software we develop is applied to our robot platform MuCAR-3 (Munich Cognitive Robot Car - 3rd Generation), a stock VW Touareg equipped with full drive-by-wire capability. The vehicle motion is estimated by a Kalman filter that fuses data from an inertial navigation system with vehicle odometry to provide jump-free estimates in an inertial integration space.

MuCAR-3 is equipped with different sensors that are beneficial for road detection. The main sensors are listed below:

- A Velodyne HDL-64 laser scanner provides 1.3 million measured points per second. This 3D point cloud provides a lot of information about the terrain profile up to 120 meters in distance. Since the laser-scanner is an active sensor that is not affected by lighting conditions, it is ideally suited for night-time application. Obstacles like bushes, trees or any kind of terrain slopes indicate the boundary of a road. In a flat-world scenario however, one is not able to detect a road using this sensor only, so we suggest using additional types of sensors.
- The second type of sensors are default color cameras. This very common sensor is used in a lot of previous work (e.g. [8], [9]) for road detection, but suffers from its sensor-typical limitations: At low light conditions the camera needs a lot of time for image exposure. High integration times leads to smearing in high dynamic scenarios. As a result we limit the integration time and take less illuminated images into account.
- The application of a color night vision (CNV) camera provides more information even in less illuminated areas. Thanks to larger pixel size and other low-light technologies this camera is better adapted for tracking roads at night.
- The last sensor is a thermal camera utilizing an uncooled microbolometer. The thermal camera is able to measure the different surface temperatures of the terrain independently of the lighting conditions. Due to the different materials of the surface the corresponding temperature is different. Figure 3 (c) gives a visual impression of the surface temperature of a dirt road. The temperature of the road is higher than that of the non-road area.

B. Sensor Fusion

In a fusion step the high amount of sensor data is combined in low level manner. We suggest to use the fusion algorithm according to [13] and [4] and extend them by a temperature and a NIR layer. We do not want to go into detail, but the main ideas can be described as follows. A local terrain map containing multiple layers of information is introduced and built as a Cartesian grid with a cell size of $0.2\,\mathrm{m}$. (A multi layer map including color, height and obstacle information is shown in Figure 4 (a).) The aim of this terrain mapping is to produce a dense local representation of the environment, making use of all data the sensors provide. Hence, we accumulate the measured data from several scans and overcome the limitation of the limited field-of-view (FOV) of the cameras and the vertical resolution of the LiDAR. We consider occluded areas, which are not visible to the camera, as well as overhanging structures, which can be interpreted as non-obstacles. The fusion step is passing several maps (such as obstacle, height, height-difference, color and thermal map) to the feature extraction.

IV. Features

In order to detect and track a road several features are defined. This chapter includes their detailed description, whereas their evaluation is part of the following chapter V. According to the nature of their main sensors they can be split up in three groups: color, 3D and thermal.

A. Color

The first group of features is based on the color information of the terrain map.

- The first features we can extract from the color map are edges, since we expect a change in color at the lane boundaries. Therefore we apply one edge operator for each direction (x and y) in order to get the edge intensity $v_{\mathrm{EI}} = \sqrt{v_{\mathrm{EI,x}}^2 + v_{\mathrm{EI,y}}^2}$. The change in color should be perpendicular to the direction of the road. As a result we use the corresponding edge phase, that can be calculated consequently with $v_{\mathrm{EP}} = \mathrm{atan}\left(\frac{v_{\mathrm{EI,y}}}{v_{\mathrm{EI,x}}}\right)$.
- The green-ratio feature assumes that the road area has a low ratio of green color compared to the other color channels ($\frac{g}{r+g+b}$). Since slightly vegetated forest roads do have a high green ratio, we also consider the area below the robot. There is no additional sensor to observe this area, but the cells are taken from the accumulated map (Section III-B). Our algorithm adapts a reference value online and calculates the cell's green-ratio feature accordingly.
- According to [9] the color of frequently-driven roads tends to have a low saturation value in the HSV color space. Because the saturation channel of the HSV space is independent of the illumination intensity, the feature depends less on shadows.
- Assuming that the vehicle drives on a road, we compare all color information inside the terrain map with the cells below the robot. The accumulation of sensor data (see Section III-B). One method to do the comparison is to setup a histogram of RGB-color values below the vehicle and perform a histogram back projection to the complete terrain map. This color comparison can be executed also by using another color space. We suggest utilizing the YUV color space, by calculating the mean and standard deviation of its components inside the area below the robot. The feature value of a cell results from the Mahalanobis distance to the color values below the vehicle. Since the Y component represents the illumination, which possibly might have negative influence at scenarios with challenging lighting conditions, it can be neglected.
These features should never be used alone, since the robot will not able to rejoin a road after having left once.

B. 3D Features

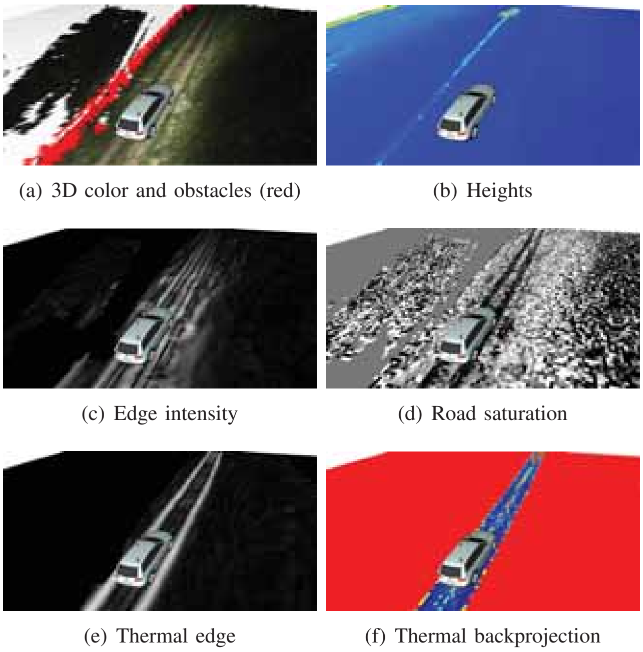
The second feature group refers to the 3D texture of the surface. Their main source of information is the LiDAR, but it can be driven with any system capable of providing 3D data.

- We expect roads to be free of static obstacles. This implies that occupied cells are definitely non-road cells and areas of high obstacle density tend not to be roads.
- The obstacle detection does not interpret a cell as an obstacle if the cell is located inside a low terrain slope. But even small slopes indicate the position of a road. Areas with a high variation in slope have a low road probability.
- The terrain height can be introduced in a similar way to the slope. The cell's information about its height is an indicator of non passable obstacles or areas that are not nice to drive on. The higher the difference between the cell's height and the robot plane, the more likely the cell is to be non-road.
- We use a classification of vegetation such as grass or bushes by analyzing the frequency of the distance signal of each laser scanner diode. This method allows us to distinguish between road and grass independently of light. This information, however, should not be a hard decision criteria, since a lot of dirt roads or forest roads are slightly covered with grass.

C. Thermal

The last feature group uses information provided by the thermal camera. In most cases the base material of the road is different to the non-road material. This causes a different heating of the surface at day-time and thus we expect differences in the temperature profile of road and non-road area. Our practical tests have confirmed this and we were even able to observe thermal differences between the white markings of a road and the road itself. See Figure 3. In order to use this we introduce the following two features:

- Similar to the edges in color, we expect edges in temperature located at the road boundaries. For this reason we apply edge operators to the thermal layer



(a) 3D color and obstacles (red)

(b) Heights

(c) Edge intensity

(d) Road saturation

(e) Thermal edge

(f) Thermal backprojection

Fig. 4. Features of road scene shown in Figure 1

of the terrain map and get measurements of the edge's intensity and direction.

- The last important observation is the homogeneous temperature distribution along the road. Based on this we calculate the temperature statistics under the robot and compare it to all cells of the terrain map. If we assume that the robot is already located on the road, we can extract cells with similar temperature and the road itself.

V. Feature Evaluation

In the following chapter we analyze the quality of the presented features and state which features are beneficial under challenging night-time lighting.

A. Data Generation

The generation of feature data is performed by applying the road tracker from Section VI to several recordings of sensor measurements. Given the tracking result, we can generate pairs of positive and negative road samples. The positive is defined by the road tracking result itself and the negative is built by falsifying the road geometry (clothoid parameters, such as displacement or curvature) of the positive one. Given the road geometry we are able to extract the sample's features - e.g. edges at the road boundary. For each sample all the feature values are finally stored with a label (positive or negative) in memory.

The different sensor recordings include different seasons (summer, winter, ...) and types of roads (paved, unpaved, dirt, forest). Each track was driven at day- and at night, so we can evaluate which features are still beneficial at less illumination. We were able to generate at least 10000 night samples and the corresponding 10000 day samples.

B. Comparison

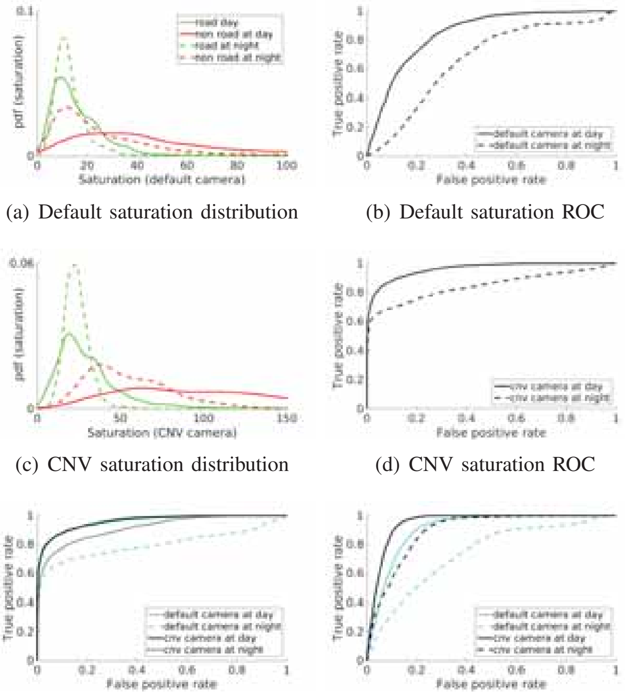
In order to perform this comparison, we define criteria which represent the feature's quality. We suggest taking a look at the distribution of the labeled feature values and the Receiver Operating Characteristic (ROC) of the features. The influence of day and night-time causes differences in the ROC of a feature.

In general the ROC's curve is created by plotting a feature's true positive rate against the feature's false positive rate at various threshold settings. High true positive rates at low false positive rates mark the most powerful features.

The ROC of one single feature was generated as follows: the feature's values of the collected data are compared to a threshold and classified as road or non-road. Since the data is labeled, we can identify false positives, false negatives, true negatives or true positives. We sample the threshold and get different numbers of the false positives and true positives for each threshold.

The following specifies the different Probability Density Functions (PDF) and ROC curves for day and night time. We do not describe all the features in detail, but we select the most representative ones.

- Evaluating the color features of the road, we noticed - as expected - a reduction of quality. In all ROC curves the degradation of the features based on the default color camera is much higher than to the CNV camera. In some cases (default camera's saturation or green ratio, Figure 5 (b) and (f)) the ROC curve approximates to the $y = x$ axis, which represents a maximum uncertainty.



(a) Default saturation distribution

(b) Default saturation ROC

(c) CNV saturation distribution

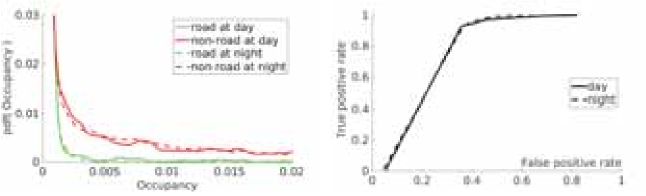
(d) CNV saturation ROC

(e) Edge intensity ROCs

(f) Green Ratio ROCs

Fig. 5. PDF of color features

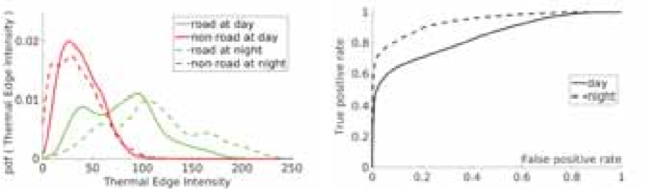
- The active LiDAR measurements allow the the 3D features to be independent of illumination. The distribution of the features is almost the same. For example see the occupancy feature at Figure 6. Also the ROCs do not change from day to night. Thus, they can be used unrestricted for tracking at night. Unfortunately a road cannot be detected by 3D features only in general. The low true positive rates at low false positive rates can be interpreted as follows: The 3D features contain the implicit information about non-road area, but no direct information about roads.



(a) Obstacle feature distribution (b) Obstacle feature ROC

Fig. 6. PDF of 3D features

- The last group to evaluate are the thermal features. Transitions at the road boundary are observable at each time of the day. This group of features has even a better performance at night because there is no direct influence of the sun. Shadows generate different surface temperatures on one surface material.



(a) Thermal edge distribution (b) Thermal edge ROC

Fig. 7. PDF of thermal features

The thermal features depend on weather conditions. Rain and fog cool down the surface and make the road area and the non-road area having less difference in temperature. In Figure 8 the ROC of the thermal edge direction is presented for dry and wet scenarios. The temperature ROC has also a strong dependency to the quality of the road. This dependency is even stronger than the influence of weather. A example of poor structured road can be seen in Figure 10.

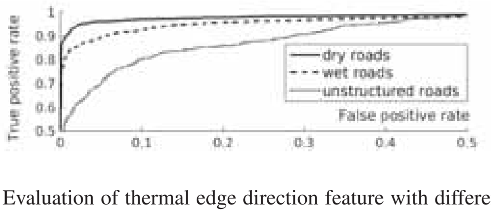


Fig. 8. Evaluation of thermal edge direction feature with different weather conditions and road quality

VI. Tracking System

In this paper we use an extended version of the tracking method suggested by [4]. This method is based on a particle filter, which projects its road hypotheses into the local terrain map and weights them according to a set of road features, similar to the presented ones. The main extension of our method is the increased number of road features and the usage of further sensors. This increases the robustness of tracking and we are able to perceive road networks even at night.

A. Particle Filter

Similar to [4] we use two different road geometries to model the road network. A clothoid with a fixed width is used as a road model. The intersections are constructed by multiple clothoids starting at a common origin. The algorithm switches between the different models with the help of a digital road map. All coordinates of the system state are kept relative to the robot's ego coordinate system. The filter can be split up in two main parts:

- A prediction step uses the estimation of the robot's motion in order to update the relative coordinates of the road and intersection (see [9]).
- In the following measurement step we rebuild the probability distribution of the state vector $\mathbf{x}_p$ by assigning a specific weight to each predicted particle. The filter projects the geometric models of all particles into the local terrain map and measures their road quality. This quality q_p is estimated by a Naive Bayes classifier, that compares n measured feature values to a trained feature distribution of roads or uses a manual generated heuristic which again depends on the measured feature values. In a last step the state vector and its covariance are generated as the weighted mean of the best particles.

The calculation time of the algorithm is small enough to run on an ordinary machine (Intel i7 2600K) in sensor rate, which is given by the Velodyne LiDAR as slowest sensor at 10 Hz. Figure 1 is showing the estimation of the road tracker.

B. Effects to Tracking Quality

In this section we want to demonstrate the benefit of the additional night sensors and features and compare it to the previous method introduced in [4].

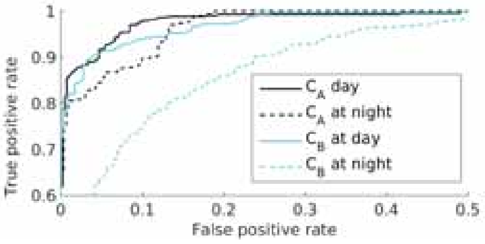
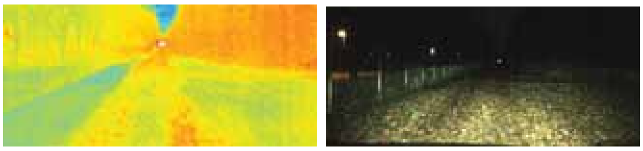


Fig. 9. ROCs of classifier with full and reduced feature capability

Two different Naive Bayes classifiers are trained: one with full feature capability (classifier C_A) and a second with the reduced number of features (classifier C_B). The second reduced classifier represents the classification power of the method presented in [4] and is only trained with default color camera features and without any information of night sensors. At day the performance of C_A is only slightly better than C_B. At night the classifier show a different degradation of their performances: C_A is influenced only a bit because it supports all the presented features. The ROC of C_B is influenced very much since it is limited to the default color camera.

C. System Limitations

A bad classification result indicates that the system is operating near the limit of its perception. One key role for our robust tracking system is the clear separation of road and non-road cells by a couple of features. The system has reached its limits if the sensor data does not indicate any existence of a road. For example in a flat environment the 3D point cloud of a laser scanner cannot be used to determine the position of the road. A similar case is given by the thermal camera and a homogeneous temperated surface. Here, the sensor does not provide useful information. Also a color camera is not a perfect sensor. One is not able to visually extract the road in a uniformly colored terrain. One example of a challenging scenario is show in Figure 10, where the road is covered by leaves. Since the particle filter uses all of the above mentioned features, the system is able to compensate sensors, which do not provide beneficial information. At least one significant feature is necessary to make tracking valid.



(a) Thermal camera

(b) CNV camera

Fig. 10. Challenging road scene

VII. Conclusion and Future Work

A. Conclusion

In this paper we describe several sensors and their relevance for road tracking at night-time. In a first step color cameras, a thermal camera and a LiDAR are fused into a multi layer local terrain map. Based on the information of this grid map, we generate a variety of features and evaluate them according to their relevance for road recognition. Obviously, the active sensors are not influenced by lighting conditions. Also the features which are based on thermal data are very informative. The data provided by the color camera gets worse with less light, but is still partially usable. Finally we presented a particle filter based tracking algorithm that utilizes the described features for a robust recognition of road networks.

Some visual impressions of the features and the presented tracking system at night can be seen at www.mucar3.de/iros2015-roadtracking.

B. Future Work

The modular weighting step of the particle filter can be extended easily by new features. The basis of a possible feature extension is given by the measured reflectivity values of the laser scanner. Since our focus is in less structured regions without any artificial boundary markings, we expect different surface reflectivities between road and (for example) grass, but no extreme difference as between paved road and its white lane markings.

Acknowledgments

The authors gratefully acknowledge funding by the Federal Office of Bundeswehr Equipment, Information Technology and In-Service Support (BAAINBw).

References

[1] T. Kühnl, F. Kummert, and J. Fritsch, “Spatial ray features for real-time ego-lane extraction,” in *Proceedings of IEEE Intelligent Transportation Systems (ITSC)*, Anchorage, AK, USA, Sept. 2012, pp. 288 – 293.

[2] G. B. Vitor, A. C. Victorino, and J. V. Ferreira, “A probabilistic distribution approach for the classification of urban roads in complex environments,” in *Workshop on IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, May 2014, pp. 1–6.

[3] J. Beck and C. Stiller, “Non-parametric lane estimation in urban environments,” in *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Dearborn, MI, USA, June 2014, pp. 43 – 48.

[4] M. Manz, M. Himmelsbach, T. Luettel, and H.-J. Wuensche, “Detection and Tracking of Road Networks in Rural Terrain By Fusing Vision and LIDAR,” in *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, San Francisco, CA, USA, Sept. 2011, pp. 4562–4568.

[5] E. Dickmanns and A. Zapp, “A Curvature-based Scheme for Improving Road Vehicle Guidance by Computer Vision,” in *SPIE Conference on Mobile Robots*, Cambridge, MA, USA, Oct. 1986.

[6] C. Jung and C. Kelber, “Lane following and lane departure using a linear-parabolic model,” in *Proceedings of IEEE Intelligent Vehicles Symposium*, Nov. 2005, pp. 1192 – 1202.

[7] H.Loose and U.Franke, “B-Spline Based Road Model for 3D Lane Recognition,” in *Proceedings of IEEE Intelligent Transportation Systems Conference*, Madeira, Spain, Sept. 2010.

[8] U. Franke, H. Loose, and C. Knoppel, “Lane Recognition on Country Roads,” in *Proceedings of IEEE Intelligent Vehicles Symposium*, Istanbul, Turkey, June 2007.

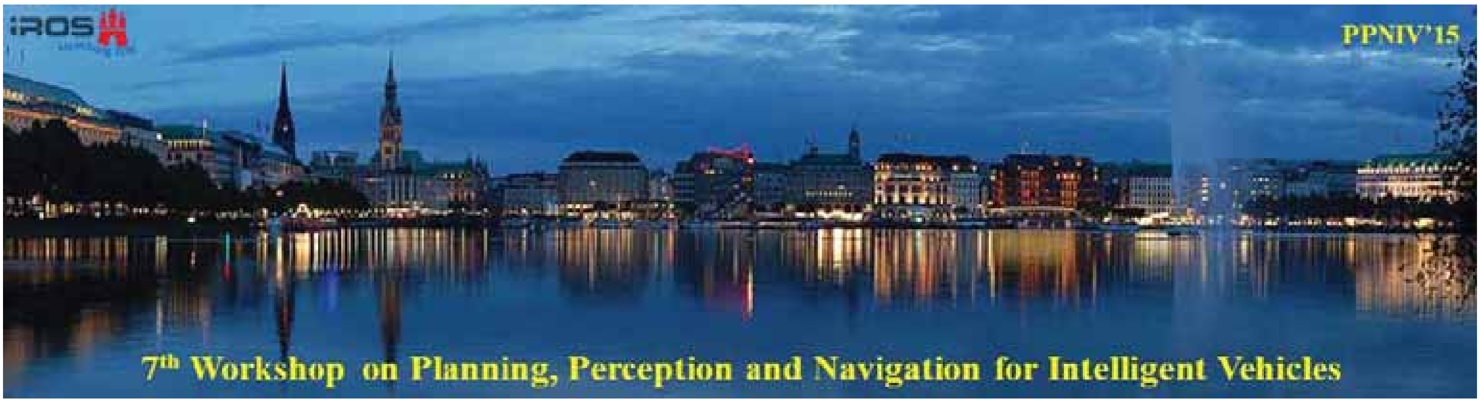
[9] M. Manz, M. Himmelsbach, T. Luettel, and H.-J. Wuensche, “Fusing LIDAR and Vision for Autonomous Dirt Road Following – Incorporating a visual feature into the tentacles approach,” in *Proceedings of 21. Fachgespraech Autonome Mobile Systeme (AMS)*, Karlsruhe, Germany, Dec. 2009.

[10] M. Serfling, R. Schweiger, and W. Ritter, “Road course estimation in a night vision application using a digital map, a camera sensor and a prototypical imaging radar system,” in *Proceedings of IEEE Intelligent Vehicles Symposium*, Eindhoven, The Netherlands, Nov. 2008.

[11] F. Schüle, R. Schweiger, and K. Dietmayer, “Augmenting Night Vision Video Images with Longer Distance Road Course Information,” in *Proceedings of IEEE Intelligent Vehicles Symposium*, Gold Coast, Australia, June 2013.

[12] O. Hartmann, R. Schweiger, R. Wagner, F. Schüle, M. Gabb, and K. Dietmayer, “Night Time Road Curvature Estimation based on Convolutional Neural Networks,” in *Proceedings of IEEE Intelligent Vehicles Symposium*, Gold Coast, Australia, June 2013.

[13] S. F. X. Bayerl and H.-J. Wuensche, “Detection and Tracking of Rural Crossroads Combining Vision and LiDAR Measurements,” in *Proceedings of IEEE Intelligent Transportation Systems Conference*, Qingdao, China, Oct. 2014.


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session III

Interactive session

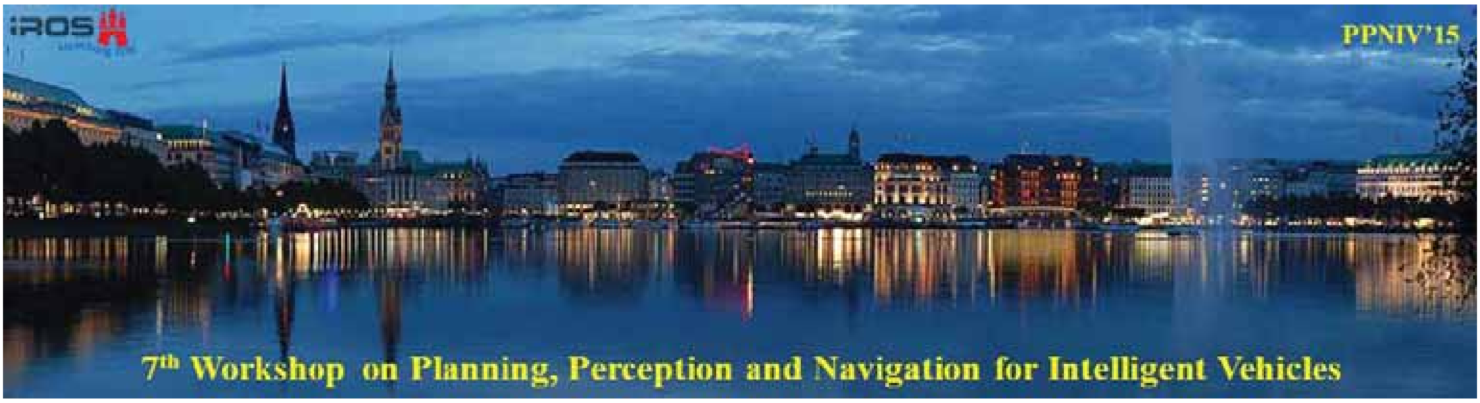
- **Title: Generating Compact Models for Traffic Scenarios to Estimate Driver Behavior Using Semantic Reasoning**
  Authors: Ilya Dianov, Karinne Ramirez-Amaro and Gordon Cheng

- **Title: 16 channels Velodyne versus planar LiDARs based perception system for Large Scale 2D-SLAM**
  Authors: Nobili S. Dominguez S. Garcia G. Martinet P.

- **Title: Discriminative Map Matching Using View Dependent Map Descriptor**
  Authors: Liu Enfu Tanaka Kanji

- **Title: Path Planning and Steering Control for an Automatic Perpendicular Parking Assist System**
  Authors: Plamen Petrov Fawzi Nashashibi and Mohamed Marouf

- **Title: Obstacle segmentation with low-density disparity maps**
  Authors: Daniela A. Ridel, Patrick Y. Shinzato and Denis F. Wolf


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Generating Compact Models for Traffic Scenarios to Estimate Driver Behavior Using Semantic Reasoning

Ilya Dianov, Karinne Ramirez-Amaro and Gordon Cheng

Abstract— **Driving through a constantly changing environment is one of the main challenges of autonomous driving. To navigate successfully, the vehicle should be able to handle a variety of possible situations on the road by constantly analyzing the traffic environment and determine which objects might influence its current behavior. This paper presents an artificial intelligence method to improve the perception and situation awareness of autonomous vehicles by detecting and extracting meaningful information from different traffic scenarios, and inferring the correct driving behavior for each of them. Our method uses a state of the art technique based on semantic reasoning previously used for recognizing human activities in cooking scenarios. This algorithm has been adapted and extended to the automotive domain by introducing new object properties such as *ObjectInFront*, *ObjectActedOn*, *MoveForward*, *Turn*. The main advantage of our proposed method is its adaptability to different mobile domains without any additional training. First, our system is trained on traffic situations. The obtained semantic models are later used to autonomously navigate a mobile robot in an indoor environment by utilizing the acquired knowledge and inference from the automotive domain. The results show that the overall positive classification rate for traffic scenarios recognition is 90.14% of the cases. In addition, the average processing and behavior generation time for the implemented system is 0.177 seconds, which allows the mobile robot to react online to the newly encountered situations.**

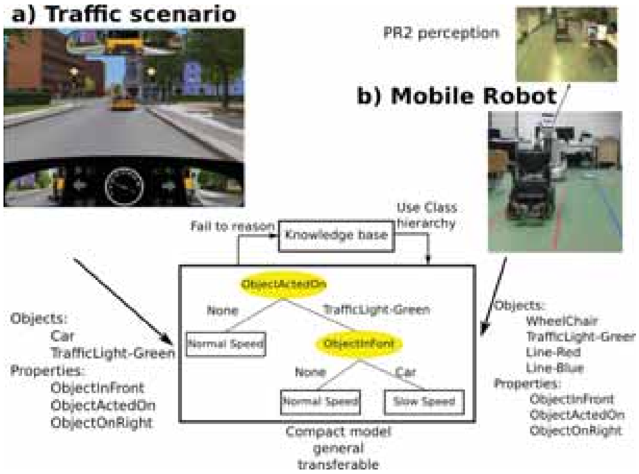


Fig. 1: General overview of our system which shows two examples where we reuse the learned semantic models: a) shows an example of the new traffic scenario and b) depicts an example of the domain transference of learned model[1].

I. INTRODUCTION

Recent developments in autonomous driving show that autonomous vehicles can bring a lot of benefits to our society e.g. reduction of traffic incidents, increased mobility for the elderly and disabled people, more efficient traffic flow, reduction of fuel consumption and many more [1]. In order to achieve that, these vehicles have to overcome many problems such as moving in a dynamic environment, processing vast amount of data from different sensors, handling driving rules, path planning and collision avoidance [2]. One of the main challenges is understanding the encountered traffic situations to estimate the proper driving behavior by considering the meaningful traffic participants and relations between them. However, finding this meaningful information represents another set of challenges such as complexity of the perceived environment, which leads to problems of having partial observable information. Typically, to solve this problem it is necessary to implement a sophisticated method which can recognize, analyze and extract contextual information about each scenario, learn and use it to determine vehicle behaviors when this scenario is encountered again [3]. The extracted contextual data is represented as spatio-temporal relationships between the ego vehicle and traffic environment [4]. This context information can be obtained from the generation of dynamic maps [5] which will enhance the tracking of the objects around the vehicle. However, there are a vast amount of objects which can influence the vehicle motion depending on their behavior, and learning all possible traffic scenarios involving all these different objects would be computationally expensive in terms of execution and memory. Thus, semantic representation and knowledge methods can greatly improve such systems.

The goal of this paper is to create a simple and general ontology model which can be used to describe different scenarios in mobile domains. For this reason, we propose a method that can automatically recognize and extract spatial relationships from such scenarios using a simple perception system and create a reasoning mechanism which can infer appropriate vehicle behavior for each situation. Fig. 1 shows an example of our proposed system, exemplifying how the model of the same behavior can be applied to two different scenarios. First, the compact general model was created for the input scenario involving the yellow car which is not moving in front of the vehicle and the traffic light is showing a green light. To avoid collision the vehicle has to slow

Faculty of Electrical Engineering, Institute for Cognitive Systems, Technical University of Munich, Germany `ilya.dianov@tum.de`, `karinne.ramirez@tum.de` and `gordon@tum.de`

[1] The traffic scenario image is a courtesy of Verlag Heinrich Vogel. www.fahren-lernen.de

down even if the traffic light is green. A similar situation was simulated using the PR2 robot: the monitor represents the traffic light and there is a wheelchair standing in front of the robot path. Using such knowledge our system can detect that both car and wheelchair are mobile entities therefore both will infer the same behavior: slow speed.

This paper is organized as follows: Section II describes related work. Section III introduces the framework consisting of the object properties, semantic rules and the ontology. Section IV shows the application of the generated framework to mobile robot domain, and Section V presents conclusion and an outlook on possible future work.

II. RELATED WORK

There are multiple approaches for analyzing and estimating traffic scenarios. The method presented in [6] models scenarios as a state space, containing information about the vehicle, properties of its surroundings, possible behaviors and trajectories of the vehicles using Dynamic Bayesian Network (DBN) to predict the driver's behavior in its current scenario. A similar approach by Agamennoni et al. [7] introduces feature functions to characterize dynamic relationships between traffic participants to form context models which are used by DBN to predict states of all objects influencing vehicle behaviors. In contrast, [8] introduce evidential grids which utilizes geographic information from digital maps to detect navigable space for the vehicles. However, this approach is used only for obstacle avoidance and do not incorporate any data about the road infrastructure (for example traffic signs). Another method is described in [9] where a tree-like hierarchy of classifiers is introduced. Each node in the hierarchy is predefined and predicts only one certain property of an input object and can activate a child node if a specific output is predicted. In [10] a Bayesian network is used to find impact of traffic situations to each participant and predict their behavior, where participants are represented as nodes connected with conditional distribution functions. However all these approaches require a complicated perception system, large amount of samples for each driving situation and do not extrapolate and exploit semantic relations for prediction use.

Another group of methods for analyzing traffic situations are based on description logic. An ontology can be used to describe road intersections by mapping atomic concepts to specific geometric primitives [11]. Another way of using an ontology is to represent lanes and vehicles moving on them as a graph like network and detect conflicts between traffic participants at the same intersection [12]. To analyze more complex situations an ontology can be represented as a knowledge base with hierarchical structure consisting of atomic concepts and relations between them [13]. This method can be extended to find dependencies in interaction of the traffic participants to infer their likely behavior in current situations [4]. However, the described methods are using manually generated rules for reasoning, which are very specific to each situation. Moreover, their ontology representations are created for driving scenarios and cannot be applied to any other mobile domains without significant changes.

Conversely, our method provides a framework for autonomous learning semantic rules from sensory data which together with the ontology, allows to transfer knowledge from the traffic domain to other mobile domains.

III. SEMANTIC REASONING AND KNOWLEDGE REPRESENTATION

The overview of the created framework is shown in Fig. 2. The contextual information perceived by the vehicle is processed to detect objects and their spatial properties and then stored in the ontology. The reasoning module utilizes knowledge from the ontology to recognise traffic scenarios and uses semantic rules to infer driving behavior with respect to the road context (traffic rules and objects in driving environment). The semantic rules are generated using the decision tree classifier trained on the driving tests obtained from an online driving school[2].

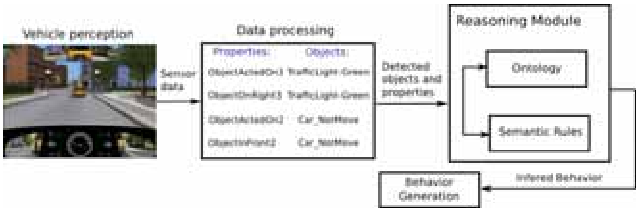


Fig. 2: Framework overview[3].

A. Identification of Object Properties

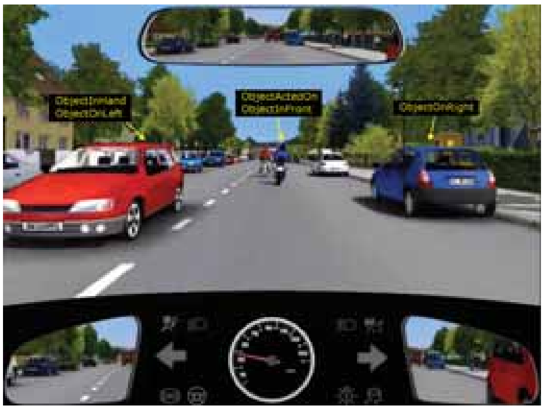


Fig. 3: Example of detecting properties from traffic situation, the input video was obtained from the online driving school[3].

To analyze different driving scenarios the video tests described above were used. Each video was manually annotated, and we found that for each traffic participant influencing the behavior of the vehicle at the current time point there is a set of common properties which are always applicable regardless of the participant type. These properties

[2] www.fahren-lernen.de

[3] The traffic scenario image is a courtesy of Verlag Heinrich Vogel. www.fahren-lernen.de

evaluates spatial relationships between the participants and the ego vehicle. Each traffic participant is represented as an abstract object and the following properties are defined:

1) *ObjectInHand*: the object is very close to the ego vehicle and can cause a collision. In Fig. 3 the red car is considered to be *ObjectInHand*, because if the vehicle would turn left it will crash with that car.
2) *ObjectActedOn*: the object is in the range of interest of the ego vehicle and might require a certain action, but neglecting the execution of this action will not lead to a collision. In Fig. 3 both motorcycles are considered to be *ObjectActedOn* because normally the vehicle should reduce speed to keep proper distance, but even if it maintain current speed there will be no collision until the vehicle would reach the motorcycles (but then they become *ObjectInHand*).

The above properties were inspired by the similar ones defined in [19] and readapted to driving scenarios. At any particular time point the object can have only one of these two properties (either *ObjectInHand* or *ObjectActedOn*) but not both of them. Additionally, we define the following properties:

3) *ObjectInFront*: the object is in front of the current driving path of the ego vehicle. In Fig. 3 both motorcycles are *ObjectInFront* because both of them are moving on the same lane as the ego vehicle.
4) *ObjectOnLeft*: the object is on the left possible path of the ego vehicle. In Fig. 3 the red car is *ObjectOnLeft* because it is moving on the lane which is to the left of the current driving path of the ego vehicle.
5) *ObjectOnRight*: the object is on the right possible path of the ego vehicle. In Fig. 3 the blue car is *ObjectOnRight* because even if there is no separation line, the car is parked on a different lane according to driving rules, and this lane is to the right of the current driving path of the ego vehicle.

For automatic detection of the above *Object Properties* we define the formulas shown in Table I. Where $l = \sqrt{x^2 + z^2}$ is the distance to the object, r_1 and r_2 are distance thresholds, x_{obj}, x_{left} and x_{right} are x coordinates of the Cartesian position of the object , left boundary and right boundary of the driving tube respectively. The framework do not require global positions of the objects or the ego vehicle for property detection and utilize only their local positions with respect to the ego vehicle. Fig. 4 shows an example of properties detection, where the blue dot represents the ego vehicle:

- The $obstacle_1$ (green dot) is detected as *ObjectInHand*.
- The $obstacle_3$ (yellow dot) is *ObjectActedOn*.
- The $obstacle_1$ (green dot) is *ObjectInFront*.
- The $obstacle_2$ (purple dot) is *ObjectOnLeft*.
- The $obstacle_3$ (yellow dot) is *ObjectOnRight*.

TABLE I: Definition of *Object Properties*.

Name	Formula
ObjectInHand	$l < r_1$
ObjectActedOn	$r_1 < l < r_2$
ObjectInFront	$x_{left} < x_{obj} < x_{right}$
ObjectOnLeft	$x_{obj} < x_{left}$
ObjectOnRight	$x_{obj} > x_{right}$

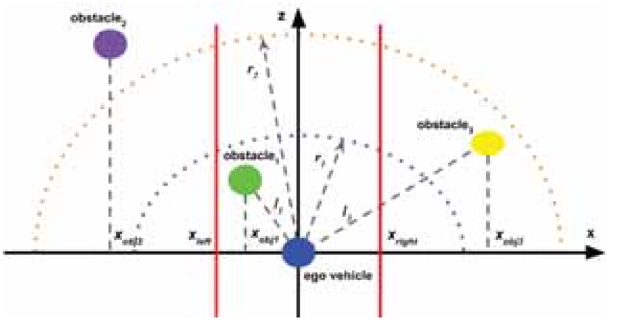


Fig. 4: Graphical Example of the *Object Properties* detection.

In addition, we identified that traffic participants of the same type can have different properties which we called *Instance Properties*. For the data obtained from the video tests we define the following *Instance Properties*:

- MoveToward: the object is moving towards the vehicle.
- MoveForward: the object is moving forward the vehicle.
- NotMove: the object is not moving.
- ChangingLane: the object is changing lane.
- Crossing: the object is crossing the path of the vehicle.
- Turn: the object is turning.

It is always possible to add additional *Instance Properties* to generate a more accurate model of traffic scenarios.

B. Semantic Rules

In order to map the *Object properties* to the ego vehicle driving activities, a decision tree classifier was build similar to [20]. The training data sample contains the current *Object Properties*:

1) *ObjectInHand (None, Something)*
2) *ObjectActedOn (None, Something)*
3) *ObjectInFront (None, Something)*
4) *ObjectOnLeft (None, Something)*
5) *ObjectOnRight (None, Something)*

where *Something* represents an object with certain instance property (for example *Vehicle_MoveToward*) and *None* is used if there is no object with that particular *Object Property*. And a target concept value which describes a current vehicle behavior:

$$Class : VehicleActivity\{NormalSpeed, SlowSpeed, NormalStop, LaneChange\}.$$

Here is an example of a training sample:

$$\{None, Vehicle\_MoveToward, None, Vehicle\_MoveToward, None, NormalSpeed\}.$$

It is possible to have two separate sets of *Object Properties* for different *Classes* in order to describe the situation in more detail. For example, the class *Vehicle* can have the properties:

$$\{ObjectInHand1, ObjectActedOn1, ObjectInFront1, ObjectOnLeft1, ObjectOnRight1\}$$

where 1 indicates that this is a first set of properties. While, the class *Pedestrian* can have a different set of properties:

$$\{ObjectInHand2, ObjectActedOn2, ObjectInFront2, ObjectOnLeft2, ObjectOnRight2\}.$$

where 2 indicates that this is a second set of properties. In this case, the training sample should contain combination of both properties, which means that each ego vehicle activity can be represented by several combinations of *Object properties* (in contrast to [20]). To learn a target concept value from the data samples we trained a decision tree classifier based on the C4.5 algorithm [14]. The information gain is defined as follows:

$$Gain(S,P) = Entropy(S) - \sum_{v \in Values(P)} \frac{|S_v|}{S} Entropy(S_v)$$

where $Values(P)$ is the set of all possible values of the *Object properties*, and $S_v = s \in S | P(s) = v$.

C. Ontology Model

The formal definition of an ontology is ”an explicit specification of a conceptualization” [15], in other words it is an unambiguous representation of the knowledge about a certain domain. An ontology usually consists of terms organized in hierarchical structure and relationships between those terms. Our proposed ontology was built as a knowledge base using Knowrob [16] and represented in the Web Ontology Language (OWL) [17]. It consists of the terminological box (*TBox*) and the assertional box (*ABox*) [18] .

1) *Tbox* describes concepts in the ontology, which are usually called *Classes*. Each concept has a set of properties called attributes. Relations between concepts are represented by rules and axioms.
2) *Abox* describes instances of concepts.

The terminology box consists of:

1) *Classes* which represent different types of objects detected from the traffic environment.
2) *Object Properties*: properties described previously, which are common for all classes or for a specific class.
3) *Instance Properties*: these properties are defined for each object, and two objects belonging to same class can have different instance properties.

The assertional box was defined as following:

- *Abox* consists of instances which belong to classes defined in *Tbox*. All perceived objects are contained in *Abox* and if any of the *Object Properties* are held, their instances are created and placed in *Abox*.

To implement the obtained semantic rules in the Reasoning module and connect them with the ontology the Knowrob *Computable Classes* were used. The *Computable* defines a semantic relation between instances of classes representing possible behavior of the ego vehicle and *Object Properties*. For example:

```
**if** $(o_i = Vehicle)$ & $ObjectInHand2(o_i)$ & $ObjectInFront2(o_i)$ & $(NotMove(o_i))$ **then**
    $NormalStop$
**end if**
```

where o_i is an object detected in the traffic environment. The *Computables* were implemented using Prolog, which provides a useful functionality for ontology description, knowledge inference, searching and pattern matching.

IV. EXPERIMENT

To prove that our proposed methodology is working in different mobile domains, first the framework described in Section III was applied to create the semantic rules for traffic scenarios. Next, to test the robustness of our system in the real environment we used it to navigate PR2 mobile robot. The robot environment contains unknown objects and we will demonstrate how without any additional training the PR2 can utilize the system build for traffic scenarios and generate correct behavior by taking advantage of knowledge and inference.

A. Semantic Tree Training

TABLE II: Confusion matrix of the recognized driving behavior.

		Predicted Class			
		NormalSpeed	NormalStop	SlowSpeed	LaneChange
Actual Class	NormalSpeed	94.4%	0	5.6%	0
	NormalStop	0	98.5%	1.5%	0
	SlowSpeed	3.6%	0	96.4%	0
	LaneChange	0	0	0	100%

For training we used 14 video samples of the driving tests (see Section III). Each sample had a length of 15 seconds and contained 150 video frames. The video tests consist of different complex traffic scenarios involving multiple road participants, traffic signs and pedestrians. In the first experiment, each frame of the input video was manually annotated to obtain training samples, containing a sequence of detected objects and its properties as well as the class of the recognized driving behavior. A decision tree was generated in the Weka data mining system [21] and was tested on a 3 new video samples of the driving tests containing previously untrained traffic scenarios. The resulting classification rate was 94.6%, and the confusion matrix is shown in Table II. The partial decision tree is shown on Fig. 5, where each driving situation is identified by a set of specific properties and instances, which allows a compact and general representation.

By analyzing the obtained results, we concluded that segmenting each frame of the video was redundant, and instead it is better to extract only 5 samples for each traffic situation recognized in the input video. To prove this theory, a second experiment was conducted using the same video samples for training and testing as before, and the resulting classification

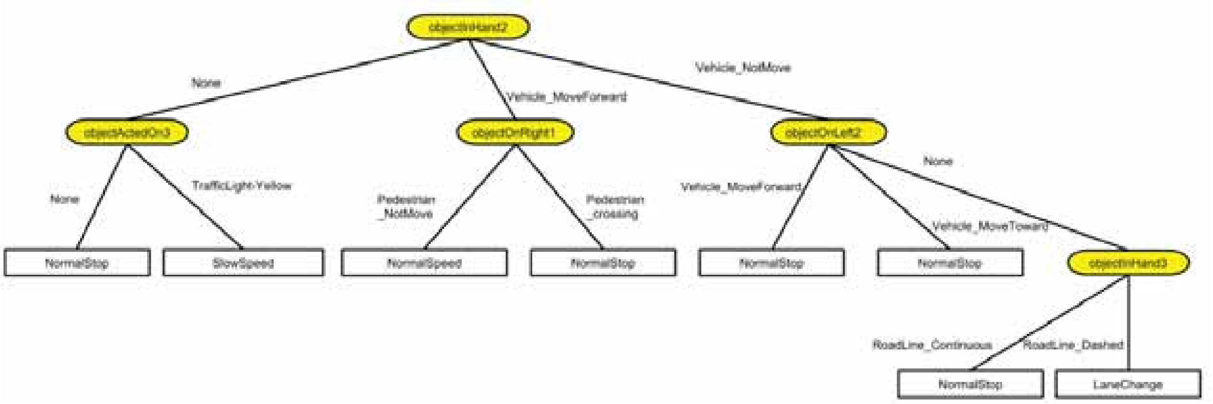


Fig. 5: Partial decision tree obtained from the first experiment instance.

rate was 90.14%. The confusion matrix generated for the new setup is shown in Table III. The resulting decision tree remained the same as the one obtained in the first experiment, which clearly indicates that our proposed method do not require large amount of data for training.

TABLE III: Confusion matrix for the second experiment with less training samples.

		Predicted Class			
		NormalSpeed	NormalStop	SlowSpeed	LaneChange
Actual Class	NormalSpeed	92.5%	0	7.5%	0
	NormalStop	0	97.8%	2.2%	0
	SlowSpeed	6.8%	0%	93.2%	0
	LaneChange	0	0	0	100%

B. Integration with the Robotics Domain

The framework was integrated with a PR2 mobile robot and the overview of the created system is shown on Fig. 6. The robot behavior is generated reactively based on the perception data obtained from the robot camera.

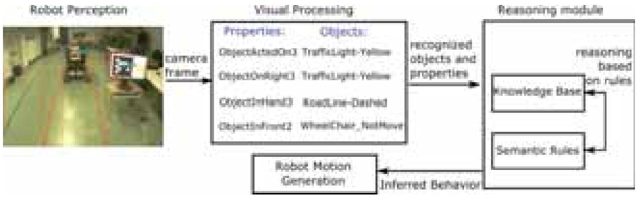


Fig. 6: Overview of the system for automatic recognition of driving scenarios implemented on the PR2.

First, each image frame obtained from the robot right camera is processed to detect existing objects and their properties. The realistic detection of objects and pedestrians is out of the scope of this paper, that is why the visual processing module was implemented using the OpenCV and the aruco library [22], which allows to detect AR markers and obtain their 3D position and orientation by using only one camera. Additionally, the color based detection was used to recognize lines of the driving tube in the testing area. For evaluation only positions in 2D space were used, because the robot and most objects it encounters cannot move in vertical direction. When objects and properties are obtained, the system uses the Reasoning module described in Section III to infer the correct behavior in a current situation (Fig. 6). To prove that the robot can reuse the semantic rules obtained from the decision tree trained on traffic situations the testing scenario shown on Fig. 7 was created. The robot can move inside the lane formed by the red lines, or by the middle red and blue lines. The middle red line represents a dashed road line, and the blue one represents a continuous road line. On his path PR2 encounters the wheelchair which is not moving and has the property *ObjectInHand2*. Moreover, the robot always perceives the dashed road line which can be crossed for overtaking and has property the *ObjectInHand3*. Using the obtained data from the tree shown in Fig. 5, the following *Computable* will be called[4]:

```
if $ObjectInHand2(o_i)$ & $ObjectInFront2(o_i)$ & $(NotMove(o_i)$ & $(o_i = Vehicle)$ & $ObjectInHand3(o_j)$ & $(o_j = RoadLine\_Dashed)$ then
    $LaneChange$
end if
```

where o_i and o_j are objects detected in traffic environment. However, the object o_i with the *ObjectInHand3* property is not the *Vehicle* but the *WheelChair* which means that the direct execution of the *Computable* would fail and additional information from the ontology will be requested. The ontology will infer the class hierarchy of the object o_i and detect that the class *WheelChair* is a subclass of *Vehicle* :

$$o_i \sqsubseteq WheelChair \sqsubset LightVehicle \sqsubset Vehicle$$

Consequently, the computable defined above will be executed and the *LaneChange* behavior will be generated.

The performance of the system is shown in Table IV which clearly indicates that the average reaction time for each perceived situation is 0.177 seconds[5]. This means that

[4] Please note that this is a simplified example of the Prolog implementation.

[5] Our system was implemented on a computer with the Intel(R) Core(TM) i5 CPU 750 2.67 GHz and 4GB memory.

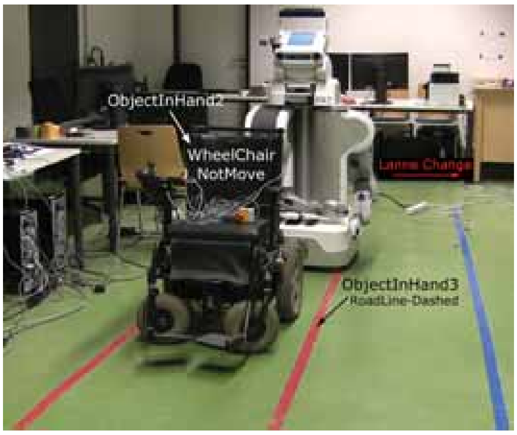


Fig. 7: Overview of the experimental setup.

the mobile robot, i.e. PR2, takes around 0.177 seconds to make a decision. It is possible that the inference time will improve when a faster perception system is used.

TABLE IV: Performance for each input frame obtained from the PR2 camera.

Type of the operation	Average execution time in sec
Image processing and object detection	0.0685529
Properties detection	0.0166563
Behavior inference	0.0917922
Total	**0.1770014**

V. CONCLUSIONS

In this paper a framework for recognition and extraction of driving situations using an artificial intelligence method was presented. This framework improves perception and situation awareness of autonomous vehicles in dynamic environments and creates general compact models of different traffic scenarios which are used to reason on road contexts. First, we trained our system in the traffic domain, and with the obtained model we tested new unknown scenarios obtaining a classification rate of 90%. Next, to demonstrate the robustness of our method we tested it on the new environment with the PR2 mobile robot. The robot reused the framework obtained from the first experiment to recognise new and previously untrained situations by taking advantage of the proposed ontology and inference methods. This framework allowed the robot to navigate successfully in an indoor environment with an average reaction time of 0.177 seconds. The results show that knowledge and semantic reasoning allows to apply our framework to different mobile domains by using the same semantic rules to enable mobile agents to correctly infer driving behaviors without additional training.

The methodology was implemented using a simple perception system, and though it used only a frontal camera, the system can be easily extended for full range detection using different types of sensors or data from dynamic maps, which will require only one additional property for objects at the back of the vehicle.

REFERENCES

[1] Beiker, S. A., ”Legal aspects of autonomous driving.” Santa Clara L. Rev., vol.52, no.4, pp.1145-1156, 2012.

[2] Aeberhard, M.; Rauch, S.; Bahram, M.; Tanzmeister, G.; Thomas, J.; Pilat, Y.; Homm, F.; Huber, W.; Kaempchen, N., ”Experience, Results and Lessons Learned from Automated Driving on Germany’s Highways.” Intelligent Transportation Systems Magazine, IEEE , vol.7, no.1, pp.42-57, Spring 2015.

[3] Armand, A.; Filliat, D.; Ibanez-Guzman, J., ”Detection of unusual behaviours for estimation of context awareness at road intersections”. In in Proceedings of the 5th Planning, Perception and Navigation for Intelligent Vehicles Workshop (IROS 2013), 2013.

[4] Armand, A.; Filliat, D.; Ibanez-Guzman, J., ”Ontology-based context awareness for driving assistance systems.” Intelligent Vehicles Symposium Proceedings, 2014 IEEE , vol.227, no.233, pp.8-11, June 2014.

[5] Florez, S. A. R.; Frmont, V.; Bonnifait, P.; Cherfaoui, V., ”Multi-modal object detection and localization for high integrity driving assistance.” Machine Vision and Applications, vol.25, no.3, pp.583-598, 2014.

[6] Gindele, T.; Brechtel, S.; Dillmann, R., ”A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments.” Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on , vol.1625, no.1631, pp.19-22, Sept. 2010.

[7] Agamennoni, G.; Nieto, J.I.; Nebot, E.M., ”Estimation of Multivehicle Dynamics by Considering Contextual Information.” Robotics, IEEE Transactions on , vol.28, no.4, pp.855,870, Aug. 2012.

[8] Kurdej, M.; Moras, J.; Bonnifait, Ph.; Cherfaoui, V., ”Map-Aided Evidential Grids for Driving Scene Understanding.” IEEE Intelligent Transportation Systems Magazine, vol.7, no.1, pp.30-41 Jan. 2015.

[9] Bonnin, S.; Kummert, F.; Schmudderich, J., ”A Generic Concept of a System for Predicting Driving Behaviors.” Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on , vol.1803, no.1808, pp.16-19, Sept. 2012.

[10] Gindele, T.; Brechtel, S.; Dillmann, R., ”Learning Driver Behavior Models from Traffic Observations for Decision Making and Planning.” Intelligent Transportation Systems Magazine, IEEE , vol.7, no.1, pp.69,79, Spring 2015.

[11] Hummel, B.; Thiemann, W.; Lulcheva, I., ”Scene understanding of urban road intersections with description logic.” Logic and Probability for Scene Interpretation, 8091, 2008.

[12] Regele, R., ”Using Ontology-Based Traffic Models for More Efficient Decision Making of Autonomous Vehicles.” Autonomic and Autonomous Systems, 2008. ICAS 2008. Fourth International Conference on , vol.94, no.99, pp.16-21, March 2008.

[13] Hulsen, M.; Zollner, J.M.; Weiss, C., ”Traffic intersection situation description ontology for advanced driver assistance.” Intelligent Vehicles Symposium (IV), 2011 IEEE , vol.993, no.999, pp.5-9, June 2011.

[14] R. Quinlan, C4.5: Programs for Machine Learning. San Mateo, CA:Morgan Kaufmann Publishers, 1993.

[15] Gruber, T. R., ”A translation approach to portable ontology specifications.” Knowledge acquisition, vol.5, no.2, pp.199-220, 1993.

[16] Tenorth, M.; Beetz, M., ”KnowRob: A knowledge processing infrastructure for cognition-enabled robots.” The International Journal of Robotics Research, vol.32, no.5, pp.566-590, 2013.

[17] Dean, M.; Schreiber, G.; Bechhofer, S.; van Harmelen, F.; Hendler, J.; Horrocks, I.; Stein, L. A.,” OWL web ontology language reference.” W3C Recommendation February, 10, 2004.

[18] De Giacomo, G.; Lenzerini, M., ”TBox and ABox reasoning in expressive description logics.” KR, vol.96, pp.316-327, 1996.

[19] Ramirez-Amaro, K.; Beetz, M.; Cheng, G., ”Understanding the Intention of Human Activities through Semantic Perception: Observation, Understanding and Execution on a Humanoid Robot.” Advanced Robotics Journal, vol. 29, issue 5, 2015.

[20] Ramirez-Amaro, K.; Beetz, M.; Cheng, G., ”Automatic segmentation and recognition of human activities from observation based on semantic reasoning.” IROS 2014, 2014 IEEE/RSJ International Conference on , vol.5043, no.5048, pp.14-18, Sept. 2014.

[21] Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; Witten, I. H., ”The WEKA data mining software: an update.” ACM SIGKDD explorations newsletter, vol.11, no.1, pp.10-18, 2009.

[22] Garrido-Jurado, S.; Muoz-Salinas, R.; Madrid-Cuevas, F. J.; Marn-Jimnez, M. J., ”Automatic generation and detection of highly reliable fiducial markers under occlusion.” Pattern Recognition, vol.47, no.6, pp.2280-2292, 2014.

16 channels Velodyne versus planar LiDARs based perception system for Large Scale 2D-SLAM

Nobili S.[1] Dominguez S.[2] Garcia G.[3] Philippe M.[4]

Abstract— **The ability of self-localization is a basic requirement for an autonomous vehicle, and a prior reconstruction of the environment is usually needed. This paper analyses the performances of two typical hardware architectures that we evaluate in our 2D Simultaneous Localization and Mapping (2D-SLAM) system for large scale scenarios. In particular, the selected configurations are supposed to guarantee the possibility of integrating at a later stage mobile objects tracking capabilities without modifying the hardware architecture. The choice of the perception system plays a vital role for building a reliable and simple architecture for SLAM. Therefore we analyse two common configurations: one based on three planar LiDARs Sick LMS151 and the other based on a Velodyne 3D LiDAR VLP-16. For each of the architectures we identify advantages and drawbacks related to system installation, calibration complexity and robustness, quantifying their respective accuracy for localization purposes. The conclusions obtained tip the balance to the side of using a Velodyne-like sensor facilitating the process of hardware implementation, keeping a lower cost and without compromising the accuracy of the localization. From the point of view of perception, additional advantages arise from the fact of having 3D information available on the system for other purposes.**

I. INTRODUCTION

An efficient and accurate solution to the Simultaneous Localization and Mapping (SLAM) problem is the basic building block for an autonomously navigating platform. The perception system employed in the architecture plays a fundamental role in determining the quality of the performance. Indeed, the reliability of the map and consequently the accuracy of the localization are highly dependent on the measurements provided by the local sensors. Nowadays there are several options when choosing the hardware architecture that allows us to apply SLAM for positioning our vehicle. The question arises when we have to choose which hardware set-up is the most appropriate for our application given some constraints about budget, ease of installation, precision, reliability against changing environment, versatility, etc. Most commonly used architectures employ Light Detection and Ranging (LiDAR) technology measuring at different angles the distance to the surrounding environment. The SLAM problem is well-known for its increasing complexity in terms of accuracy, runtime and computational resources required while covering long outdoor distances and mapping. We regard these aspects as particularly relevant for precise localization of memory-restricted systems and we propose a multi-map LiDAR-based 2D-SLAM solution. We implemented an extended version of the *GMapping* Robot Operating System (ROS) package. In particular, we adapted it and integrated it into our architecture for being used in Large Scale multi-map 2D SLAM. Our version allows to start building a new map when required, saving previously the map under construction. A map-manager is in charge of deciding when a map must be stored and start building a new one. Later, during the localization phase, the sub-maps are loaded as they are required along the pre-recorded journey and Monte Carlo localization techniques using a probabilistic particle filter are applied to find the most likely position given the map, laser scan and odometry measurements. *Connection points* connect a sub-map with its neighbour and delimit when a sub-map ends and a new one starts. The local reference frame of a sub-map is normally positioned on a *connection point* (See figure 1).

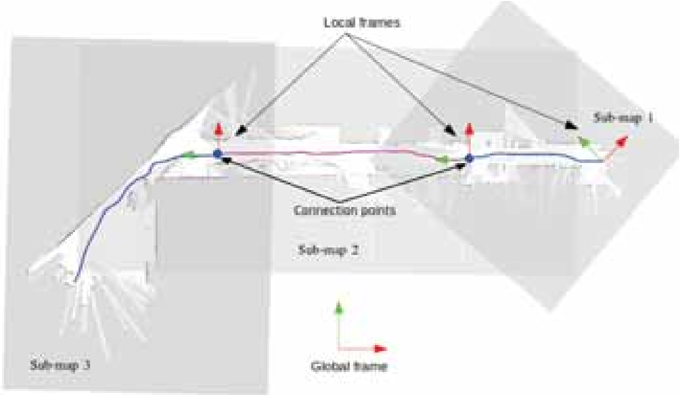


Fig. 1. We represent the vehicle's path (from right to left) in a chain of sub-maps. Each submap is connected to the previous and the next one through *connection points*.

In this paper, we present experimental results obtained in a urban context using two distinct laser-based hardware architectures typically used in SLAM but the same software for localization. In Table I, we summarize the main features of both types of sensors. Our main contribution focuses on a comparison between the performances of the two systems in terms of map quality, localization accuracy and robustness to temporarily static elements like parked vehicles.

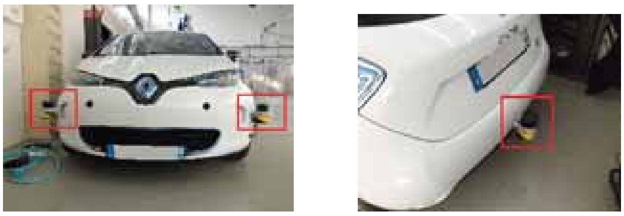
The first experimental platform is an electric car *Renault Zoe ZE* equipped with three LiDARs (SICK LMS 151) placed at 50 cm from the ground level in a configuration

[1] [2] [4] Nobili S., Dominguez S. and Philippe M. are with IRCCyN, Institut de Recherche en Communication et Cybernétique de Nantes, École Centrale de Nantes, 1 rue de la Noë, 44321 Nantes, France

[3] Garcia G. is with ECN École Centrale de Nantes, 1 rue de la Noë, 44321 Nantes, France

[1] simona.nobili@eleves.ec-nantes.fr
[2] salvador.dominguez@irccyn.ec-nantes.fr
[2] Gaetan.Garcia@ec-nantes.fr
[3] philippe.martinet@irccyn.ec-nantes.fr

that guarantees a 360° Field of View (FoV) about the car vertical axis. Specifically, two of the LiDARs are mounted on the two front corners of the car in order to cover straight and side views, whereas the third one covers the back side view (See figure 2 and 4). An extrinsic calibration process is performed to ensure that all the three LiDARs lie on a plane as closed as possible to a common horizontal plane such that the scans can be merged and given as input measurement to the SLAM solver.



Sick LMS151 installed on the front

Sick LMS151 on the rear side

Fig. 2. The LiDARs Sick LMS151 are installed in strategical positions to ensure a 360°FoV around the vehicle.

The second experimental platform is a vehicle *Renault Fluence* equipped with a 16 channels Velodyne LiDAR PUCK (VLP-16) placed some centimeters above the roof surface and scanning 360° about the car's vertical axis (see Figure 3). This sensor is a 360° revolving-head 3D LiDAR with 16 laser beams vertically separated along a range of 30° with 2° of angular resolution. The ith laser, after a full rotation, sweeps a cone in the 3D space. In this context, we take advantage of the vertical FoV (from -15° to 15° with respect to the sensor reference frame) to infer a 2D laser-scan information merging data belonging to a vertical range between 1.8 and 2.8 meters from the ground. In this way the scan measurements will not be influenced by the most common moving elements, which in a urban context are assumed to be cars, people or small objects. This height also provides robustness to slopes and defects of the road as the laser plane is less likely to intersect the ground. Moreover, note that the decision of working in two dimensions is justified by one main applicability reason. From a practical point of view, 2D information are sufficient for self-localization on a local flat map, as the car moves locally in two dimensions, and are manageable in the general case of restricted availability of computational resources.

TABLE I
SENSORS' FEATURES

-	Field View	Max Range	Layers	≈ Price
SICK LMS 151	270°	$50m$	1	3000 $
Velodyne VLP-16	360°	$> 100m$	16	8000 $

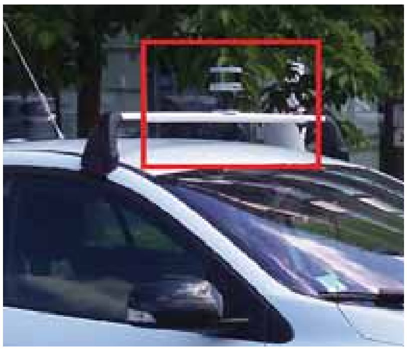


Fig. 3. The Velodyne VLP-16 is installed on the Renault Fluence ZE's roof surface.

The remainder of this paper is organized as following. In the next section, we present some of the most effective perception systems currently employed for SLAM applications, along with some relevant state-of-the-art methods for large-scale SLAM. In section III we explain how we obtain a 360° planar laser scan with a certain angular resolution where the measurements are relative to the vehicle's reference frame in both cases. Additionally we explain how we generate the ground truth, as well as, how we perform the comparison between computed position and ground truth position. In section IV we present the experiments performed on this study and their purpose. In section V we present the results of the experiments performed giving some partial conclusions. And finally in section VI, we summarize the main points of our results.

II. RELATED WORK

In the past, highly effective SLAM techniques have been developed and state-of-the-art SLAM solvers are now able to achieve good performances in terms of accuracy and real-time processing (e.g. *GMapping* [1] and *Hector SLAM* [2]).

The first implementations of SLAM methods were based on combined motion control and features observations with an Extended Kalman Filter (EKF) [3], [4]. However, as reported in [5], the solution to the EKF-based SLAM is consistent in the linear-Gaussian case but diverges in general situations. Subsequently, Rao-Blackwellized particle filters have been introduced as effective SLAM problem solvers under conditions of non-linearity. The approach proposed in [1] and [6] uses a particle filter in which each particle carries an individual map of the environment and treats the consequent requirement of reducing the number of particles. This algorithm is open source for the community of researchers under the name of *GMapping* and is currently employed for many SLAM-based applications. However, the problem of computational complexity over large-scale environments, of the order of tens of kilometers, has not been directly addressed in this work.

Closely related to the solution we propose are hierarchical SLAM methods to deal with large-scale applications. *Atlas* [7] is a framework which builds a two-levels hierarchy combining a local Kalman Filter with global graph optimization. Similarly, *Hierarchical SLAM* [8] is a mapping technique which uses statistically independent local maps interconnected at an upper level through an adjacency graph. Subsequent proposals employ independent local maps with robot-centred representation [9], local metric maps associated with topological places in a topological map [10], submapping methods in which the current submap summarizes all the information [11], local maps with feature positions and final robot pose associated with a global map without robot poses [12]. In [13] the authors present a SLAM technique requiring a small memory footprint. This feature makes this solution particularly suited for large-scale problems. In the case of this paper, we present an adaptation of the *GMapping* framework to deal with computational complexity problems while covering long distances.

Of fundamental importance for a SLAM-based architecture is the perception system. While quality and accuracy of the sensors are basic requirements for the reliability of the measurements, the costs in terms of system installation effort and finances have to be taken into account at *conception time*. During the *DARPA Urban Grand Challenge* in 2007 [14], fully equipped autonomous vehicles performed complex SLAM-based tasks to compete in a 96km course in a urban area. *Boss*, by the Carnegie Mellon University and General Motors Corporation team [15], took advantage of three 3D LiDARs (IBEO Alasca XT) mounted at about 50cm from the ground assuming to cover a relatively flat road surface, processing each layer independently and assuming infrequent changes in the static environment. *Junior*, by the Stanford University team [16], used a Velodyne HDL-64 mounted on the car's roof surface to perform 2D mapping and localization. *Odin*, by the Virginia Tech team [17], used the coupled information provided by three LiDARs (two IBEO Alasca XT and a IBEO Alasca A0) mounted at about 50cm from the ground level. In the last few years *Google* developed hardware and software architectures for a self-driving car [18]. The heart of the system is a Velodyne HDL-64 generating a detailed 3D map of the environment which requires high processing capabilities.

In this context, the aim of our work is to provide a comparison of the results obtained for the localization in an unknown environment using two differently equipped vehicles. In particular, we analyse the performances of our solution to a large-scale SLAM problem in the case of a (three) 2D LiDARs based versus a 16 planes Velodyne LiDAR based perception architecture.

III. METHODOLOGY

The required inputs to the SLAM system are odometry and planar laser scans, providing the information about the car's motion and the surrounding environment respectively. Specifically, we generate 360° FoV scans with 0.5° of angular resolution @ 10 Hz from each of the laser-based perception architecture and odometry @ 100Hz from the data collected by the OBD-II connectors and IMUs integrated in the cars. For the sake of simplicity, we consider both odometry and scans data with respect to the car reference frame, placed at the center of the rear axis.

In order to provide an unbiased comparison, we run the same SLAM system on both our vehicles. However, given two distinct hardware architectures, the approaches adopted to generate the scans and their quality vary. In the following, we present the two different techniques developed to generate laser scan information, first from the three Sick LiDARs and then from the VLP-16.

A. Sick LiDARs

In the case of Zoe (Figure 4), we convert the individual scan measurements into the 360° output scan by knowing the pose of each of the sensors with respect to the car reference frame. In particular, we determine the exact pose of each sensor using a process of extrinsic calibration, fitting the overlapping parts of the individual scans and then we convert each of the measurements from the sensor's local frame to the car's frame by simple reference frame transformation (Equation 1).

$$T_{scan} = T_{sensor} * T_{point} \tag{1}$$

where T_{point} is the transformation matrix of a point with respect to the sensor's frame, T_{sensor} is the fix transformation of the sensor with respect to the car's reference frame and T_{scan} is the transformation of the measured point with respect to the car's reference frame which is composed by a translation vector $\{x, y, z\}$ and a rotation matrix. From the transformed position we can extract the angular position α with the expression (2)

$$\alpha = \arctan \frac{x}{-y} \tag{2}$$

Depending on the scan's angular resolution, for a given α, a unique corresponding index in the output scan vector is given by (Equation 3).

$$i(\alpha) = round(\frac{\alpha}{\Delta\alpha}) \tag{3}$$

where $\Delta\alpha$ is the chosen angular resolution. In our case $\Delta\alpha$ is 0.5°and $\alpha \in \{0, 360°\}$.

On the other hand, we obtain the output range as

$$range_i = \sqrt{x^2 + y^2} \tag{4}$$

Notice that, for the same α there can be more than one measurement. In this case, we select the nearer point, i.e. the one with smaller range.

B. 16 Planes Velodyne

In the case of Fluence (Figure 5), the VLP-16 is placed above the roof surface. In particular, we compute the exact position of the VLP-16 as 1.457 meters from the car's rear axis and 1.565 from the ground after a process of extrinsic calibration. Similar to the three Sick LiDARs' case, we convert each of the measurements from the sensor's local frame to the car's frame by simple reference frame transformation (Equation 1). Notice that in this case the raw data provided by the VLP-16 correspond to 3D measurements. Therefore,

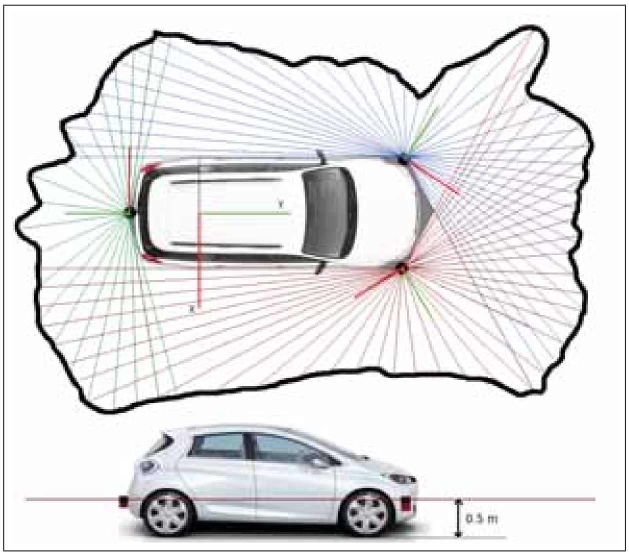


Fig. 4. Sensor configuration for Zoe. In different colours, the coverage of each planar LiDAR. In grey, the uncovered area. At the bottom, the profile of the scans horizontal plane.

so as to obtain a planar scan information, we project the points belonging to the vertical range $\{1.8, 2.8\}$ meters from the ground to a plane passing through the VLP-16 reference frame center and perpendicular to its vertical axis. Once this transformation is performed, the scan data type can be identified by an index and a corresponding range value as explained in the case of Zoe (Equation 3 and 4).

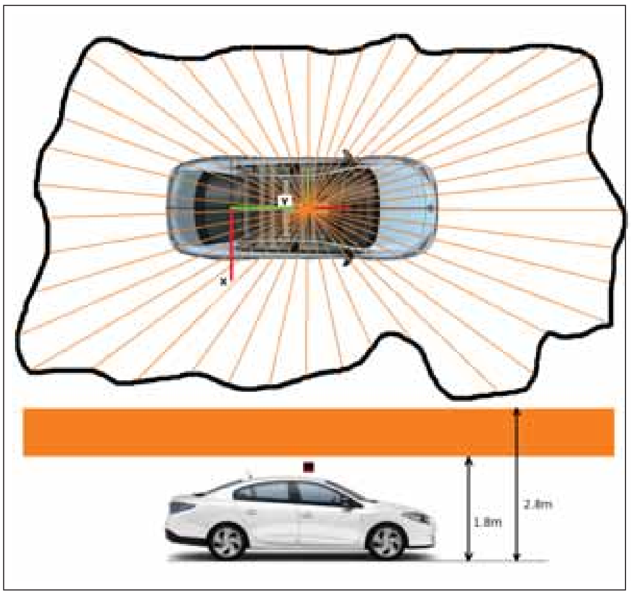


Fig. 5. Sensor configuration for Fluence. A 360° laser scan is obtained from the 3D point-cloud generated with VLP-16. In the bottom, the profile point-cloud range involved in the generation of the planar scan is shown.

C. Comparison with ground truth

The ground truth is generated using a Proflex 800 RTK-GPS receiver that applies the differential corrections obtained from a DGPS station located on the roof of IRCCyN building. The measurements provided by the RTK-GPS have an error of less than 1 cm in position when in Fixed mode. We estimate the orientation (heading), by computing the direction of the movement.

We compare the position obtained by the SLAM system at time t with the ground truth interpolated to that time. The interpolation method used is through splines of position coordinates with the time as independent variable.

IV. EXPERIMENTAL SET-UP

A couple of experiments have been performed to quantify the localization accuracy and robustness of both the systems. The precision of each of the results is evaluated comparing the position of the car with a ground-truth generated using the measurements provided by the high precision RTK-GPS on the points where it is available. During both the experiments, the vehicles travel in convoy, that is, one following the other, in order to ensure the same environmental conditions.

A. First Experimental Set-up

For the first experiment, the vehicles travel for about 3 kilometers. The chosen trajectory covers the area around the campus of École Centrale de Nantes, which is a standard urban sector. Using the data recorded by both the vehicles, we perform the task of map building. Later Monte Carlo localization in these maps is performed. The aim of this experiment is to compare the positioning accuracy of both systems under the same conditions.

B. Second Experimental Set-up

A second experiment has been conducted in a highly changing environment such as the parking area of the campus of École Centrale de Nantes, where the position and number of parked vehicles constantly changes. We performed recordings over two different days in order to ensure changes in the temporarily-static elements (e.g. parked vehicles) met along the path. In particular, the first day we recorded in the early morning (when the parking was almost empty) and the second day during the day-time (when the parking was crowded). In this context, on the first day the vehicles are asked to perform a task of localization and mapping. Subsequently, on a second day, the vehicles have to localize themselves along the same trajectory but with the maps previously built. The aim of this experiment is to check the robustness of each set-up against environmental changes.

V. RESULTS

A. First Experiment. Same trajectory, same environment

In figure 6 we show the accumulative and differential histogram of the error resulting from localization under the conditions detailed in Section IV-A, for both the cars.

As we can see the precision of Zoe's localization (equipped with planar LiDARs) is slightly better than Fluence's localization (equipped with VLP-16). For Zoe 95% of the measurements have less than 0.6 meters of error while for Fluence 95% have less than 0.7 meters. In this case, Zoe performs better because, since the LiDARS detections originally belong to a plane, the measurements are more stable and less noisy. Instead, the fact that each VLP-16 scan

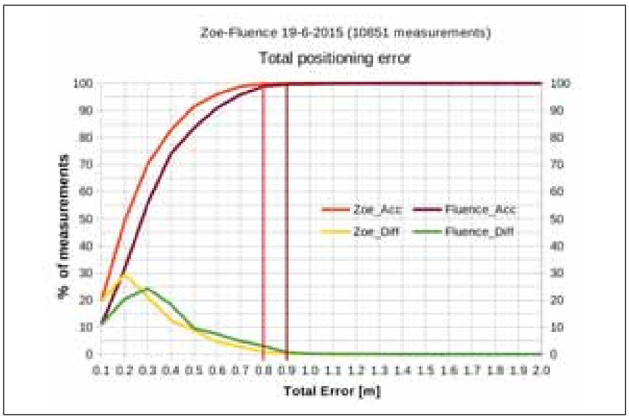
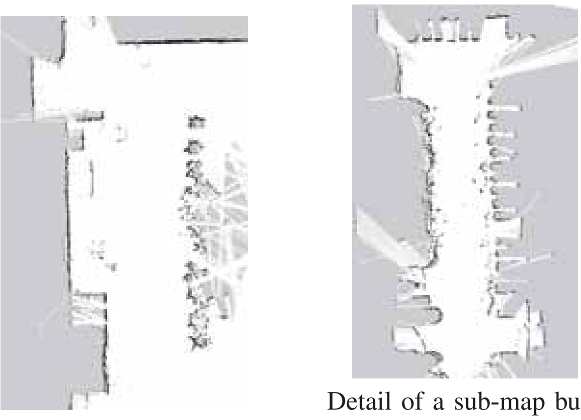


Fig. 6. Accumulative and differential histogram of positioning error between both cars for the same trajectory and same environment. The distance covered is about 3 Km. The coloured vertical line show the maximum error obtained along the journey for the respective car.

results from the projection of a collection of points onto a plane, causes the map not to be overall as well shaped as in the case of the planar LiDARs.

B. Second Experiment. Same trajectory, highly changing environment

Figure 7 shows part of the map built on the same area by both systems. In the left image, the map built by the VLP-16-based system doesn't show short objects like the cars parked, but the walls of the sourrounding buildings and trees. In the right one, the map build by Sick LMS151-based system show all the objects that can be seen at a height of 50 centimeter from the floor level. Just by looking at both maps we can expect significant differences on the analysis of the positioning accuracy.



Detail of a sub-map built using the LiDARs VLP-16

Detail of a sub-map built using the LiDARs Sick LMS151

Fig. 7. Detail of the same working area seen by each LiDAR system. Note that, while in the case of the Sick LMS151 the cars parked on the sides of the road are visible while in the case of the VLP-16 only the static environment like the walls of the buildings and the trees in the right side are considered.

Figure 8 shows the accumulative and differential positioning error histograms for the case of the planar LiDAR-based system. As expected the precision obtained for the second day using the maps of the first day is worst than in the case of the first day. However, an acceptable precision even under highly changing conditions was not an obvious result. In the first day 95% of the measurements lie under 0.4 meters while in the second that percentage moves to 0.8 meters. In this case we can assess that the common environmental changes affect the precision but still more than 97% of the measurements lie under 1 meter of error.

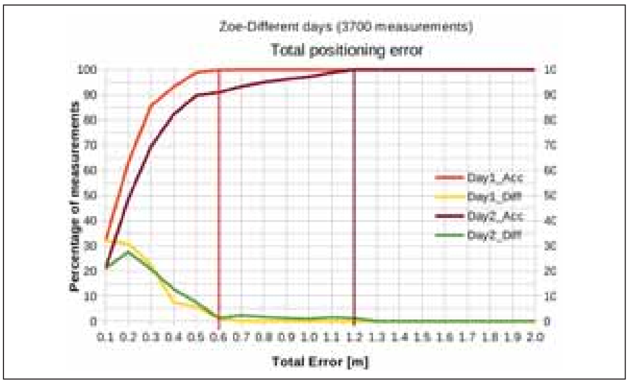


Fig. 8. Accumulative and differential histogram of positioning error for the car equipped with three Sick LMS151 on the same trajectory but different days where the environment has changed strongly. In coloured vertical lines are marked the maximum error obtained for the respective day.

Figure 9 shows the accumulative and differential histograms of the error in position between both days for the case of the Velodyne-based system (Renault Fluence). As expected, there is no significant difference on the precision between localizing with the same maps in both days as, from the point of view of the sensor, the environment has not changed at the projected range of heights. This means that this system is more robust against high environment changes under heights below 1.8 meters than the case of the planar LiDARs. Regarding the accuracy, in both days 95% of the measurements have less than 0.6 meters of error in absolute position.

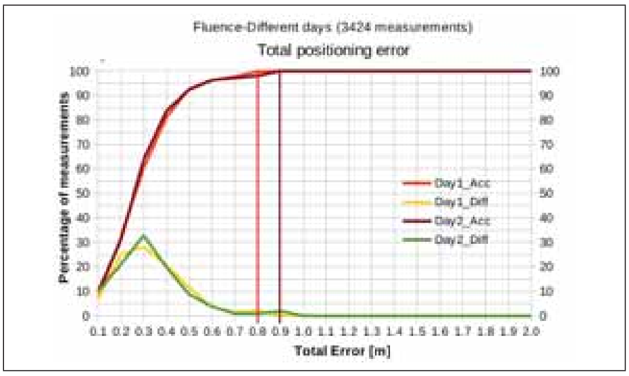


Fig. 9. Accumulative and differential histogram of positioning error for the car equipped with Velodyne VLP-16 on the same trajectory but different days where the environment has changed strongly. In coloured vertical lines are marked the maximum error obtained for the respective day.

VI. CONCLUSIONS

In this paper we perform a quantitative comparison, using our 2D-SLAM localization system for large scale scenarios, of two typical LiDAR-based hardware configurations: one based on several LiDARs strategically installed around the car at a low height, and the other one based on a single 3D-LiDAR installed on the roof of the car. Both systems generate a single 360° scan centered on the car's reference frame which is used for map-building and/or map-localization. The qualitative results from studies conducted over our datasets are summarized in Table II. In the same conditions and with an static environment, the set-up using planar LiDARs performs slightly better than with the 3D sensor as the scans are less noisy on the range dimension. The positioning error obtained is, in general, 10 cm. smaller in the first case. The reason for being more noisy in the 3D sensor's case is because we are projecting onto the same plane a range of heights (1 meter of range in our case), so that, the probability of variability is higher than in the case of one single height. Another conclusion is that the maps built with VLP-16 data provide more robustness against common medium term environment changes, like cars parked on the sides of the road, describing mostly the static details, like the walls of the buildings, trees or urban structures. This is a desirable feature on map localization. Furthermore, there are other advantages of using a single sensor regarding the simplicity of installation, extrinsic calibration and general set-up, speeding up considerably the set-up process. Regarding the cost-effectiveness, the newly released sensor VLP-16 is affordable for most of research projects in comparison with using 3 outdoors planar LiDARs. Finally, in the case of the 3D-sensor the fact of having 3D information available on the system opens a wide range of possibilities from the point of view of the perception.

TABLE II
COMPARISON TABLE

-	Planar LiDARs	3D LiDAR
Accuracy same environment	↑	↓
Robustness env. changes	↓	↑
Ease installation	↓	↑
Ease extrinsic calibration	↓	↑
Total cost	↑	↓
Provides 3D data	No	Yes

ACKNOWLEDGMENT

This work has been funded by the European Fund in Regional Development FEDER-ROBOTEX trough the *Pays de la Loire* regional development from june 2013 to october 2015. Thanks also to the ICARS team of IRCCyN.

REFERENCES

[1] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, April 2005, pp. 2432–2437.

[2] S. Kohlbrecher, O. von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *Safety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium on*, Nov 2011, pp. 155–160.

[3] P. Moutarlier and R. Chatila, "An experimental system for incremental environment modelling by an autonomous mobile robot," in *Experimental Robotics I, The First International Symposium, Montréal, Canada, June 19-21, 1989*, ser. Lecture Notes in Control and Information Sciences, V. Hayward and O. Khatib, Eds., vol. 139. Springer, 1989, pp. 327–346.

[4] R. Smith, M. Self, and P. Cheeseman, "A stochastic map for uncertain spatial relationships," in *Proceedings of the 4th International Symposium on Robotics Research*. Cambridge, MA, USA: MIT Press, 1988, pp. 467–474. [Online]. Available: http://dl.acm.org/citation.cfm?id=57425.57472

[5] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (slam) problem," *Robotics and Automation, IEEE Transactions on*, vol. 17, no. 3, pp. 229–241, Jun 2001.

[6] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *Robotics, IEEE Transactions on*, vol. 23, no. 1, pp. 34–46, Feb 2007.

[7] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller, "An atlas framework for scalable mapping," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 2, Sept 2003, pp. 1899–1906 vol.2.

[8] C. Estrada, J. Neira, and J. Tardos, "Hierarchical slam: Real-time accurate mapping of large environments," *Robotics, IEEE Transactions on*, vol. 21, no. 4, pp. 588–596, Aug 2005.

[9] J. Castellanos, R. Martinez-Cantin, J. Tards, and J. Neira, "Robocentric map joining: Improving the consistency of ekf-slam," *Robotics and Autonomous Systems*, vol. 55, no. 1, pp. 21 – 29, 2007, simultaneous Localisation and Map Building. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889006001448

[10] F. Ferreira, I. Amorim, R. Rocha, and J. Dias, "T-slam: Registering topological and geometric maps for robot localization in large environments," in *Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on*, Aug 2008, pp. 392–398.

[11] P. Pinies, L. M. Paz, and J. Tardos, "Ci-graph: An efficient approach for large scale slam," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, May 2009, pp. 3913–3920.

[12] S. Huang, Z. Wang, G. Dissanayake, and U. Frese, "Iterated d-slam map joining: evaluating its performance interms of consistency, accuracy and efficiency," *Autonomous Robots*, vol. 27, no. 4, pp. 409–429, 2009. [Online]. Available: http://dx.doi.org/10.1007/s10514-009-9153-8

[13] B. Suger, G. Diego Tipaldi, L. Spinello, and W. Burgard, "An approach to solving large-scale slam problems with a small memory footprint," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 3632–3637.

[14] (2007) Darpa urban challenge. [Online]. Available: http://archive.darpa.mil/grandchallenge/index.html

[15] C. Urmson, J. Anhalt, H. Bae, J. A. D. Bagnell, C. R. Baker, R. E. Bittner, T. Brown, M. N. Clark, M. Darms, D. Demitrish, J. M. Dolan, D. Duggins, D. Ferguson , T. Galatali, C. M. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, S. Kolski, M. Likhachev , B. Litkouhi, A. Kelly , M. McNaughton, N. Miller, J. Nickolaou, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, V. Sadekar, B. Salesky, Y.-W. Seo, S. Singh, J. M. Snider, J. C. Struble, A. T. Stentz, M. Taylor , W. R. L. Whittaker, Z. Wolkowicki, W. Zhang, and J. Ziglar, "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge, Part I*, vol. 25, no. 8, pp. 425–466, June 2008.

[16] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, and S. Thrun, "Junior: The stanford entry in the urban challenge," *Journal of Field Robotics*, 2008.

[17] A. Bacha, C. Bauman, R. Faruque, M. Fleming, C. Terwelp, C. Reinholtz, D. Hong, A. Wicks, T. Alberi, D. Anderson, S. Cacciola, P. Currier, A. Dalton, J. Farmer, J. Hurdus, S. Kimmel, P. King, A. Taylor, D. V. Covern, and M. Webster, "Odin: Team victortango's entry in the darpa urban challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 467–492, 2008. [Online]. Available: http://dx.doi.org/10.1002/rob.20248

[18] (2011) Ieee spectrum: How google's self-driving car works. [Online]. Available: http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works

Discriminative Map Matching Using View Dependent Map Descriptor

Liu Enfu Tanaka Kanji

Abstract— **The problem of matching a local occupancy grid map built by a mobile robot to previously built maps is crucial for autonomous navigation in both indoor and outdoor environments. In this paper, the map matching problem is addressed from a novel perspective, which is different from the classic bag-of-words (BoW) paradigm. Unlike previous BoW approaches that trade discriminativity for viewpoint invariance, we develop a local map descriptor that is view-dependent and highly discriminative. Our method consists of three distinct steps: (1) First, an informative local map of the robot's local surroundings is built. (2) Next, a unique viewpoint is planned in accordance with the given local map. (3) Finally, a synthetic view is described at the designated viewpoint. Because the success of our local map descriptor (LMD) depends on the assumption that the viewpoint is unique given a local map, we also address the issue of viewpoint planning and present a solution that provides similar views for similar local maps. Consequently, we also propose a practical map-matching framework that combines the advantages of the fast succinct bag-of-words technique and the highly discriminative LMD descriptor. The results of experiments conducted verify the efficacy of our proposed approach.**

I. Introduction

The problem of matching a local occupancy grid map built by a mobile robot to previously built maps is crucial for autonomous navigation in both indoor and outdoor environments [1]–[7]. This paper addresses a general 1-to-N matching problem in which a 2D pointset map is given as a query, and the system searches over a size N map database to find similar database maps that are relevant under rigid transformation.

The classical approach to the map-matching problem is to describe the appearance of each local map using high-dimensional local invariant feature descriptors such as shape features (e.g., polestar feature [8]), and perform feature matching between query and database maps. One major limitation of such an approach is the time consumed comparing the high-dimensional descriptors [9]. One of the most popular approaches used to address this computational cost is the bag-of-words (BoW) approach, in which an unordered collection of vector quantized feature descriptors (e.g., shape context, polestar), which are extracted at random, dense, or interest points (e.g., FLIRT), is used for compact map representation and efficient matching to pre-built maps. Thus far, the BoW approach has been utilized in various map-matching tasks, ranging from view image sequence maps to 3D point cloud maps [5]–[7]. Our proposed approach is also built on the BoW system in [10], in which the BoW framework is successfully applied to the retrieval of 2D occupancy maps using rotation invariant polestar descriptors.

In this paper, we consider the local map descriptor (LMD), which involves the generation of text descriptions of local map content to facilitate fast succinct text-based map matching. Unlike previous local feature approaches that trade discriminativity for viewpoint invariance, we develop a holistic view descriptor that is view-dependent and highly discriminative. Our method consists of three distinct steps:

1) First, an informative local map of the robot's local surroundings is built.
2) Next, a unique viewpoint is planned in accordance with the given local map.
3) Finally, a synthetic view is described at the designated viewpoint.

The success of our holistic view descriptor is based on the assumption that the viewpoint is unique given a local map. Therefore, we also address the issue of viewpoint planning and present a solution that provides similar views for similar local maps. We also propose a practical map-matching framework that combines the advantages of the fast succinct BoW techniques (e.g., [11]), and the highly discriminative LMD holistic view descriptor. The results of experiments conducted using the publicly available radish dataset [12] and our own collected dataset confirm the efficacy of our proposed approach.

In this paper, we focus on methods that describe not only local feature descriptors but also the local keypoint configuration among them. Among these methods, the part model [13], in which a scene is modeled as a collection of visual parts, is very popular. The model uses information on relative positions as spatial cues to improve the discriminative power of representation. However, existing part-based models primarily focus on a small set of pre-learned parts. Our approach is somewhat similar in concept to the spatial pyramid matching approach in [14], as opposed to the focus on kernel definition and improvement to discriminative power of previous solutions. Most of the works cited above either explicitly or implicitly assume that the viewpoint trajectory of the mapper robot w.r.t. the local map is unavailable. In contrast, we explicitly use the viewpoint information produced by our viewpoint planner as a cue to compute the holistic view descriptor. The success of our approach is based on the assumption that the viewpoint planner provides a unique viewpoint given a local map; therefore, we also consider the issue of viewpoint planning. To the best of our knowledge, these two issues have not been explored in previous approaches.

Our work has been supported in part by JSPS KAKENHI Grant-in-Aid for Young Scientists (B) 23700229, and for Scientific Research (C) 26330297.

E. Liu and K. Tanaka are with Graduate School of Engineering, University of Fukui, Japan. tnkknj@u-fukui.ac.jp

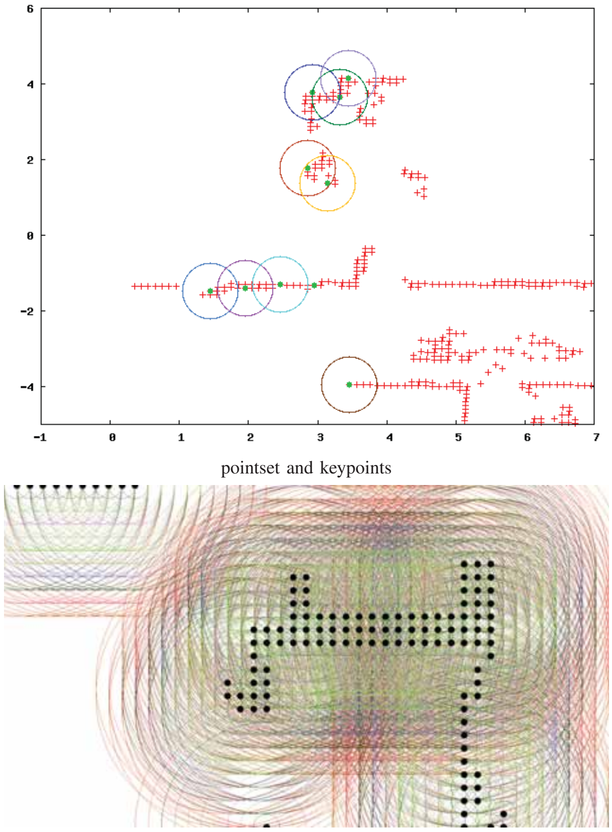
II. Local Map Descriptor

A. Baseline System

This section describes the baseline map-matching system, on which our proposed approach is built, and which is also used as a benchmark for performance comparison in the experimental section, Section III. The main steps in the procedure carried out by the system are as follows: (1) Extraction of appearance features from each local map, (2) translation of the extracted features to a BoW descriptor, and (3) construction/retrieval of the map database from the BoW descriptors. These three steps are explained in detail below.

1) Feature Extraction: We adopt the polestar feature for our purpose because it has several desirable properties, including viewpoint invariance and rotation independence, and has proven effective as a landmark for map matching in previous studies [10]. The extraction algorithm consists of three steps (Fig.1): (1) First, a set of keypoints are sampled from the raw 2D scan points. (2) Next, a circular grid is imposed and centered at each keypoint with different $D = 10$ radius. (3) Finally, the points falling into each circular grid cell are counted and the resulting D-dim vector outputted as the polestar descriptor.

2) BoW Descriptor: Next, we quantize each D-dim polestar vector to a 1-dimensional code termed “visual word”. This quantization process consists of three steps: (1) normalization of the D-dim vector by the vector’s L1 norm, (2) binarization of each i-th element of the normalized vector into $b_i \in \{0,1\}$, and (3) translation of the binarized D-dim vector into a code or a visual word: $w_a = \sum_i 2^i b_i$. Currently, the threshold for binarization is determined as the mean of all the elements of the vector. In consequence, a map is represented by an unordered collection of visual words $\{w_a \mid w_a \in [1,K]\}$, called BoW. Because we consider D-dim binarized polestar descriptors, the vocabulary size is $K = 2^{10}$.



pointset and keypoints

circular grids

Fig. 1. Extraction of 2D polestar features from a 2D pointset map.

3) Database Construction/Retrieval: We use the BoW representation for both the database construction and retrieval processes. In the former process, each local map is indexed by the inverted file system, by using each word w_a belonging to the map as an index. In the latter process, all the indexes that have words in common with the query map are accessed and the resulting candidate database maps are ranked based on the frequency or the number of words in common. A frequency histogram of visual words is represented by a K-dim vector when we have K words in the vocabulary. Similarity between a pair of BoW frequency histograms is evaluated in terms of the histogram intersection.

B. Proposed Extension

In this section, we outline our proposed extension. As mentioned earlier, we built on the baseline system described in Section II-A, and developed a novel holistic view descriptor. Our method consists of three distinct steps: (1) build a local map, (2) plan a unique viewpoint given the local map, and (3) describe a synthetic view at the planned viewpoint. These three steps as well as the modified map-matching algorithm are detailed in below.

1) Map Building: We first build a local map from a short sequence of perceptual and odometry measurements; each measurement sequence must be sufficiently long to cover rich photometric and geometric information about the robot’s local surroundings. In implementation, each sequence corresponds to the robot’s 3 m run. Any map-building algorithm (e.g., FastSLAM, scan matching) can be used to register a measurement sequence into a local map. We start a local map every time the robot’s viewpoint moves along the path. This results in a collection of overlapping local maps along the path.

2) Viewpoint Planning: We wish to design a robust planner that provides a unique viewpoint given a local map. (Note that the viewpoint is not necessarily one of the actual viewpoints.) An occupancy grid map is constructed from the 2D pointset map and used as input to our viewpoint planner. Currently, we plan the unique viewpoint near to the center of gravity (CoG) of all the occupancy grid cells. This strategy is inspired by the observation that the CoG can be unique given a local map both in narrow corridors and in rooms.

In implementation, all the viewpoints on the free space cells on the local occupancy grid map are viewed as candidate viewpoints, and among them, the closest candidate to the CoG is selected as the viewpoint for the holistic view descriptor. Subsequently, we determine the viewing direction based on the “dominant direction” [15] of the occupancy grid cells. An intuitive example of the dominant direction is

Manhattan world-like environments, where the two dominant directions should be the two orthogonal directions of the manhattans world. To estimate the dominant directions, we adapt the Manhattan world assumption criteria in [15].

3) Holistic View Descriptor: Let us now look at the holistic view at the planned viewpoint and represent it in the BoW form. A key difference of our BoW representation from that of previous works is that we no longer need to rely on view invariant local features that trade discriminativity for view invariance. Instead, we can exploit the knowledge of viewpoint w.r.t. the ego-centric local map coordinate to make the holistic descriptor view-dependent, and thus highly discriminative. Our BoW representation comprises appearance words and pose words. The former represents the appearance descriptor of each local feature w.r.t. the local map coordinate. Currently, we simply use the descriptor of each local feature and quantize it into an appearance word, as we did in Section II-A.2. The latter, pose word, represents the keypoint of each local feature w.r.t. the local map coordinate. During implementation, we quantize the keypoint (x,y) w.r.t. the local map's coordinate to obtain the pose word (w_x,w_y) with resolution quantization step size of 0.1 m. As a result, our visual word is in the form:

$$\langle w_x, w_y, w_a \rangle. \tag{1}$$

4) Map Matching: To index and retrieve the BoW map descriptors, we use the appearance word w_a as the primary index for the inverted file system, while using the pose word (w_x,w_y) as an additional cue for fine matching. The retrieval stage begins with a search of the map collection using the given appearance word w_a as a query to obtain all the memorized feature points with common appearance words, and filter out those feature points whose pose word (w'_x,w'_y) is distant from that of the query feature (w_x,w_y):

$$|w_x - w'_x| > D_{x,y}, \tag{2}$$

$$|w_y - w'_y| > D_{x,y}, \tag{3}$$

to obtain the final shortlist of maps. Currently, we use a large threshold, $D_{x,y} = 1[m]$, to suppress false negatives, i.e., incorrect identification of relevant maps as not being relevant.

III. Experiments

We conducted map-matching experiments to verify the efficacy of the proposed approach. In the ensuing subsections, we first describe the datasets and the map-matching tasks used in the experiments, then present the results obtained and conduct performance comparison against the baseline system.

A. Dataset

For map matching, we created a large-scale map collection from the publicly available radish dataset [12], which comprises odometry and laser data logs acquired by a car-like mobile robot in indoor environments (Fig.2). We created a collection of query/database maps using a scan matching algorithm from each of six different datasets—namely, "abuilding," "albert," "fr079," "run," "fr101," and "kwing"—which were obtained by the mobile robot's 79–295 m travel, corresponding to 521–5299 scans. Fig.3 shows examples of the query and database maps. The map collection comprises more than 13,000 maps. Our map collections contain many virtually duplicate maps, which makes map matching a challenging task.

B. Qualitative Results

Recall that the objective of map matching is to find a relevant map from the map database for a local map given as a query. The relevant map is defined as a database map that satisfies two conditions: (1) Its pose is near the query map's pose within a predefined range, where the pose of a map is defined as the CoG of the map's pointset; and (2) its distance traveled along the robot's trajectory is distant from that of the query map, such as in a "loop-closing" situation in which a robot, after traversing a loop-like trajectory, returns to a previously explored location.

For each relevant map pair, a map-matching task is conducted using a query map and a size N map database, which consists both of the relevant map and $(N-1)$ random irrelevant maps. The spatial resolution of the occupancy map is set to 0.1m. We implemented the map-matching algorithm in C++, and successfully tested it on various maps. Figs. 3 show the results of map matching using the baseline ("BoW") and the proposed ("LMD") systems. As can be seen, fewer false positives appear in the case of the proposed LMD method than the BoW method. This is because many of the incorrect matches are successfully filtered out by the proposed feature, which uses the keypoint configuration as a cue. Quantitative evaluation results for our approach are provided in the next subsection.

C. Quantitative Results

For performance comparison, we evaluated the averaged normalized rank (ANR) [16] for both the BoW and LMD methods. ANR is a ranking-based performance measure in which a lower value is better. To determine ANR, we conducted a number of independent map-matching tasks with different queries and databases. For each task, the rank assigned to the ground-truth database map by a map matcher of interest was investigated and normalized by the database

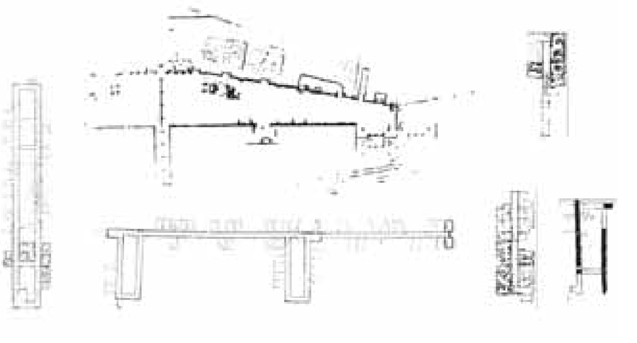
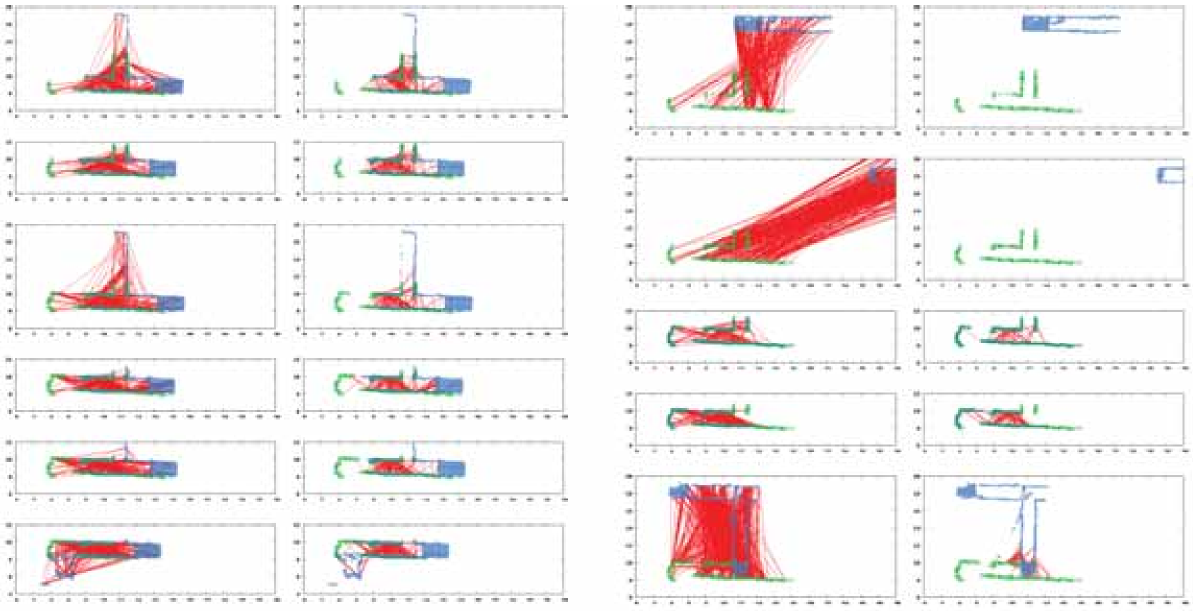


Fig. 2. Datasets used in the experiments: "abuilding," "albert," "fr079," "run," "fr101," and "kwing" from the radish dataset [12].



BoW LMD

Fig. 3. Examples of map matching. Left: relevant map pairs. Right: irrelevant map pairs.

TABLE I

SUMMARY OF ANR PERFORMANCE [%].

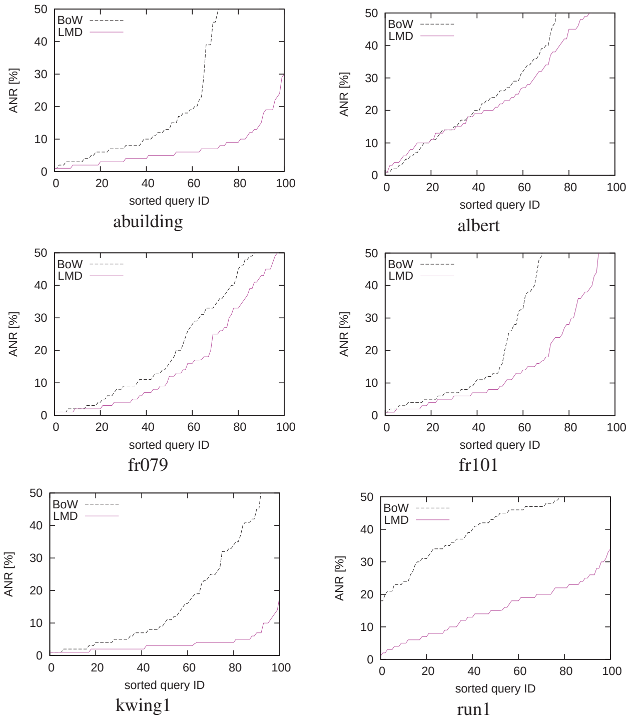
dataset	abuilding	albert	fr079	fr101	kwing1	run1	OCD
BoW	29.3	35.0	24.0	32.6	18.7	41.7	37.1
LMD	**7.0**	**26.6**	**17.1**	**16.7**	**3.6**	**15.2**	**22.6**

size N. ANR was subsequently obtained as the average of the normalized ranks over all the map-matching tasks. All map-matching tasks were conducted using 13,592 different queries and map databases.

Table I and Fig.4 summarize the ANR performance. The proposed LMD system clearly outperforms the baseline BoW system. By filtering out incorrect matches using the keypoint configuration as a cue, the LMD method was able to successfully perform map matching in many cases, as shown in the figure. In contrast, the BoW system based on appearance words alone often does not perform well, mainly because of the large number of false matches. The above results verify the efficacy of our approach. Table I also reports results of additional experiments using our own collected dataset, termed “OCD” in the table. In this experiments, we used pioneer 3 DX mobile robot equipped with LMS200 laser scanner and collected a set of 38 local maps in our university building 1F, 2F, and 5F. All the maps used in the experiment are shown in Fig.5 and shown in Table I are ANR performance. As can be seen, the proposed method outperforms the previous BoW method for almost all maps considered in the experiments.

IV. CONCLUSIONS

In this paper, we focused on generating text description of local map content for fast succinct text-based map matching. In particular, we presented a novel holistic view descriptor that describes a synthetic view at a planned viewpoint. We addressed the issues involved in building a local map, planning viewpoints, and computing the holistic view descriptor. The results of experiments conducted with the publicly available radish dataset confirm the efficacy of our proposed approach. In the future, we plan to use the



abuilding albert fr079 fr101 kwing1 run1

Fig. 4. ANR performance for each dataset (horizontal axis: sorted query map ID, vertical axis: ANR in [%]).

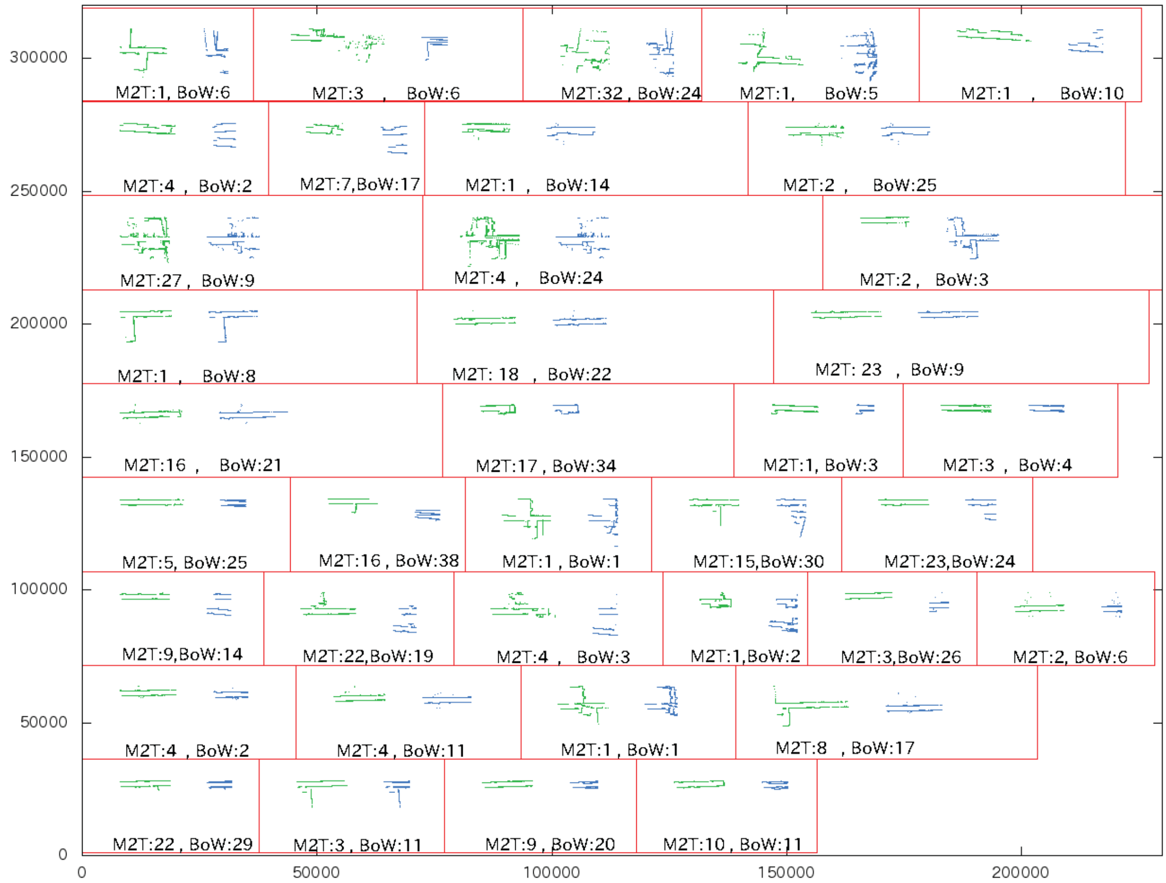


Fig. 5. Results for own collected dataset (OCD). Shown in the figure are from top to bottom, pairs of query map #0-#4, #5-#8, #9-#11, #12-#14, #15-#18, #19-#24, #25-#30, #31-#34, and #35-#38, and its ground-truth database map. Performance is evaluated in terms of rankings of ground-truth database map and the ranking values are shown in the figure for either proposed LMD method (“M2T”) or conventional BoW method (“BoW”). The details are better seen by zooming on a computer screen.

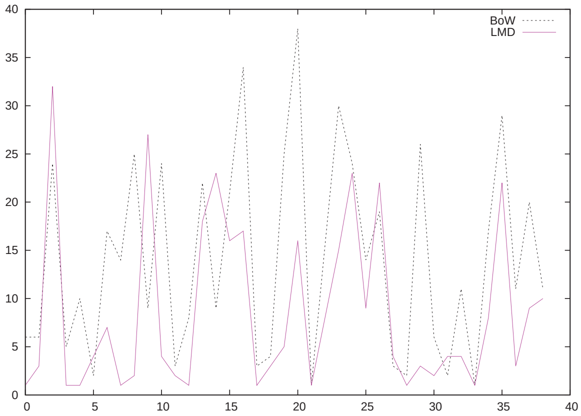


Fig. 6. Ranks for each query from our own collected dataset. (horizontal axis: query map ID, vertical axis: ANR in [%]).

presented LMD system for long-term operation of robots in familiar environments. Although this paper focused on the standard 2D pointset map, we believe our approach is sufficiently general to be applicable to a broad range of map formats, such as the 3D point cloud map, as well as general view-based maps.

References

[1] T. Botterill, S. Mills, and R. Green. Speeded-up bag-of-words algorithm for robot localisation through scene recognition. In *IVCNZ08*, pages 1–6, 2008.

[2] Arnau Ramisa, Adriana Tapus, David Aldavert, Ricardo Toledo, and Ramon Lopez de Mantaras. Robust vision-based robot localization using combinations of local feature region detectors. *Autonomous Robots*, 27(4):373–385, 2009.

[3] John McDonald, Michael Kaess, Cesar Dario Cadena Lerma, José Neira, and John J. Leonard. Real-time 6-dof multi-session visual SLAM over large-scale environments. *Robotics and Autonomous Systems*, 61(10):1144–1158, 2013.

[4] Sebastian Thrun, Dieter Fox, Wolfram Burgard, and Frank Dellaert. Robust monte carlo localization for mobile robots. *Artif. Intell.*, 128(1-2):99–141, 2001.

[5] M. Cummins and P. Newman. Appearance-only slam at large scale with fab-map 2.0. *Int. J. Robotics Research*, 30(9):1100–1123, 2011.

[6] Adrien Angeli, David Filliat, Stephane Doncieux, and Jean arcady Meyer. A fast and incremental method for loop-closure detection using bags of visual words,h conditionally accpeted for publication in. *IEEE Trans. Robotics, Special Issue on Visual SLAM*, 2008.

[7] Tom Botterill, Steven Mills, and Richard Green. Speeded-up Bag-of-Words algorithm for robot localisation through scene recognition. pages 1–6, 2008.

[8] E. Silani and M. Lovera. Star identification algorithms: Novel approach & comparison study. *IEEE Trans. Aerospace and Electronic Systems*, 42(4):1275–1288, 2006.

[9] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Packing bag-of-features. In *Proc. IEEE Int. Conf. Computer Vision*, 2009.

[10] Kanji Tanaka and Kensuke Kondo. Multi-scale bag-of-features for scalable map retrieval. *JACIII*, 16(7):793–799, 2012.

[11] Sivic J. and Zisserman A. Video google: a text retrieval approach to object matching in videos. *Proc. IEEE Int. Conf. Computer Vision*, pages 1470–1477, 2003.

[12] Andrew Howard and Nicholas Roy. The robotics data set repository (radish), 2003.

[13] Dongfeng Han, Wenhui Li, and Zongcheng Li. Semantic image classification using statistical local spatial relations model. *Multimedia Tools Appl.*, 39(2):169–188, 2008.

[14] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2169 – 2178, 2006.

[15] Sven O. and Markus V. Robust single view room structure segmentation in manhattan-like environments from stereo vision. In *IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 5315–5322, 2011.

[16] Hanada Shogo and Tanaka Kanji. Partslam: Unsupervised part-based scene modeling for fast succinct map matching. In *IEEE/RSJ Int. Conf. IROS*, pages 1582–1588. IEEE, 2013.

Path Planning and Steering Control for an Automatic Perpendicular Parking Assist System

Plamen Petrov, Fawzi Nashashibi, *Member, IEEE*, and Mohamed Marouf

***Abstract*— This paper considers the perpendicular reverse parking problem of front wheel steering vehicles. Relationships between the widths of the parking aisle and the parking place, as well as the parameters and initial position of the vehicle for planning a collision-free reverse perpendicular parking in one maneuver are first presented. Two types of steering controllers (bang-bang and saturated *tanh*-type controllers) for straight-line tracking are proposed and evaluated. It is demonstrated that the saturated controller, which is continuous, achieves also quick steering avoiding chattering and can be successfully used in solving parking problems. Simulation results and first experimental tests confirm the effectiveness of the proposed control scheme.**

I. Introduction

The perpendicular parking is the most efficient and economical since it accommodates the most vehicles per linear meter [1], and is especially effective in long term parking areas. Due to the special constraint environments, much attention and driving experience is needed to control the vehicle, and this parking maneuver may be a difficult task. For this reason, automated operation attracts significant attention from research view point, as well, and from the automobile industry. One of the difficulties in achieving automatic parking is the narrow operating place for collision-free motion of the vehicle during the parking maneuver, and planning of optimal trajectories is often used in the applications. In [2], an optimal stopping algorithm was designed for parking using an approach combining an occupancy grid with planning optimal trajectories for collision avoidance. The geometry of the perfect parallel parking maneuver is presented in [3]. In [4], a practical reverse parking maneuver planner is given. A trajectory planning method based on forward path generation and backward tracking algorithm, especially suitable for backward parking situations is reported in [5]. A car parking control using trajectory tracking controller is presented in [6]. In [7], a saturated feedback control for an automated parallel parking assist system is reported. In recent years, automatic parking systems have been also developed by several automobile manufacturers [9, 10].

In this paper, we focus on geometric collision-free path planning, and feedback steering control for perpendicular reverse parking in one maneuver. Geometric path planning based on admissible circular arcs within the available parking spot is presented in order to steer the vehicle in the direction of the parking place in one maneuver. Two steering controllers (bang-bang and saturated *tanh*-type) for path tracking are proposed and evaluated. The rest of the paper is organized as follows: In Section II, geometric considerations for planning perpendicular reverse parking in one maneuver are presented. In Section III, two feedback steering controllers are proposed. Simulation results and first experimental tests are reported in Section IV. Section V concludes the paper.

II. Geometric Considerations for Collision-Free Perpendicular Parking in One Maneuver

A. Vehicle Model

In this paper, a rectangular model of a front-wheel passenger vehicle is assumed. The vehicle parameters which affect the parking maneuver, as well as the parameter values used in the simulations, are presented in Table I.

TABLE I. **Vehicle Parameters**

Vehicle parameters	*Notation*	*Value*
Longitudinal vehicle base	l	*2.6m*
Wheel base	b	*1.8m*
Distance between the front axle and the front bumper	l_1	*0.94m*
Distance between the rear axle and the rear bumper	l_2	*0.74m*
Maximum steering angle	α_{max}	*π/6rad*

B. Collision-Free Path Planning with a Constant Turning Radius

The geometry of the reverse perpendicular parking in one collision–free maneuver is shown in Fig. 1. In the perpendicular parking scenario considered in this paper, the vehicle starts to move backward from an initial position *1* in the parking aisle, with constant steering angle α_c, which may be smaller than the maximum steering angle ($|\alpha_c| \leq |\alpha_{max}|$), and has to enter in the parking place (position *2*) without colliding with the boundary c_1 of parking lot L1 and boundaries c_2, and c_3 of parking lot L2. In position 2 the orientation of the vehicle is parallel with respect to the parking place. After that, the vehicle continues to move backward in a straight line into the parking place until it reaches the final position *3* (Fig. 1). Assuming a circular motion of the vehicle (with turning radius ρ_c), with center *O* (Fig. 1). The radius ρ_c is calculated from the formula

$$\rho_c = \frac{l}{\tan \alpha_c} \cdot \qquad (1)$$

P. Petrov is with the Faculty of Mechanical Engineering, Technical University of Sofia, 1000 Sofia, Bulgaria, (e-mail: ppetrov@ tu-sofia.bg).

F. Nashashibi is with the Robotics & Intelligent Transportation Systems (RITS), INRIA - Rocquencourt, 78153 Rocquencourt, France, (e-mail: fawzi.nashashibi@inria.fr).

M. Marouf is with the Robotics & Intelligent Transportation Systems (RITS), INRIA - Rocquencourt, 78153 Rocquencourt, France, (e-mail: mohamed.marouf@inria.fr).

The boundaries of the turning path during the perpendicular parking are determined by the dimensions of the traces (circular arcs) formed by the left corner of the front bumper B_2 with radius r_{B2}, the left corner of the rear bumper B_4 with radius r_{B4}, and the end of the rear wheel axle C_1, respectively, as shown in Fig.1. Since the vehicle executes a plane rotation, the trajectories of these points form arcs of concentric circles.

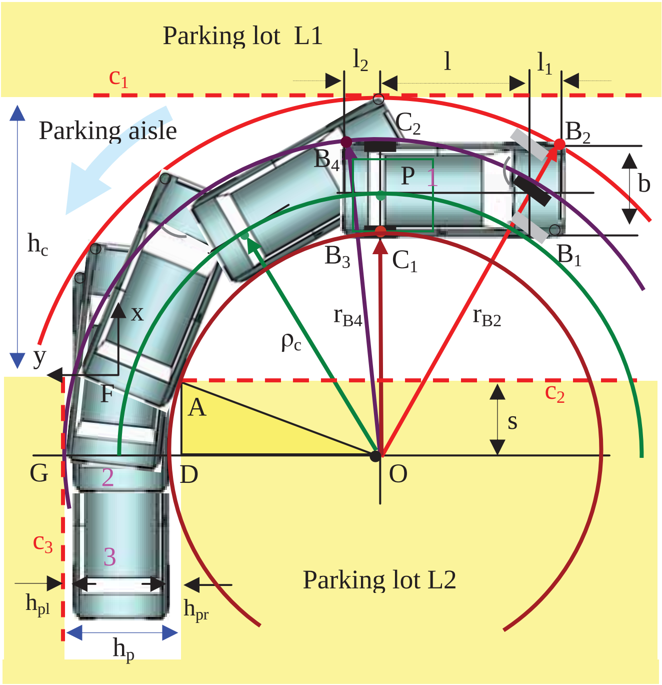


Figure 1. Geometry of the collision-free perpendicular parking maneuver

From the ΔOC_2B_2, applying the Pythagorean Theorem, we obtain an expression for the radius r_{B2} of the circular arc traced by the left corner of the front bumper B_2 in terms of the vehicle parameters l, l_1, b , and the turning radius ρ_c, as follows

$$r_{B2} = OB_2 = \sqrt{(l+l_1)^2 + \left(\rho_c + \frac{b}{2}\right)^2} \cdot \tag{2}$$

From the ΔOC_2B_4, we determine the radius r_{B4}, of the circular arc traced by the left corner of the rear bumper B_4

$$r_{B4} = OB_4 = \sqrt{l_2^2 + \left(\rho_c + \frac{b}{2}\right)^2} \cdot \tag{3}$$

We assign an inertial frame Fxy attached to the parking place, where the center F is placed in the middle between the borders of the parking place, which has its y-axis aligned with the boundary c_2 of parking lot L2, as shown in Fig. 1. Let O denotes the center of rotation of the vehicle (the Instantaneous Center of Rotation (ICR)) when it starts the parking maneuver with constant steering angle α_c. Depending on the sign of x-coordinate of ICR (point O) with respect to the Fxy frame, i.e., the offset s (Fig. 1), different formulas can be derived in order to determine the required width h_p of the parking place and the width of the parking aisle (the corridor) h_c as functions of s in order to ensure collision-free perpendicular parking in one maneuver. We consider right turning of the car in the following two cases:

- The ICR O belongs to the interval: $s \in [-(\rho_c - b/2), \quad 0]$

The lower value of the interval corresponds to the case when the right side of the vehicle B_1B_3 (Fig.1) lies on the boundary line c_2 of parking lot L2.

In order to avoid collision between the left corner B_2 of the front bumper with the boundary c_1 of L1 (Fig. 1), using (2), we obtain an expression for the width of the parking aisle h_c, as follows

$$h_c = r_{B2} - |s| = \sqrt{(l+l_1)^2 + \left(\rho_c + \frac{b}{2}\right)^2} - |s| \cdot \tag{4}$$

The function h_c = f(s) defined by (4) is linear in s, positive and monotonically increasing in the above-mentioned closed interval for s. Therefore, it takes its minimum and maximum values at the ends of this interval.

To avoid a collision between the right point C_1 of the rear axle with the vertex A of obstacle L2, from the ΔOAD, applying the Pythagorean Theorem, the distance OD (Fig. 1) is calculated as follows

$$OD = \sqrt{\left(\rho_c - \frac{b}{2}\right)^2 - s^2} \cdot \tag{5}$$

In order to avoid a collision between the left corner B_4 of the rear bumper with the edge c_3 of the parking place, using (3) and (5), the following expression for the width h_p of the parking space is obtained

$$h_p = r_{B4} - OD = \sqrt{l_2^2 + \left(\rho_c + \frac{b}{2}\right)^2} - \sqrt{\left(\rho_c - \frac{b}{2}\right)^2 - s^2} \cdot \tag{6}$$

The function h_p = f(s) defined by (6) is continuous on the closed interval of s mentioned above. This function is differentiable on the open interval $s \in (-(\rho_c - b/2), \quad 0)$, and its derivative is given by

$$\frac{\partial h_p}{\partial s} = \frac{s}{\sqrt{\left(\rho_c - \frac{b}{2}\right)^2 - s^2}} < 0 \cdot \tag{7}$$

Therefore, the function h_p = f(s) is strictly decreasing on the closed interval [-(ρ_c – $b/2$), 0]. The maximum and minimum values of h_p can be found by replacing in (6) the boundary values of the interval: s = - (ρ_c – $b/2$) and $s = 0$.

- The ICR O belongs to the interval: $s \in [0, l_2]$

The upper bound l_2 corresponds to the case when the rear bumper lies on the Fy-axis at the instant when the orientation of the vehicle is parallel to the parking place.

In order to avoid a collision between the left corner B_2 of the front bumper with the boundary c_1 of L1, using (2), we obtain an expression for the width of the parking aisle h_c

$$h_c = r_{B2} + s = \sqrt{(l+l_1)^2 + \left(\rho_c + \frac{b}{2}\right)^2} + s \cdot \qquad (8)$$

Again, the function h_c = f(s) defined by (8) is linear in s, positive and monotonically increasing in the above-mentioned close interval of s. Therefore, it takes its minimum and maximum values at the ends of this interval.

To avoid a collision between the left corner B_4 of the rear bumper with the edge c_3 of the parking place, and between the right point C_1 of the rear vehicle axle with the vertex A of obstacle L2, we obtain the following expression for h_p

$$h_p = \sqrt{l_2^2 + \left(\rho_c + \frac{b}{2}\right)^2 - s^2} - \left(\rho_c - \frac{b}{2}\right) \cdot \qquad (9)$$

The function h_p = f(s) defined by (9), is continuous on the closed interval of $s \in [0, l_2]$. This function is differentiable on the open interval $s \in (0, l_2)$ and the derivative is

$$\frac{\partial h_p}{\partial s} = -\frac{s}{\sqrt{l_2^2 + \left(\rho_c + \frac{b}{2}\right)^2 - s^2}} < 0 \cdot \qquad (10)$$

Therefore the function is strictly decreasing on the closed interval $s \in [0, l_2]$. The maximum and minimum values of h_p can be found by replacing the limit values $s = 0$ and $s = l_2$ of the interval, respectively, in the expression (10). It should be noted that for $s = 0$, the two functions defined by (6) and (9) take the same maximum value. For $s = l_2$, the function h_p = f(s) takes minimum value, which is exactly the width b of the vehicle.

From a practical point of view, it is important to determine the starting positions of the vehicle for parking without collision in one maneuver in the case when the widths h_c and h_p of the parking aisle and the parking space, respectively, are specified in advanced. Suppose that the widths of the parking aisle and the parking place are set as $h_c = h_{cd}$ and $h_p = h_{pd}$, respectively, and also that $h_{cd} < r_{B2}$. In this case, from (2) and (4), it follows that

$$-|s|_{\max} = h_{cd} - r_{B2} \cdot \qquad (11)$$

From (3) and (6), we obtain a formula for the minimum value of s as follows

$$-|s|_{\min} = -\sqrt{\left(\rho_c - \frac{b}{2}\right)^2 - (r_{B4} - h_{pd})^2} \; . \qquad (12)$$

Simulation results were performed to illustrate the relationships between the widths h_c and h_p of the parking aisle and the parking space, respectively, as functions of the offset s in the interval [-(ρ – $b/2$), 0] by using parameters of the test vehicle (Table I) with $\alpha_c = \alpha_{max}$, ($\rho_c = \rho_{min}$). The values of h_c and h_p, (h_{cd} and h_{pd}), were chosen as follows: $h_{cd} = 6m$ and $h_{pd} = 2.4m$.

As seen from Fig.2, the function h_p = f(s) (the solid blue line) decreases in the interval and converges to $b=1.8m$ (the red dotted line), which is exactly the length of the wheel base of the vehicle. Meanwhile, the graph intersects the horizontal line for the assigned value of $h_{pd} = 2.4m$ (the blue dotted line) at s = - $|s|_{min}$ = $-1.91m$, which is the minimum value of s obtained from (12) for collision-free parking. In order to park the vehicle in one maneuver for s = - $|s|_{min}$ = $-1.91m$, from (8), the required minimum width h_c of the parking aisle is obtained to be $h_c = 4.55m$ which is less than the specified value of $h_{cd} = 6m$.

The function h_c = f(s) (the green solid line) increases linearly in the interval and the graph intersects the horizontal line for the assigned value of $h_{cd} = 6m$ (the green dotted line) at s = - $|s|_{max}$ = $-0.46m$, which is the maximum value of s, obtained from (11). For s = - $|s|_{max}$ = $-0.46m$, from (6), the required minimum width h_p of the parking place has to be h_p = 1.88m, which is less than the assigned value of $h_{pd} = 2.4m$.

Therefore, given specified values $h_c = h_{cd} = 6m$ and $h_p = h_{pd} = 2.4m$ for the parking aisle and the parking space, respectively, for collision-free parking, the offset s can take values in the interval [- $|s|_{min}$, - $|s|_{max}$] = [$-1.91m$, $-0.46m$], where the boundary values are determined by (12) and (11), respectively.

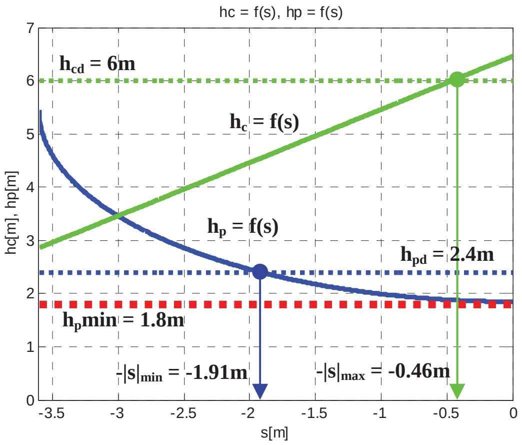


Figure 2. Collision-free interval for s

The distances between the car and the boundaries of the parking space h_{pl} and h_{pr} (Fig. 1), when the vehicle is parallel to the parking space, are determined as follows

$$h_{pr} = \left(\rho_c - \frac{b}{2}\right) - \sqrt{\left(\rho_c - \frac{b}{2}\right)^2 - s^2} \; , \qquad (13)$$

$$h_{pl} = h_{pd} - b - h_{pr} \; . \qquad (14)$$

From the simulations, for s = -$|s|_{min}$ = $-1.91m$, the obtained values of h_{pr} and h_{pl} are $h_{pr} = 0.55m$ and $h_{pl} = 0.05m$.

From a practical view point, it is better to park the car symmetrically with respect to the boundaries of the parking place, since it is not very wide. For this end, we calculate the minimum value of the offset $s = s_m$, in order to park the vehicle symmetrically in the center of the parking space (Fig. 3). We set

$$h_{ps} := h_{pr} = h_{pl} = \frac{h_{pd} - b}{2} . \tag{15}$$

From the ΔOAD (Fig. 3), the distance OD is determined as

$$OD = \sqrt{\left(\rho_c - \frac{b}{2}\right)^2 - s_m^2} . \tag{16}$$

Since the turning radius can be expressed as

$$\rho = \frac{b}{2} + h_{pr} + OD , \tag{17}$$

and substituting h_{pr} from (13) and OD from (16) into (17), we arrive to an expression for s_m, as follows

$$-|s|_m = -\sqrt{\left(\rho_c - \frac{b}{2}\right)^2 - \left(\rho_c - \frac{h_{pd}}{2}\right)^2} . \tag{18}$$

The new offset -|s_m| is bigger than those given by (12) (-|s|$_m$ > -|s|$_{min}$). In the simulation results, -|s|$_m$ = *-1.44m* > -1.91*m*. In general, it must be checked whether the new offset -|s|$_m$ is smaller than -|s|$_{max}$ given by (11). If it is the case, the car can park symmetrically without collision in reverse when s is at least s = -|s|$_m$. In this case, however, the boundary c_3 of the parking place will not be tangent to the arc of circle traced by point B_4 of the left corner of the rear bumper; nevertheless, point A (vertex A of obstacle L2) will lie again on the arc of circle traced by point C_1 of the rear vehicle axle. Therefore, given specified dimensions of the parking aisle and parking place $h_c = h_{cd}$ and $h_p = h_{pd}$, respectively, the offset s can take values in the closed interval -|s|∈[-|s|$_m$, -|s|$_{max}$], where -|s|$_m$ and -|s|$_{max}$ are determined from formulas (18) and (11), respectively, (Fig.3).

Hence, in order to perform reverse perpendicular parking in one maneuver and to place the vehicle symmetrically in the parking place, the starting position, i.e., the reference point P of the vehicle has to be on any one of the arcs of circles with radius ρ of center $O(x_O, y_O)$, where x_O∈[-|s|$_m$, -|s|$_{max}$] and y_O = -ρ_c, with respect to an inertial frame Fxy attached to the parking place. The initial orientation has to be tangent to the arc (Fig. 3). The reference path of the parking maneuver consists of two parts. The first one is a circular arc with center O connecting the staring position of the vehicle and the tangent point T between the arc and the x-axis of Fxy. At that point, the car will be parallel to the parking place. The second part of the reference path is a straight line along the y-axis of the coordinate frame Fxy between point T and the goal position G of the parking place, where point G lies on the x-axis of Fxy, (Fig. 3).

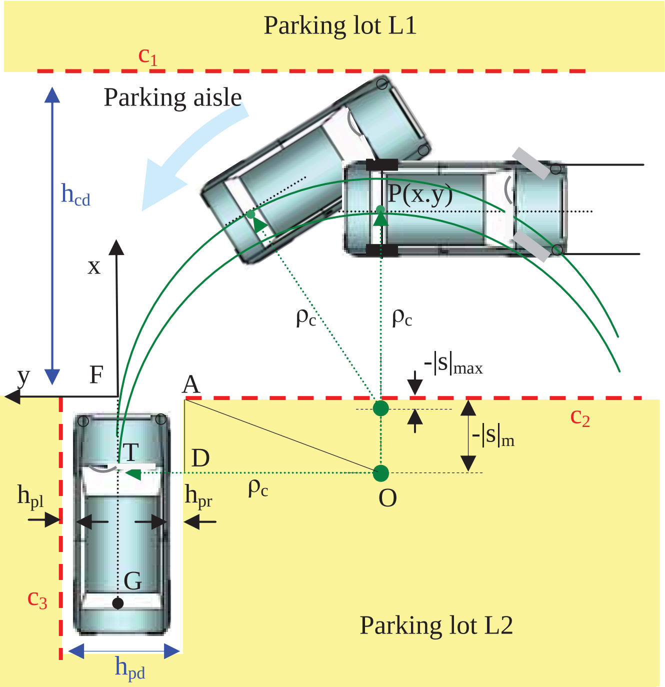


Figure 3. Geometry of collision-free perpendicular parking

III. Steering Control

For a low speed motion, which is the case of the parking maneuver, we assume that the wheels of the vehicle roll without sliding, and the velocity vectors are in the direction of the orientation of the wheels. We consider a simplified (bicycle model) of the vehicle, where the front and rear wheels are replaced by two virtual wheels, placed at the longitudinal axis of the vehicle. An inertial coordinate system is attached to the parking place (Fig. 3). The coordinates of the reference point P in Fxy are denoted by (x_P, y_P). The orientation of the vehicle θ is defined as an angle between the x-axis of Fxy and the longitudinal vehicle base. The front wheel steering angle is denoted by α. The equations of motion of the vehicle in the plane have the form [7]

$$\begin{aligned} \dot{x}_P &= v_P \cos\theta \\ \dot{y}_P &= v_P \sin\theta , \\ \dot{\theta} &= \frac{v_P}{l} \tan\alpha \end{aligned} \tag{19}$$

where v_P is the velocity of point P. We consider a practical stabilization of the vehicle in the parking place. Our approach is based on controlling the motion of the vehicle along a straight line (the x-axis of Fxy) passing through the goal point G (Fig.3) in the parking place and aligned with the orientation of the place with velocity of the car, which is dependent of the distance between the vehicle and the goal position [7]. Since the reference path for the first part of the parking maneuver is a circular arc, first a bang-bang controller is proposed, where the front wheel steering angle is constrained by magnitude and takes only two constant values. As a consequence, the vehicle trajectories represent circular arcs. However, in practice, due to the discontinuity of the control law, an undesirable behavior of the system (chattering) will occur when the position of the vehicle is in the vicinity of the tracking line, and the orientation error is also small. In order to avoid the chattering, a saturated control based on hyperbolic tangent function is also proposed, which is constrained by magnitude, but the control function is continuous.

A. Bang-Bang Control

In this paper, we propose a bang-bang control in the case when the vehicle is moving backward, ($v_P = -|v_P| < 0$). The vehicle has to track a straight line which coincides with the *x*-axis of coordinate frame *Fxy*. The design of the control low is based on the second and third equations of (19). The steering angle of the front wheels is constraint and takes values $\pm\alpha_c$. For brevity of exposition, we will present the final form of the bang-bang control. The control design procedure for backward driving of the vehicle is similar to those presented in [8], but the form is slightly different, since the vehicle velocity has negative sign. The bang-bang controller for backward driving has the form

$$u = \begin{cases} u & \text{if} \quad y_P > 2\dfrac{l}{\tan\alpha_c}\sin\dfrac{\theta}{2}\left|\sin\dfrac{\theta}{2}\right| \\ & \text{or} \quad y_P = 2\dfrac{l}{\tan\alpha_c}\sin\dfrac{\theta}{2}\left|\sin\dfrac{\theta}{2}\right| \quad \text{and} \quad y_P > 0 \\ -u & \text{if} \quad y_P < 2\dfrac{l}{\tan\alpha_c}\sin\dfrac{\theta}{2}\left|\sin\dfrac{\theta}{2}\right| \\ & \text{or} \quad y_P = 2\dfrac{l}{\tan\alpha_c}\sin\dfrac{\theta}{2}\left|\sin\dfrac{\theta}{2}\right| \quad \text{and} \quad y_P < 0 \end{cases}, \tag{20}$$

where

$$u = \frac{\tan\alpha_c}{l}\,. \tag{21}$$

B. Saturated Control

In order to avoid chattering in the system when a pure bang-bang control is used, we propose a differentiable saturation in the form of hyperbolic tangent (*tanh*(.)) constraint. This function is bounded by ± 1. Also $tanh(x) \geq 0$ if $x \geq 0$, and $tanh(x) < 0$ if $x < 0$. *Tanh*(*x*) is close to the signum function, when in $tanh(K_t x)$ the gain K_t is large, as shown in Fig. 4.

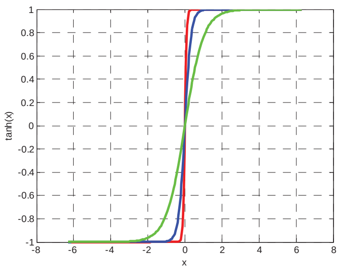


Figure 4. The function $tanh(K_t x)$ for $K_t = 1,3$, and 10

We propose the following feedback bounded steering controller

$$\alpha = a\tan\left[lu\tanh(K_t v)\right], \tag{22}$$

where *u* is given by (21),

$$v = K(\theta - a_0 y_P)\,, \tag{23}$$

and K_t, *K* and a_0 are positive constants.

IV. Simulation and Experimental Results

Simulation results using MATLAB are presented to illustrate the effectiveness of the proposed steering controllers for perpendicular reverse parking in one maneuver. The parameters of the vehicle are given in Table I. For the simulations, the constant steering angle of the front wheels was chosen to be $\alpha_c = \alpha_{max} = \pi/6 rad$. Using (1), for the minimum turning radius ρ is obtained the value of $\rho = 4.5m$. The parking aisle h_{cd} was *6m* wide, while the width of the parking place h_{pd} was *2.4m*. The initial coordinates of the vehicle reference point *P* with respect to the inertial frame *Fxy* attached to the parking place were $(x_P(0), y_P(0)) = (3.5m, -4.5m)$. In this case, the offset *s* is equal to $s = -1m$ and belongs to the interval $[-|s|_m, -|s|_{max}] = [-1.44m, -0.46m]$ for symmetric parking in one maneuver. The initial orientation of the vehicle was chosen to be $\theta(0) = -\pi/2rad$. The initial coordinates of the vehicle reference point *P* with respect to an inertial frame *Gxy* with center placed in the goal position *G* of the vehicle in the parking place (Fig. 3), and which has its *x*-axis aligned with the *x*-axis of *Fxy* are $(x_P(0), y_P(0)) = (7.5m, -4.5m)$. The maximum value of the vehicle velocity was chosen to be $|v_P| = 0.3m/s$. The values of the saturated *tanh*-type controller were $K_t = 8, K = 5.85, a_0 = 0.17$.

Starting from identical initial conditions, the planar paths of the vehicle using bang-bang control and saturated (*tanh*-type) control are presented in Fig. 5. As seen from the simulation, the vehicle trajectories are quite similar. This result shows that the saturated control can be used instead of bang-bang control in order to steer the vehicle into the parking place according to the geometrical considerations for collision-free reverse perpendicular parking in one maneuver presented in Section II.

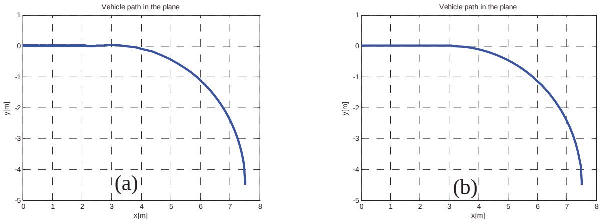


Figure 5. Perpendicular parking: Planar paths of the vehicle using bang-bang control (a) and saturated control (b).

Evolution in time of the front-wheel steering angle by using bang-bang control and saturated control is presented in Fig. 6. The simulation results show the advantage of the saturated control: the chattering occurring using bang-bang-control, when the position of the vehicle is in the vicinity of the tracking line, and the orientation error is also small, is avoided.

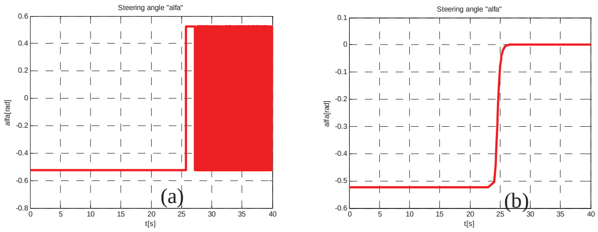


Figure 6. Perpendicular parking: Evolution in time of the front-wheel steering angle using bang-bang control (a) and saturated control (b)

An animation of the perpendicular reverse parking in one maneuver using saturated *tanh*-type steering control is shown in Fig. 7.

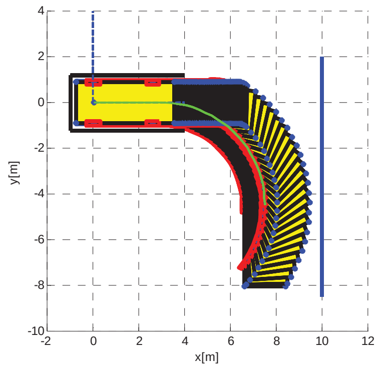


Figure 7. Perpendicular reverse parking using saturated control

The saturated *tanh*-type controller has been implemented on an experimental automatic electric vehicle CyCab and initial tests of perpendicular reverse parking has been initialized (Fig. 8). In the first tests, only information from the encoders mounted on the wheels were used for determining the position of the vehicle with respect to an inertial frame attached to the goal position into the parking place. The dimensions of the CyCab are: $l = 1.2m$; $b = 1.2m$; $l_1 = l_2 = 0.35m$; $\alpha_c = \alpha_{max} = \pi/6rad$ and $\rho_c = \rho_{min} = 2.08m$. The assigned values for the parking aisle and the parking place were chosen to be $h_{cd} = 3m$ and $h_{pd} = 2m$, respectively. For symmetric parking into the parking place, according to (18) and (11) the offset s can take values in the closed interval $-|s| \in [-|s|_m, -|s|_{max}] = [-1.01m, -0.095m]$. For the experiment shown in Fig. 8, the initial coordinates of the vehicle with respect to *Gxy* with center placed at the goal position in the parking place were approximately $(x_P(0), y_P(0)) = (3m, -2.1m)$. The first experiments confirm the effectiveness of the proposed controller.

V. Conclusion

In this paper, the problem of perpendicular reverse parking of front wheel steering vehicles was considered. Geometric considerations for collision-free perpendicular parking in one reverse maneuver were first presented, where the shape of the vehicle and the parking environment were expressed as polygons. Relationships between the widths of the parking aisle and parking place, as well as the parameters and the initial position of the vehicle have been given, in order to plan a collision-free maneuver, in the case, when the car has to be symmetrically positioned into the parking place. Two types of steering controllers (bang-bang and saturated controllers) for straight-line tracking have been proposed and evaluated. It was demonstrated that, the saturated *tanh*-type controller, which is continuous, was able to achieve also quick steering avoiding chattering and can be successfully used in solving parking problems. Simulation results and the first experiments with a test vehicle confirm the effectiveness of the proposed control scheme.

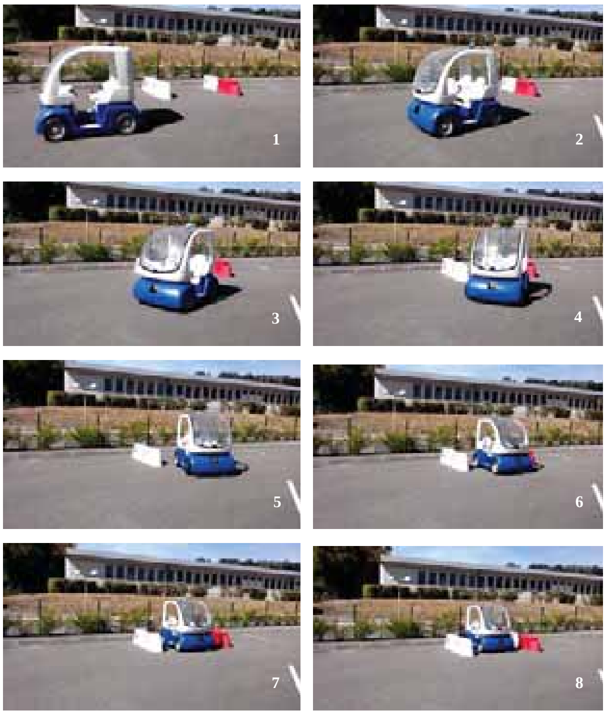


Figure 8. Automatic perpendicular parking of a CyCab vehicle

References

[1] USAF - LANDSCAPEDESIGN GUIDE, available at: http://www.ttap . mtu.edu/publications/2007/ParkingDesignConsiderations.pdf.

[2] B. Gutjahr and M. Werling, Automatic collision avoiding during parking maneuver – an optimal control approach, *In Proc. 2014 IEEE Intel.. V. Symposium*, 2014, pp. 636-641.

[3] S. Blackburn, The geometry of perfect parking, Available at: http://personal.rhul.ac.uk/uhah/058/perfect_parking.pdf.

[4] C. Pradalier, S. Vaussier and P. Corke, Path planning for parking assistance system : Implementation and experimantation, *In Proc. Austr. Conf. Rob. Automation*, 2005.

[5] J. Moon, I. Bae, J. Cha, and S. Kim, A trajectory planning method based on forward path generation and backward tracking algorithm for automatic parking systems, *In Proc. IEEE Int. Conf. Intell. Transp. Systems*, 2014, pp. 719-724.

[6] K. Lee, D. Kim, W. Chung, H. Chang, and P. Yoon, Car parking control using a trajectory tracking controller, In Proc SICE_ICASE Int. J. Conference, 2006, pp. 2058-2063.

[7] P. Petrov and F. Nashashibi, Saturated feedback control for an automated parallel parking assist system, *In Proc. IEEE Conf. Contr. Autom. Rob. Vision*, 2014, pp. 577-582.

[8] P. Petrov, C. Boussard, S. Ammoun, and F. Nashashibi, A hybrid control for automatic docking of electric vehicles for recharging, *In Proc. IEEE Int. Conf. Rob. Automation*, 2012, pp. 2966-2971.

[9] Avalable at: https:// www.youtube.com/watch?v=YbrwUrxFybQ

[10] Avalable at: https://www.youtube.com/watch?v=b_m8DqTlOLE

Obstacle segmentation with low-density disparity maps

Daniela A. Ridel, Patrick Y. Shinzato and Denis F. Wolf[1].

Abstract— **Autonomous vehicles aim at decreasing the high level of accidents that have occurred year after year. Several requirements as localization, mapping, recognition of traffic signs and detection of obstacles must be satisfied for the development of such intelligent systems. This paper presents a method for the detection of general obstacles by stereo vision. The system uses a disparity value and a cost associated with each pixel and its 24-neighborhood information to segment the image into obstacles. It was evaluated in two different datasets and with hand-labeled images as ground truth. The quantitative and qualitative results showed satisfactory in different scenarios and the average computation time of 55 ms for KITTI dataset images (disparity processing included) enables the application of the method on our platform for the online detection of obstacles.**

I. INTRODUCTION

Autonomous vehicles have drawn the interest of the academic community and automotive industry over the past years, as they can assist in solving problems faced by the current transportation system, such as collision and heavy traffic flow. This technology also enables the development of Intelligent Transportation Systems to improve the time of use of vehicles and reduce the growing automobile fleet.

Several requirements, as localization, mapping, recognition of traffic lights and signs must be satisfied for a vehicle to be driven by itself. Detection and tracking algorithms enable a dynamic modeling of obstacles around a vehicle for the planning and execution of actions, as lane changing and overtaking, and prevention of any type of collision.

Some approaches on obstacle detectors have focused on specific types of obstacles and generally require a training step. The use of such methods in real applications would lead to high computational costs for the support of a specific classifier for each type of obstacle. Moreover, some new obstacles may not be detected, as no classifier has been trained for it. Other approaches require the tuning of a large number of parameters and use of high-cost sensors.

This paper proposes a general obstacle detection system based on vision sensors, widely addressed in the academic community and commercially viable for applications in industries for assisting autonomous vehicle navigation. The system considers any obstacle, as pedestrians, cars, cyclists, poles, trees and traffic signs in the environment operated by the autonomous vehicle. Such a detection system will be further joined to a full obstacle tracking system and CaRINA platform [14] (Fig. 1), designed by the LRM[1] (Mobile Robotics Lab) at USP (University of Sao Paulo).

Fig. 1: CaRINA platform.

Since the relative position of any obstacle is important for the planning of actions, the system will use a sensor, i.e. a stereoscopic camera to provide information on metrics. The sensor consists of a pair of monocular cameras and enables the construction of a disparity map representing the depth of each image pixel. From the image pixel coordinates (u,v) combined with the disparity value (d), the system generates a map of costs for each pixel based on [3] and [1], which represents a cost for each pixel to be part of either an obstacle or a navigable area. It is segmented according to the cost and disparity value for the detection of obstacles inside the camera view, as shown in Fig. 2. For evaluation purposes, we used images from the KITTI object detection dataset [5] (left side of Fig. 2) and images captured on our platform (right side of Fig. 2).

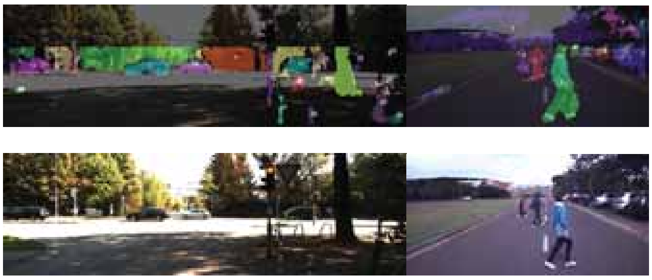


Fig. 2: Clustered points. Each color represents an obstacle..

The main advantages of the proposed approach are it requires neither dense disparity maps, which improves the

[1]The authors are with Institute of Mathematics and Computer Science (ICMC), University of São Paulo (USP), São Carlos, Brazil. danielaridel@usp.br, shinzato@icmc.usp.br and denis@icmc.usp.br

[1] http://www.lrm.icmc.usp.br/

time of computation, nor a training phase or prior knowledge on the environment, and makes no assumptions about the environment, as flatness of surfaces.

II. RELATED WORK

Several approaches have been developed to solve the problem of detection of obstacles through the use of active sensors, such as LIDAR (Light Detection and Ranging) and radar and passive sensors, as cameras, or even the fusion of different types of sensors [15].

Many algorithms detect specific obstacles, as people or cars (a review on vision-based vehicle detection is provided in [9]) and some approaches separate dynamic and static obstacles ([7], [16] and [8]).

Other methods perform the detection task in a general manner, i.e. they do not distinguish between different obstacle classes, but between obstacles and non obstacles. Erbs et al ([10]) segmented obstacles using stereo vision and stixel representation. In [6], the obstacles were detected in a general manner by v-disparity and clustering using DBSCAN (Density-Based Algorithm for Discovering Clusters) [13] algorithm.

III. SYSTEM OVERVIEW

Given a disparity map, a cost is calculated for a subsample of pixels, according to [3] and [1] and followed by a clustering step for the segmentation of those points into obstacles.

The entire system follows a sequence of four processing steps. The first consists in running a correspondence algorithm for the creation of a disparity map. In the next step, pixels equally distributed in the image are selected for the creation of a graph that contains nodes $p(x,y,z,u,v,d)$, where x,y,z are the 3D coordinates from a reconstruction process, u,v are screen coordinates and d is the disparity value. In the third step a cost that represents how likely a single pixel could be an obstacle is computed. The fourth step corresponds to the clustering, which will transform points into object entities. Figure 4 shows intermediate results of this process and details on how to compute the cost values are available in [1].

A. Stereo Correspondence

Prior to the execution of a stereo correspondence algorithm, a histogram equalization method is applied to both images (left and right) to smooth noise caused by shadows. A region of interest of $height = \frac{2}{3} original\_height$ is selected and only the disparity for lines with step λ is computed, for improvements in the speed performance. The disparity image was constructed by a modified version of *Efficient Large-Scale Stereo Matching* (ELAS) [11] with no post processing filters. The resulting disparity map (Fig. 4b) was then used in the selection step for the construction of a graph. Other methods that calculate disparity maps can also be used.

B. Selection of Points

An undirected graph $G = \{P,E\}$ is created in this step. P is the set of nodes that represents sparse points (x,y,z,u,v,d) and E is the set of all edges that guarantees there exists no other point between two points connected by an edge. Differently from previous works, the edges of G are not generated by The Planar Delaunay Triangulation Method [12], but according to the valid 24-neighborhood of each pixel (Fig. 3). This modification enables the system to respond at higher frequency rates, because of the constant time access.

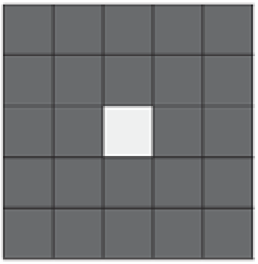


Fig. 3: 24-neighborhood.

P must contain points equally distributed in the image (i.e., distance-to-any-neighbour $\geq \lambda$) with a valid disparity value (higher than zero) for the extraction of a sparse set of points.

C. Cost Computing of Points

In this step, a cost value C is calculated for each node p from graph G, where $0.0 < C \leq 1.0$ and represents a confidence level of node p being an obstacle. Using graph G from the previous step, our approach calculates C by Equation 1 (more details in [3] [1]), which depends only on the local spatial-condition represented by Equation 2 for each pair of nodes connected by an edge. The resulting image (Fig. 4c) is used in the point clustering step. Equation 1 is defined as

$$C(p) = \frac{1}{||Q(p)||} \sum_{n \in Q(p)} sigmoid\left(\frac{|p_y - n_y|}{||p-n||}, \cos(\theta), \sigma\right), \quad (1)$$

where $|p_y - n_y|$ corresponds to the absolute difference of height values between these two points, $||p-n||$ is the distance between points p and n, θ and σ are constants, $Q(p)$ is a set of all 24 neighbours of p, and $sigmoid()$ is a function defined in Equation 2:

$$sigmoid(x,\mu,\sigma) = 0.5 + \frac{0.5(x-\mu)\sigma}{\sqrt{1+(x-\mu)^2\sigma^2}}, \quad (2)$$

where μ specifies the effective threshold and σ is a scale parameter that influences the tangent slope at the threshold. The intuition behind this method is derived from [3], in which the obstacle estimation was binary.

D. Point Clustering

Both disparity and cost values were used as inputs for the grouping of points into obstacles. The first point, p, is selected and, if it is not part of a cluster, a new cluster label is assigned to it. All points that belong to its neighborhood are retrieved and for each neighbor q ($q \subset N$, where N is the set of neighbors of point p) to be considered part of the same cluster of p, the three following conditions must be satisfied:

1) q is not part of a cluster
2) $|(p_d - q_d)| \leq \eta$
3) cost of $q > \alpha$,

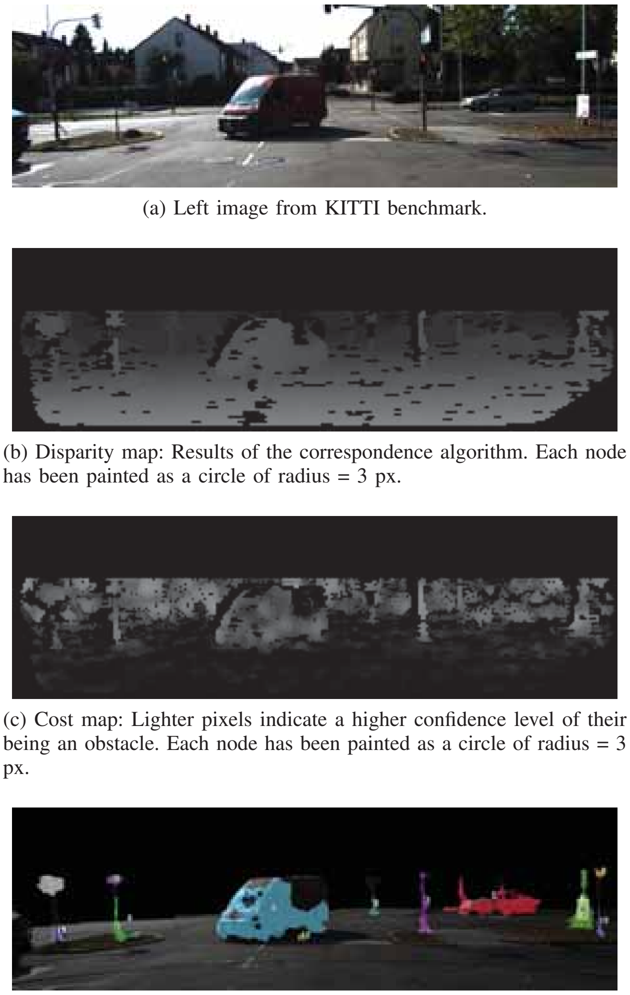
where p_d and q_d correspond to the disparity values of the points. If point q has suited the conditions, it becomes part of the same cluster of p. The process continues for each neighbor that has become part of the same group until all neighbors have been processed. After all neighborhood of p has been analyzed, another point is chosen and the same process is repeated. The algorithm differs from the well known DBSCAN because it does not use the concepts of border or core points and checks the condition of the minimum number of points for the validation of a cluster. If a cluster has enough points, it is considered an obstacle, otherwise the entire cluster is considered noise.

The condition that takes into account the chance of a point being an obstacle is used to separate points in obstacles from points in a navigable area. The disparity condition groups points of a certain uniformity and sudden changes in the disparity map will not cluster neighboring points to the same obstacle. The values used for η and α were 1 and 0.1, respectively. At the end of the algorithm, a certain number of clusters corresponding to obstacles will have been detected (Fig. 4d).

IV. Experimental Results and Discussion

The quantitative evaluation of a general obstacle detector algorithm is not an easy task, as even for humans, it might be difficult to segment obstacles in complex scenarios, and no benchmark, without the classification step, is available for the comparison of different approaches. This section addresses the evaluation method and the construction of ground truth data and discusses some of the experimental results.

Two hundred images were selected - a hundred from the object detection benchmark of KITTI and a hundred from a sequence captured on our platform. They contain different types of obstacles interacting with the vehicle, which implies it performs some action, such as braking or overcoming. All visible obstacles were hand-labeled for the construction of the ground-truth. They were identified based on the spatial condition in the scene, e.g. if there is a pole between the camera and a car, (which, in the camera view, splits the car in the middle), the pole and the car will be considered three obstacles, namely left side of the car, the pole and right side of the car. If the boundaries of more than one obstacle are not clear, they are labeled as same obstacle (e.g., two people walking together). Obstacles far from the road or sidewalk are considered background and are grouped into a single obstacle. The range was limited to a maximum number of 30 meters, experimentally determined because the disparity values are not very accurate after this distance.



(a) Left image from KITTI benchmark.

(b) Disparity map: Results of the correspondence algorithm. Each node has been painted as a circle of radius = 3 px.

(c) Cost map: Lighter pixels indicate a higher confidence level of their being an obstacle. Each node has been painted as a circle of radius = 3 px.

(d) Obstacles detected. Each color represents a cluster. Each node has been painted as a circle of radius = 3 px.

Fig. 4: Data from Figs. 4b and 4c were used as input to the clustering algorithm (Fig. 4d).

In Experiment 1 (section IV-B), the LIDAR data available on KITTI were used for the assignment of a distance to each pixel. Pixels whose distance was longer than 30 meters were not considered. In Experiment 2 (section IV-C), we used the disparity data to project the distance to each pixel. All experiments were conducted in an Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz of 8GB RAM memory running ubuntu 14-04.

A. Evaluation method

For each group of points detected by the clustering algorithm, a convex hull polygon representation was constructed, according to [4]. For each polygon detected, we associated a ground truth polygon in which the polygon detected showed a more overlapped area. More than one detected polygon could be associated with the same ground truth polygon.

Therefore, for each ground truth polygon, we calculated the intersection area with each associated polygon. The intersection areas between the detected polygons were decreased and the resulting area was divided by the area of the ground truth polygon. A hit was considered when the resulting area was greater than $\varphi = 0.4$. The metrics of precision and recall were calculated for each image.

Figure 5 illustrates the evaluation method - green represents true positives (TP - obstacles correctly detected), blue represents false negatives (FN - ostacles that should have been detected), red represents false positives (FP - detections that are not obstacles) and pink represents detections associated with a polygon of the ground truth of a score smaller than φ.

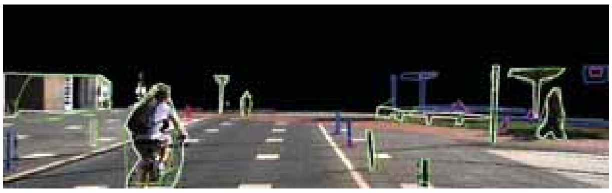


Fig. 5: Green represents TP, blue represents FN and red and pink represent FP (pink represents detections associated with a polygon of the ground truth of a score smaller than φ).

The unique different parameter in the two experiments was λ. In Experiment 1, $\lambda = 5$ and in Experiment 2, $\lambda = 10$, because the cameras of the datasets have different focal length and size of CCD sensor, therefore an adjustment had to be made to compensate for this difference. Due to this difference between the two sensors, the width (in pixels) of a detected person was smaller in KITTI than in our dataset.

B. Experiment 1

A hundred images from KITTI dataset were used in Experiment 1. The images have 1242×375 (465750) pixels and the computational cost running of the algorithm was 55 ms. The averages of precision and recall for all images are, respectively, 0.58 and 0.68. A high recall value is desired in an autonomous vehicle system, because all obstacles near the vehicle must be detected. The obstacles were correctly identified in most images, however, some were not identified due to a failure in the disparity maps and mainly because such maps lose precision in function of the distance. When the camera approaches the obstacle, the chance of its being detected increases. In some cases, visible obstacles can not be segmented by the disparity condition, which causes them to be merged into a single cluster.

In the hand-labeled images, when the boundaries of obstacles were not clear, they were labeled as a single obstacle. Therefore, when some people are detected as a single obstacle, it can also be considered a hit, depending on their proximity. Otherwise, when they have a visible boundary, they are grouped separately, e.g. cluster 14 in Fig. 10a.

The obstacles were correctly segmented in most images (people walking alone, Fig. 10a (10, 11, 12, 15, 17, 19 and 24), Fig. 10c (7 and 8), cars Fig. 10b (4, 5, 6 and 7), Fig. 10d (7), cyclists Fig. 10d (6 and 10), poles Fig. 10a (22, 25 and 28), Fig. 10b (1, 8 and 9) and Fig. 10d (11, 12, 16, 18, 20, 21, 24, 25 and 26), trees Fig. 10c (2 and 5)).

In some cases, the obstacles were not identified because of errors in the disparity map, or the group clustering of some obstacles by the algorithm (people walking side by side) and individual clustering in the ground truth (Fig. 10a (obstacle 14)). As we set the maximum disparity parameter to 100 in ELAS, some obstacles very close to the cameras were not detected (Fig. 6).

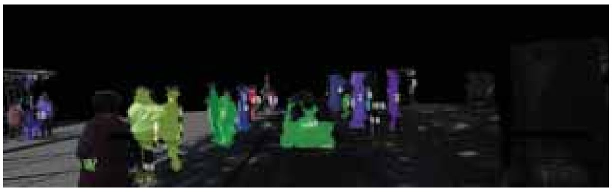


Fig. 6: Obstacles very close to the camera were not detected because of the maximum disparity parameter.

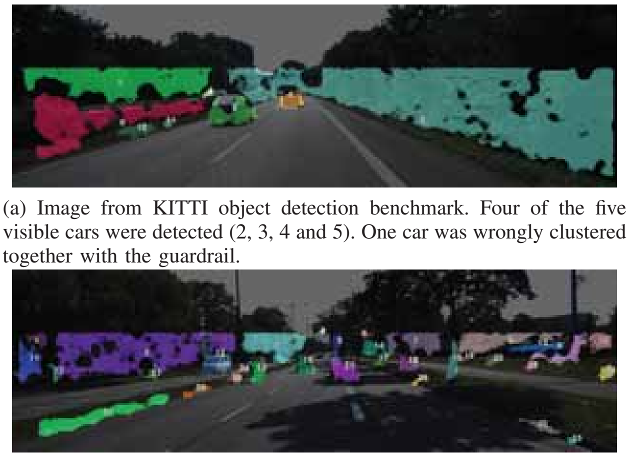
C. Experiment 2

We also evaluated the proposal using CaRINA dataset, our own data collected on our autonomous platform. The images were captured by the extended baseline of a Bumblebee XB3 (BBX3-13S2C-38) Point Grey (Fig 7). The evaluation procedure was the same conducted in Experiment 1; however, as we do not have the LIDAR 3D data to ensure the distance at each pixel, we projected it using disparity. People were correctly detected in Figs. 10e (3, 4 and 5) 10f (5, 6 and 7) and 10g (5 and 6), and cars in Figs. 10e (7 and 8), 10f (8 and 9) and 10g (7, 8 and 9). The precision and recall achieved were 0.63 and 0.76, respectively. The images have 1280×960 (a total of 1228800) pixels and the computational cost was 114 ms, including rectification, disparity map construction, cost computation and clustering.

Fig. 7: The images were captured by the extended baseline of a Bumblebee XB3 Point Grey.

D. Final considerations

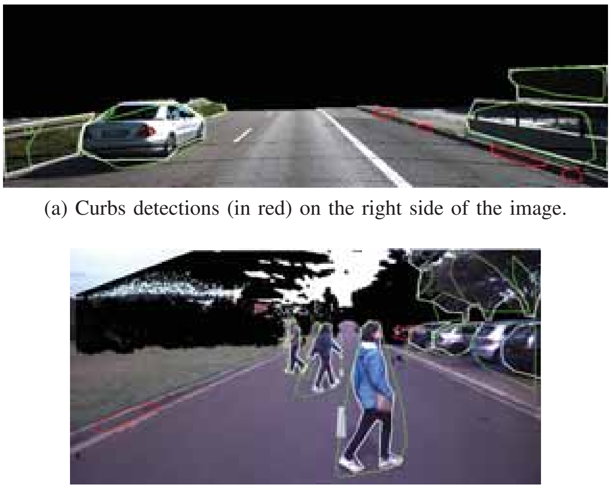
Although we have limited the range to 30 meters to evaluate the detection, our approach can reach farther obstacles, as shown in Fig. 8. Another reason for low precision is curbs were not considered obstacles in our ground truth (Fig.9), as they have different heights and shapes in the perspective view, which hampers their detection as a single obstacle. In a future evaluation, the inclusion of a ground detection can help the classification of such obstacles as curbs, for a higher precision rate.



(a) Image from KITTI object detection benchmark. Four of the five visible cars were detected (2, 3, 4 and 5). One car was wrongly clustered together with the guardrail.

(b) Image from KITTI object detection benchmark. Two distant cars (4 and 5) were identified.

Fig. 8: Figures with no range limitation.



(a) Curbs detections (in red) on the right side of the image.

(b) Curbs detections (in red) on the left side of the image.

Fig. 9: Example of images with a detected curb. The red and green polygons represent the result of the convex hull transform in the set of points of each cluster. Green represents true positives, red represents false positives and white represents the ground truth answer.

All results can be seen at `https://youtu.be/CD6X99RacgM`.

V. CONCLUSIONS

This paper has addressed the construction of an obstacle detection system to ensure safety to agents operating in the same environment of an autonomous vehicle. A stereo system was used as a perception sensor of the environment, partly because of the lower costs of cameras in comparison to other sensors, as LIDAR, and the increase in the computing power, which enables the processing of images at higher frequency rates. The approach was tested in two hundred images, half from KITTI Benchmark and half from CaRINA dataset. Preliminary results indicate this is a promising approach to be used in real time and future studies will focus on the implementation of tracking and distinction between dynamic and static obstacles.

ACKNOWLEDGMENT

The authors acknowledge the support granted by CNPq processes 132228/2014-9 and 168217/2014-7.

REFERENCES

[1] Shinzato, P. Y. (2015). *Estimation of obstacles and road area with sparse 3D points*. Doctoral Thesis, Institute of Mathematics and Computer Science (ICMC), University of Sao Paulo, Sao Carlos. Retrieved 2015-08-14, from http://www.teses.usp.br/teses/disponiveis/55/55134/tde-07082015-100709/en.php

[2] P. Y. Shinzato, D. F. Wolf and C. Stiller. ”Road terrain detection: Avoiding common obstacle detection assumptions using sensor fusion”. *Intelligent Vehicles Symposium Proceedings*, pages 687-692, 2014.

[3] P. Y. Shinzato and D. Gomes and D. F. Wolf. ”Road estimation with sparse 3D points from stereo data”. *Intelligent Transportation Systems (ITSC)*, pages 1688-1693, 2014.

[4] J. Sklansky. ”Finding the convex hull of a simple polygon”. *Pattern Recognition Letters*, pages 79 - 83, 1982.

[5] A. Geiger, P. Lenz and R. Urtasun. ”Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite”. *Conference on Computer Vision and Pattern Recognition (CVPR)* 2012.

[6] S. Kramm and A. Benshair. ”Obstacle detection using sparse stereovision and clustering techniques”. *Intelligent Vehicles Symposium (IV)*, pages 760-765, 2012.

[7] V. Guiziline and F. Ramos. ”Online self-supervised segmentation of dynamic objects”. *International Conference on Robotics and Automation*, pages 4720-4727, 2013.

[8] P. Lenz, J. Ziegler, A. Geiger and M. Roser. ”Sparse scene flow segmentation for moving object detection in urban environments”. *Intelligent Vehicles Symposium (IV)*, pages 926-932, 2011.

[9] S. Sivaraman and M. M. Trivedi. ”Looking at Vehicles on the Road: A Survey of Vision-Based Vehicle Detection, Tracking, and Behavior Analysis”. *Transactions on Intelligent Transportation Systems*, pages 1773-1795, 2013.

[10] F. Erbs, A. Witte, T. Scharwaechter, T. Mester and U. Franke. ”Spider-based Stixel object segmentation”. *Intelligent Vehicles Symposium Proceedings*, pages 906-911, 2014.

[11] A. Geiger, M. Roser and R. Urtasun. ”Efficient Large-Scale Stereo Matching”. *Computer Vision - ACCV 2010 - 10th Asian Conference on Computer Vision*, pages 25-38, 2010.

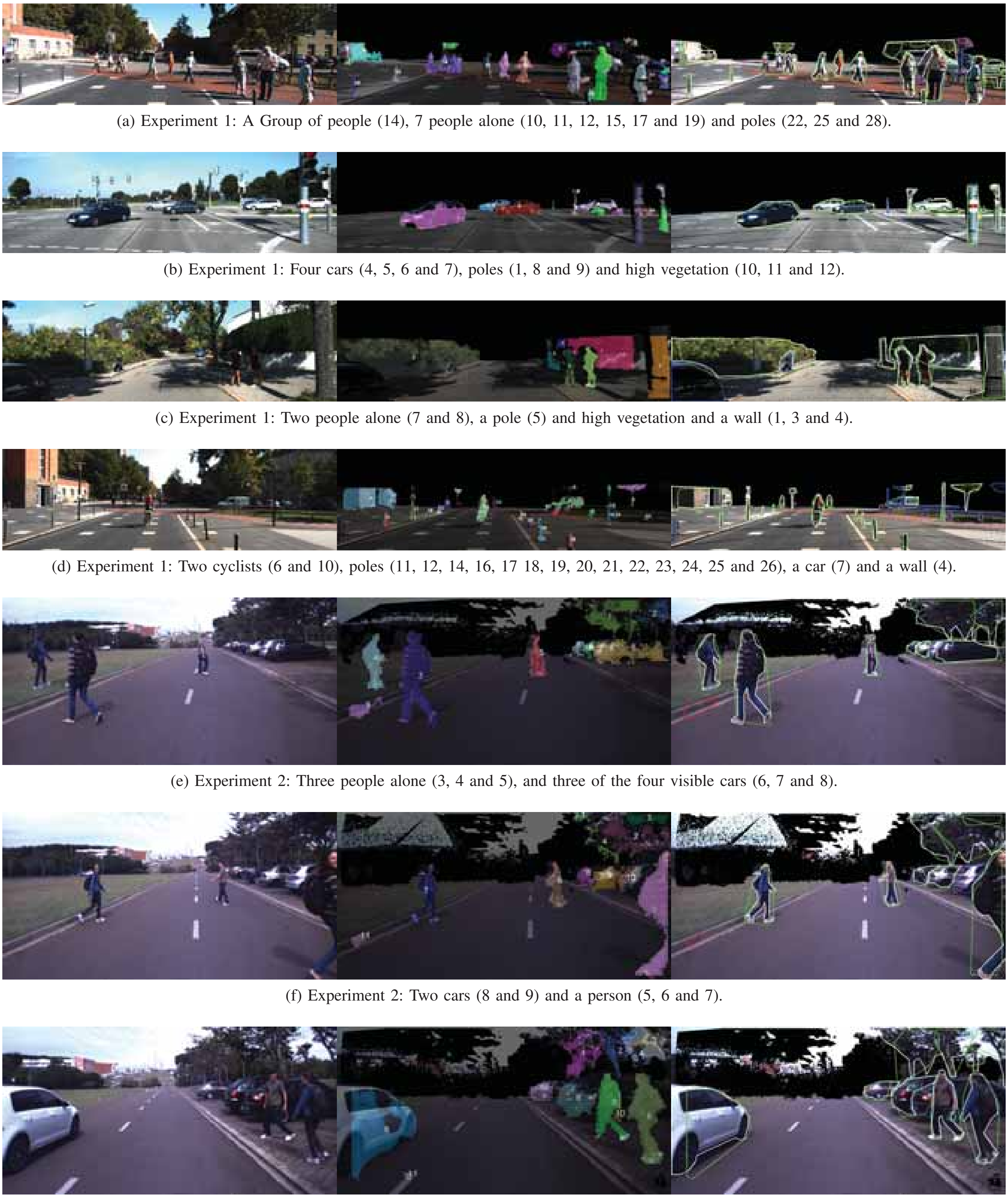
[12] D. T. Lee and B. J. Schachter. ”Two algorithms for constructing a Delaunay triangulation”. *International Journal of Computer and Information Sciences*, pages 219-242,1980.

[13] M. Ester, H. P. Kriegel, J. Sander and X. Xu. ”A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226-231, 1996.

[14] L. C. Fernandes, J. R. Souza, G. Pessin, P. Y. Shinzato, D. Sales, C. Mendes, M. Prado, R. Klaser A. C. Magalhes, A. Hata, D. Pigatto, K. C. Branco, V. Grassi Jr., F. S. Osorio and D. F. Wolf. ”CaRINA Intelligent Robotic Car: Architectural design and applications”. *Journal of Systems Architecture*, pages 372-392, 2014.

[15] T.-D. Vu, O. Aycard and F. Tango. ”Object perception for intelligent vehicle applications: A multi-sensor fusion approach”. *Intelligent Vehicles Symposium Proceedings*, pages 774-780, 2014.

[16] A. Bewley,V. Guizilini, F. Ramos and B. Upcroft. ”Online self-supervised multi-instance segmentation of dynamic objects”. *Robotics and Automation (ICRA)*, pages 1296-1303, 2014.



(a) Experiment 1: A Group of people (14), 7 people alone (10, 11, 12, 15, 17 and 19) and poles (22, 25 and 28).

(b) Experiment 1: Four cars (4, 5, 6 and 7), poles (1, 8 and 9) and high vegetation (10, 11 and 12).

(c) Experiment 1: Two people alone (7 and 8), a pole (5) and high vegetation and a wall (1, 3 and 4).

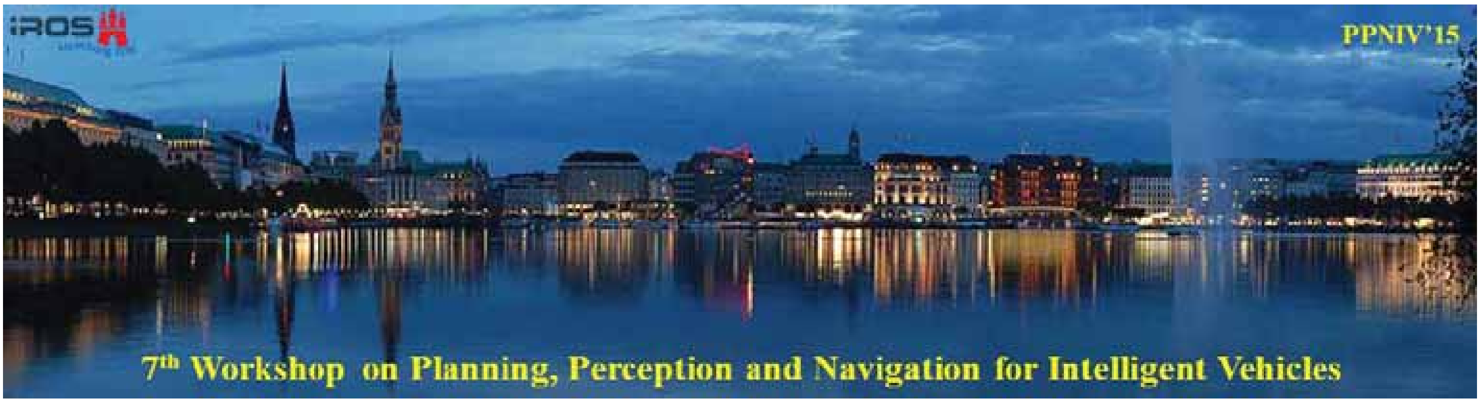
(d) Experiment 1: Two cyclists (6 and 10), poles (11, 12, 14, 16, 17 18, 19, 20, 21, 22, 23, 24, 25 and 26), a car (7) and a wall (4).

(e) Experiment 2: Three people alone (3, 4 and 5), and three of the four visible cars (6, 7 and 8).

(f) Experiment 2: Two cars (8 and 9) and a person (5, 6 and 7).

(g) Experiment 2: Three of the four visible cars (7, 8 and 9) and a person (5 and 6).

Fig. 10: Results of detection. The first column is the image from the left camera. The second column represents each cluster formed by its set of points and the third column shows the results from the evaluation with range limitation. White and blue represent ground truth obstacles, green represents obstacles considered correct and pink and red represent false detections.


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

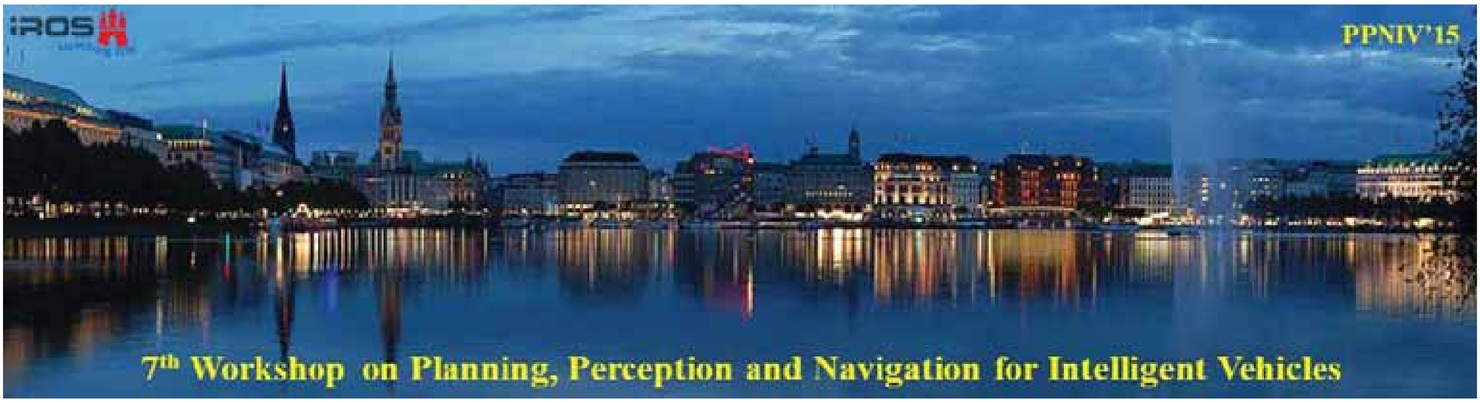
Session IV

Planning & Navigation

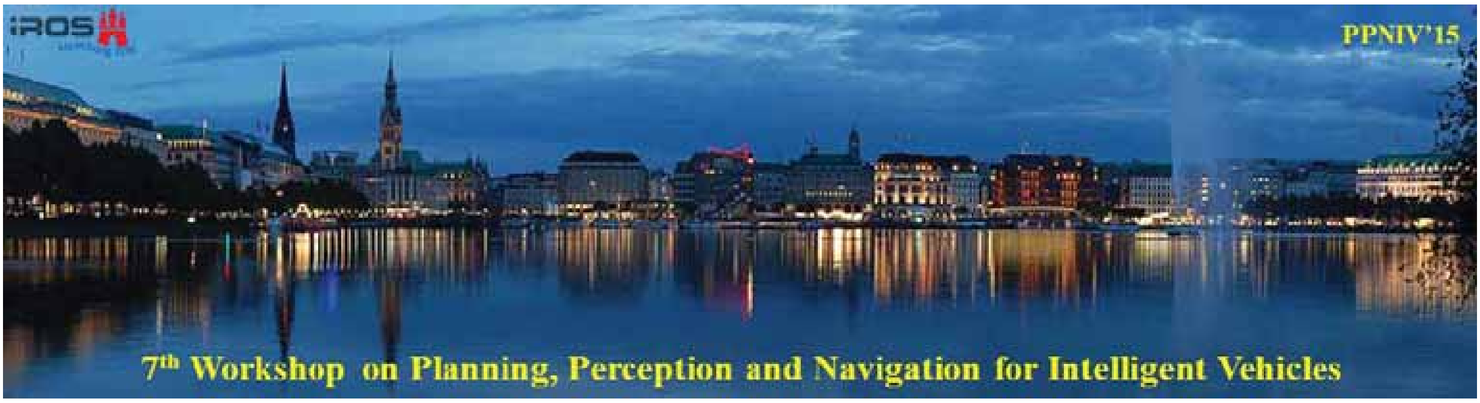
- **Keynote speaker: Pr Matthias Althoff (Technische Universität München, Germany)**
  Title: Determining the Nonexistence of Evasive Trajectories for Collision Avoidance Systems

- **Title: Safe prediction-based local path planning using obstacle probability sections**
  Authors: Tanja Hebecker and Frank Ortmeier

- **Title: Improving Monte Carlo Localization using Reflective Markers: An Experimental Analysis**
  Authors: Francesco Leofante, Gwénolé Le Moal, Gaëtan Garcia , Patrice Rabaté


PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems


IROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session IV

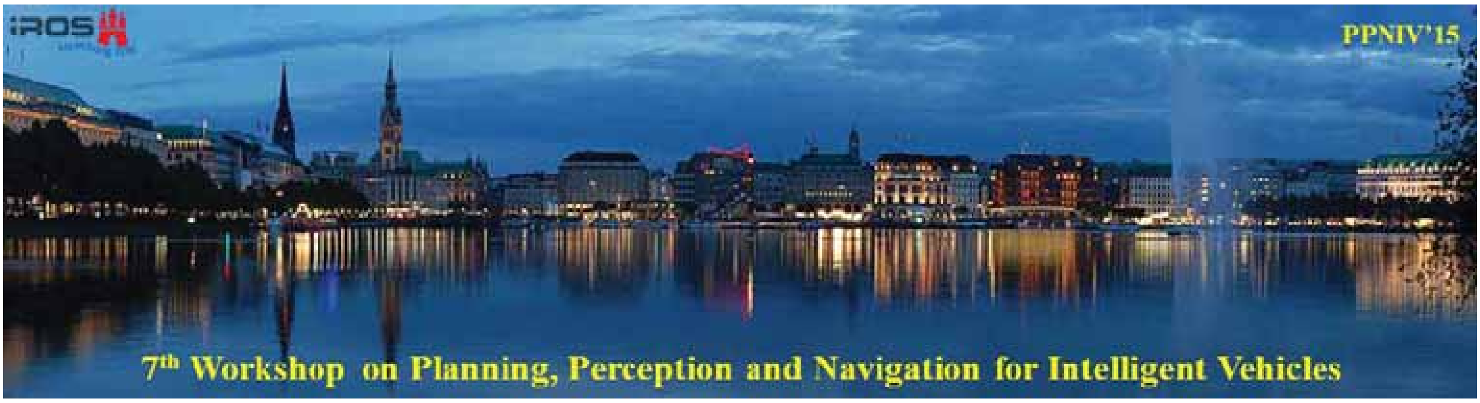
Keynote speaker: **Pr Matthias Althoff**
(Technische Universität München, Germany)

Determining the Nonexistence of Evasive Trajectories for Collision Avoidance Systems

Abstract: It is of utmost importance for automatic collision avoidance systems to correctly evaluate the risk of a current situation and constantly decide, if and what kind of evasive maneuver must be initiated. Most motion planning algorithms find such maneuver by searching a deterministic or random subset of the state space or input space. These approaches can be designed to be complete in the sense that they converge to a feasible solution as sampling is made denser. However, they are not suitable to determine whether a solution exists. In this paper, we present an approach which overapproximates the reachable set of the host vehicle considering workspace obstacles. Thus, it provides an upper bound of the solution set and it can report if no solution exists. Furthermore, the calculated set can be used for guiding the search of an underlying planning algorithm to find a solution as each trajectory of the host vehicle is ensured to be contained within this set.

Biography: Matthias Althoff received the diploma in Mechatronics and Information Technology from the department of mechanical engineering at the Technische Universität München, Germany, in 2005. He received his PhD degree (summa cum laude) in electrical engineering from the same university under the supervision of Univ.-Prof. Dr.-Ing./Univ. Tokio Martin Buss in 2010. From 2010 - 2012 he was a postdoctoral researcher at Carnegie Mellon University, USA, with a joint appointment in electrical engineering and the Robotics Institute. He joined the computer science department at Ilmenau University of Technology, Germany, in 2012 as assistant professor for automation systems. Since 2013 Matthias Althoff is assistant professor in computer science at the Technische Universität München.

His research interests include the design and analysis of cyber-physical systems, formal verification of continuous and hybrid systems, reachability analysis, planning algorithms, robust and fault-tolerant control. Main applications of his research are automated vehicles, robotics, power systems, and analog and mixed-signal circuits.


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Determining the Nonexistence of Evasive Trajectories for Collision Avoidance Systems

Matthias Althoff, Sebastian Söntges

Technische Universität München

September 28, 2015

Assessing the Risk of Traffic Situations

When to initiate an emergency maneuver?

- Collision "almost" unavoidable
- Approach: Test if there is none or only "very few" evasive trajectories

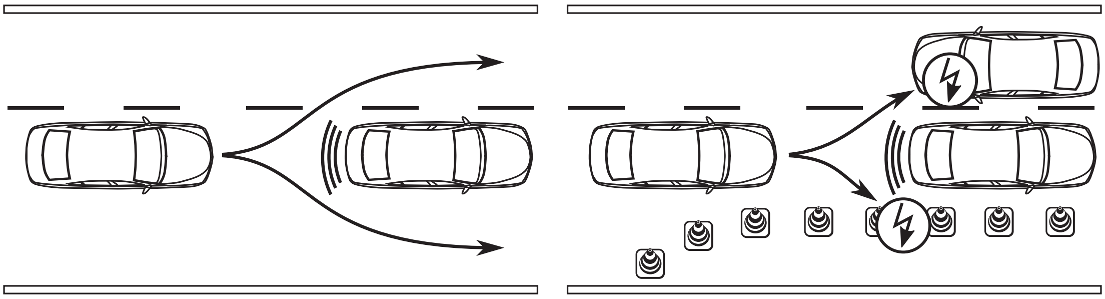


Figure: Existence of evasive trajectories

Uncountably Many Possible Situations

- How “many” future trajectories of other traffic participants are possible?
- How “many” evasive trajectories are there?

Answer for both: Infinitely many.

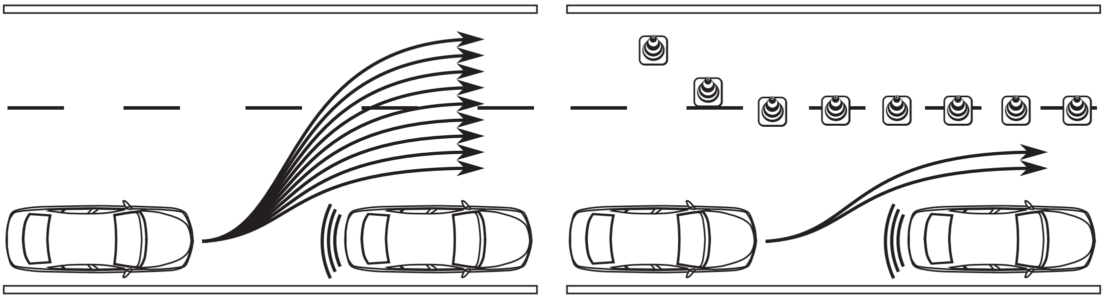
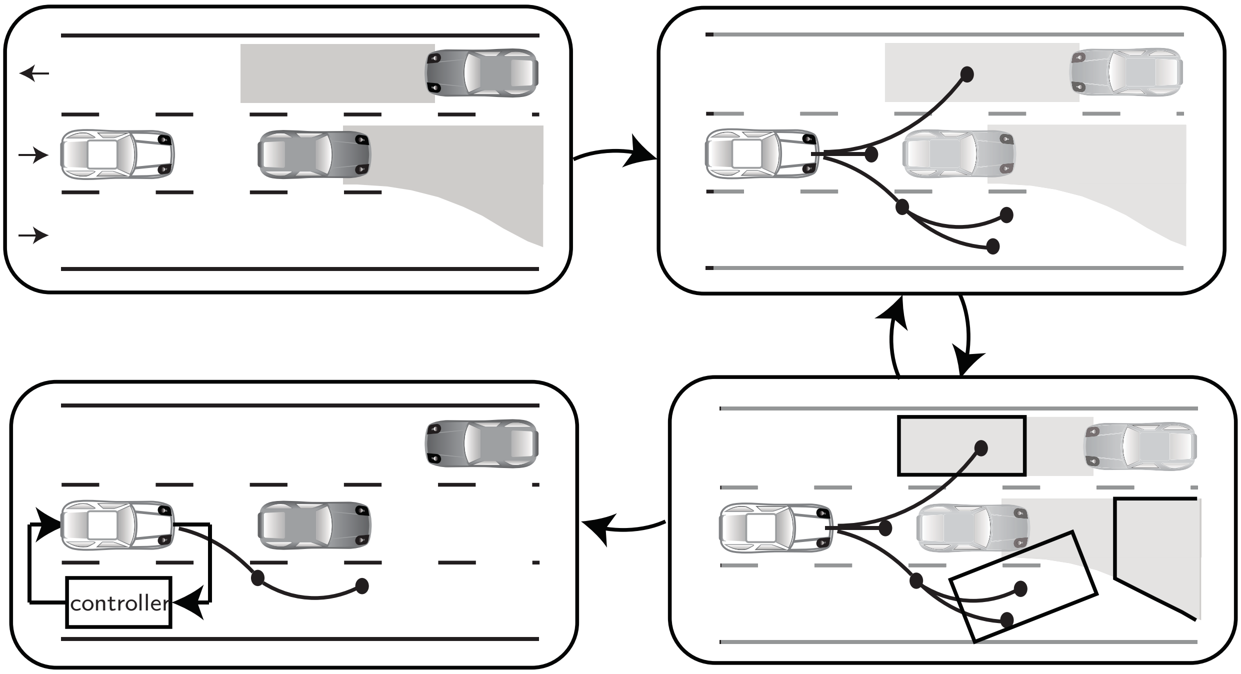


Figure: Set of evasive trajectories

Overview of Our Previous Approach

① occupancy prediction

② trajectory planning



④ trajectory tracking

③ collision checking

Outline

1. Model of the ego vehicle and other traffic participants
2. Set-based behavior prediction
3. Determining the nonexistence of evasive trajectories
   - Set representation for reachability analysis
   - Computational tricks
4. Test results
5. Further use of reachability analysis: Verification of single trajectories

Constraints for Traffic Participants

Initially the following constraints are considered:

$C1$: positive longitudinal acceleration is stopped when a parameterized speed v_{max} is reached.

$C2$: driving backwards in a lane is not allowed.

$C3$: positive longitudinal acceleration is inversely proportional with speed above a parameterized speed v_S (modeling a maximum engine power).

$C4$: maximum absolute acceleration is limited by a_{max}.

$C5$: actions that cause leaving the road/lane boundary are forbidden.

When a violation of a constraint of a traffic participant is sensed, it is no longer considered in future predictions for that particular traffic participant.

Physical Modeling (1)

Traffic participants are a point mass with

$$\sqrt{(a^{\mathrm{lat}})^2 + (a^{\mathrm{long}})^2} \leq a_{\max} \qquad \text{(Kamm's circle).}$$

The lateral acceleration is provided by a normalized steering input u_1:

$$a^{\mathrm{lat}} = a_{\max} u_1.$$

Due to the maximum absolute acceleration $(C4)$, the longitudinal acceleration is bounded by

$$a^{\mathrm{long}}_{\mathrm{c1}} = \sqrt{(a_{\max})^2 - (a^{\mathrm{lat}})^2}.$$

Physical Modeling (2)

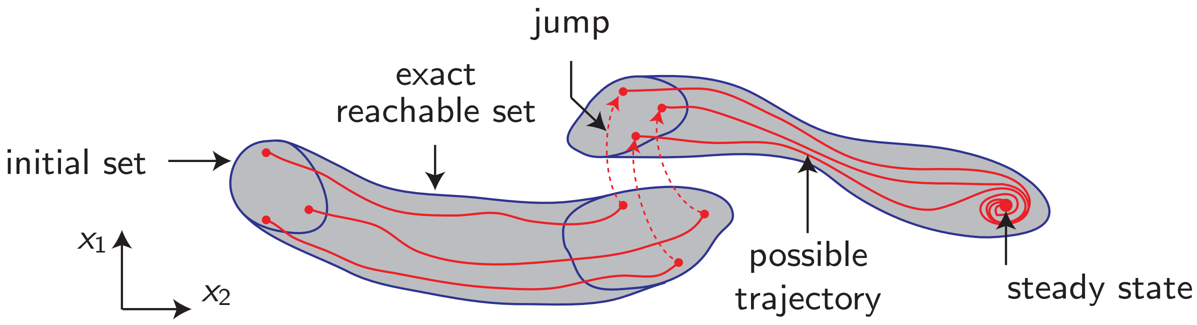
Constraints $C1$ (max. velocity), $C2$ (no driving backwards), and $C3$ (max. engine power) are considered in another bound:

$$a^{\mathrm{long}}_{\mathrm{c2}} = \begin{cases} a_{\max}\frac{v_S}{v}, & v_S < v < v_{\max} \wedge u_2 > 0 \\ a_{\max}, & (0 < v \leq v_S \vee (v > v_S \wedge u_2 \leq 0)) \\ 0, & v \leq 0 \vee (v \geq v_{\max} \wedge u_2 > 0) \end{cases}$$

The longitudinal acceleration combining both bounds is for a normalized acceleration input $u_2 \in [-1, 1]$

$$a^{\mathrm{long}} = \begin{cases} a^{\mathrm{long}}_{\mathrm{c2}} u_2, & a^{\mathrm{long}}_{\mathrm{c2}} |u_2| \leq a^{\mathrm{long}}_{\mathrm{c1}}, \\ a^{\mathrm{long}}_{\mathrm{c1}} \,\mathtt{sgn}(u_2), & a^{\mathrm{long}}_{\mathrm{c2}} |u_2| > a^{\mathrm{long}}_{\mathrm{c1}}. \end{cases}$$

Reachability Analysis

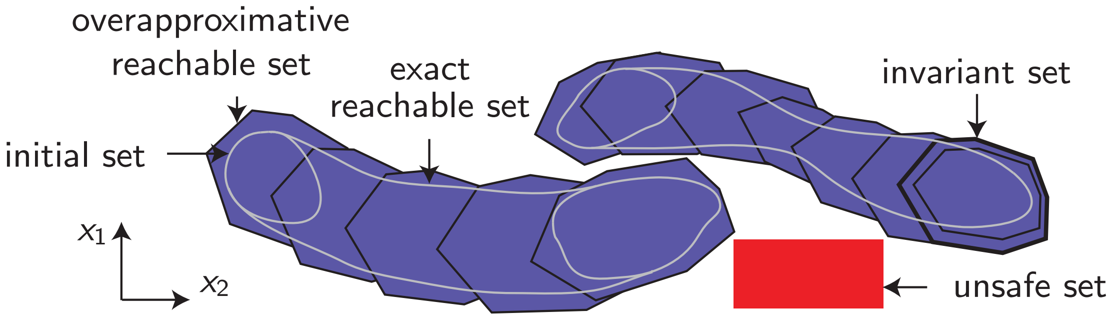


Informal Definition

A reachable set is the set of states that can be reached by a dynamical system in finite or infinite time for a

- set of initial states,
- uncertain inputs,
- and uncertain parameters.

Overapproximative Reachable Sets



- Exact reachable set only for special classes computable
  → overapproximation computed for consecutive time intervals.
- Overapproximation might lead to spurious counterexamples.
- Simulation cannot prove correctness.

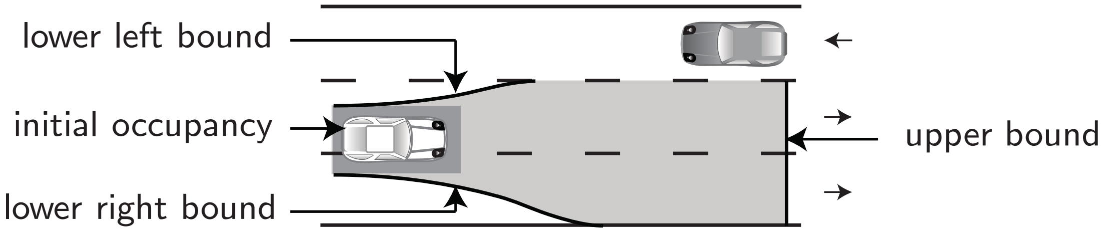
Abstraction Technique for Other Traffic Participants

Overapproximative Occupancy

Given are models M_i, $i = 1 \dots m$ which are abstractions of model M_0, i.e., $\mathrm{reach}(M_0) \subseteq \mathrm{reach}(M_i)$. The occupancy of the model M_0 can be overapproximated by

$$\mathrm{proj}\big(\mathrm{reach}(M_0)\big) \subseteq \bigcap_{i=1}^{m} \mathrm{proj}\big(\mathrm{reach}(M_i)\big). \qquad \square$$

Two models: Longitudinal dynamics along road boundaries (upper bound), lateral dynamics towards road boundaries (left/right bound).



Occupancy Along Road Boundaries

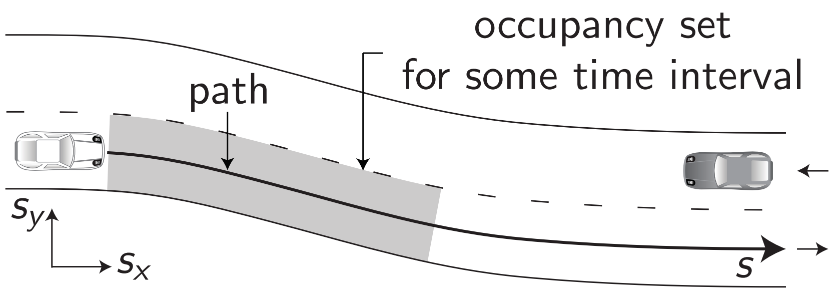
The dynamics becomes monotone when following a lane center.

Definition (Monotone dynamics)

For the initial state $x(0) \in \mathcal{R}(0)$ and inputs $u(t) \in \mathcal{U}$ the dynamics is monotone when the following holds for the solution $\chi(t, x(0), u(\cdot))$:

$$\text{if } \forall i, j, t \geq 0 : x_i(0) \leq \bar{x}_i(0), \quad u_j(t) \leq \bar{u}_j(t) \text{ then}$$
$$\forall i, t \geq 0 : \chi_i(t, x(0), u(\cdot)) \leq \chi_i(t, \bar{x}(0), \bar{u}(\cdot)). \qquad \square$$

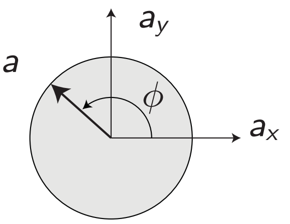
From this follows that e.g. the upper bound is provided by max. position, max. velocity, and max. acceleration:

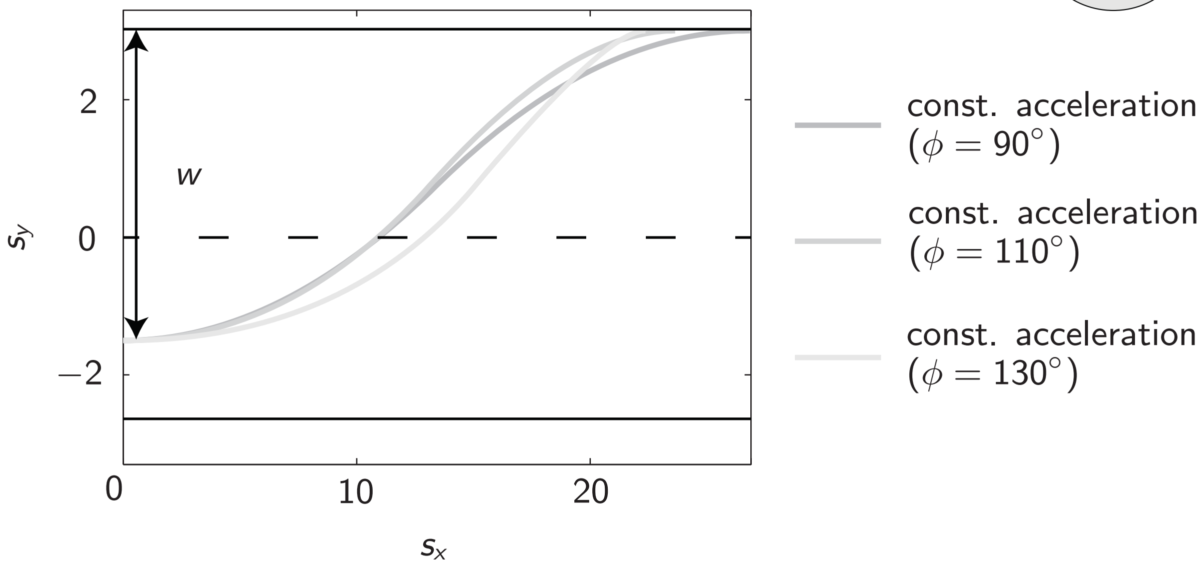


Occupancy Towards Road Boundaries

For lateral dynamics there exists no single combination of an initial state and an input trajectory determining the boundary.

Given the vehicle-fixed angle of the acceleration vector a, possible trajectories are:





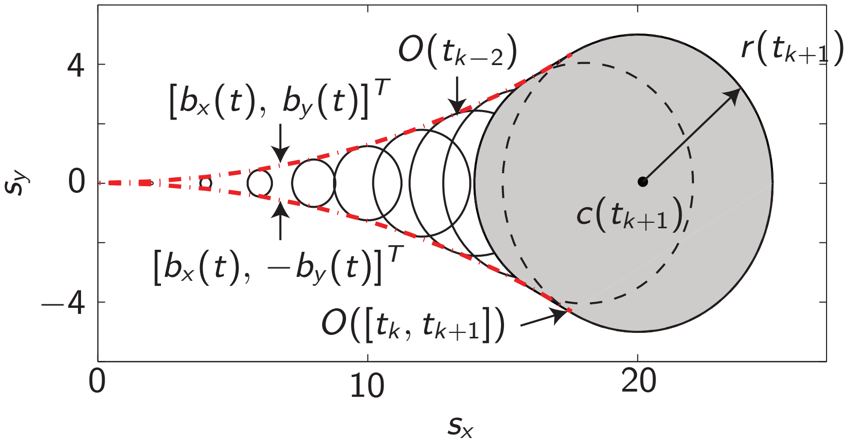
Occupancy Towards Road Boundaries: Method A

Using limit of absolute acceleration (constraint $C4$): Occupancies are circles with center $c(t)$ and radius $r(t)$:

$$c(t) = \begin{bmatrix} s_x(0) \\ s_y(0) \end{bmatrix} + \begin{bmatrix} v_x(0) \\ v_y(0) \end{bmatrix} t, \quad r(t) = \frac{1}{2} a_{\max} t^2.$$

From this follows the boundary of occupation:

$$b_x(t) = v_0 t - \frac{a_{\max}^2 t^3}{2 v_0}, \quad b_y(t) = \sqrt{\frac{1}{4} a_{\max}^2 t^4 - \left(\frac{a_{\max}^2 t^3}{2 v_0} \right)^2}.$$

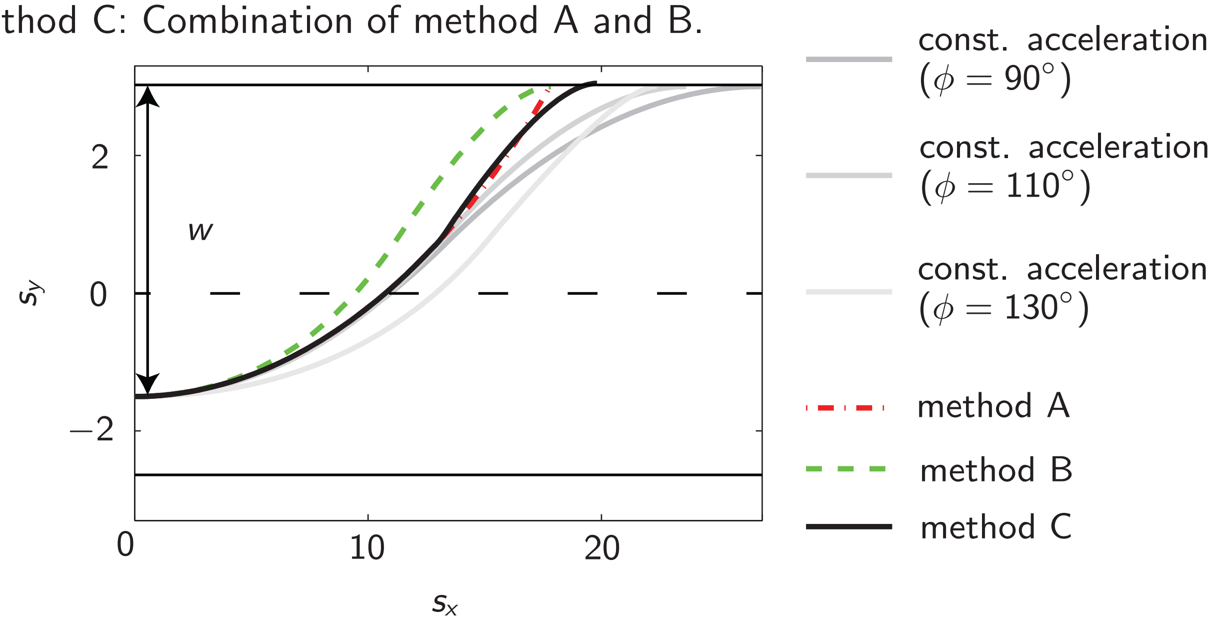


Occupancy Towards Road Boundaries: Method B and C

Method B: Assume independence of lateral and longitudinal acceleration → Solution of time t_s for switching the steering angle to avoid road departure:

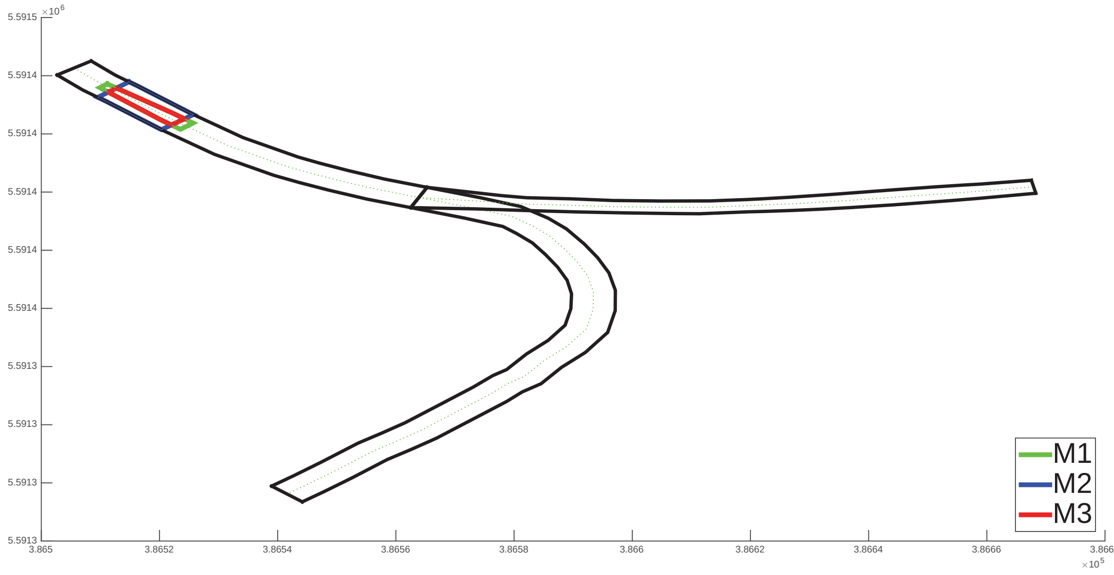
$$t_s = \frac{\sqrt{a_{\max} w + \frac{1}{2} v_0^2} - v_0}{a_{\max}}$$

Method C: Combination of method A and B.



Examples: Lane 1

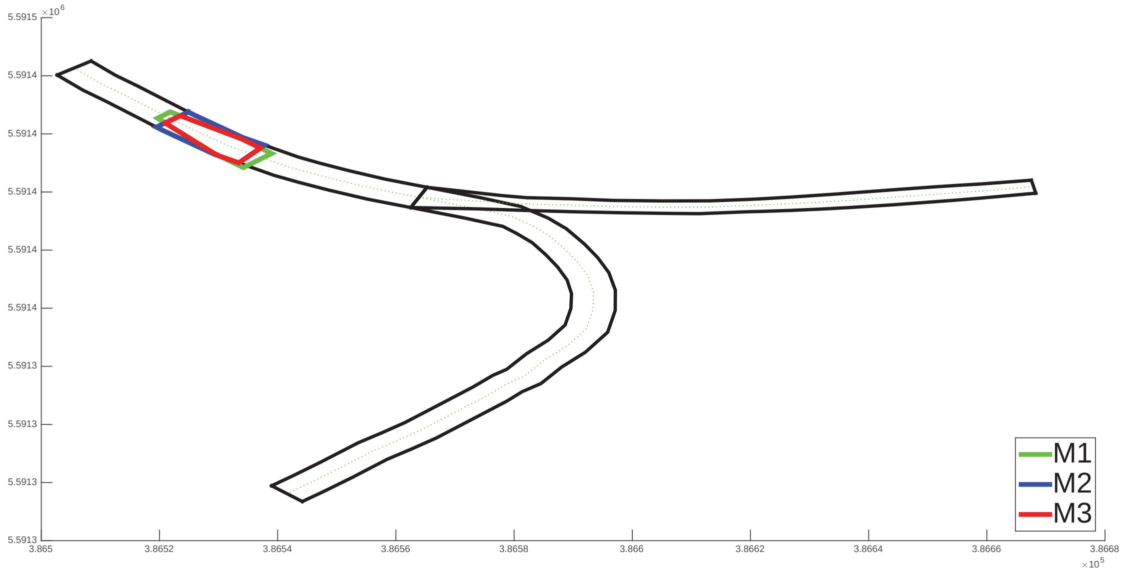
Step 1:



- M1: restricted absolute acceleration.
- M2: restricted acceleration and velocity in longitudinal direction.
- M3: staying within road boundaries.

Examples: Lane 1

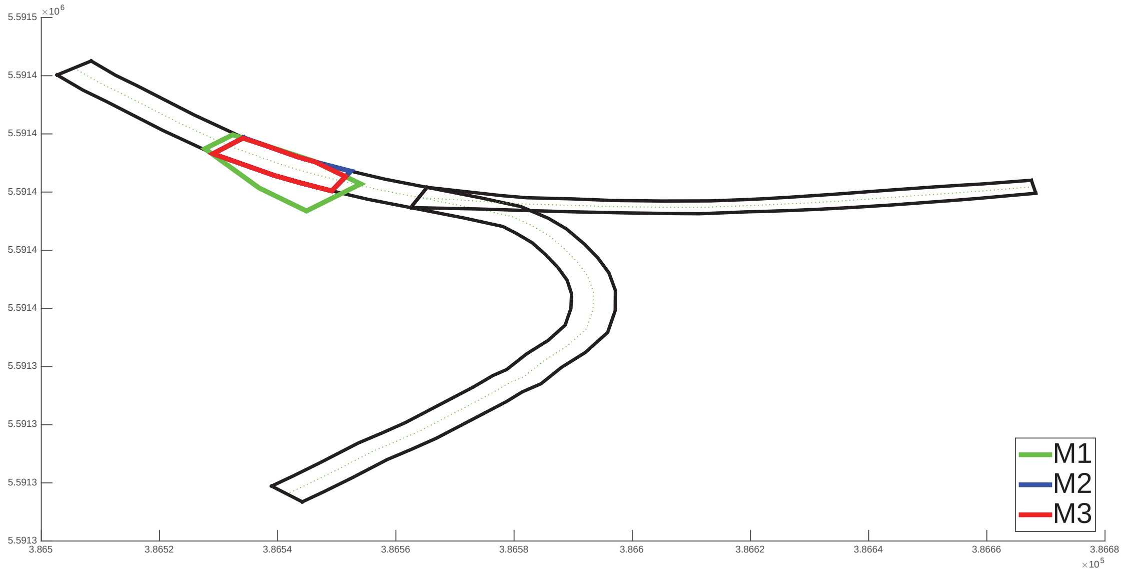
Step 2:



- M1: restricted absolute acceleration.
- M2: restricted acceleration and velocity in longitudinal direction.
- M3: staying within road boundaries.

Examples: Lane 1

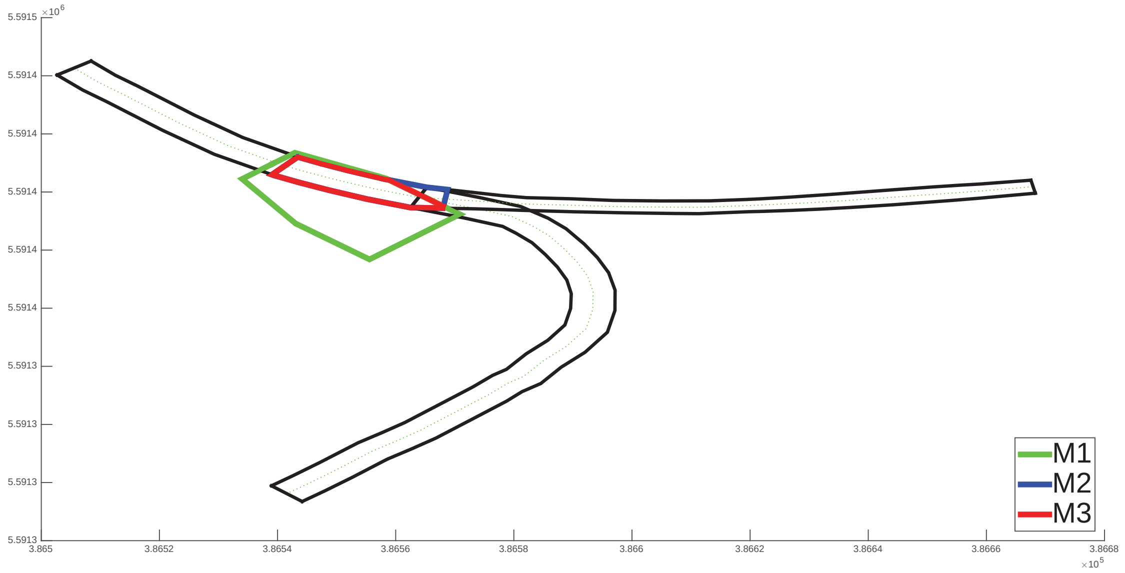
Step 3:



- M1: restricted absolute acceleration.
- M2: restricted acceleration and velocity in longitudinal direction.
- M3: staying within road boundaries.

Examples: Lane 1

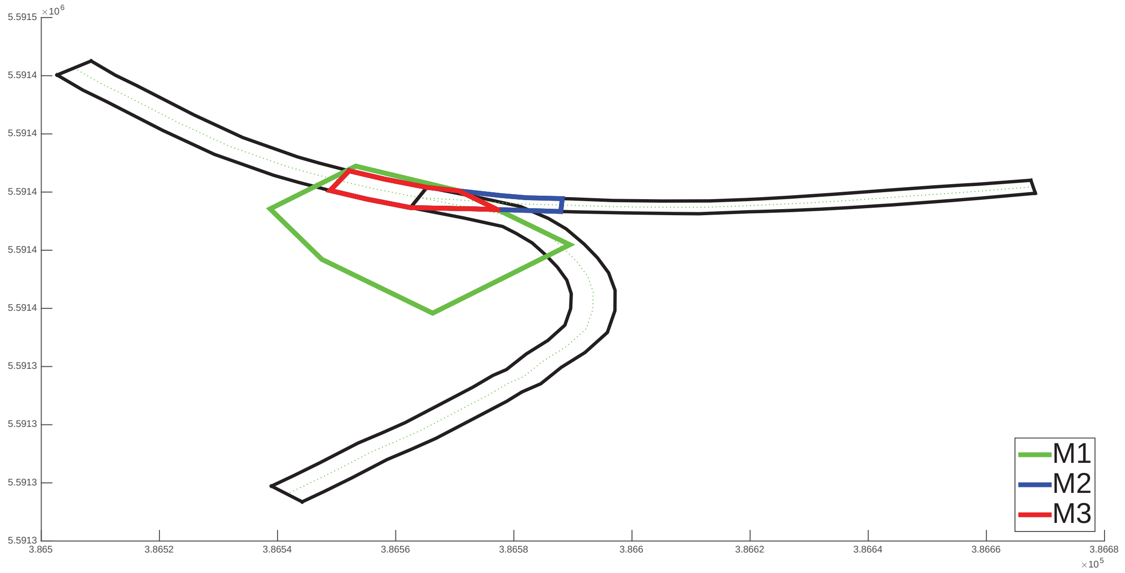
Step 4:



- M1: restricted absolute acceleration.
- M2: restricted acceleration and velocity in longitudinal direction.
- M3: staying within road boundaries.

Examples: Lane 1

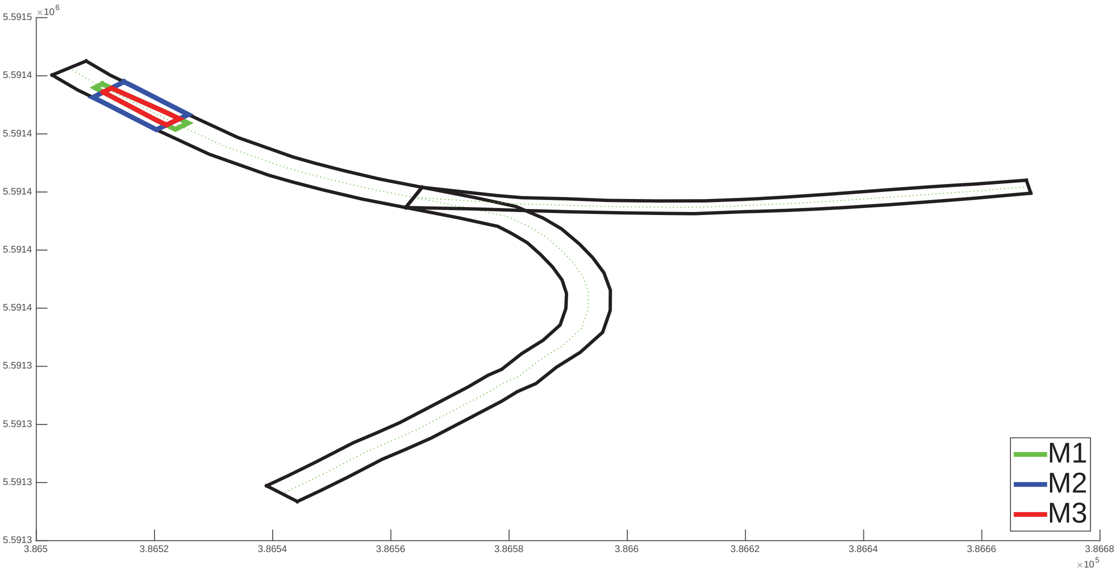
Step 5:



- M1: restricted absolute acceleration.
- M2: restricted acceleration and velocity in longitudinal direction.
- M3: staying within road boundaries.

Examples: Lane 2

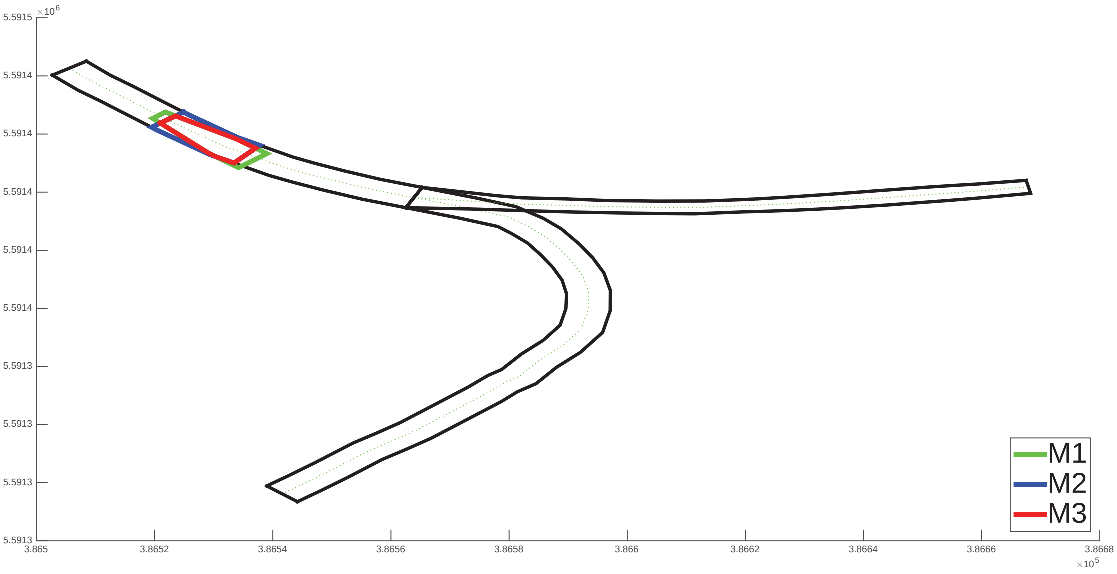
Step 1:



- M1: restricted absolute acceleration.
- M2: restricted acceleration and velocity in longitudinal direction.
- M3: staying within road boundaries.

Examples: Lane 2

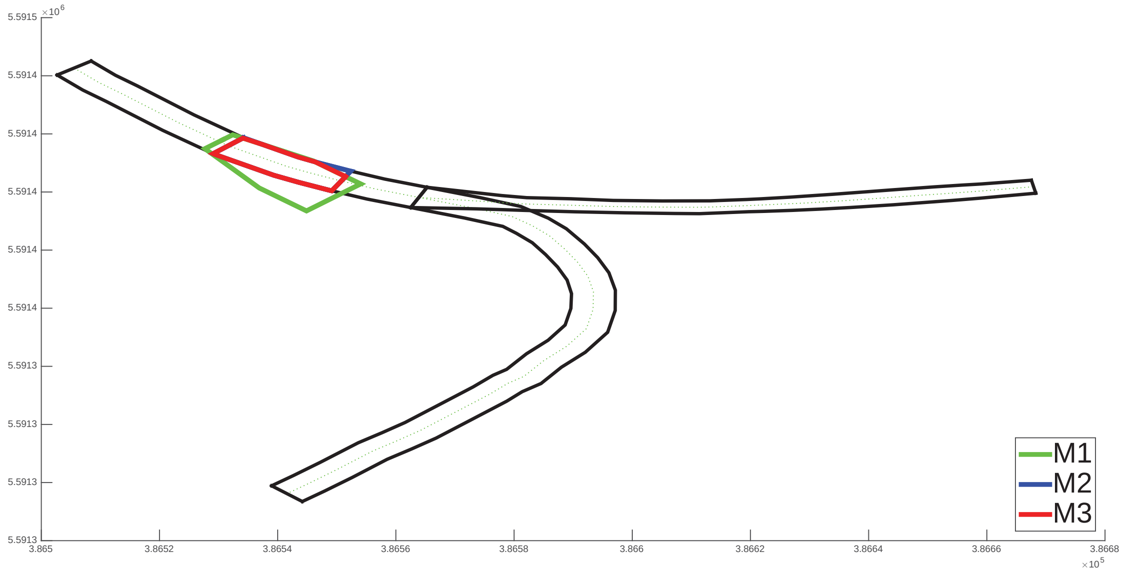
Step 2:



- M1: restricted absolute acceleration.
- M2: restricted acceleration and velocity in longitudinal direction.
- M3: staying within road boundaries.

Examples: Lane 2

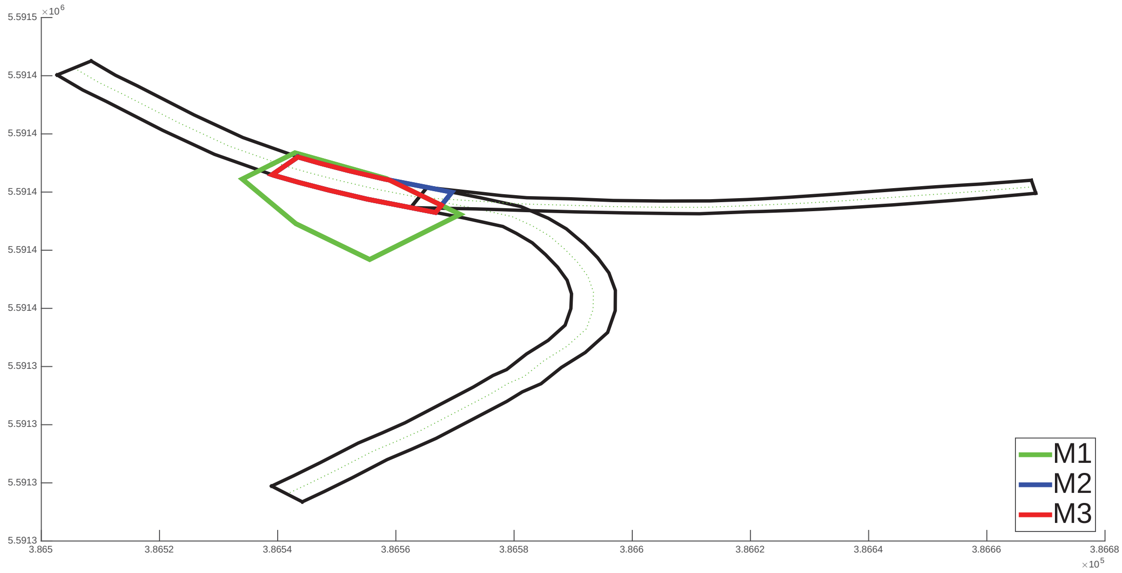
Step 3:



- M1: restricted absolute acceleration.
- M2: restricted acceleration and velocity in longitudinal direction.
- M3: staying within road boundaries.

Examples: Lane 2

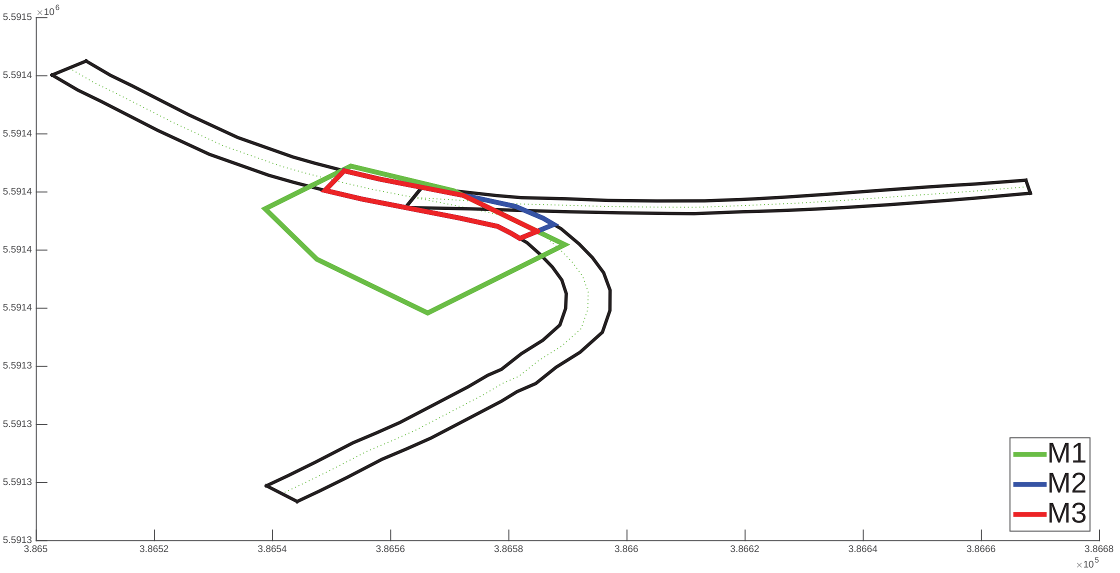
Step 4:



- M1: restricted absolute acceleration.
- M2: restricted acceleration and velocity in longitudinal direction.
- M3: staying within road boundaries.

Examples: Lane 2

Step 5:



- M1: restricted absolute acceleration.
- M2: restricted acceleration and velocity in longitudinal direction.
- M3: staying within road boundaries.

Selection of an Appropriate Set Representation

Reachable set of the ego vehicle

We restrict the reachable set to states that can be reached by a collision-free trajectory

$$\forall t : \ \mathcal{A}(x(t)) \ \cap \ \mathcal{O}(t) = \emptyset.$$

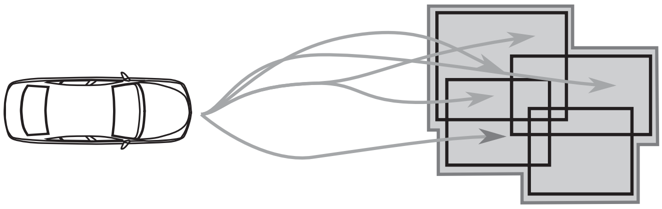
where $\mathcal{A}(x(t))$ is the footprint of the vehicle at state $x(t)$ and $\mathcal{O}(t)$ is the obstacle region.

The representation of the set is crucial for efficient calculations:

- Propagation of set after one time step must suite motion model (e.g. convex polyhedron for LTI)
- Intersection tests with forbidden states (e.g. obstacles)
- Complement set operation to cut forbidden regions from reach set

Overapproximation of all Reachable States

- We overapproximate the reachable set of the previous vehicle model (now: only consideration of acceleration limits; for computational reasons: $\|a\|_\infty \leq a_{max}$) by the union of boxes:



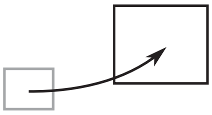
$$\mathbf{reach}(x_0, t) \subset \bigcup_j \text{box}_t^{(j)}$$

- Each box is four-dimensional and stores an interval of position and velocity in x, y-direction:

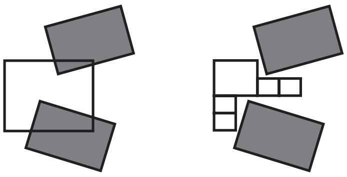
$$[s_{x,min}\ s_{x,max}] \times [s_{y,min}\ s_{y,max}] \times [v_{x,min}\ v_{x,max}] \times [v_{y,min}\ v_{y,max}]$$

Main Algorithm

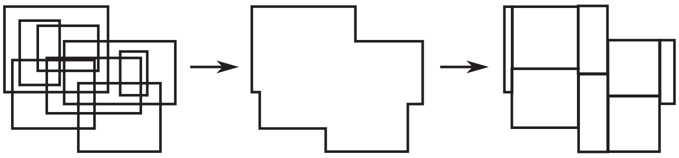
Propagate:



Split (dark regions are obstacles):

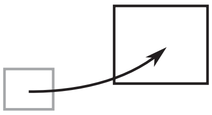


Repack:



```
for $i \leftarrow 1$ to steps do
    $\mathcal{R}_i \leftarrow \emptyset$
    for all box$_{(i-1)}$ in $\mathcal{R}_{i-1}$ do
        box$_{(i)} \leftarrow$ PROPAGATE(box$_{(i-1)}$)
        $\mathcal{R}_i \leftarrow \{\mathcal{R}_i$, SPLIT(box$_{(i)}$)$\}$
        if $|\mathcal{R}_i| >$ max. number boxes then
            $\mathcal{R}_i \leftarrow$ REPACK($\mathcal{R}_i$)
        end if
    end for
end for
return $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_{steps}$
```

Propagation of Boxes

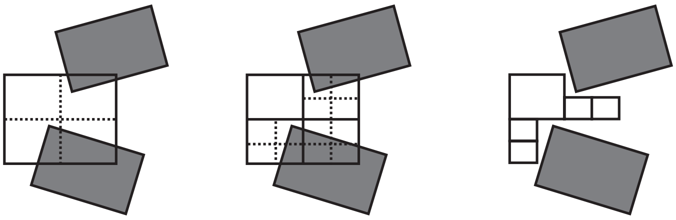


- Overapproximation by propagation of extremal values

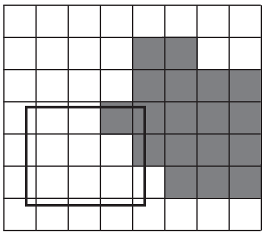
$$\begin{aligned}
s_{t+\Delta t}^{(L)} &= s_t^{(L)} + v_t^{(L)}\Delta t - \frac{1}{2}a\Delta t^2 \\
s_{t+\Delta t}^{(H)} &= s_t^{(H)} + v_t^{(H)}\Delta t + \frac{1}{2}a\Delta t^2 \\
v_{t+\Delta t}^{(L)} &= v_t^{(L)} - a\Delta t \\
v_{t+\Delta t}^{(H)} &= v_t^{(H)} + a\Delta t
\end{aligned}$$

a is maximum acceleration/braking, Δt is time step, (L) and (H) denote the lower and upper bound.

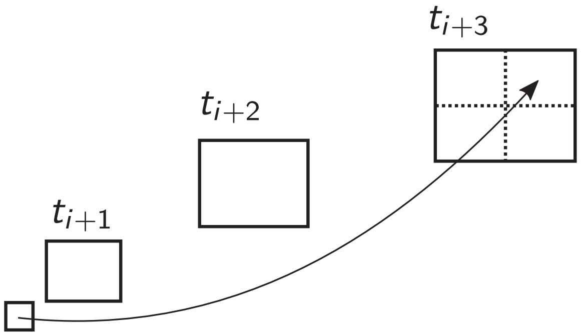
Collision Check of Boxes and Split Boxes



- Split rectangle along both axes if it intersects with obstacle
- Similar to quadtrees
- Repeat splitting until split rectangles do not intersect any obstacle or the diagonal is less than inner circle of vehicle
- For occupancy grids collision check can be done in O(1) using a summed area table

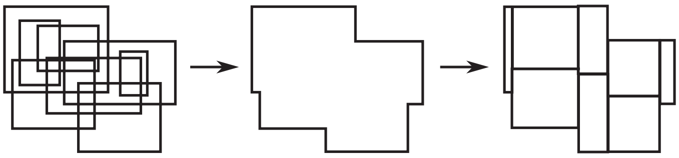


Restrict Velocities in Split Boxes



- Find reachable velocities within a position subregion
- Using Pontryagin's maximum principle to find candidate functions to minimize/maximize final velocity
- Bang-bang input
- Find start and goal state with extremal final velocities

Repack Reachable Set to Non-Overlapping Boxes



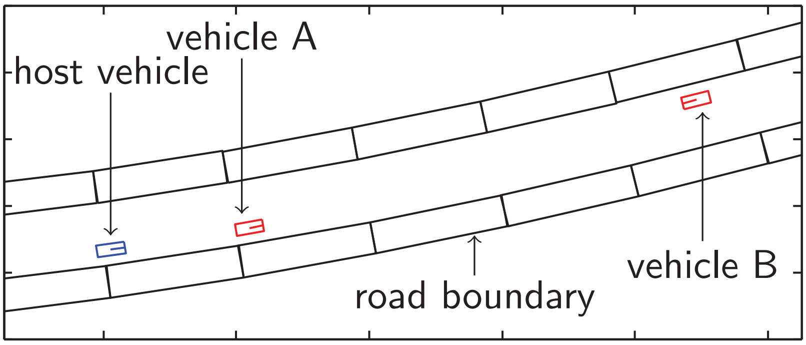
- Union of axis-aligned rectangles can be done in

$$O(m \log m + p \log(2m^2/p))$$

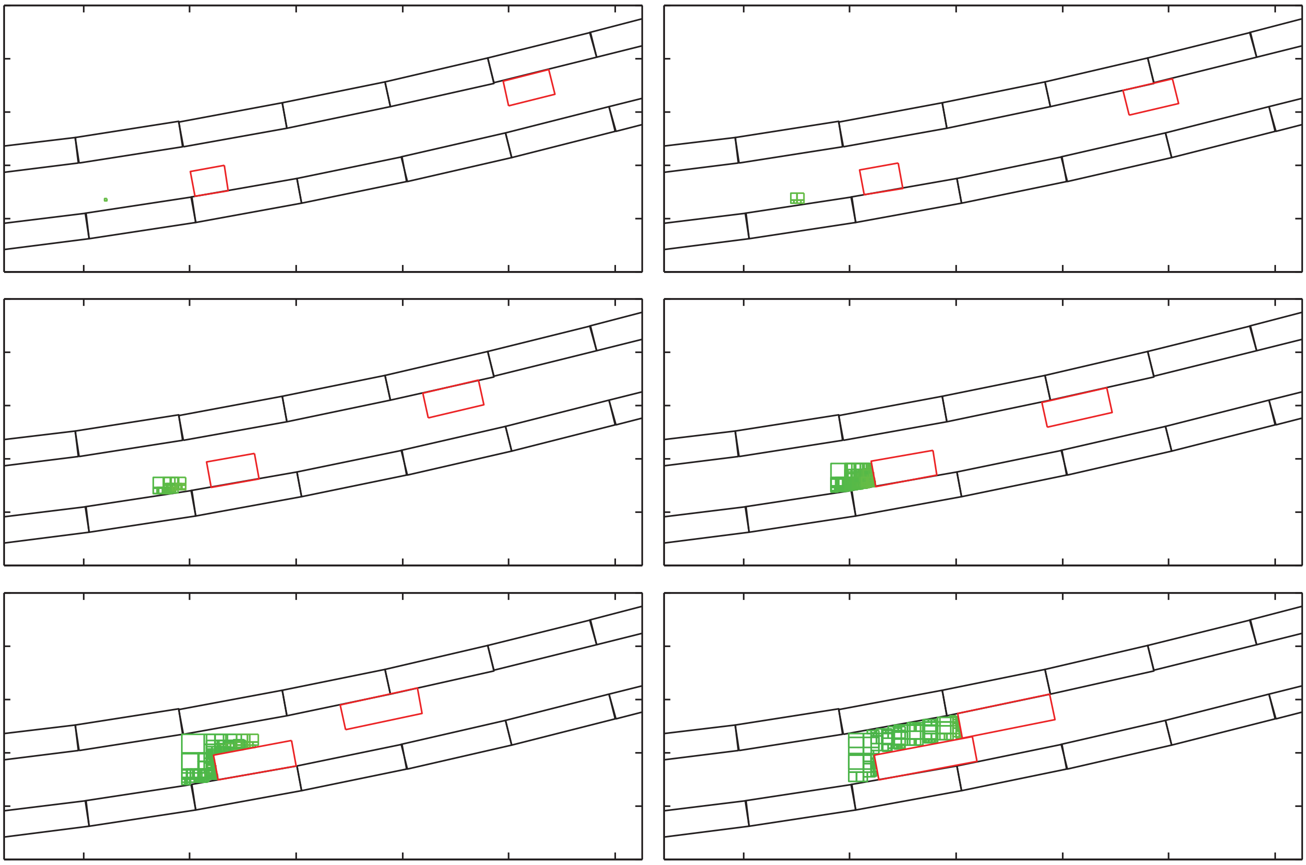
  p: number of edges in the contour
  m: number of rectangles
- Sweep line algorithm using a segment tree to do fast interval union
- Modified algorithm to split into non-overlapping boxes in the same sweep
- Velocity set for each new box is obtained from minimum and maximum velocity of all intersecting original boxes

Example Scenario

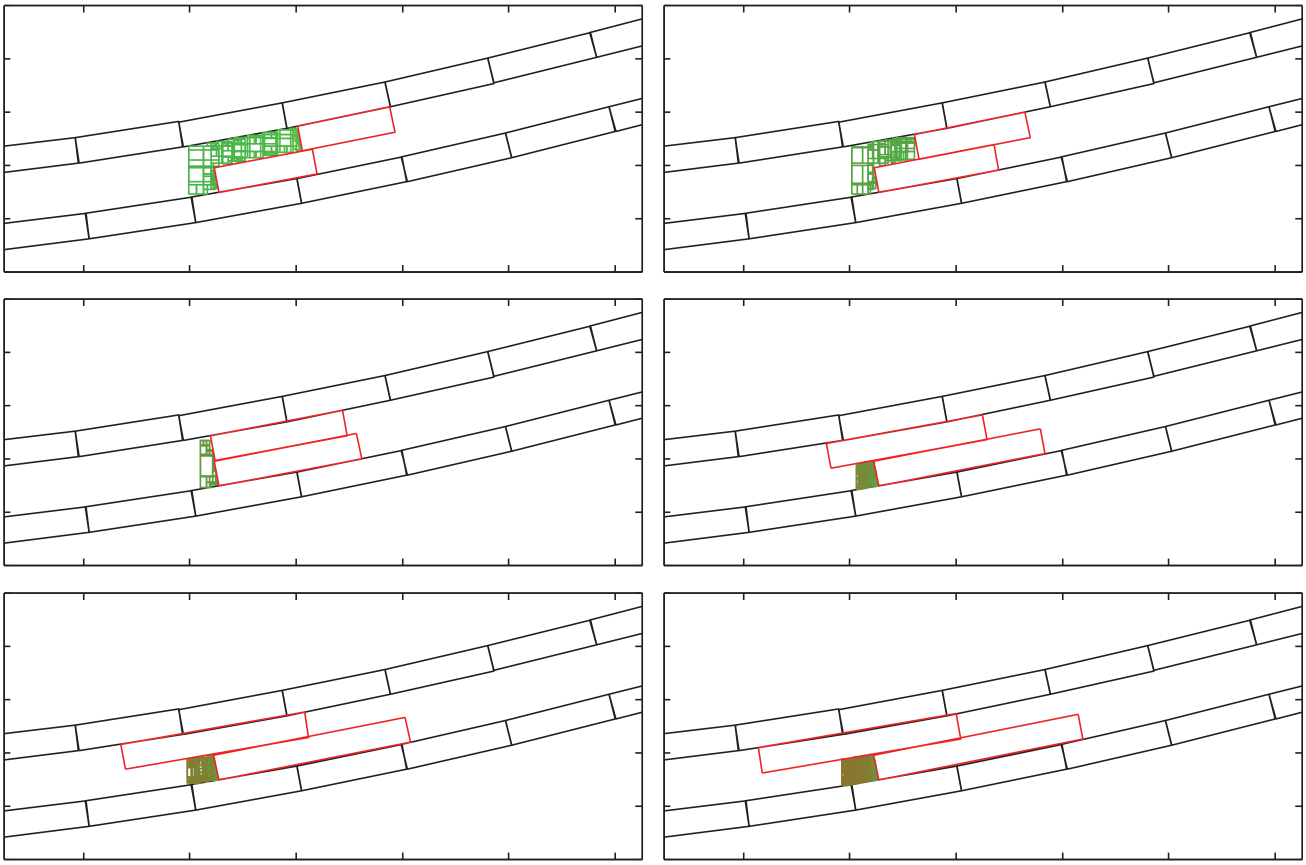
- Scene with ego vehicle and two other vehicles
- Other vehicles are assumed to stay on lane, but may accelerate or brake arbitrarily within some bound



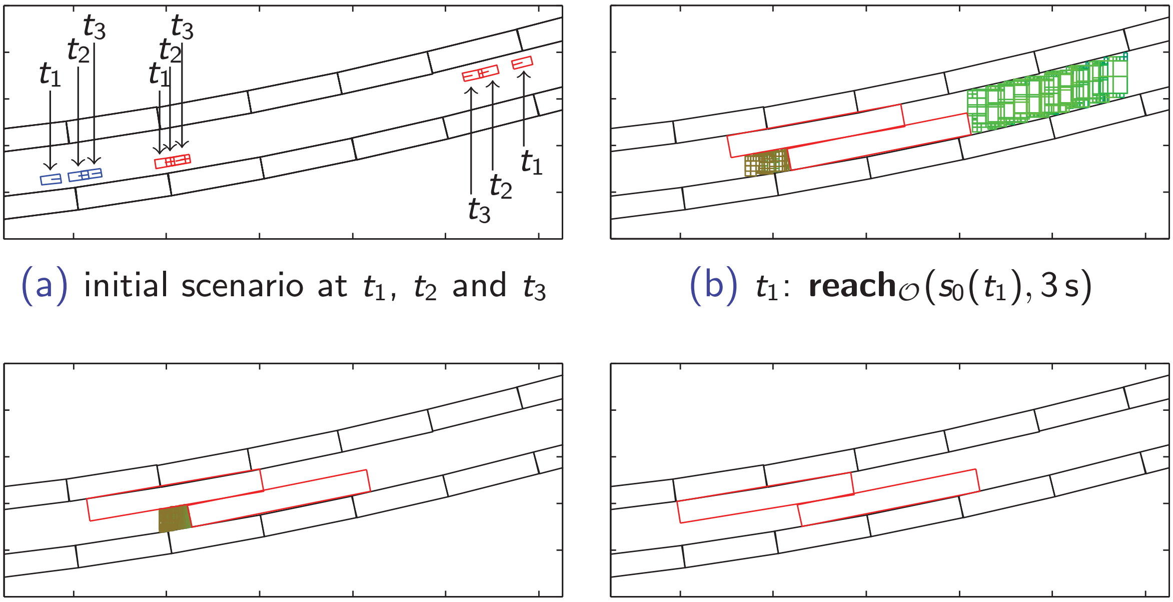
Example Scenario



Example Scenario



Example Scenario



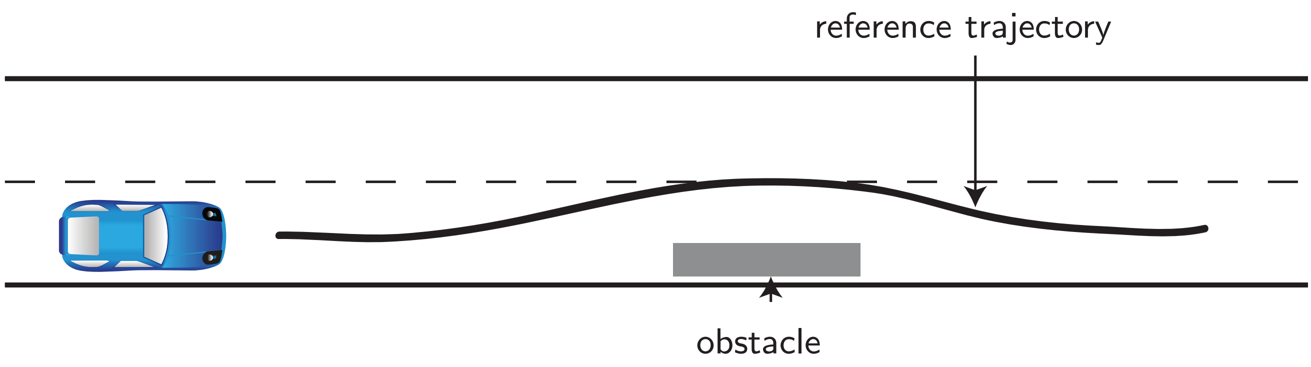
(a) initial scenario at t_1, t_2 and t_3

(b) t_1: $\mathbf{reach}_{\mathcal{O}}(s_0(t_1), 3\,\mathrm{s})$

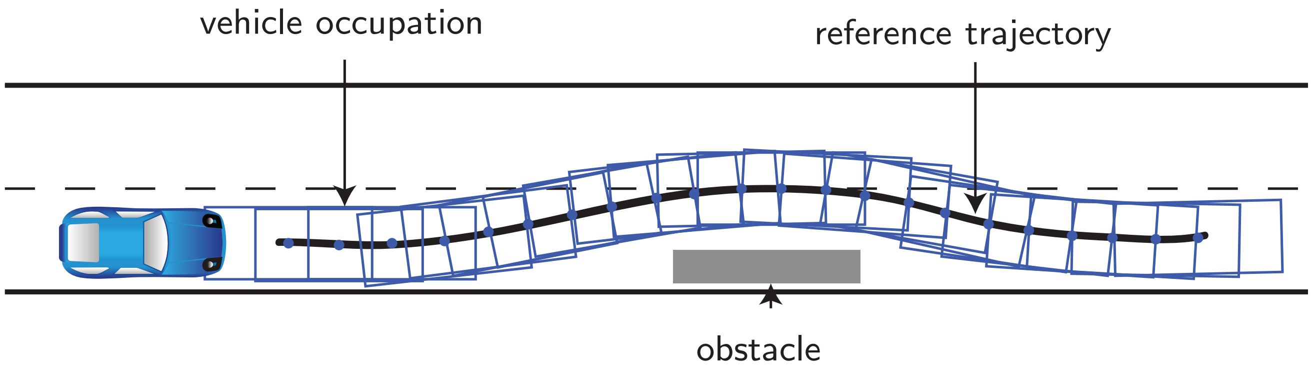
(c) t_2: $\mathbf{reach}_{\mathcal{O}}(s_0(t_2), 3\,\mathrm{s})$

(d) t_3: $\mathbf{reach}_{\mathcal{O}}(s_0(t_3), 3\,\mathrm{s})$

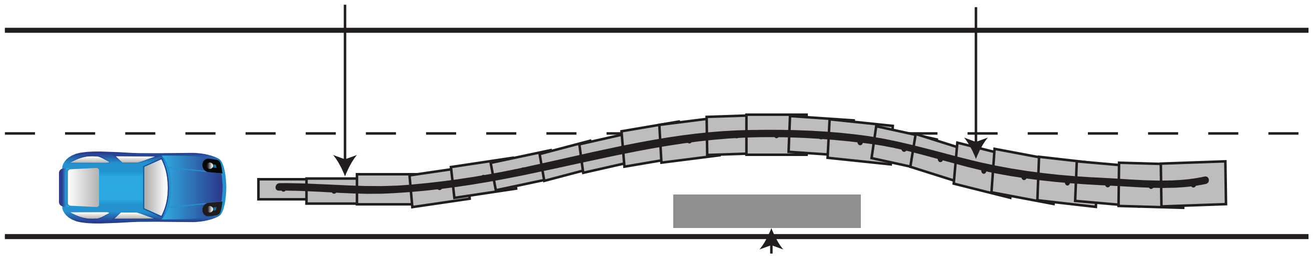
Trajectory Verification: Situation



Trajectory Verification: Standard Approach



Trajectory Verification: Considering Uncertainties



reachable set of the center

reference trajectory

obstacle

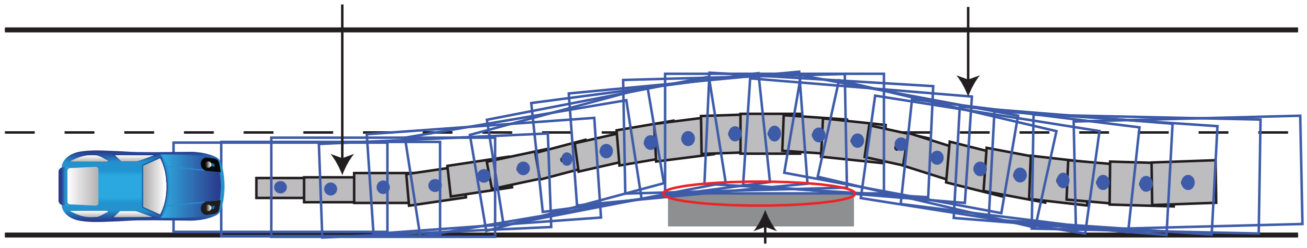
Robust Safety Problem

Is the planned maneuver of the autonomous vehicle still safe under

- uncertain initial states,
- uncertain measurements,
- and disturbances?

Objective: Guarantee safety when bounds on uncertainties are known.

Trajectory Verification: Formal Verification Reveals Problems



reachable set of the center

vehicle occupation

possible collision

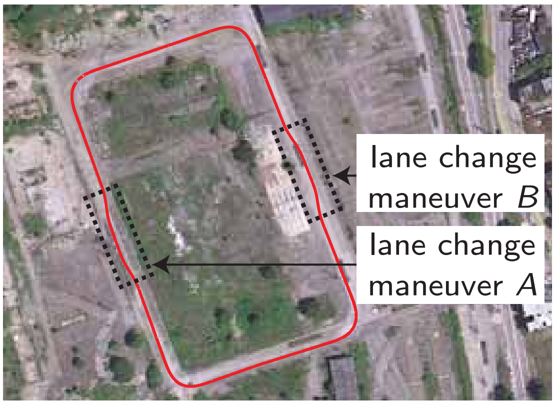
Robust Safety Problem

Is the planned maneuver of the autonomous vehicle still safe under

- uncertain initial states,
- uncertain measurements,
- and disturbances?

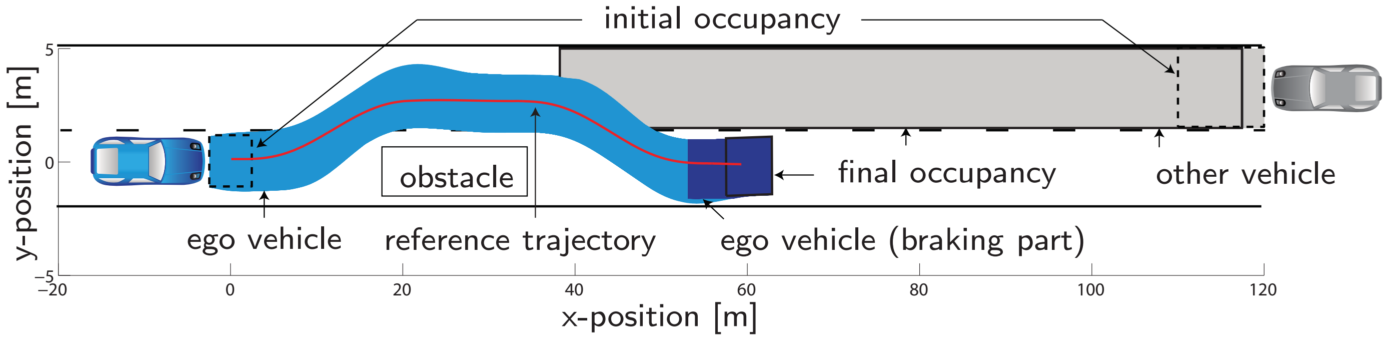
Objective: Guarantee safety when bounds on uncertainties are known.

Online Verification Of Automated Driving

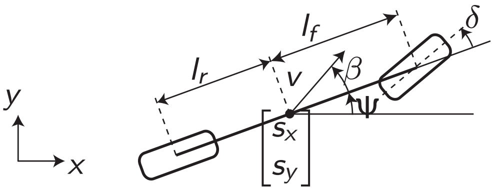


Test site

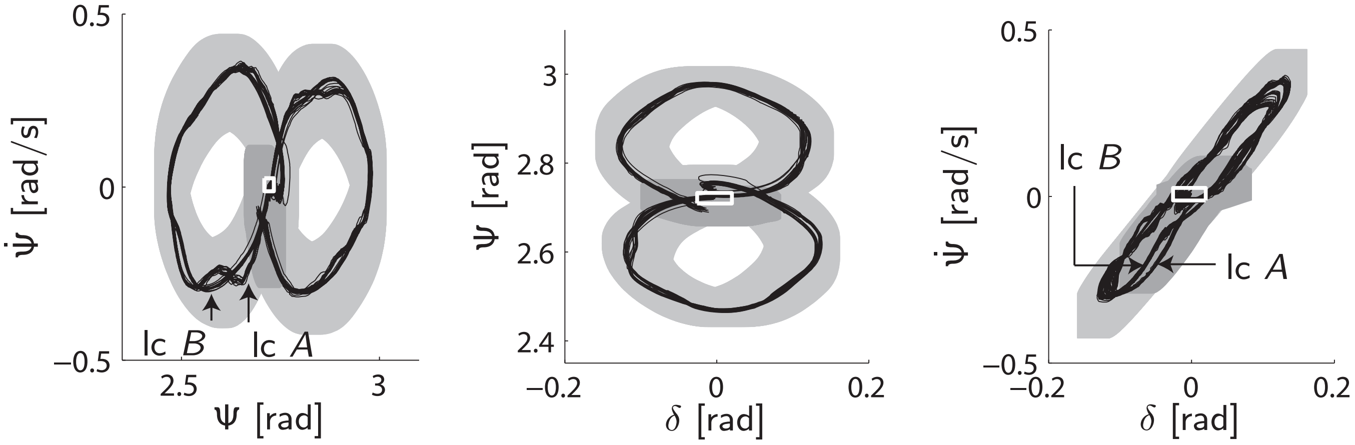
Test vehicle



Test Drive Results



s_x, s_y [m]	x- and y-position
Ψ [rad]	orientation
β [rad]	slip angle at center of mass
δ [rad]	front wheel angle
v [m/s]	velocity



computation time: ≈ 1.8 times faster than maneuver time (Intel i7, 1.6GHz)

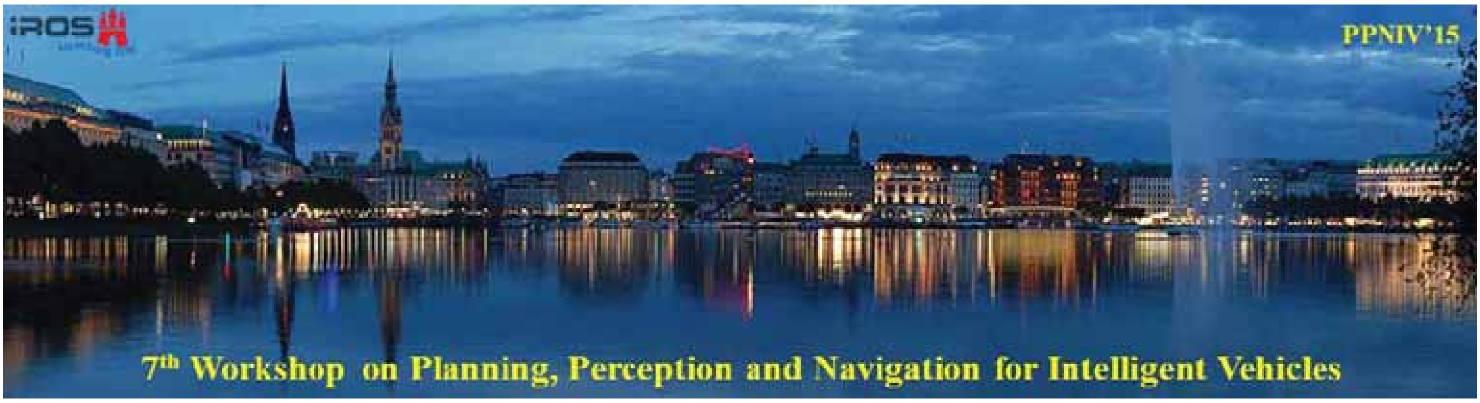
Conclusions and Future Work

Conclusions

- All possible behaviors can be computed.
- The approach considers uncertain measurements of other traffic participants and the ego vehicle.
- Assuming that the mathematical model captures all real behaviors, we can prove that no evasive trajectory exists.
- The computation is constantly improving and already less than a second.

Future Work

- Comparison of results with real world measurements.
- Implementation in C++ for deployment in a vehicle.


PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

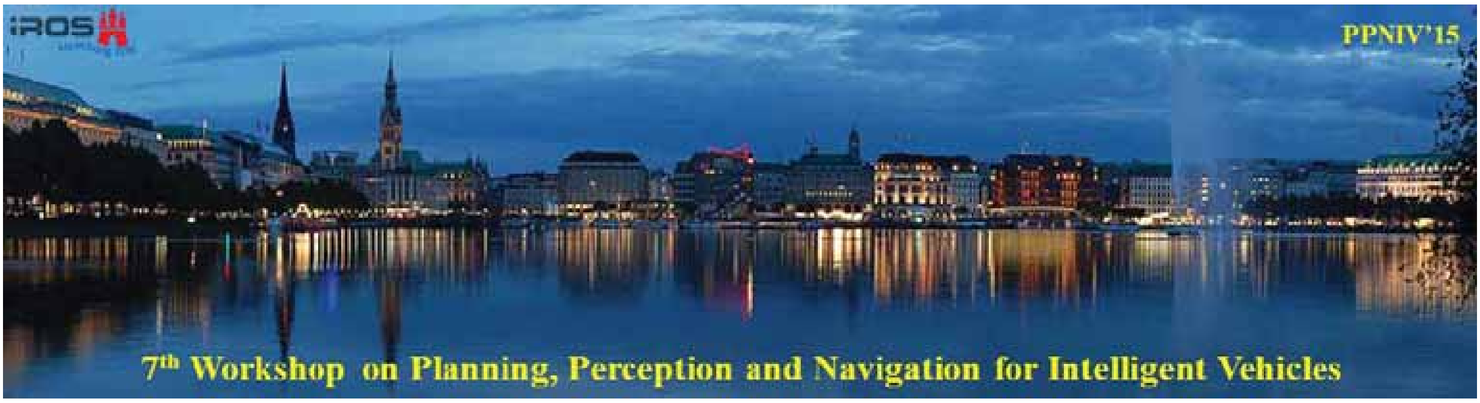
2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session IV

Planning & Navigation

- **Title: Safe prediction-based local path planning using obstacle probability sections**
  Authors: Tanja Hebecker and Frank Ortmeier

- **Title: Improving Monte Carlo Localization using Reflective Markers: An Experimental Analysis**
  Authors: Francesco Leofante, Gwénolé Le Moal, Gaëtan Garcia , Patrice Rabaté


PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Safe prediction-based local path planning using obstacle probability sections

Tanja Hebecker[1] and Frank Ortmeier[2]

Abstract— **Autonomous mobile robots gain more and more importance. In the nearest future they will be a part of everyday life. Therefore, it is critical to make them as reliable and safe as possible. We present a local path planner that shall ensure safety in an environment cluttered with unexpectedly moving obstacles. In this paper, the motion of obstacles is predicted by generating probability sections, and collision risks of path configurations are checked by determining whether these configurations lead inevitably to a collision or not. The presented approach worked efficiently in scenarios with static and dynamic obstacles.**

I. INTRODUCTION

Due to the fast development in the field of autonomous mobile robotics the importance of safe motion planning increases. Even in situations with many irrationally moving obstacles local path planners have to guarantee a safe avoidance of obstacles. Therefore, it is necessary to consider the robot's kinematics, the motion of dynamic obstacles and also risks that can arise in the future even if a robot is located at a point of a path (waypoint) that is itself collision-free.

Within the last years many new path planning approaches for autonomous robots were presented but to our knowledge there is no approach that considers both future collision probabilities of a waypoint in the presence of moving obstacles, and also atypical obstacle motion. For guaranteeing freedom from collision, even promising path planning approaches still have some need for improvement.

Schmidt and Berns [1] presented an approach in which an extended A* path planning algorithm is applied to a 2D gridmap with growing regions for obstacles as a minimum safe distance. This approach does only consider static obstacles.

A method named Minimal Risk Motion Planning is presented in [2]. A route cost function minimizes possible failures to reach a goal state. By applying the wavefront algorithm [9], a cost-to-go function within the sensor field of view is calculated. Future collision risks of path states are not considered.

There are also approaches considering the possibilities of a future collision with static and dynamic obstacles. In [5] Petti and Fraichard proposed the Rapidly-Exploring Random Tree (RRT [13])-based motion planning scheme Partial Motion Planning (PMP). The Inevitable Collision State (ICS) approach is applied to check if it is possible in a state to avoid a collision with an obstacle (otherwise the state is an ICS), and to guarantee safety by checking a state whether at least one control command exists which is not leading to a collision state. This approach promises well but atypical obstacle motions are not considered.

Bouraine et al. [6] presented a passive motion safety approach that guarantees that, if a collision takes place, the robot will be at rest. The future obstacle motion is modeled by reachable sets [9] occupying with growing time the whole workspace. Therefore, a Braking ICS-Checking algorithm is integrated in a navigation scheme called PASSAVOID to check whether for all future braking trajectories of a state a collision occurs before the robot is at rest. It is not the task of PASSAVOID to drive the robot to a given goal and in some cases collisions occur that could have been avoided.

Likewise, methods to predict the motion of moving obstacles already exist. Vasquez and Fraichard [3] presented a technique to estimate the motion of a moving object in a structured environment in the long term by determining typical motion patterns of obstacles to predict their future motion. The presented approach is not able to predict atypical trajectories and for a short observation time frame the predicted motion tremendously differs from an obstacle's real motion.

Kushleyev and Likhachev proposed in [4] a representation of dynamic obstacles that models their predicted trajectories and considers the prediction uncertainty with error ellipses. It is assumed that the obstacle maintains constant controls at all future times. Atypical motion of obstacles is not regarded and if dynamic obstacles remain stationary it is possible that the robot gets stuck.

In this paper, we present a local path planning approach for mobile robots in unknown dynamic environments that considers (i) the robot's dynamic properties, (ii) motion probability of obstacles and (iii) future risks. The consideration of the robot's kinematics is important for generating only traceable paths and for a better evaluation of critical situations. This is because planning safe paths that a real robot cannot track due to a high velocity or constrained rotating movement can lead to collisions. We involve motion probabilities of obstacles by motion probability sections to consider realistic and, to our knowledge in contrast to other related works, also atypical obstacle motions with variable velocities in order to plan collision-free paths. Future risks of a chosen waypoint of a path will be estimated to not lead a robot into a situation where a collision is inevitable. Therefore, we apply ICS but we simplify the ICS checks similar to the PMP method for requiring less computation time. For our ICS checks we allow very low collision risks

[1] Chair of Software Engineering, Faculty of Computer Science, Otto-von-Guericke-University, Germany, Tanja.Hebecker@ovgu.de

[2] Chair of Software Engineering, Faculty of Computer Science, Otto-von-Guericke-University, Germany, Frank.Ortmeier@ovgu.de

of the future states to not make path planning impossible because of too strict constraints.

Currently, this path planner is designed for holonomic and non-holonomic ground robots and, therefore, first, it is only required to involve the two-dimensional case. An environmental model representing a mobile robot within a room with static and moving obstacles is implemented in OpenRAVE.

We assume that the parameters and kinematics of detected obstacles are known to our algorithm. Currently, we start from the premise that obstacles have the same kinematics as our robot. Another assumptions are that the obstacles' unseen part is not bigger than the part that is detected by the sensor and that their motions are independent from other obstacles aside from collision avoidance.

II. METHOD DESCRIPTION

In this section, we present the concept of our path planning approach which is summarized in pseudocode 1. First, cells in a 2D grid map are generated following the strategy in Section II-A. Then, motion probability sections are determined in Section II-B to predict the future position of detected moving obstacles. The modified wavefront algorithm [12] assigns cost values to the grid map cells after a collision-free local goal is determined within the sensor field of view (see Section II-A). Then, a path is determined by determining a cell sequence. The neighboring cells of the lastly to the cell sequence added cell are checked if they have a lower cost value. This process continues as long as the path has not reached the local goal. If the condition is true the cell centers of these cells are checked if they do not lead to an inevitable collision (see Section II-C.3), keep a safety distance to obstacles and the sensor range (see Section II-C.1), and are reachable (see Section II-C.2). In this context, we define reachability as the existence of a control command such that the robot can reach this waypoint safely and without unneeded detours. If these criteria are fulfilled the considered cell is added to the path cell sequence. The path is generated by the cell centers from the cell sequence.

Algorithm 1: Generating a collision-free path

```
Input: sensor range $r_{sensor}$, initial state of robot $s_{init}$
Output: Path $P$
while global path not safe do
    $Cells \leftarrow CellGeneration(r_{sensor})$;
    $RiskArea \leftarrow MotionProbabilitySections(r_{sensor})$;
    $LocalGoal \leftarrow AssignGoal(RiskArea, Cells)$;
    $Costs \leftarrow ModifiedWavefront(Cells, LocalGoal)$;
    $P \leftarrow FindingPath(Costs, Cells, RiskArea, s_{init})$;
end
return $P$;
```

A. Generating Potential Waypoints

The first step is to apply an algorithm that provides potential waypoints within the field of view. Many path planning approaches can be applied, e.g., Virtual Force Field by [10] or the enhanced Vector Field Histogram method VFH* by [11]. We propose to use the modified wavefront algorithm presented in [12]. It is an extension of the wavefront algorithm that we previously applied for a UAV online planner in 3D [8]. The modified wavefront algorithm is a real-time capable approach that is applicable in 2D and 3D space and capable of finding shortest paths. It guarantees the achieving of the goal if it is possible to reach, i.e., it does not lead to a local minimum. This method requires a grid map representation of the workspace. Because the applied sensor has a circular field of view we construct the grid map with polar coordinates.

An example of a grid map representation in a polar coordinate system is depicted in Fig. 1. The radii get larger by a constant value and the straight lines along the polar angles have a constant distance to each other. This way cells are generated that are smaller at short distance to the robot and larger with growing distance. A local goal is a cell center which is determined after checking all cell centers for the following conditions: (i) the distance to the global goal is as short as possible, (ii) the distance to static obstacles and to the sensor range limit is not too short and (iii) there is no risk that an obstacle occupies this cell center within the required time frame (see Section II-C.1).

The modified wavefront algorithm labels the cells of the work space with cost values based on their distance to the local goal. First, all these cells in the work space get the cost value zero, and then, the cell containing the local goal receives the cost value one. Orthogonal neighbor cells of the goal get the cost value of the goal plus 3 and diagonal neighbor cells get the cost value of the goal plus 4. This process continues for their neighbors with the cost value of the currently considered cell (see Fig. 1).

Cells that are occupied by obstacles or due to obstacles beyond the sensor's view get an invalid cost value. When assigning cost values we consider only the sensor detected position of dynamic obstacles and not their motion. We include their possible future positions in the determination of motion probability sections that are considered when choosing a waypoint later. The algorithm stops when no cell has the cost value zero anymore.

Beginning from a starting cell, centers from neighboring cells with the lowest (and positive) value are potential waypoints. Before selecting one of these potential waypoints we have to ensure that it does not lead into a collision.

B. Probability Sections

We consider only the last detected position of moving obstacles for the application of the modified wavefront algorithm, hence, we have to include their motion when choosing waypoints for a path. We represent the obstacles' motion with motion probability sections. The possible future position of obstacles is described by an approximated reachable set around the detected part of the moving obstacles. This approximated reachable set is partitioned into areas containing the probabilities of keeping or changing a direction.

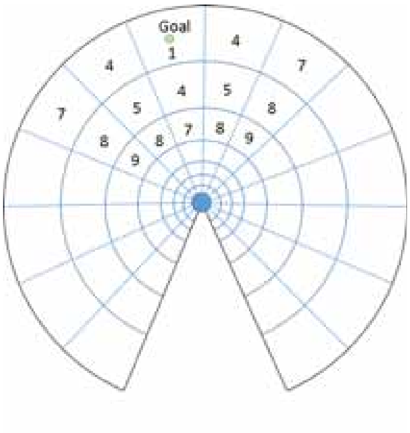


Fig. 1: Generating cells and applying the modified wavefront algorithm within the robot's sensor range

These probability sections are approximations to enable fast calculations.

If an obstacle with a certain velocity direction moves freely in a room the probability for this obstacle to continue with the same velocity and in the same direction is higher than a turning or a braking maneuver (except, e.g., it has an obstacle in front of it). The probability that an obstacle brakes and moves with a sharp curve to the left or to the right is in this case lower, and the probability that an obstacle moves suddenly in the direction where it came from is the lowest one because it is completely irrational (but not impossible) except it has to fulfill a task that demands such a motion behavior.

We refer a probability value to a section giving the probability that in the next time steps the obstacle will be within this area. These probabilities are computed with the standard normal distribution for the motion angle. The peak is in the direction of the current motion.

Examples of probability sections are depicted in Fig. 2. The obstacle O1 of the robot is moving towards the robot and the obstacle O2 is about to move out of the field of view (the velocity arrows show the velocity direction). Both obstacles are unknown and, therefore, the robot has only the information about the obstacles that it gets from its sensor. Around the seen part of the obstacles an area is determined that this obstacle part is able to reach. Considering their current speed a distance is calculated that these obstacles can cover within a certain time frame which is explained in Section II-C.1. This distance determines the limit of the probabilistic future position area. This area for obstacle O1 is divided by maximum possible turning maneuvers to the left and to the right with a constant velocity and by the current motion direction. The reason for the section limitation by these turning maneuvers is that a change of the obstacle motion direction without braking is in unconstrained space almost as possible as continuing in the initial direction. If an obstacle has to avoid another object the probability for continuing in the same direction decreases. But if this object is not in the front but at a side of the obstacle it is still highly probable that the obstacle continues without reducing the velocity. Therefore, the front section is separated by the current direction line into two sections.

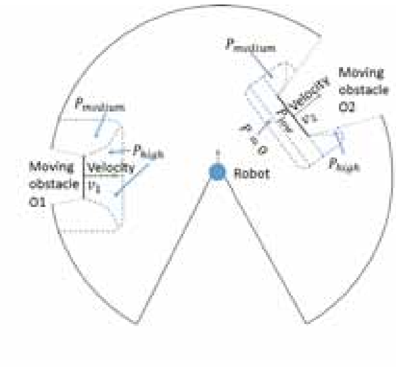


Fig. 2: Probability sections for a dynamic obstacle within the laser range

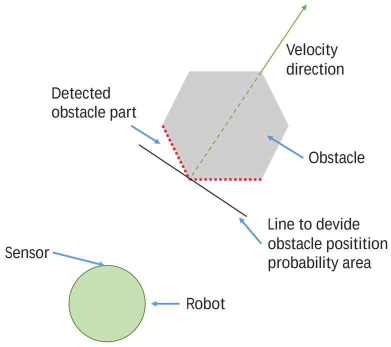


Fig. 3: Determination of a straight line that divides the lowest probability section from the whole future position area

In Fig. 2 the robot can see the back view of obstacle O2 which is about to leave the field of view and, therefore, the part of the area around the obstacle that is behind the obstacle has a low occupancy probability. Hence, we divide the area around the detected obstacle part by a straight line going through the obstacle's point with the closest distance to the robot and being at an angle of 90° to the velocity direction. For a better understanding, the determination of the straight line is depicted in Fig. 3. To avoid unnecessary huge areas a part that definitely is not reachable within the given time frame gets the probability 0. This area part is calculated by assuming the obstacle to suddenly move directly in the opposite direction. Within the given time frame it is not possible that the obstacle reaches the area boundary due to its initial velocity. This negligible area part starts at the maximum reachable distance in the reversed direction of the initialized obstacle motion. As in the case of obstacle O1 the section with the highest probability is determined by calculating maximum possible turning maneuvers with a constant velocity.

The probability values are influenced by the presence of other obstacles. If obstacle O2 or its approximated reachable set intersects the probability section of obstacle O1 the probability value for the concerned section of O1 is calculated with the formula for conditional probabilities.

The motion probability sections are involved for selecting waypoints (see Section II-C.1). Parts of the environment that cannot be perceived by the sensors because they are hidden by obstacles are considered as potential obstacles.

C. Selection of Waypoints

Beside the criterion that a waypoint must have a lower cost value than its predecessor other conditions have to be fulfilled to be chosen as a point of a path. The waypoint has to be collision-free, reachable and with low future collision risks.

1) Safe Distance to Obstacle Regions: A waypoint must not be occupied by an obstacle when the robot reaches it. In our specification, this means that a waypoint is not allowed to be located within an obstacle position probability section for a time frame the robot would need to arrive there. The only exception is the unreachable section with probability equal to zero because there is no collision risk.

The extents of the probability sections depend (beside the obstacle's current velocity and kinematics) on the distance of a waypoint which is checked for freedom from collision to the robot's initial position. That is because we choose the time frame t_{frame} for the calculation of the motion probability area by this distance and a certain velocity. We determine this velocity as the lower bound of the robot's velocity for reachability (see Section II-C.2) v_{bound} because t_{frame} is an assumed time frame the robot needs to get to the considered waypoint in the exclusion of unexpected detours.

Additionally, a safety distance to the obstacle motion area has to be included because if an obstacle really reaches the limit of this area it must not be too close to the robot. This safety distance is added to the outer limits of the obstacle sections and it depends on whether the obstacle is in motion direction of the robot or not for not rendering planning in narrow areas unnecessarily impossible. For obstacles in a planned robot motion direction we determine the safety distance as

$$d_{safe} = \begin{cases} term1 + \frac{l_{robot}}{2}, & \text{if } t_{break} > t_{frame} \\ term2 + \frac{l_{robot}}{2}, & \text{else} \end{cases}, \qquad (1)$$

where $term1 = -\frac{a_{max}}{2} \cdot t_{frame}^2 + v_0 \cdot t_{frame}$ and $term2 = -\frac{a_{max}}{2} \cdot t_{break}^2 + v_0 \cdot t_{break}$. The parameter t_{break} is the time the robot needs to come to a standstill, v_0 is the initial velocity and l_{robot} is the length of the robot. Equation 1 is based on uniform acceleration. The reason for adding $\frac{l_{robot}}{2}$ is that we assume the center of the robot to be located on the waypoint, then half of its extent exceeds this waypoint. We consider the maximum extent of the robot due to safety reasons and, therefore, we include half of the robot's length in the safety distance.

For obstacles that are not located in the planned motion direction of the robot the probability sections are additionally only extended with $d_{safe} = \frac{l_{robot}}{2}$. The occupancy probability of a waypoint is determined for the time when the robot is able to reach the waypoint.

The safety distance d_{safe} holds also for static obstacles. Due to possible obstacles beyond the field of view also a safety distance d_{safe} to the outer sensor range has to be kept. A waypoint must not be located within an obstacle section with a probability higher than zero and has to keep a safety distance d_{safe} to static obstacles and the full sensor range. If these conditions are complied the waypoint is checked for fulfilling the following criteria, if not another waypoint is analyzed.

Even if a waypoint is collision-free, two other criteria have to be fulfilled before it becomes a part of the path: (i) the waypoint has to be reachable by the robot without unnecessary detours due to motion safety and without braking to a too slow velocity, and (ii) the robot's state at the waypoint must not be an ICS.

2) Reachability: To determine if a waypoint is reachable without the need that the robot comes below a certain speed, we determine an approximated reachable set that was already presented in our previous work [8]. The limits of this reachable set are calculated by computing trajectories representing maximum possible turning maneuvers to the left and to the right and the maximum distance d_{reach} that the robot with its initial velocity can cover.

We consider the motion direction that the robot would have if it moves from the waypoint that was lastly added to the path to the potential new waypoint. Turning maneuvers are determined by setting the initial speed into this motion direction starting from the potential waypoint. We do not want to plan a path that is only traceable when the robot stops in front of obstacles or to render path planning impossible due to too high velocities. Therefore, the trajectories concern the case that the robot brakes with $a = -k_{reach} \cdot a_{max}$ not below $v_{bound} = k_{reach} \cdot v_{max}$. The constant value k_{reach} is defined in the interval [0,1] and in our case $k_{reach} = 0.5$. We chose this value because we assume it to represent a lower bound to ensure a smooth motion. Only cell centers between these two trajectories and within the distance d_{reach} to the previous waypoint are reachable.

3) Inevitable Collision States: According to [7] an ICS is a state in which it is not possible to avoid a collision with an obstacle, no matter what the control input of a robot is. ICS checks are important because a waypoint must not be added to a path if it is impossible to avoid a collision after reaching it. Precise ICS checks require a lot of computational time, hence, we apply a simple way of determining ICS. Similar to [5], we check if a predicted state at a waypoint is an ICS by computing three trajectories with the help of the robot's dynamic model for a certain time frame. One trajectory is for the straight forward motion and two trajectories are the ones already calculated for the reachability check with the difference that the trajectories are now determined for the case of full braking with $a = -a_{max}$. These trajectories are depicted in Fig. 4. Sampled points of these trajectories are checked for collision risks as the waypoint itself with one difference. If the collision risk for these trajectories is higher than a probability border, which we for the moment set to 5% (because the area behind an obstacle that is not

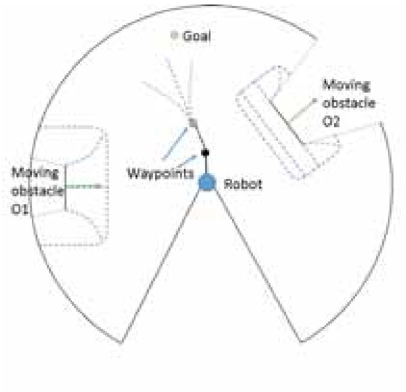


Fig. 4: ICS check of a potential waypoint

influenced by intersections of other obstacles never exceeds this probability value), than the considered waypoint is not safe enough and another waypoint has to be found.

III. EVALUATION

In this section, the evaluation results of our path planning approach are presented and compared to the PMP algorithm. Both algorithms were implemented in OpenRAVE[1].

A. Kinematics and Parameters

For testing the two approaches in simulations, we assume as an example that the robot as well as the obstacles move according to the following simple model:

$$\dot{\mathbf{x}} = \begin{pmatrix} v \cdot cos(\gamma) \\ v \cdot sin(\gamma) \\ a \\ \dot{\gamma}_{max} \end{pmatrix} \tag{2}$$

with

$$\mathbf{x} = (x_1, x_2, x_3, x_4)^T ,$$

x_1 and x_2 as positions in x- and y-direction, x_3 as speed and x_4 as the yaw angle. The state space model contains the velocity v, the acceleration a and the yaw angle γ as well as the maximum angular velocity $\dot{\gamma}_{max}$.

In our simulations, we represent obstacles and the robot with our local path planner as KUKA youBots[2]. For the evaluated scenarios, the motion of these robots is constrained according to the mathematical model 2. The KUKA youBot omni-directional mobile platform has the following parameters: a length of 0.580m, a width of 0.380m and a maximum speed of v_{max} = 0.8m/s. Additionally, we set for our simulations the maximum acceleration to a constant value of 0.2m/s^2.

The applied sensor is a 2D laser scanner with the following chosen specifications: a scan time of 0.1s, a sensor range of 5m, an angle interval [-135°,135°] and a resolution of 0.35°.

B. Simulation Results

In this section we present evaluation results for our probability section algorithm. We compare the proposed algorithm with the PMP method on scenarios where obstacles move atypically. Path smoothing was for the moment neglected in the simulations. Our environment consists of an 8×8 meters plane limited with walls from every side. The obstacles are other youbots. The environment is depicted in Fig. 5.

In the scenario of Fig. 5 three obstacles are moving with a current speed $v_{obstacle} = 0.75 \cdot v_{max}$ and the robot has a starting velocity $v_{robot} = 0.5 \cdot v_{max}$. The green lines starting at each obstacle indicate the assumed future motion direction due to the observed previous movement. The red curves show the real future motion of two obstacles (the third one $obstacle2$ moves in the assumed direction). Fig. 5(a) depicts a path planning result of our algorithm. A grid map overlays the geometry of the maximum possible sensor field of view. Due to the inclusion of unexpected obstacle motion, the planned path, which is represented by a yellow line, avoids $obstacle1$ though with its current position and velocity direction there is no collision risk. The local goal is not within one of the furthest cells in consequence of the consideration of the safety distance to the field of view's limit.

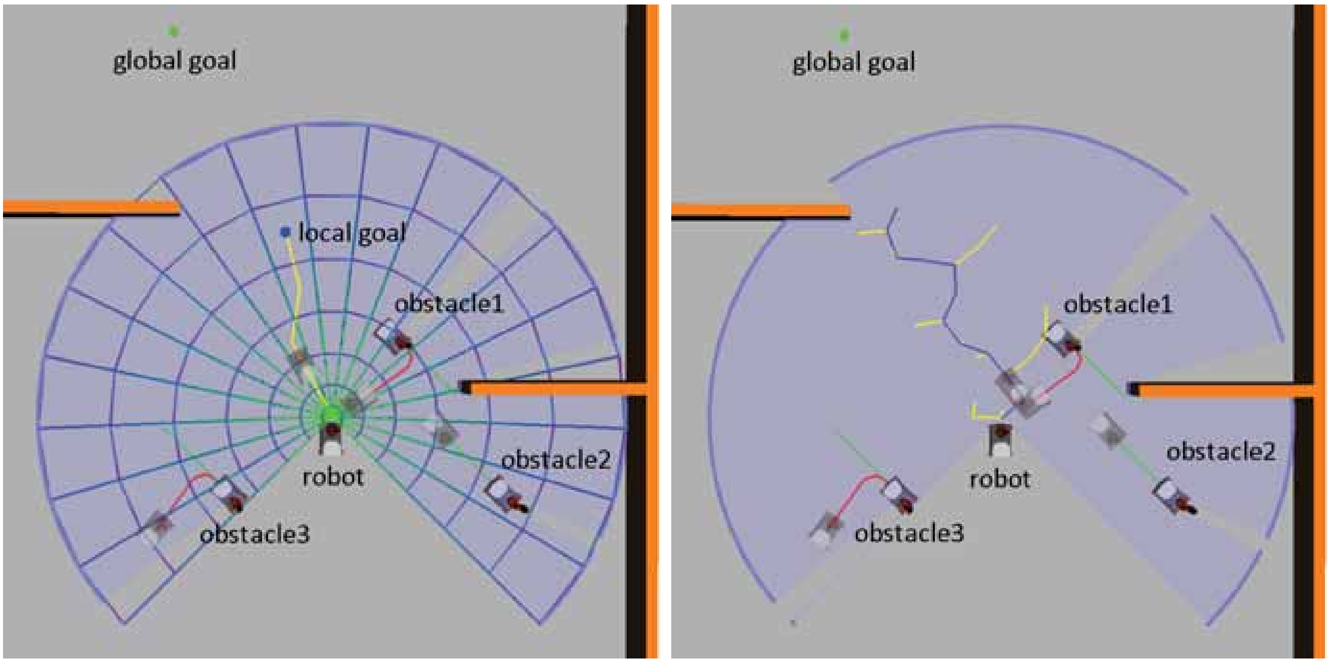
Fig. 5(b) shows the path planned by the PMP algorithm with a blue line. Negligible branches, which are generated in a RRT search towards the global goal, are represented with yellow arms at the path. The chosen path is collision free considering only the current obstacle states but with the actual future motion of $obstacle1$ the robot collides with the obstacle when reaching the critical waypoint. The reason of this is the different inclusion of the obstacle motion because PMP used the predicted trajectories approach presented in [3] that does not consider atypical obstacle motion. In contrast to our algorithm, the missing consideration of the robot's dynamics effects an avoidable need for velocity reduction. The unnecessary detours are caused by applying the RRT with 50 nodes.

In other test cases with more than one obstacle located on the upper part of the grid map, and with obstacles moving directly between the robot and the global goal our algorithm generates a local goal that is closer to the robot than to the global goal. The reason behind this is if the local goal would be further afar from the robot and closer to the global goal the time frame for the probability sections would increase and thus also the extents of these sections. Hence, obstacles in the upper field of view part which are moving in path direction can occupy a local goal with a longer distance to the robot before our robot is able to reach it. Therefore, only short collision-free paths are calculated until the robot escapes the critical situation.

In the tested scenarios, our algorithm always found a safe path to a local goal as long as it was possible to reach the local goal. The adaptation of the local goal to environmental conditions contributes to the planning of a complete path. The only test cases when the path planner fails to reach the

[1] OpenRAVE: see http://www.openrave.org, accessed on August 4, 2015

[2] KUKA youBot: see http://www.youbot-store.com, accessed on August 4, 2015



(a)

(b)

Fig. 5: We compared our approach with the PMP algorithm in the same scenarios. (a) shows the in this paper presented algorithm and (b) the PMP method.

local goal were when the robot was already at initialization time in an unsafe situation with obstacles moving all around and too close to the robot and blocking all possible ways to the local goal.

IV. CONCLUSIONS

In this paper, we presented a local path planning method for mobile robots that chooses a local goal within the field of view according to the environmental situation, considers future risks of a potential waypoint by applying ICS checks and includes possible irrational motion of obstacles by motion probability sections. We tested the algorithm in the OpenRAVE simulation environment with moving and static obstacles and compared it to the PMP method. In simulations, our algorithm always found a safe path to a local goal. In future works, we have to include the following issues in our algorithm:

- In the case that no safe waypoint can be added to a path a state will be determined where the collision risk is as minimal as possible and where the robot will get the fewest harm in case of a collision (e.g., the robot turns in a position that a collision has the fewest consequences).
- Another future work direction is the extension to 3D cases, e.g., to be capable to plan flight paths for unmanned aerial vehicles (UAVs).

REFERENCES

[1] D. Schmidt, K. Berns, ”Construction site navigation for the autonomous excavator Thor”, in 6th International Conference on Automation, Robotics and Applications (ICARA), IEEE, 2015, pp. 90–97.

[2] C. Goerzen, M. Whalley, ”Minimal risk motion planning: a new planner for autonomous UAVs”, in AHS International Specialists Meeting on Unmanned Rotorcraft, 2011.

[3] D. Vasquez, T. Fraichard, ”Motion Prediction for Moving Objects: a Statistical Approach”, in IEEE International Conference on Robotics and Automation, vol. 4, IEEE, 2004, pp. 3931 – 3936.

[4] A. Kushleyev, M. Likhachev, ”Time-bounded Lattice for Efficient Planning in Dynamic Environments”, in IEEE International Conference on Robotics and Automation, IEEE, 2009, pp. 1662 – 1668.

[5] S. Petti, T. Fraichard, ”Safe Motion Planning in Dynamic Environments”, in IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005, pp. 2210 – 2215.

[6] S. Bouraine, T. Fraichard, H. Salhi, ”Provably Safe Navigation for Mobile Robots with Limited Field-of-Views in Dynamic Environments”, in International Conference on Robotics and Automation, IEEE, 2012, pp. 174 – 179.

[7] T. Fraichard, H. Asama, ”Inevitable collision states. A step towards safer robots?”, in IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 1, IEEE, 2003, pp. 388 – 393.

[8] T. Hebecker, R. Buchholz, F. Ortmeier, ”Model-Based Local Path Planning for UAVs”, in Journal of Intelligent and Robotic Systems, Springer, 2014

[9] S. M. Lavalle, ”Planning Algorithms”, Cambridge University Press, 2006

[10] J. Ni, W. Wu, J. Shen, X. Fan, ”An Improved VFF Approach for Robot Path Planning in Unknown and Dynamic Environments”, in Mathematical Problems in Engineering, 2014

[11] I. Ulrich, J. Borenstein, ”VFH*: Local obstacle avoidance with look-ahead verification”, in IEEE International Conference on Robotics and Automation, vol. 3, IEEE, 2000, pp. 2505 – 2511

[12] J. S. Oh, Y. H. Choi, J. B. Park, Y.F. Zheng, ”Complete Coverage Navigation of Cleaning Robots using Triangular Cell Based Map”, in IEEE Transactions on Industrial Electronics, vol. 51, IEEE, 2004, pp. 718 - 726

[13] S. LaValle, J. Kuffner, ”Randomized kinodynamic planning”, in International Conference on Robotics and Automation, MAy 1999, pp. 473-479

Improving Monte Carlo Localization using Reflective Markers: An Experimental Analysis

Francesco Leofante^{1,2}, Gwénolé Le Moal^{3}, Gaëtan Garcia^{1} , Patrice Rabaté^{2}

Abstract— **Robust localization is a basic requirement for many applications in mobile robotics. Despite many techniques have been devised to find solutions to the localization problem, symmetrical or featureless environments represent a great challenge for most of the commonly used approaches. In this paper, we investigate how artificial landmarks can be used to reduce the chances of failure of the localization process. More specifically, an experimental analysis of a probabilistic sensor model designed to employ reflective markers is presented. The analysis will focus on two central parameters of the model, several experiments are carried out to evaluate their effect on the overall localization process.**

I. INTRODUCTION

Mobile robot localization is the problem of determining the pose of a robot relative to a given map of the environment. Localization plays an important role in nearly all robotics tasks involving autonomous navigation, unfortunately the pose of a robot can usually not be sensed directly as most robots do not have noise-free sensors for measuring pose. Therefore pose-related information must be inferred from different sources of data (e.g. odometry, additional sensors etc.), sometimes making it difficult to achieve precise localization. Moreover, structurally symmetrical or featureless environments increase the chances of failure of these techniques: the ambiguity characterizing such environments may indeed result in different robot poses being indistinguishable based on sensor data. In this paper, we focus on the problem of employing artificial landmarks to counteract the effects of the aforementioned ambiguity on the overall localization process. In particular, we wish to propose an experimental analysis of a sensor model developed to take into consideration the additional information provided by reflective markers placed in the environment at known locations. Our main contribution is to provide experimental results related to the choice of central parameters of the model by considering a concrete instantiation of the localization problem in featureless environments. Furthermore, several experiments are performed, both in simulated and real environments, to show how the proposed model improves the overall localization performance.

The remainder of this paper is organized as follows. Section II gives a brief overview on several techniques used to improve localization using landmarks. The localization algorithm used is described in Section III, which is followed in Section IV by a formal description of the sensor model employed. Section V illustrates the results collected from simulated and real experiments. Finally, Section VI provides final conclusions and directions for future works.

II. RELATED WORKS

A variety of techniques has been proposed to find solutions to the robot localization problem. Traditional approaches typically use natural landmarks present in the environment in which the robot navigates [1], [2]. These techniques, which have the advantage of not implying any change in the environment, are however prone to fail when the environment does not present suitable characteristics. This is when the use of artificial landmarks becomes beneficial. Some approaches consider landmarks which can be uniquely identified, such as RFID transponders [3], or 2D bar codes on the floor which can be detected with a camera [4] (see [5], [6] for other examples). These landmarks are used to guarantee accurate localization, greatly simplifying the localization problem. However, the mentioned solutions require specific landmark coding and the use of identification systems which are not always available on a robot. Using indistinguishable artificial landmarks like retro-reflective tape greatly alleviates this problem, as most robots are equipped with laser range finders. In [7], an approach is presented to compute a configuration of reflectors that decreases the overall ambiguity of the environment, thus increasing the robustness of the localization process. The authors addressed the landmark placement problem, formulated as the problem of selecting a subset of landmarks out of a finite set of possible configurations. This represents a fundamental problem, as there exist a relation between the localization error and the configuration of the selected landmarks as proven in [8]. The authors of [7] also developed a sensor model which allows to employ the reflectivity of the the measured objects, in addition to their range and bearing, to obtain robust localization. The sensor model, contrarily to other approaches [9], [10], does not rely on geometrical reasoning but instead operates in a probabilistic localization framework. This proves to be particularly beneficial in applications where occlusions and other uncontrollable events may make it impossible to use landmark information in a pure geometrical fashion. While the authors mostly focussed on the problem of finding an optimal placement for landmarks, our aim is to investigate the properties of the model. In this paper, we present the results of an experimental analysis of the sensor model, and study the effects it produces on the localization performance.

[1] Institut de Recherche en Communications et Cybernétique de Nantes, Ecole Centrale de Nantes, 44321 Nantes, France

[2] Airbus Group Innovations, 44340 Nantes, France

[3] CIMPA, 44340 Nantes, France

III. MONTE CARLO LOCALIZATION

Monte Carlo localization (MCL) [11] has been employed to estimate the pose of the robot. Given a map of the environment, the algorithm estimates the position and orientation of a robot as this moves and senses the environment. The algorithm uses a particle filter to represent the distribution of likely states, with each particle representing a possible state i.e. the belief $bel(x_t)$ is represented by a set of particles $\mathcal{X}_t = x_t^{[1]}, \ldots, x_t^{[M]}$. The algorithm starts with a given distribution of particles over the configuration space. Whenever the robot moves, particles are shifted to predict the new state after the movement. Whenever a new sensor reading is available, particles are resampled based on recursive Bayesian estimation. Different sensor models have been implemented to compute the likelihood of measurements, likelihood field model [12], [13] has been used here. According to this model, in order to evaluate how well the actual sensed data correlate with the predicted state, the end points of a sensor scan z_t are first projected into the global coordinate space of the map. Then, for each measurement coordinate $(x_{z_t^k}, y_{z_t^k})^T$, the likelihood of measurement z_t^k is computed using the distribution

$$p_{hit}(z_t^k) \sim \mathcal{N}(d_{hit}, \sigma_{hit}^2) \tag{1}$$

based on the distance d_{hit} between the end point of the measurement z_t^k and the closest obstacle on the map. The model assumes two additional sources of uncertainty i.e. failures, given by the point-mass distribution p_{max} and unexplained random measurements, modelled by the uniform distribution p_{rand}. The three distributions are finally combined together to compute the *importance factor* of each particle i.e. the likelihood of the estimate they carry.

MCL, in its basic implementation, solves the global localisation problem but cannot recover from robot kidnapping, or global localisation failures [12]. This is a consequence of the fact that as a position is acquired, particles at places other than the most likely pose gradually disappear. After some iterations, particles only survive near a single pose, and the algorithm is unable to recover if this pose happens to be incorrect.

This problem can be solved by injecting a number of random particles based on the estimate of localization performances. One possible way to implement this is to monitor the probability of sensor measurements $p(z_t \mid z_{1:t-1}, u_{1:t}, m)$ and relate it to the average sensor probability . This quantity can be approximated by the average of importance factor (by definition):

$$p(z_t \mid z_{1:t-1}, m) = \frac{1}{M} \sum_{m=1}^{M} w_t^{[m]} \tag{2}$$

The estimate is usually smoothed by averaging it over several time steps, since there exist multiple reasons why the measurement probability may be low, besides a localisation failure. For this reason, a short-term average of the measurement likelihood is maintained, and related to the long-term average when determining the number of random samples to add.

IV. THE SENSOR MODEL

The sensor model presents similarities to the one developed by the authors of [7], that is a variant of the likelihood field model introduced in Section III. However, contrarily to the model of [7],we decided to implement the sensor model in two steps to have more control on each phase of the localization process.

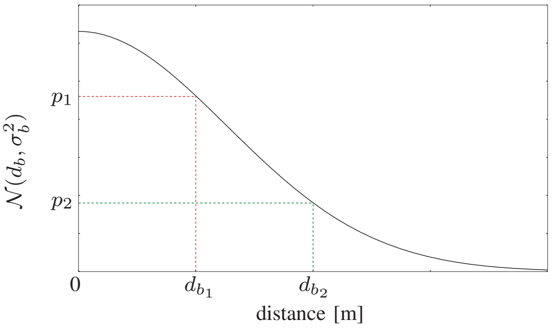


Fig. 1: In the second update step, a new coefficient is computed if markers are detected. The smaller the distance, the more likely the measurement is.

First, the likelihood of each range measurement is computed using the distribution

$$p_{hit}(z_t^k) \sim \mathcal{N}(d_{hit}, \sigma_{hit}^2) \tag{3}$$

where d_{hit} is again the distance between the end point of the measurement z^k and the closest obstacle on the map. This likelihood is used to perform a first update of the weights of the particle cloud. Next, a correction factor is applied to the weights of each particle according to the distribution

$$p_b(z_t^k) \sim \mathcal{N}(d_b, \sigma_b^2) \tag{4}$$

where d_b represents the distance to the marker which is the closest, as seen in figure 1. Here $\mathcal{N}$ represents a normal distribution with mean μ and standard deviation σ.

The resulting likelihood is therefore computed as follows:

$$p(z_t \mid x, m) = \prod_{k=1}^{N} p_{hit}(z^k) \prod_{k=1}^{N} \gamma p_b(z^k) \tag{5}$$

where γ denotes a coefficient used to weight the importance of the second update step.

V. EXPERIMENTAL EVALUATION

In order to evaluate the performances of the model considered, experiments were conducted in simulation and on a real robot. Different set-ups have been chosen, with a special focus on featureless and highly symmetrical environments. As already stated, our purpose was to carry out a thorough study of the sensor model introduced in Section IV. To this end, parameters γ and σ_b have been studied. Different combinations have been tested in order to assess how these parameters influence the localization process.

A. Simulation

A simulation has been implemented using the V-REP [14] simulation environment. The software allowed to build a fully operational simulation with which we could carry out different tests, each time focussing on a specific parameter.

The sensors used for the simulation were two Hokuyo URG 04LX laser range scanners, modified so as to return the reflectivity of the measured objects. The simulator implements laser models by means of proximity sensors or vision sensors, as physical simulations of light are not supported. In particular the model we used is based on proximity sensors, which can return the ID of the detected object. Given this, we simply decided what surface property to use depending on the object ID returned. As for the landmarks, we used round markers with a diameter of $100mm$.

The robot can be controlled using the ROS Navigation packages [15], localization was performed using the ROS implementation of MCL, called Augmented MCL (AMCL). The modifications of section IV have been applied to the general algorithm. The simulator provides built-in methods to output odometric data for the robot's pose and velocity, together with their covariance matrices.

Accurate occupancy maps of the simulated environments have been obtained using HECTOR SLAM techniques [16].

A preliminary evaluation of the sensor model was carried out using simulations in which the robot had to move along a fixed trajectory in the environment . The error on the final position was recorded, for different values of the parameters γ and σ_b, and different parameter choices were then compared.

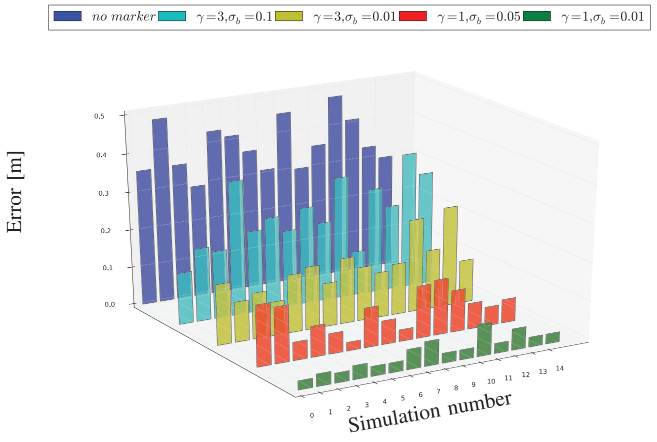


Fig. 2: Error on the final pose obtained in different simulation runs. Different values for γ and σ_b result in errors of different amplitudes. The model always produced better results when compared to traditional techniques.

As expected, the use of artificial landmarks improves the localization performance. Figure 2 shows that the average error obtained using one reflector is always lower than the one given by the traditional likelihood field model (i.e. without markers). Concerning the choice of the parameters, it is possible to notice that high precision is obtained when low values of σ_b are selected. This is normal as a sharper Gaussian in the sensor model removes more particle that are far from the real position. Such high degree of precision could be tested since experiments were run in a completely controlled environment, where uncertainties were kept to a minimum. Increasing values of the weighting coefficient γ led to a decrease in accuracy, which became more significant when higher σ_b values were chosen. This is normal as more particles survive when the Gaussian is wider, and their weight results increased by γ even if their estimate is not accurate.

B. Real Experiments

The sensor model has also been tested using real data gathered with an experimental mobile robot equipped with two SICK S300 Expert CMS laser range finders. An occupancy map of the environment was built, again using HECTOR SLAM. Experiments were carried out in a corridor of approximately $12 \times 2m$ size. In the following, the pose of the robot is defined in terms of (x, y, θ).

Three different scenarios have been considered:

a) The particle filter is initialized with a Gaussian distribution centred in $(6.10, -14.90, 0)$, the real pose of the robot. Here $\sigma_x = 0.5$, $\sigma_y = 0.5$ and $\sigma_\theta = \frac{\pi}{12}$.
b) The particle filter is initialized using a mixed distribution, with three Gaussian distributions centred respectively in $(6.10, -14.90, 0)$, $(6.70, -14.90, 0)$, $(7.40, -14.90, 0)$, all of them with $\sigma_x = 0.05$, $\sigma_y = 0.1$ and $\sigma_\theta = 0.1$. The first cluster (i.e. the one centred in $(6.10, -14.90, 0)$) represents the actual starting point of the robot.
c) The particle filter is initialized as above, but the actual starting position of the robot is actually $(12.60, -14.90, 0)$ i.e. we initialize with an initial error on x of $6m$.

Figure 5 shows the experimental setup described above.

When present, beacons have been placed at $(9.53, -13.51)$ and $(9.23, -15.57)$, as shown in figure 6.

To evaluate the model, we moved the robot along fixed trajectories on the major axis of the corridor (forward motion for cases 5a, 5b and backward motion in case 5c). The localization algorithm was executed 10 times per set of parameters, using 10000 particles initially distributed as described before. Odometry was used as ground truth to identify the correct pose of the robot at each iteration of the algorithm. When studying cases 5a and 5b, AMCL managed to provide good results both with and without reflective markers. However, we could still notice improvements in the localization performances obtained when reflectors were used, as particles are more quickly converging to the correct estimate. An analysis of the effects of different parameter sets on the the average localization error during position tracking has been carried out. Even in cases where the standard likelihood field model provided good results, the proposed approach led to improvements as shown in figure 3.

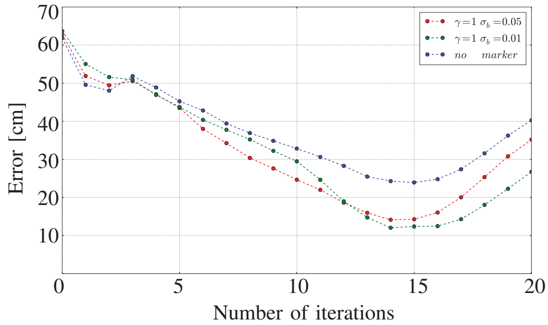


Fig. 3: Case 5a - Average localization error with and without reflective markers, and for different values of γ and σ_b. The marker is visible until iteration 15 approximatively.

Case 5b was used to gain a better understanding of the effects of the second update step introduced by the model (as explained in section IV). The evolution of the weights of the three particle clusters was monitored throughout several experiments. A performance index r_c has been defined as:

$$r_c = \frac{\omega_{first}}{\omega_{second}} \tag{6}$$

where ω_{first} is the weight of the cluster centred in the correct position, and ω_{second} is the second most weighted cluster. The presence of landmarks increased the confidence on the estimate as shown in figure 4.

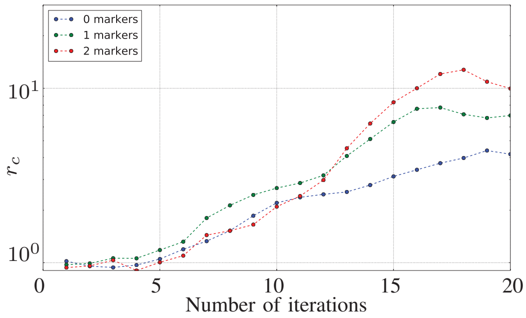


Fig. 4: Case 5b - Confidence in the estimate given by the filter.

It is possible to notice that a higher number of markers does not always result in better performances. Previous works suggest that parameters such as the number of markers and their location heavily influence the localization performance. If these parameters are not tuned properly (i.e. if the placement of landmarks is not optimal), the ambiguity of the environment may not be successfully reduced. This ultimately results in performances which are comparable to the ones obtained without markers.

When we considered a wrong initial estimate instead (case 5c), AMCL failed when no additional information was provided. Adding markers to the process helped improving the performances, allowing a significant decrease in the error. Tables I and II show the error on the final position, obtained using one marker.

Error on x (m)	Mean	Median	STD	Min. Value	Max. Value
No markers	5.17	4.59	3.50	0.29	9.92
One marker	1.96	0.81	2.41	0.06	13.17

TABLE I: Case 5c - Statistics on the x axis

Error on y (m)	Mean	Median	STD	Min. Value	Max. Value
No markers	0.57	0.18	0.82	0.01	2.66
One marker	0.25	0.25	0.15	0.03	0.50

TABLE II: Case 5c - Statistics on the y axis

Even though the robot was not able to recover completely, it is possible to see that the use of reflective markers helped reducing the localization error induced by a wrong initial estimate. In particular, table I shows that the average error on the x axis decreases from $5.17m$ when no beacons are used, to $1.96m$ when one beacon is introduced. This value would be even lower if few outliers were discarded: the median error is indeed $0.81m$, against $4.59m$ obtained without beacons. In both cases, localization provided good results on the y axis, as seen in table II.

VI. Conclusions

In this paper a study has been proposed on a sensor model which allows to use indistinguishable artificial landmarks to reduce the overall ambiguity of featureless environments. To this end, a simulation has been developed to study the performances of the model, for different configurations of the landmarks, and for different parameter values. We evaluated the model for various environments using real data. The results demonstrate that the model yields to considerable improvements in the localization performance. We expect that a more sophisticated sensor model employing laser readings to their full potential could provide more robust result. For instance, this could be done by incorporating information related to the angle of incidence of the laser at the moment of hitting the marker.

References

[1] Leonard, J.J.; Durrant-Whyte, H.F., ”Mobile robot localization by tracking geometric beacons,” Robotics and Automation, IEEE Transactions on , vol.7, no.3, pp.376,382, Jun 1991.

[2] Schmidt, T.; Hennig, D.; Fox, D.; Burgard, W., ”Estimating the absolute position position of a mobile robot using position probability grids,” Proceedings of the National Conference on Artificial Intelligence, 1996

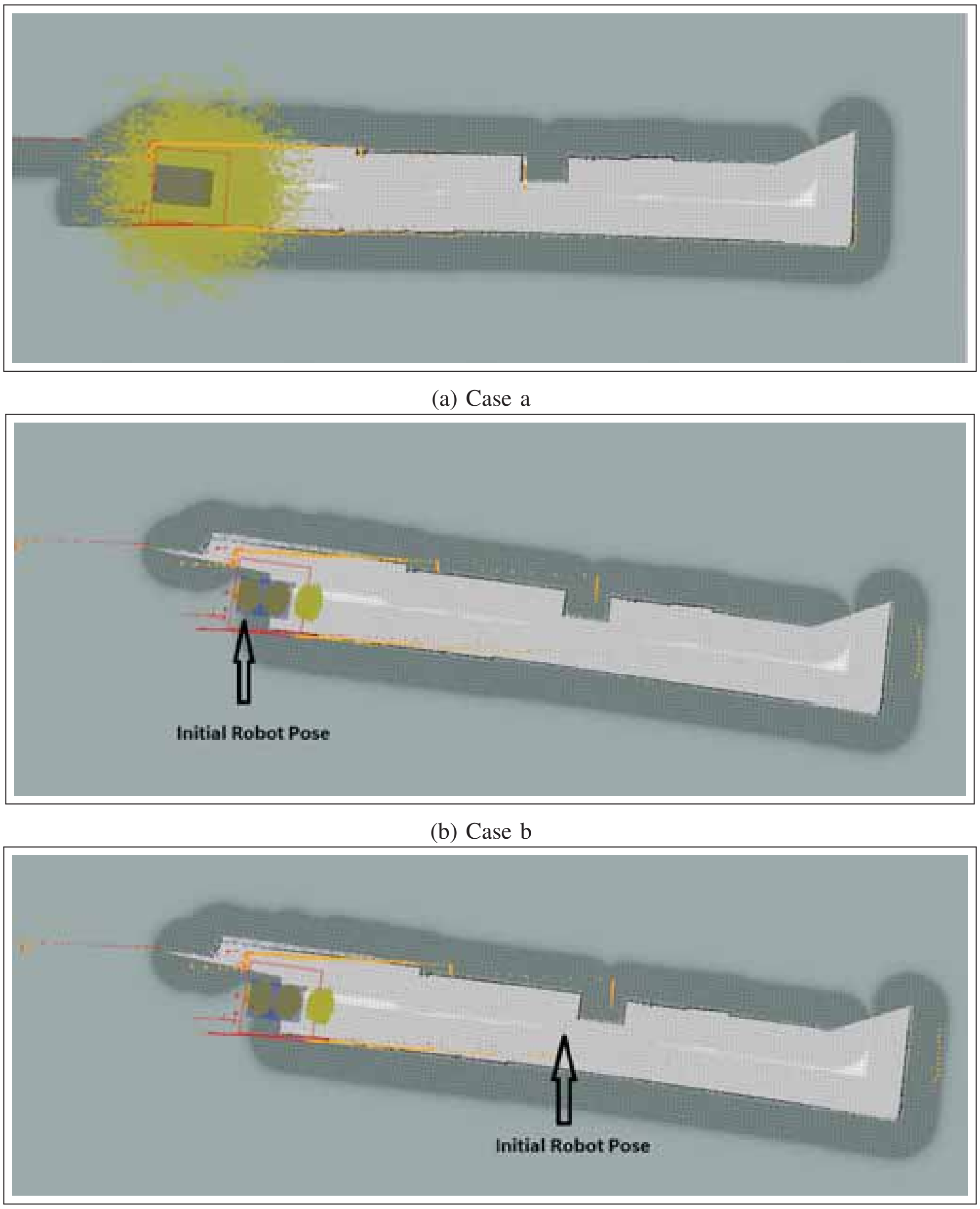
[3] Roehrig, C.; Heller, A.; Hess, D.; Kuenemund, F., ”Global Localization and Position Tracking of Automatic Guided Vehicles using passive RFID Technology,” ISR/Robotik 2014; 41st International Symposium on Robotics; Proceedings of , pp.1,8, 2-3 June 2014.

[4] Guizzo, E., ”Three Engineers, Hundreds of Robots, One Warehouse,” Spectrum, IEEE , vol.45, no.7, pp.26,34, July 2008.

[5] Howard, A.; Mataric, M.J.; Sukhatme, G., ”Relaxation on a mesh: a formalism for generalized localization,” Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on , vol.2, no., pp.1055,1060 vol.2, 2001.

[6] Rafflin, C.; Fournier, A., ”Learning with a friendly interactive robot for service tasks in hospital environments,” Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on , vol.3, pp.492,497, 1995.

[7] Meyer-Delius, D.; Beinhofer, M.; Kleiner, A.; Burgard, W., ”Using artificial landmarks to reduce the ambiguity in the environment of a mobile robot,” Robotics and Automation (ICRA), 2011 IEEE International Conference on, pp.5173,5178, 9-13 May 2011.



(a) Case a

(b) Case b

(c) Case c

Fig. 5: Different initial conditions tested to evaluate the localization performance.

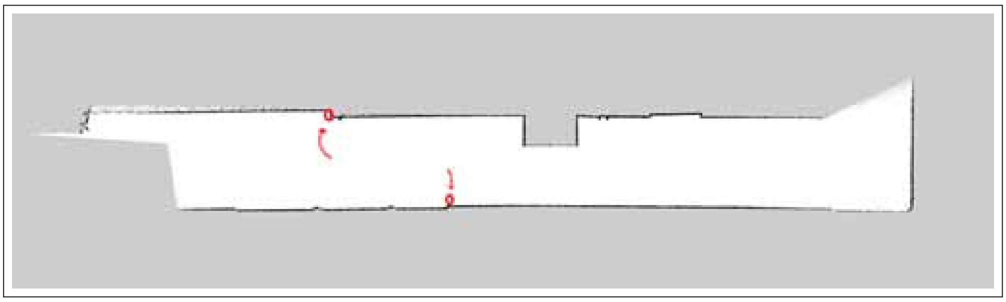


Fig. 6: Placement of the markers in the map

[8] Sutherland, K.T.; Thompson, W.B., ”Inexact navigation,” Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on , pp.1,7 vol.1, 2-6 May 1993.

[9] Salas, J.; Gordillo, J.L., ”Placing artificial visual landmarks in a mobile robot workspace,” in Proc. of the Ibero-American Conf. on Artificial Intelligence (IBERAMIA), 1998.

[10] Sinriech, D.; Shoval, S., ”Landmark configuration for absolute positioning of autonomous vehicles,” IIE Transactions, vol. 32, pp 613-624, 2000.

[11] Dellaert, F.; Fox, D.; Burgard, W.; Thrun, S., ”Monte Carlo localization for mobile robots,” Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on , vol.2, pp.1322,1328, 1999.

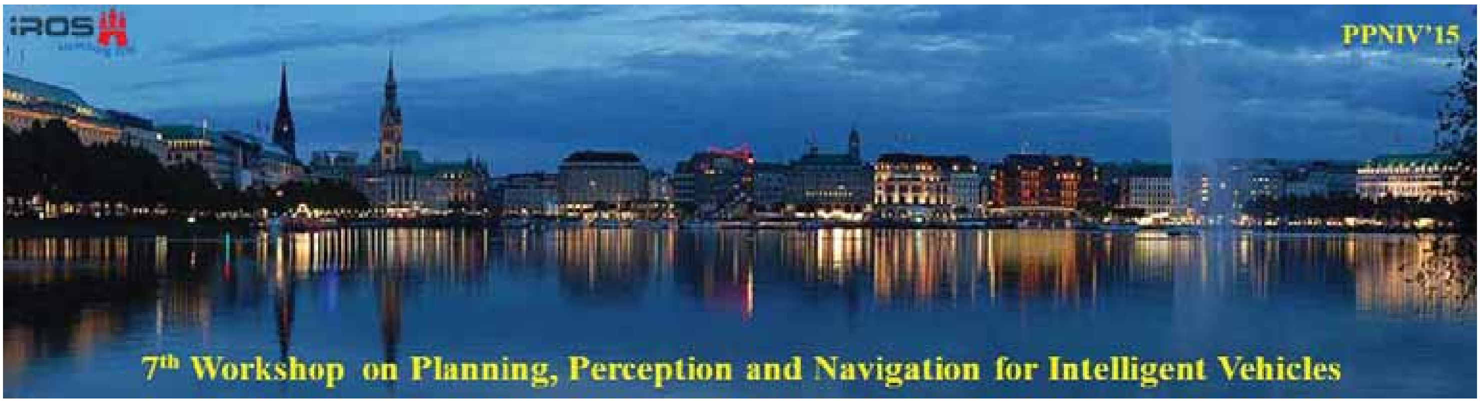
[12] Thrun, S.; Burgard, W.; Fox, D., ”Probabilistic Robotics,”, MIT Press, 2005.

[13] S. Thrun, ”A probabilistic online mapping algorithm for teams of mobile robots,” International Journal of Robotics Research, vol. 20, no. 5, pp. 335-363, 2001.

[14] Rohmer, E.; Signgh S.P.N.; Freese, M., ”V-REP: a Versatile and Scalable Robot Simulation Framework,” IEEE/RSJ International Conference on Intelligent Robots and Systems, 2013.

[15] ROS Navigation Stack, http://wiki.ros.org/navigation. Last accessed June, 18 2015.

[16] A Flexible and Scalable SLAM System with Full 3D Motion Estimation, S. Kohlbrecher and J. Meyer and O. von Stryk and U. Klingauf, Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR) 2011.


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

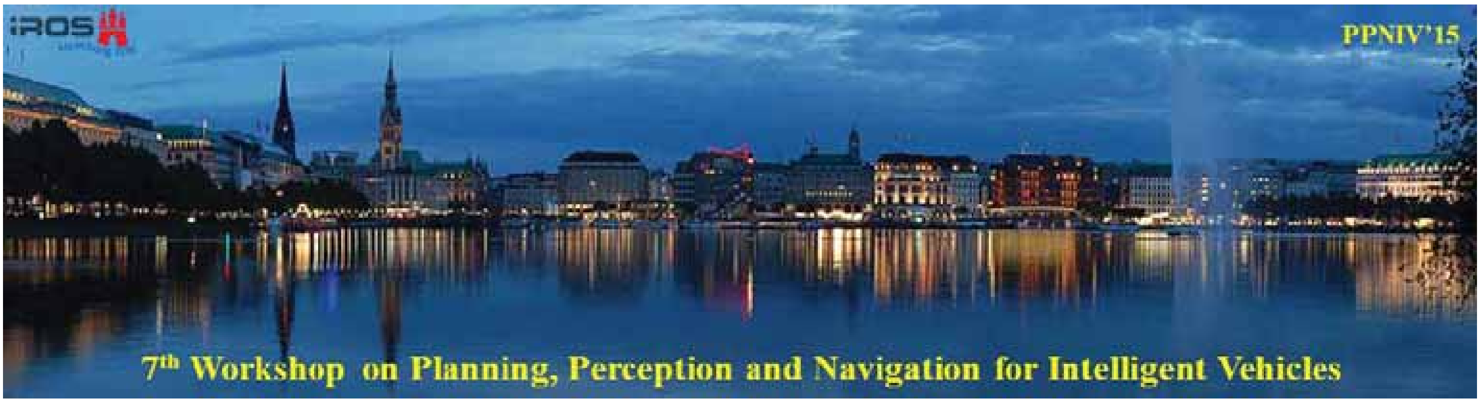
Session V

Sensing

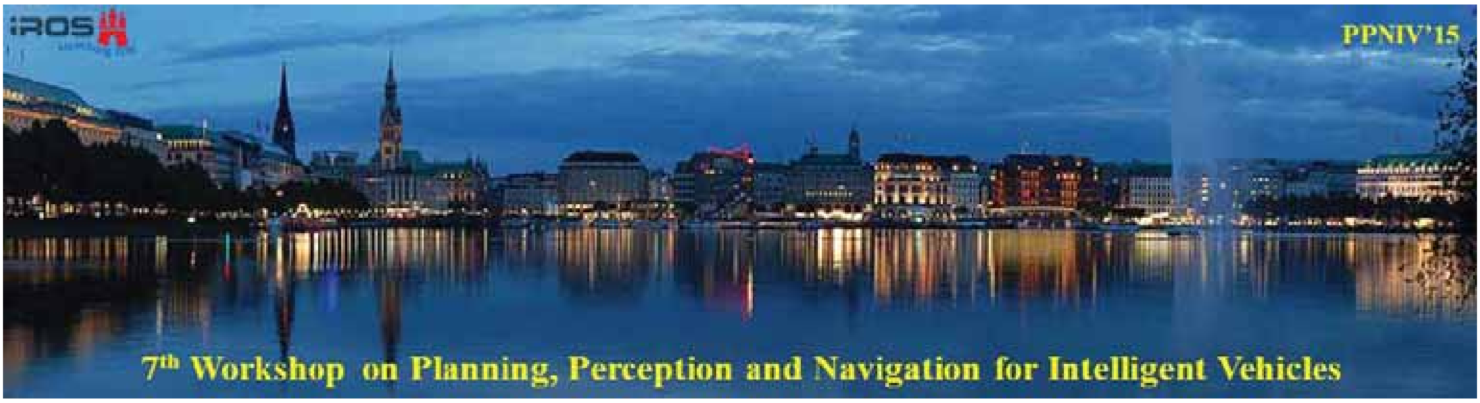
- **Keynote speaker: Christian Laugier (INRIA, France)**
  Title: Embedded Bayesian Perception & Risk Assessment for ADAS & Autonomous Cars

- **Title: 360 degre 3d ground surface reconstruction using a single rotating camera**
  Authors: Motooka, Sugimoto, Okotumi, Shima

- **Title: Towards Characterizing the Behavior of LiDARs in Snowy Conditions**
  Authors: Sebastien Michaud Jean-Francois Lalonde, and Philippe Giguere


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

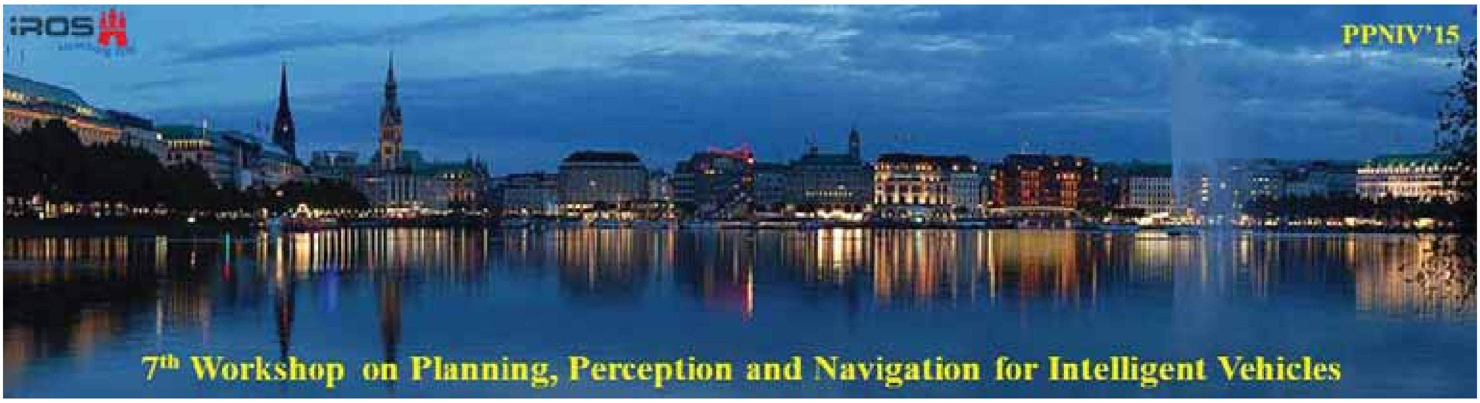
Session V

Keynote speaker: **Dr Christian Laugier**
(INRIA, France)

Embedded Bayesian Perception & Risk Assessment for ADAS & Autonomous Cars

Abstract: This talk addresses both the socio-economic and technical issues which are behind the development of the next generation of cars. These future cars will both include enhanced Advanced Driving Assistance Systems and Driverless Car functionalities. In the talk, new Bayesian approaches for Autonomous Vehicles will be presented, with an emphasis on Situation Awareness, Collision Risk Assessment, and Decision-making for safe navigation and maneuvering. It will be shown that Bayesian approaches are mandatory for developing such technologies and for obtaining the required robustness in presence of uncertainty and complex traffic situations. Results obtained in cooperation with Toyota and with Renault will also been presented.

Biography: Dr. Christian LAUGIER is first class Research Director at Inria. His current research interests mainly lie in the areas of *Motion Autonomy, Intelligent Vehicles, Embedded Perception, Decisional Architectures and Bayesian Reasoning*. He is a member of several international scientific committees and he has organized or co-organized numerous IEEE workshops and major conferences in the field of Robotics. In particular he has been General Chair, Program Chair or Program co-Chair of the international conferences IEEE/RSJ IROS’97, IROS’00, IROS’08, IROS’10, IROS’12, FSR’07 and ARSO’15. He is co-chair of the IEEE RAS Technical Committee on “Autonomous Ground Vehicles and Intelligent Transportation Systems”, and he is also member of the Steering Committee and Senior Editor of the IEEE Transactions on Intelligent Vehicle. Christian Laugier has co-edited several books and handbooks in the fields of Robotics and Intelligent Vehicles, and he also co-edited several special issues in high impact Robotics journals. He recently brought recognized scientific contributions and patented innovations to the field of Perception & Decision for Autonomous Robots and Intelligent Vehicles. He is the recipient of several awards, including the IEEE/RSJ Harashima award 2012. He has also co-founded four start-up companies.


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Embedded Bayesian Perception & Risk Assessment for ADAS & Autonomous Cars

Christian LAUGIER, Research Director at Inria

Christian.laugier@inria.fr

Contributions from

Mathias Perrollaz, Christopher Tay Meng Keat, Stephanie Lefevre, Javier-Ibanez Guzman
Amaury Negre, Lukas Rummerlhard

Keynote

Workshop PPNIV-7 , IEEE/RSJ IROS 2015, September 28th 2015, Hamburg

Content of the talk

❑ Socio-economic context & Addressed problem

❑ Bayesian Perception *(Key Technology 1)*

❑ Bayesian Risk Assessment & Decision-making *(Key Technology 2)*

❑ Conclusion & Perspectives

Automobile plays a big role in our human society

A Social & Industrial revolution in the 20th century

The car ?

A technological machine designed for enhancing individual Mobility ?

For most of cars owners it's more than that !

✓ Synonymous to **motion freedom**

✓ Often considered as a **Precious Personal Goods &** showing a particular **Social position**

✓ Often synonymous to **Driving Pleasure** *(including speed feeling)… but this is progressively changing because of rules enforcements*

✓ **Look / Performances & Comfort / Safety** are more and more considered as important criteria **….**

But the reality is somewhat different !

in particular in cities

Intelligent Mobility & Next Cars Generation

A drastic change of the Societal & Economic context

❑ Huge expected growth of the number of **Vehicles** *(~3 billions in 2050)* & of **People in cities** *(~75% of population in 2050)*

❑ Human Society is **no more accepting** all the **nuisances** & the incredible socio-economic cost of traffic **accidents** *=> 50 millions injuries & 1.3 million fatalities/Year in the world [1] … 93% of road accidents are caused by human errors !*

❑ **Driving Safety & Efficiency** are now becoming **major issues** for both **governments** *(regulations & supporting plans)* and the **automotive industry** *(technology & commercial issues)*

❑ **Growth of ADAS market:** *$16 billions at the end of 2012 … $261 billions by 2020 [2]*

❑ **New Technologies can strongly help for** *(e.g. for ADAS & Autonomous Driving)*

- ✓ *Constructing **Cleaner** & **more Intelligent cars** => Next cars generation*
- ✓ *Developing **Sustainable Mobility solutions for smart cities** => Cybercars*

[1] G.Yeomans. Autonomous Vehicles, Handling Over Control: Opportunities and Risks for Insurance. Lloyd's 2014
[2] ABI Research on Intelligent Transportation Systems and Automotive Technologies Research Services

The good news

❑ Thanks to the last decades advances in the fields of *ICT & Robotics*, ***Smart Cars & ITS*** are **gradually becoming a reality**
=> Driving assistance & Autonomous driving, Passive & Active Safety systems, V2X communications, Green technologies for reducing fuel consumption & pollution **…** *and also significant advances in **Embedded Perception & Decision-making systems***

❑ **Legal issue** is also progressively addressed by governmental authorities
*=> June 22, 2011: Law Authorizing Driverless Cars on **Nevada** roads … and this law has also been adopted later on by **California** and some **other states** in USA*

*=> Several **other countries** (including Europe, France, Japan …) are also currently analyzing the way to **adapt the legislation** to this new generation of cars*

Automotive industry

Expected evolution from ADAS to Driverless Cars?

Horizon 2020-30 ?

Nissan promises a driverless car for 2020

LE FIGARO
Date : 29/08/2013

Autonomous car: An industrial challenge for tomorrow !
French Minister of Industry & Carlos Ghosn (CEO Renault-Nissan)

Google Car 2011
=> 140 000 miles covered

Toyota
Automated Highway Driving Assist
=> Demo Tokyo 2013, Product 2016 ?

But also most of the major Automotive Constructors !
e.g.

Tesla (90% Autonomous in 2016)

Volvo, Mercedes Class S, BMW
….

Still some open questions: *Why driverless cars ? Intelligent co-Pilot v/s Full Autonomy ? Acceptability ? Legal issue ? Driver / Co-Pilot Control transitions ?*

Today talk: Addressed Problem & Challenges

Safe & Socially Compliant Vehicle Navigation
in Open & Dynamic Human Environments
Focus on Perception & Risk Assessment & Decision-making

Place Charles de Gaulle (Paris), every day

Road Safety campaign, France 2014

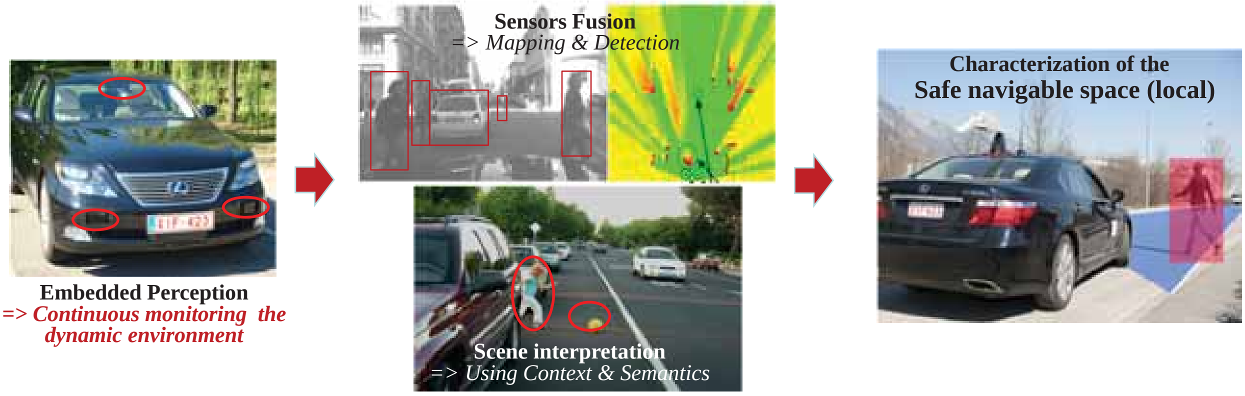
Situation Awareness & Decision-making in complex situations

Anticipation & Prediction

Main features

✓ Dynamic & Open Environments
✓ Incompleteness & Uncertainty *(Model & Perception)*
✓ Human in the loop *(Social & Interaction Constraints)*

Key Technology 1: Bayesian Perception



Embedded Perception
=> Continuous monitoring the dynamic environment

Sensors Fusion
=> Mapping & Detection

Scene interpretation
=> Using Context & Semantics

Characterization of the
Safe navigable space (local)

❑ Main difficulties

*Noisy data, Incompleteness, Dynamicity, Discrete measurements + **Real time !***

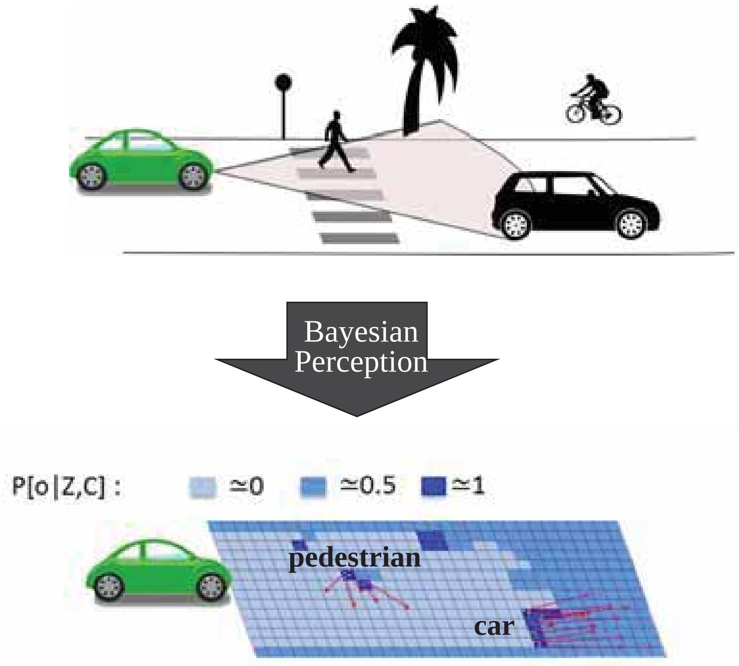
❑ Approach: Bayesian Perception

- ➢ *Reasoning about **Uncertainty** & **Time window** (Past & Future events)*
- ➢ ***Improving robustness** using **Bayesian Sensors Fusion***
- ➢ ***Interpreting the dynamic scene** using **Semantic & Contextual** information*

Bayesian Perception : Basic idea

Sensors Observations
Lidar, Stereo camera, IMU …

Environment Model
- *Sensor Fusion*
- *Occupancy grid integrating uncertainty*
- *Velocities representations*
- *Prediction models*

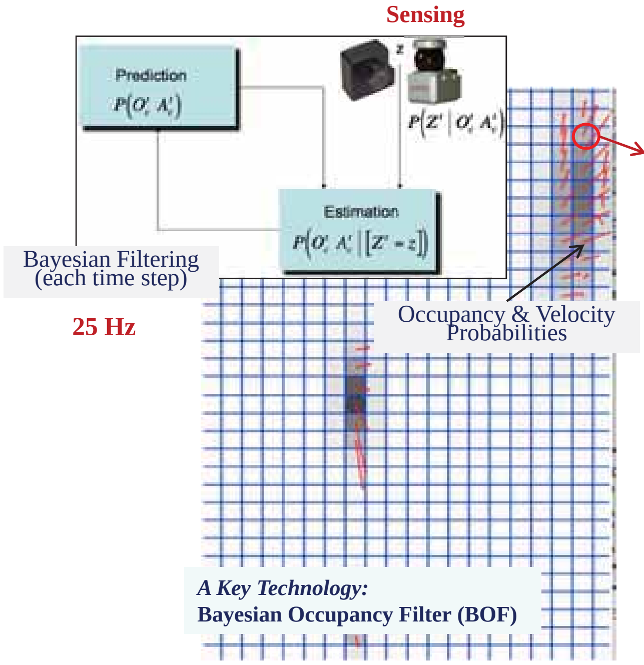


Bayesian Perception

Occupancy probability + Velocity probability
+ Motion prediction model

A new framework: *Dynamic Probabilistic Grid*

A clear distinction between Static & Dynamic parts



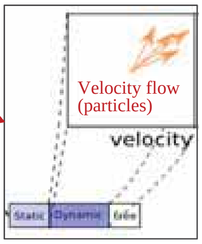
Sensing

Bayesian Filtering
(each time step)

25 Hz

Occupancy & Velocity
Probabilities

A Key Technology:
Bayesian Occupancy Filter (BOF)



Velocity flow
(particles)

❖ *Patented by Inria & Probayes, Commercialized by Probayes*

❖ *Used by: Toyota, Denso, Probayes, IRT Nanoelec / CEA*

❖ Processing Dynamic Environments using **DP-Grids** *(Occupation & Velocity Probabilities)*

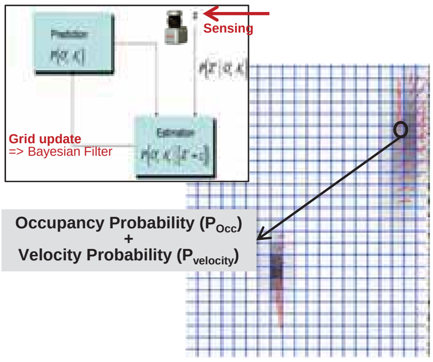
❖ Bayesian Inference + Probabilistic Sensor & Dynamic Models *(Robust to sensing errors & occultation)*

❖ Highly parallel processing *(Hardware implementation : GPU, Many-core architecture, SoC)*

Bayesian Occupancy Filter (BOF): *Outline*

Main features:

- Estimate **Spatial occupancy**
- Analyze **Motion Field** *(using Bayesian filtering)*
- **Reason at the Grid level** *(i.e. no object segmentation at this level)*



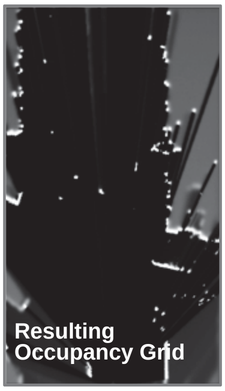
Sensing

Grid update
=> Bayesian Filter

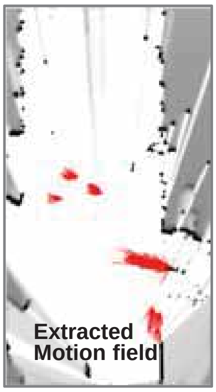
Occupancy Probability (P_{Occ})
+
Velocity Probability ($P_{velocity}$)

Camera view

Sensors data fusion
+
Bayesian Filtering



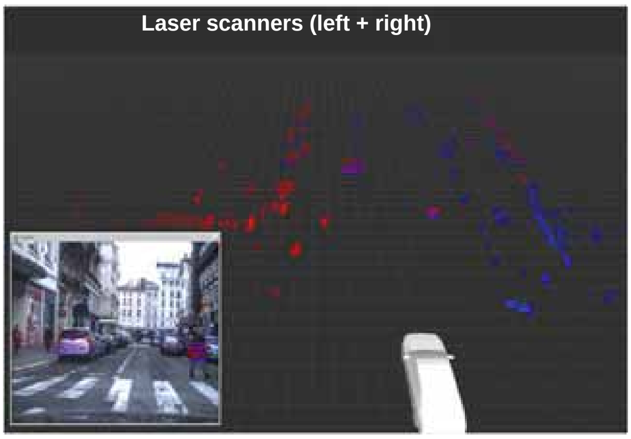
Resulting Occupancy Grid

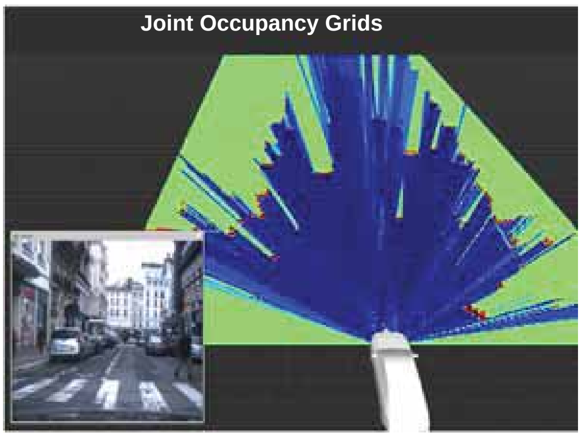


Extracted Motion field

Data fusion: *The joint Occupancy Grid*

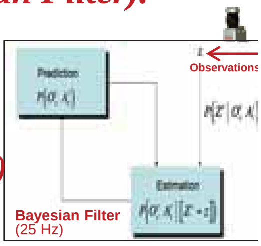
- **Observations Z_i** are given by each sensor *i* *(Lidars, cameras, etc)*
- For each set of observation Z_i , **Occupancy Grids are computed:** ***$P(O \mid Z_i)$***
- Individual grids are **merged into a single one: *P*(O | Z)**

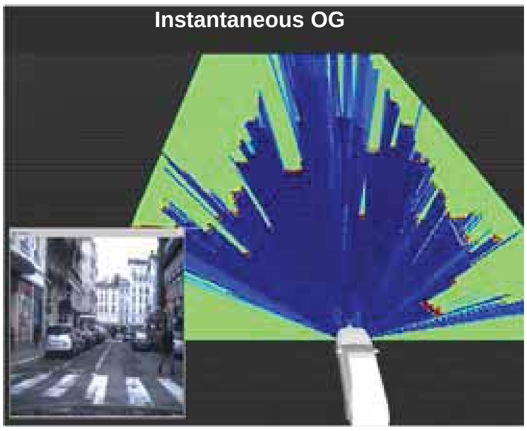


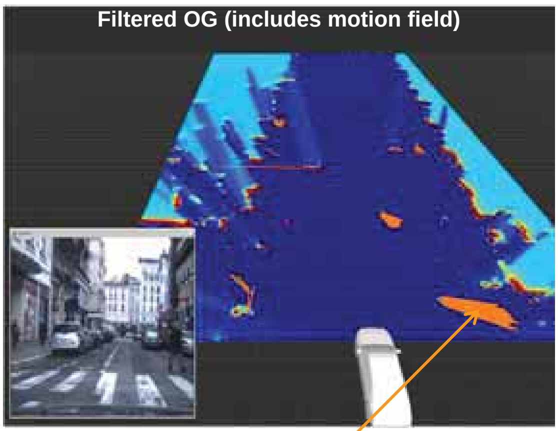


Taking into account dynamicity: *Filtered Occupancy Grid (Bayesian filtering)*

- **Filtering** is achieved through the ***prediction/correction loop*** ***(Bayesian Filter).***
  It allows to take into account grid changes over time
- **Observations** are used to update the environment model
- **Update is performed in each cell in parallel** *(using BOF equations)*
- **Motion field** is constructed from the resulting filtered data





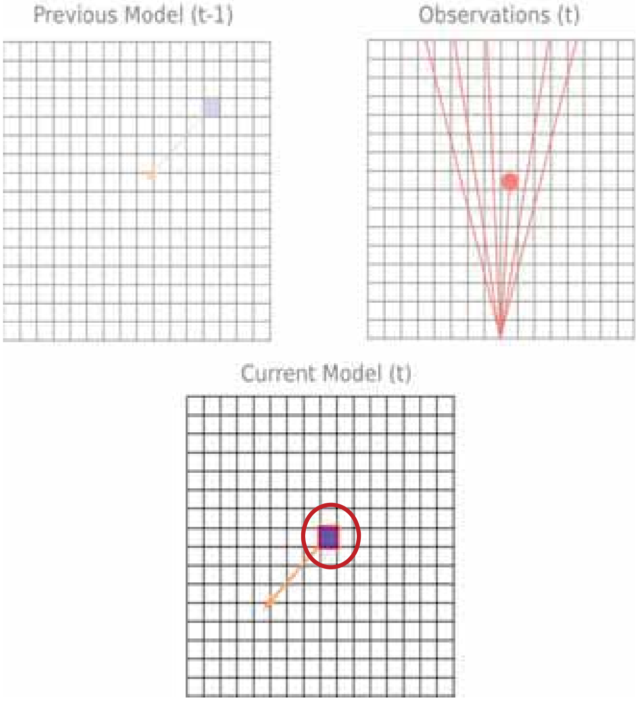


Motion field is represented in orange color

Bayesian Occupancy Filter – *Formalism*

Variables:

- C : current cell
- A : antecedent cell, i.e. the cell from which the occupancy of the current cell comes from
- O : occupancy of the current cell C
- O^{-1} : previous occupancy in the antecedent cell
- V : current velocity
- V^{-1} : previous velocity in the antecedent
- Z : observations (sensor data)



Objective:

Evaluate P(O V | Z C) : Probability of **Occupancy & Velocity** for each cell C, Knowing *the **observations Z** and the cell **location C** in the grid*

Bayesian Occupancy Filter

How to theoretically compute P(O V | Z C) ?

$$P(O\,V|Z\,C)=\lambda \sum_{A\,O^{-1}V^{-1}} P(C\,A\,O\,O^{-1}\,V\,V^{-1}\,Z)$$

Sum over the possible antecedents A and their states (O^{-1} V^{-1})

The joint probability term can be re-written as follows:

$$P(C\ A\ O\ O^{-1}\ V\ V^{-1}\ Z) = P(A)\quad P(O^{-1}\ V^{-1}\ |\ A)\quad P(O\ V\ |\ O^{-1}\ V^{-1})\quad P(C\ |\ A\ V)\quad P(Z\ |\ O\ C)$$

Joint probability => *used for the update of P(O V | Z C)*

P(A) : Selected as **uniform** (every cell can a priori be an antecedent)

P(O^{-1} V^{-1} | A) : Result from the previous iteration

P(O V | O^{-1} V^{-1}) : Dynamic model

P(C | A V) : **Indicator function** of the cell **C** corresponding to the **"projection"** in the grid of the antecedent **A** at a given velocity **V**

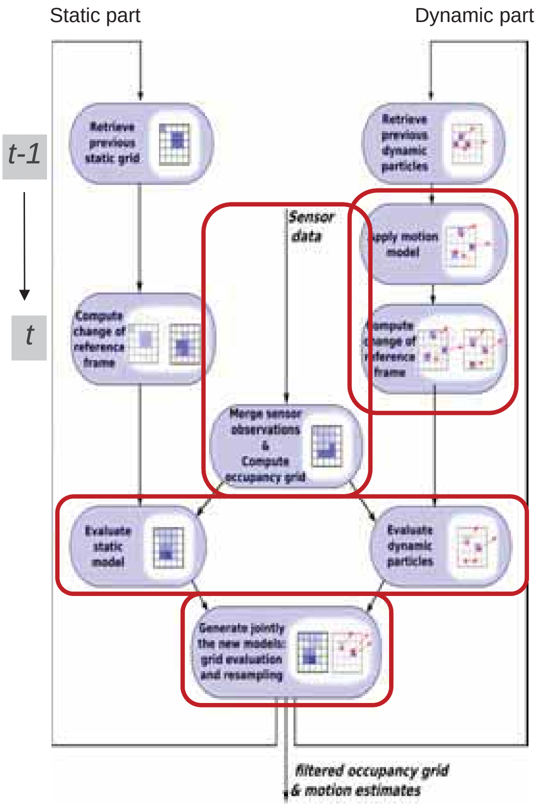
P(Z | O C) : Sensor model

Bayesian Occupancy Filter

How to compute P(O V | Z C) in practice (HSBOF process)

Main steps in the updating process

- Dynamic part (particles) is "**projected**" in the grid using motion model *(motion prediction)*
- Both Dynamic & Static parts are expressed in the **new reference frame** *(moving vehicle frame)*
- The two resulting representations are confronted to the **observations** *(estimation step)*
- **New representations (static & dynamic)** are jointly evaluated and particles re-sampled

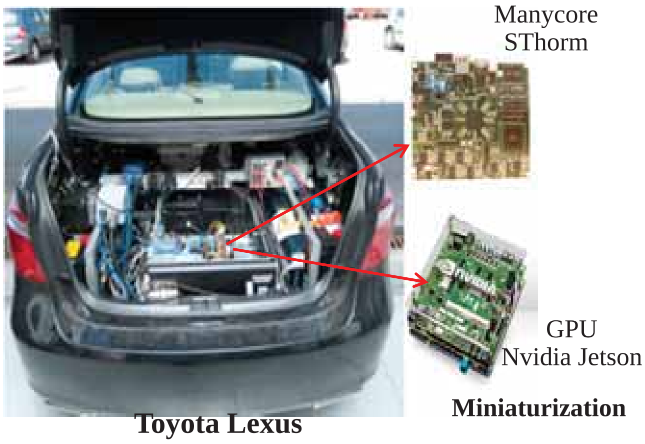

Static part
Dynamic part
t-1
t
Retrieve previous static grid
Retrieve previous dynamic particles
Sensor data
Apply motion model
Compute change of reference frame
Compute change of reference frame
Merge sensor observations & Compute occupancy grid
Evaluate static model
Evaluate dynamic particles
Generate jointly the new models: grid evaluation and resampling
filtered occupancy grid & motion estimates

Underlying Conservative Prediction Capability

=> Application to Conservative Collision Anticipation

Thanks to the prediction capability of the BOF technology, the Autonomous Vehicle "anticipates" the behavior of the pedestrian and brakes *(even if the pedestrian is temporarily hidden by the parked vehicle)*

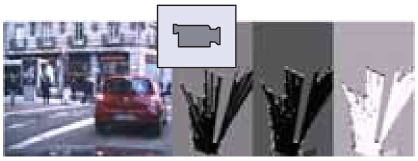
Implementation & Experimentation



Toyota Lexus

Miniaturization

Renault Zoé

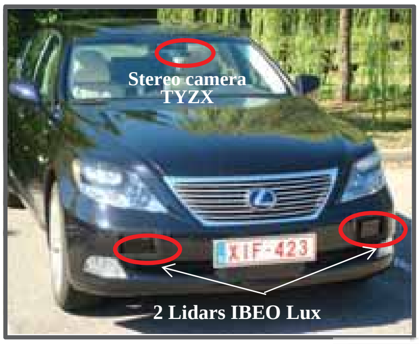


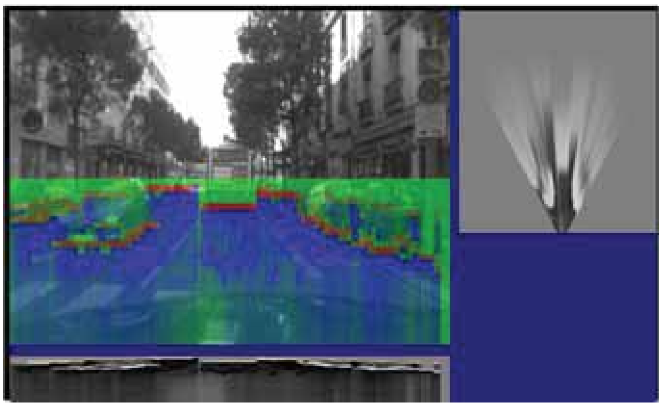
HSBOF & 2 Lidars

Bayesian Sensor Fusion (Inria / Toyota Lexus)

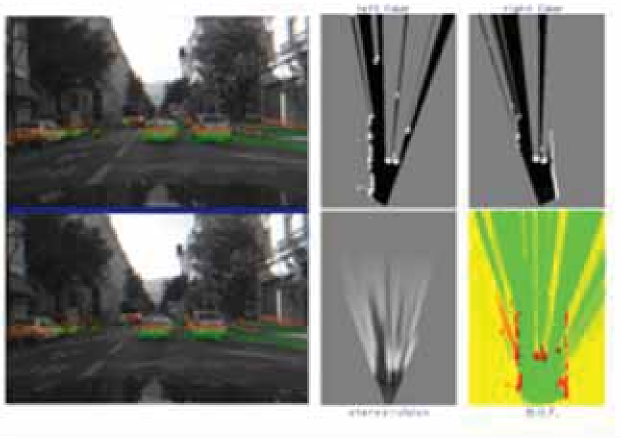
CPU+GPU+ROS / Stereo + 2 Lidars + GPS + IMU

[Perrollaz et al 10] [Laugier et al ITSM 11]
IROS Harashima Award 2012





Stereo Vision



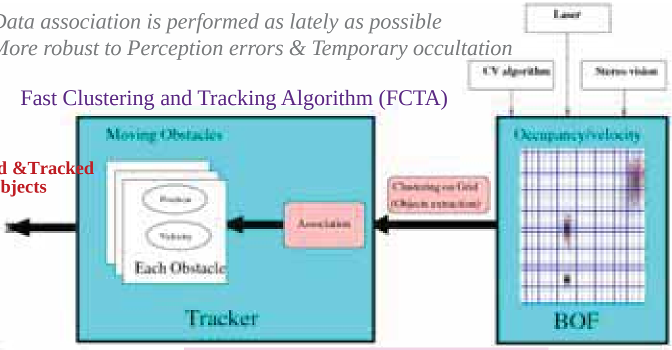
Bayesian Sensor Fusion

Grid & Object level processing architecture

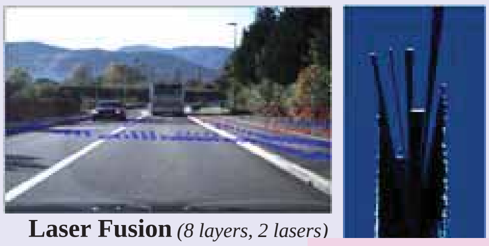
Bayesian Sensor Fusion + Detection & Tracking

- *Data association is performed as lately as possible*
- *More robust to Perception errors & Temporary occultation*

Fast Clustering and Tracking Algorithm (FCTA)

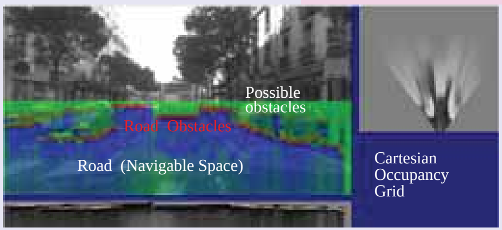


[Mekhnacha 09, Laugier et al ITSM'11]



Laser Fusion *(8 layers, 2 lasers)*

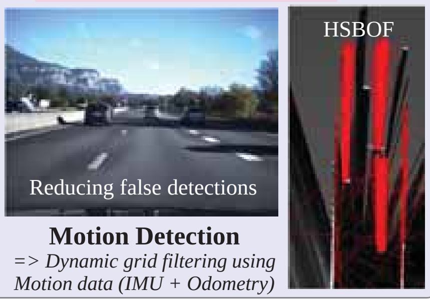
[Perrollaz et al 10-12]



Stereo-vision *(U-disparity OG+ Road/obstacle classif.)*

Multi-Lane tracker

[Qadeer et al 12, Negre et al 14]

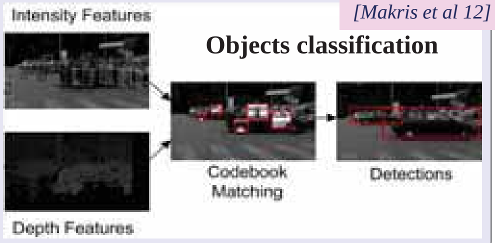


Motion Detection

=> Dynamic grid filtering using Motion data (IMU + Odometry)

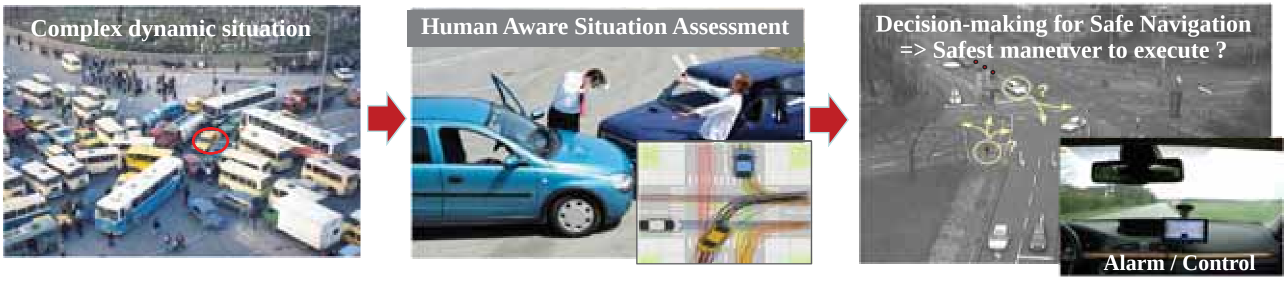
[Makris et al 12]

Objects classification



Key Technology 2: Risk Assessment & Decision

=> Decision-making for avoiding Pending & Future Collisions


Complex dynamic situation
Human Aware Situation Assessment
Decision-making for Safe Navigation
=> Safest maneuver to execute ?
Alarm / Control

❑ Main difficulties

*Uncertainty, Partial Knowledge, World changes, **Human in the loop** + **Real time***

❑ Approach: Prediction + Risk Assessment + Bayesian Decision

- ➢ *Reasoning about **Uncertainty & Contextual Knowledge** (History & Prediction)*
- ➢ ***Avoiding Pending & Future collisions** (Probabilistic Collision Risk at **t+δ**)*
- ➢ ***Decision-making** by taking into account the **Predicted behavior** of the observed mobile agents (cars, cycles, pedestrians …) & the **Social / Traffic rules***

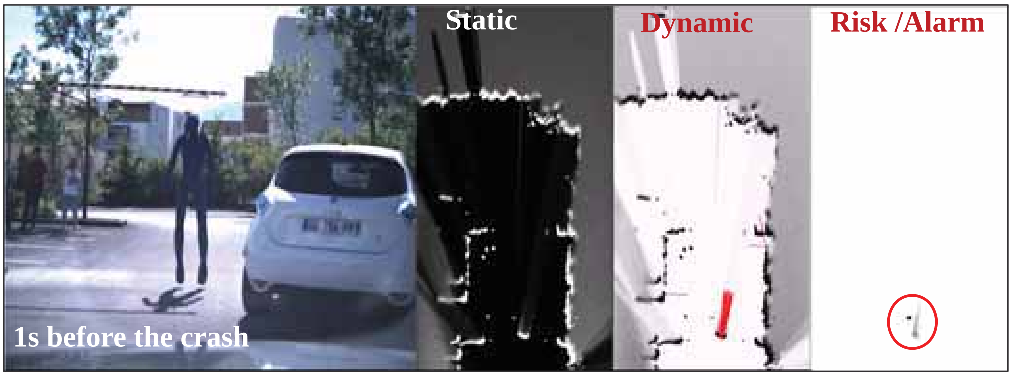
Short-term collision risk *(Grid level, Conservative)*

Objective:

- *Detect "Risky Situations" a few seconds ahead (0.5 – 3 s)*
- *Risky situations are localized in Space & Time*
- *Conservative motion prediction in the grid (Particles & Occupancy)*
- *Collision checking with Car model (shape & velocity) for every future time steps (horizon t+δ)*

δ= 0.5 s => Precrash
δ= 1 s => Collision mitigation
δ = 1.5 s => Warning / Braking

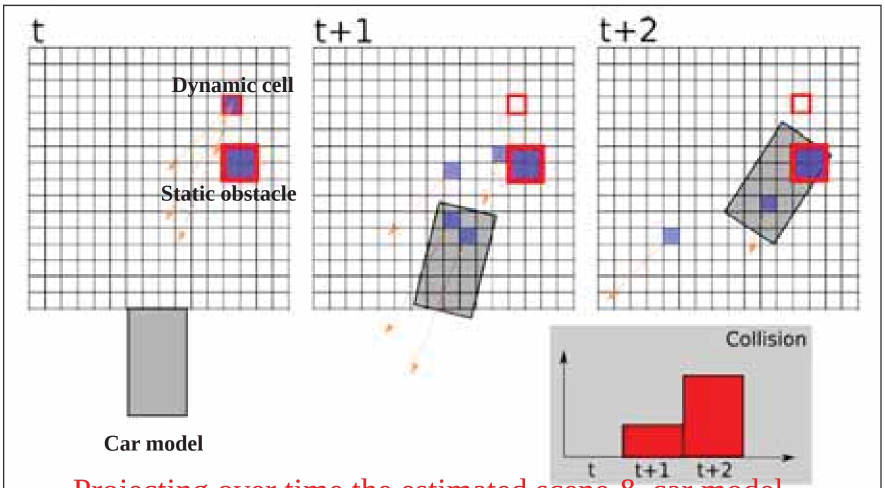
System outputs:



Short-term collision risk *(Grid level, Conservative)*

Approach *(using conservative prediction)*

- ✓**Projecting over time** the Estimated scene *(Particles & Occupancy)* & Car model *(Shape & Velocity)* *=> Apply a* ***conservative motion model (using current car motion data)***
- ✓Collision assessment for every next time step
- ✓**Integration of Risk** over a time range [t t+δ]



Projecting over time the estimated scene & car model

Short-term collision risk – *Experimental results*

Urban street experiments

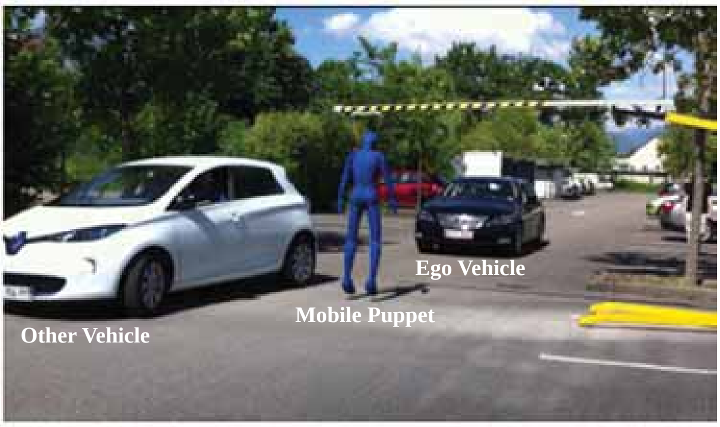
=> Almost no false alarm (car, pedestrians…)

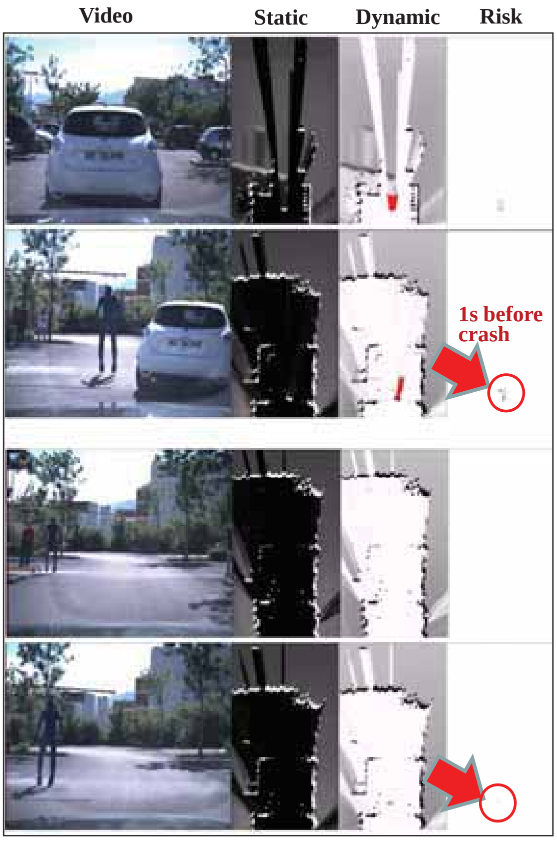
Crash scenario on test tracks

=> Almost all collisions predicted before the crash (0.5 – 2 s before)

video

Short-term collision risk – Crash scenario

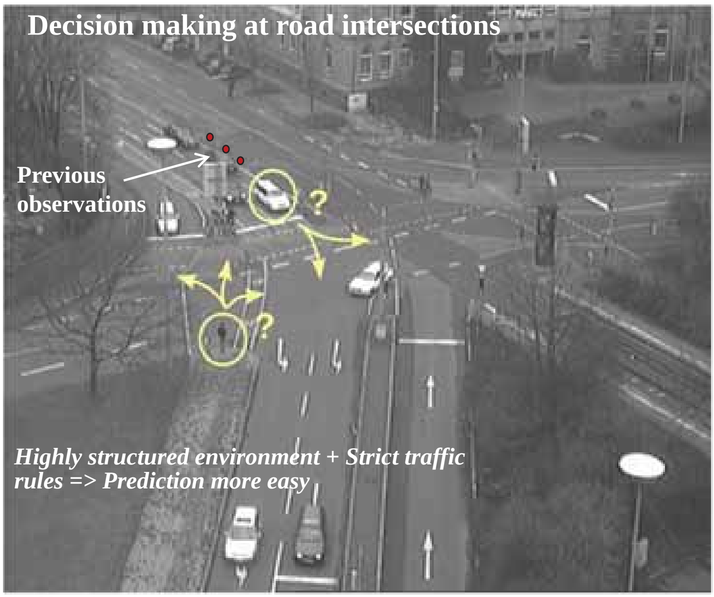




Generalized Risk Assessment (Object level)

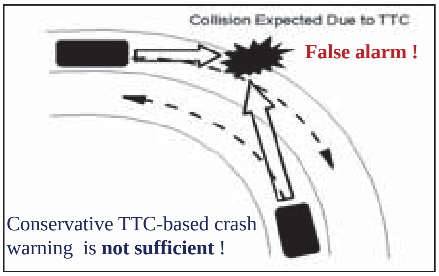
=> Increasing time horizon & complexity using **semantics**

⇒ Understand the **Current Situation** & its **likely Evolution**
⇒ Evaluate the **Risk** of future Collision for **Safe Navigation Decision**
⇒ Highly structured environment & Traffic rules make prediction more easy



Decision making at road intersections

Highly structured environment + Strict traffic rules => Prediction more easy



Conservative TTC-based crash warning is **not sufficient** !

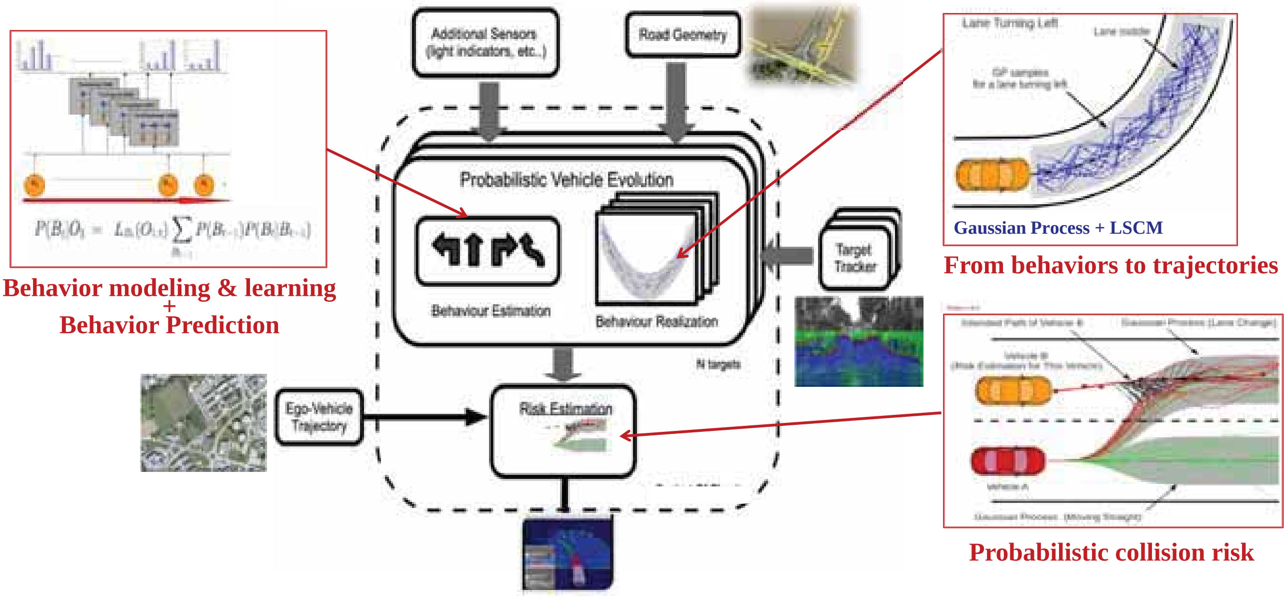
Context & Semantics
(History & Space geometry & Traffic rules)
+
Behavior Prediction
(For all surrounding traffic participants)
+
Probabilistic Risk Assessment

Behavior-based Collision risk (Object level)

Trajectory prediction & Collision Risk Assessment

[Tay thesis 09] [Laugier et al 11]

Patent Inria & Toyota & Probayes 2010



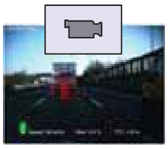
$P(B_t|O_t) = L_t(O_{1:t}) \sum_{B_{t-1}} P(B_{t-1}) P(B_t|B_{t-1})$
Additional Sensors (light indicators, etc..)
Road Geometry
Probabilistic Vehicle Evolution
Behaviour Estimation
Behaviour Realization
N targets
Target Tracker
Ego-Vehicle Trajectory
Risk Estimation
Lane Turning Left
Lane middle
GP samples for a lane turning left
Intended Path of Vehicle B
Gaussian Process (Lane Change)
Vehicle B (Risk Estimation for This Vehicle)
Vehicle A
Gaussian Process (Moving Straight)

Behavior modeling & learning
+
Behavior Prediction

Gaussian Process + LSCM

From behaviors to trajectories

Probabilistic collision risk



Video

Behavior prediction & Risk
Probayes & Inria & Toyota

Behavior-based Collision risk *(Object level)*

Intention & Expectation approach

Intersection:

- ✓ *Complex Geometry & Traffic context*
- ✓ *Large number of Vehicles & Possible Maneuvers*
- ✓ *Vehicle behaviors are **Interdependent***
- ✓ ***Human Drivers are in the loop !***

Human in the loop & Interdependent behaviors

⇒ Detect drivers errors & Colliding behaviors

⇒ **Risk** = Comparing maneuvers Intention & Expectations (using DBN)

Behavior-based Collision risk *(Object level)*

Intention & Expectation approach

Patent Inria & Renault 2012 (intersections) + Patent Inria & Berkeley 2013 (generalization)

[Lefevre thesis 13] [Lefevre & Laugier IV'12, Best student paper]

A Human-like reasoning paradigm *=> detect Drivers Errors & Colliding behaviors*

✓ *Estimating **"Drivers Intentions"** from Vehicles States Observations (X Y θ S TS)*

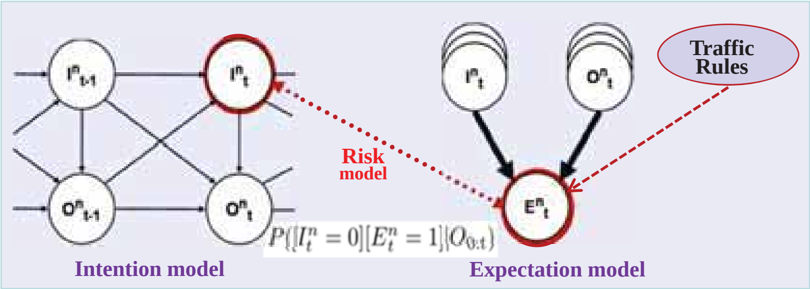
=> Perception or V2V communication

✓ *Inferring **"Behaviors Expectations"** from Drivers Intentions & Traffic rules*

✓***Risk*** = *Comparing Maneuvers **Intention & Expectation** using a **"Dynamic Bayesian Network"*** $P([I_t^n = 0][E_t^n = 1]|O_{0:t})$

=> Taking traffic context into account (Topology, Geometry, Priority rules, Vehicles states)

=> Digital map obtained using "Open Street Map"

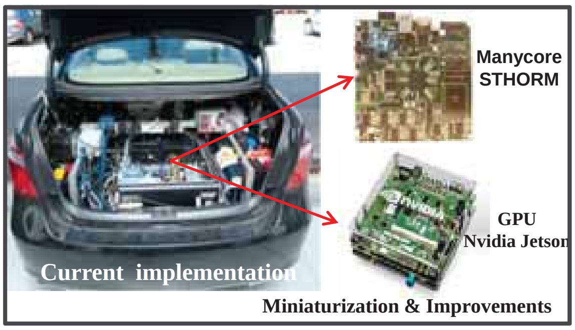


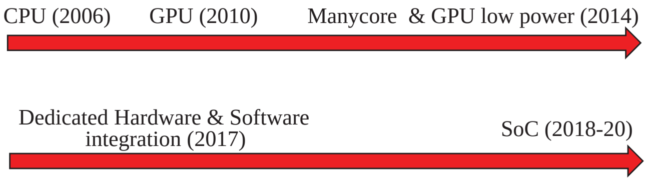
Dynamic Bayesian Network

Current & Future work

- ❑ **Approaches for Software & Hardware integration** *(Embedded Perception)*

  => *Reduce drastically Size, Weight, Energy consumption, Cost … while improving Efficiency*

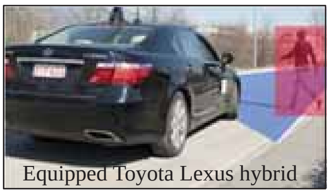


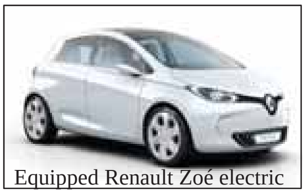

CPU (2006) GPU (2010) Manycore & GPU low power (2014)

Dedicated Hardware & Software integration (2017) SoC (2018-20)

*=> Coop. CEA LETI (common projects & **PhD student)***

- ❑ **Technologies for Autonomous Driving** *(Perception + Decision + Control + Learning)*
  - ✓ *PhD on **"Driving Decisional Process"** => Coop. Berkeley & Renault*
  - ✓ *PhD on **"Models & Algorithms for Autonomous Driving"** => Toyota*

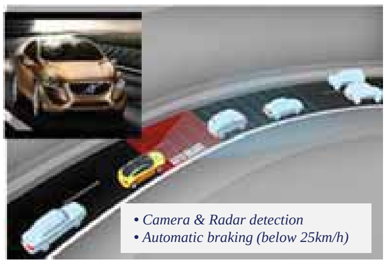

Equipped Toyota Lexus hybrid


Equipped Renault Zoé electric

Conclusion & Perspectives

- ❑ **Intelligent Cars (ADAS & Future Driverless Cars)** are gradually becoming a reality

Parking Assistant (2004)

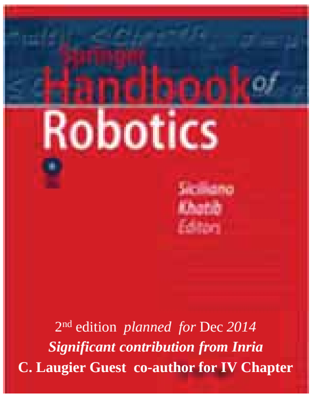

• Camera & Radar detection
• Automatic braking (below 25km/h)
Volvo Pedestrian avoidance system (2011)

Fully Autonomous Driving (2025-30 ?)

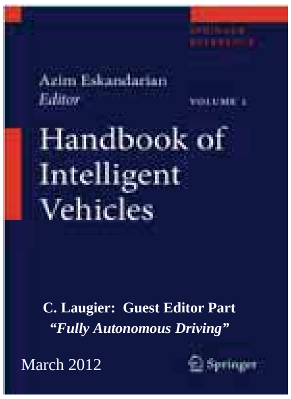
- ❑ **Bayesian Perception & Situation Awareness & Bayesian Decision** are ***key Technologies for dealing with uncertainty & addressing the Challenge of Autonomous Vehicles***

- ❑ **Several implementations** on commercial cars & **Tests in realistic traffic situations** have successfully been performed

  *…. However system **Robustness & Efficiency** have still to be improved, in particular when human is in the loop (Share control & Interaction)*

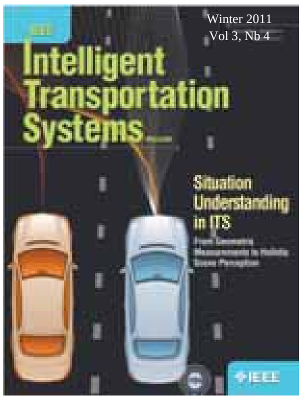


2nd edition *planned for Dec 2014*
Significant contribution from Inria
C. Laugier Guest co-author for IV Chapter

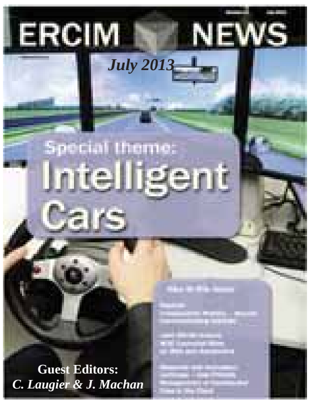


C. Laugier: Guest Editor Part
"Fully Autonomous Driving"

March 2012


IEEE
Intelligent Transportation Systems
Situation Understanding in ITS
IEEE

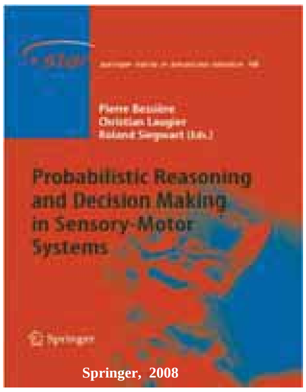
Winter 2011
Vol 3, Nb 4



July 2013

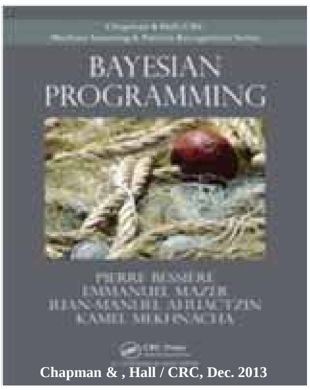
Guest Editors:
C. Laugier & J. Machan

Thank You Any questions ?


Pierre Bessière
Christian Laugier
Roland Siegwart (Eds.)
Probabilistic Reasoning and Decision Making in Sensory-Motor Systems
Springer

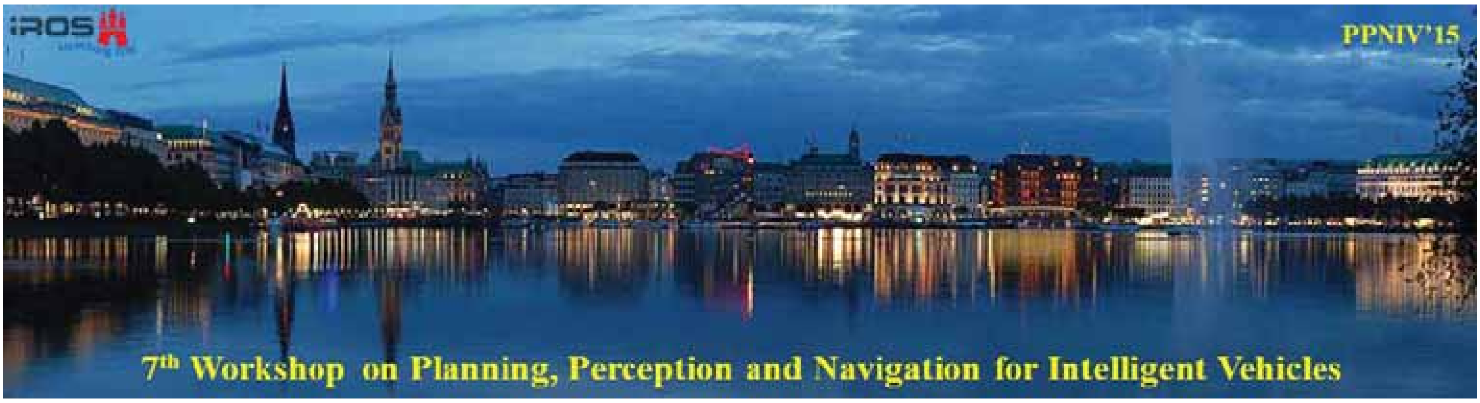
Springer, 2008

IEEE RAS Technical Committee on "AGV & ITS"
Numerous Workshops & Special issues since 2002

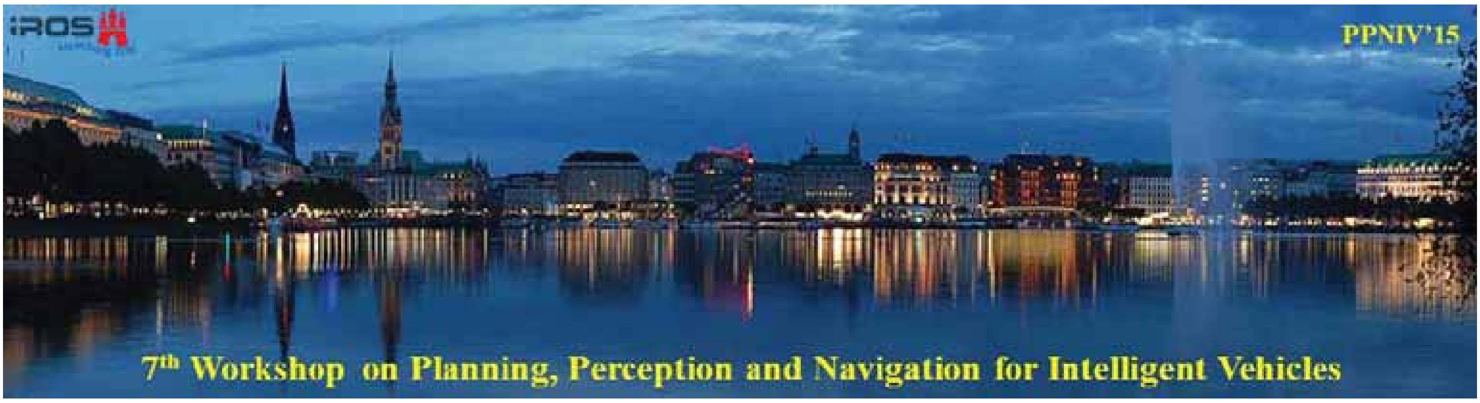


Chapman & , Hall / CRC, Dec. 2013

christian.laugier@inria.fr - http://emotion.inrialpes.fr/laugier


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

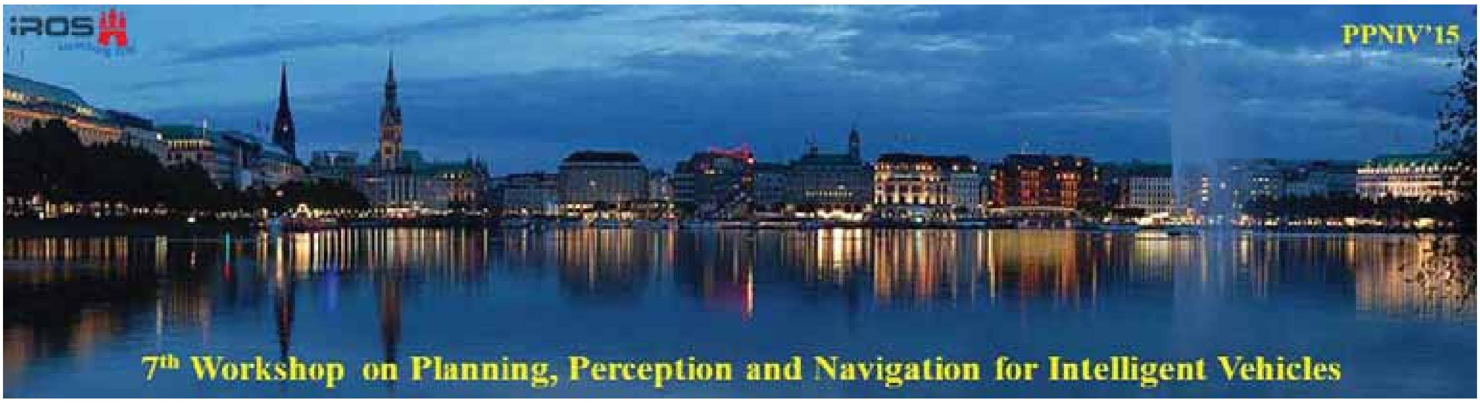
2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session V

Sensing

- **Title: 360 degre 3d ground surface reconstruction using a single rotating camera**
  Authors: Motooka, Sugimoto, Okotumi, Shima

- **Title: Towards Characterizing the Behavior of LiDARs in Snowy Conditions**
  Authors: Sebastien Michaud Jean-Francois Lalonde, and Philippe Giguere


iROS
PPNIV'15
7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems

360-Degree 3D Ground Surface Reconstruction Using a Single Rotating Camera*

Kouma Motooka[1], Shigeki Sugimoto[1], Masatoshi Okutomi[1], Takeshi Shima[2]

***Abstract*—We propose a method for reconstructing 360-degree 3D ground surfaces in high precision from images captured by a single rotating camera, assuming that the camera is mounted at an off-centered position on a construction equipment whose upper body is rotatable around a single axis (e.g. power shovel). We estimate a regular-grid ground surface, whose coordinate system is determined from the camera positions estimated by a standard structure from motion (SfM) technique. To produce high-quality ground surfaces, we first initialize the ground surface by fitting to the 3D points from SfM, then we minimize the variance of pixel values over the whole ground surface, where all contributable pixels are treated equally for the measurements, while removing outlier pixels which usually appear due to self-shadows and lens-flares under shiny weather conditions. The validity of the proposed method is demonstrated through experiments using synthetic and real images.**

I. INTRODUCTION

Estimating 3D ground surface is an important task for heavy machinery working in construction sites, for measuring earth volumes and detecting traversable areas. The 360-degree surrounding ground surface is often required for the machinery which is able to move in any direction by using two caterpillars. Considered with the tough works and harsh conditions in construction fields, the measurement device should be as simple and easy for maintenance as possible.

In this paper we propose a method for reconstructing a 360-degree 3D ground surfaces from the images captured by a single rotating camera, assuming that the camera is mounted at an off-centered position on a heavy construction equipment whose upper structure can rotate more than 360 degrees in a single axis (e.g. power shovel with hydraulic swivel joint), as shown in Fig.1.

The main contribution of this paper lies in our robust direct ground surface reconstruction stage for producing a high quality ground surface, represented by a fine regular mesh for generating a digital elevation map (DEM) with surface normals [16]. We minimize the variance of pixel values over the ground surface, where every pixel contributes to the measurements while every image is treated equally. However, the direct reconstruction method is adverse to outlier pixels engendered by self shadows and lens flares, which appear quite often when we rotate a camera in shiny outdoor conditions. We effectively detect the outlier pixels by taking the median of the pixel values instead of the mean value in the variance computation. To make this direct method feasible, we first estimate an initial ground surface by fitting to the sparse 3D points obtained by a standard structure from motion (SfM) technique with loop closing. Then the surface is robustly refined by a hierarchical meshing technique in the direct reconstruction stage. The validity of the proposed method is demonstrated through experiments using synthetic and real images.

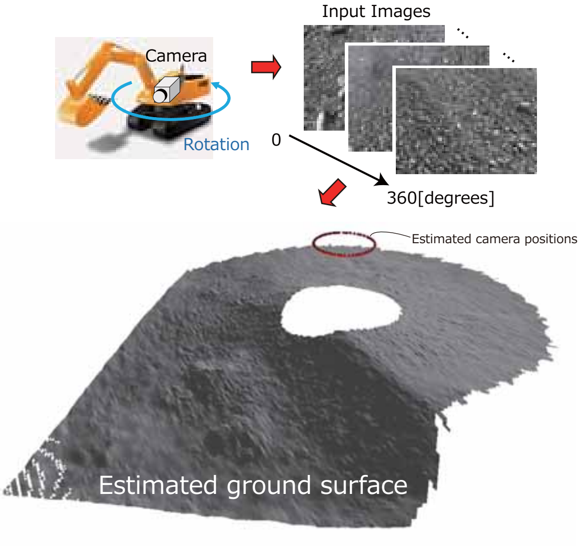


Fig. 1. Overview.

II. RELATED WORKS

The methodology for panoramic (360-degrees) 3D reconstruction has been extensively studied in the literature, including methods using multiple omnidirectional cameras (*e.g.* [15], [18]) and using a rotating standard or specially designed camera(s) (*e.g.* [8], [11], [17]). For our scenario where the imaging system is mounted on a construction equipment which is exposed to chronic vibrations in daily works, we need a sturdy and easy-to-calibrate standard imaging system instead of specially-designed naive one. Unfortunately, the previous works using a rotating standard camera (*e.g.* [17]) have adopted traditional window matching techniques for front-parallel surfaces, not taking the highly slanted ground surfaces into consideration.

On the other hand, calibrated binocular stereo cameras are widely used to estimate dense depth maps for indoor and outdoor scenes including ground surfaces (*e.g.* [1], [6], [20]). In exploration scenarios, stereo image sequences are

*This work was partly supported by the Grants-in-Aid for Scientific Research (No. 25240025) from JSPS.

[1] Dept. of Mech. and Control Eng., Graduate School of Sci. and Eng., Tokyo Institute of Technology. Ōokayama, Meguro-ku, Tokyo, 152–8550 Japan {kmotooka,shige,mxo}@ok.ctrl.titech.ac.jp

[2] Hitachi Research Laboratory, Hitachi Ltd. Ōmikacho, Hitachi-shi, Ibaraki, 319–1292 Japan takeshi.shima.rb@hitachi.com

successfully used in SLAM (simultaneous localization and mapping) for incrementally building ground metric maps [2], [10], [14], [16]. In our case a possible choice is to apply a stereo-SLAM technique to the rotational camera motion using commercially available compact stereo system. Although a synchronized stereo system is profitable for incremental 3D map estimation, the images are actually redundant in the batch processing for the images from rotating stereo cameras, because the maximum baseline length is determined by the camera rotation diameter rather than the baseline length of the compact stereo system [8]. Otherwise the stereo calibration of a wide-baseline stereo system is troublesome.

Recovering camera poses and 3D points from image collections becomes a feasible task according to the success in the recent developments of SfM techniques [12], [19], [21]. This technique is followed by dense multiview stereo (MVS) reconstruction [5] and surface model fitting [9] for reconstructing 3D surfaces. Instead of taking separated steps after SfM, our approach directly estimates the ground surface represented by the vertex heights of the regular grid mesh drawn on a level plane. In addition, other than the standard approach in the recent MVS technique, where a special *reference* image is selected for computing pairwise error criteria (*e.g.* squared pixel value differences or normalized cross correlation), we adopt the minimization of pixel value variance [4], where every image pixel equally contributes to the measurements [3]. However, in our scenario where the camera rotates 360 degrees in outdoor environments, we cannot avoid the camera's self-shadows and lens flares which the recent SfM-MVS framework does not take into careful consideration.

III. 3D GROUND SURFACE RECONSTRUCTION USING ROTATING CAMERA

We assume that the camera is mounted on the rotatable upper structure of a construction equipment and that the camera intrinsic parameters, the camera rotation radius, and the height of the camera mount position are known.

Our method is composed of four steps as shown in Fig.2. In the first step, we apply a standard SfM technique for estimating the camera positions and 3D points. In the next step, we fit the estimated camera positions to a 3D circle, whose rotation radius gives the scene absolute scale. We also set a reference (ground) coordinate system by using the position of the estimated camera rotation circle, followed by setting a regular mesh composed of piecewise triangular patches on the x-y plane in the reference coordinate system. Then the 3D points estimated by SfM are used for estimating the regular-grid ground surface which is parameterized with the vertex heights of the regular mesh, $\boldsymbol{z} = (z_1, z_2, \cdots, z_V)$. Finally, in the last step, we refine the ground surface by minimizing the variances of pixel values.

A. Structure from motion

We start with the standard single-camera SfM technique [19] with simple loop closing for estimating camera positions and 3D points of features on the off-road ground from all images, $I_n, (n = 1, 2, \cdots, N)$, captured while the camera rotates about 360 degrees. Since a typical off-road ground surface has a well-textured but homogeneous texture pattern (*e.g.* by pebbles), which engenders many mismatches under the standard SIFT matching parameter setting used for urban city environments, we use tight thresholds for the SIFT matching score and ratio test [13] for obtaining highly unique feature matches on the ground, resulting in a relatively small number of the 3D points against to the rich texture.

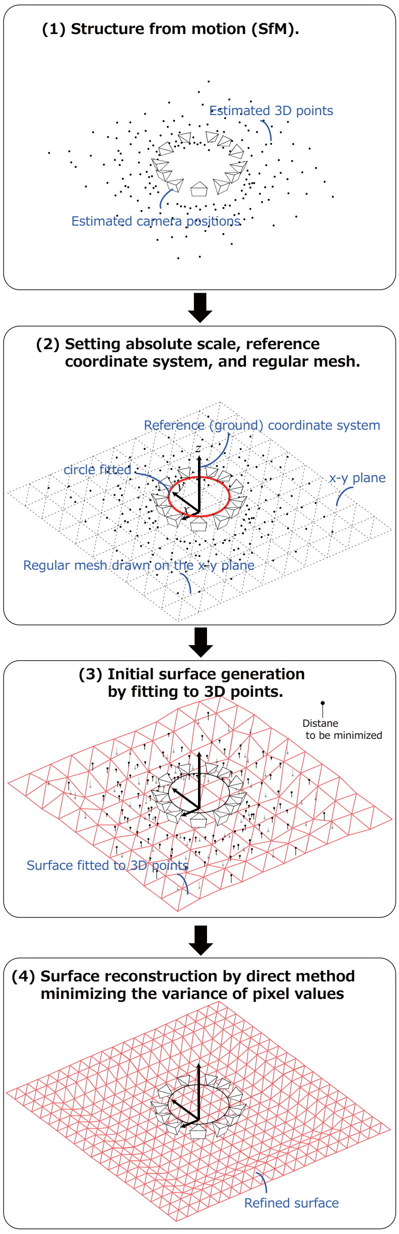
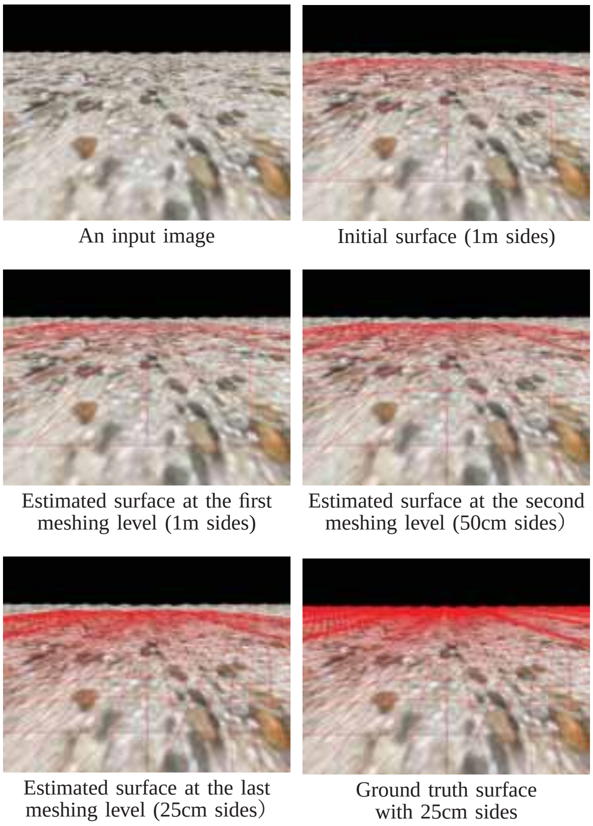

(1) Structure from motion (SfM).
Estimated 3D points
Estimated camera positions
(2) Setting absolute scale, reference coordinate system, and regular mesh.
Reference (ground) coordinate system
circle fitted
x-y plane
Regular mesh drawn on the x-y plane
(3) Initial surface generation by fitting to 3D points.
Distane to be minimized
Surface fitted to 3D points
(4) Surface reconstruction by direct method minimizing the variance of pixel values
Refined surface

Fig. 2. Algorithm flow.

B. Setting absolute scale, reference coordinate system, and regular mesh

In this step we compute the scene absolute scale by equalizing the known camera rotation radius with the radius of a camera rotation circle, which is obtained by fitting to the camera positions estimated by SfM. In the 3D circle fitting


An input image

Initial surface (1m sides)

Estimated surface at the first meshing level (1m sides)

Estimated surface at the second meshing level (50cm sides)

Estimated surface at the last meshing level (25cm sides)

Ground truth surface with 25cm sides

Fig. 3. Input image and its overlaps with surfaces.

process, we first estimate a plane which the camera positions lie in, by minimizing the distance between the plane and the estimated camera positions, followed by a standard 2D circle fitting technique using the projected camera positions on the plane. We also determine the position of the reference coordinate system, where the z-axis corresponds to the rotation axis and the x-y plane is arranged so that the circle plane is at the height of the camera mount position. On the x-y plane we also define a regular grid mesh which is composed of piecewise triangular patches (See Fig.2(2)).

C. Initial surface estimation by fitting to 3D points

We estimate an initialize ground surface, parameterized with the vertex heights of the regular mesh, $\boldsymbol{z} = (z_1, z_2, \cdots, z_V)$, by fitting to the 3D points, $\boldsymbol{x}_m = (x_m, y_m, z_m)^T, (n = 1, 2 \cdots, M)$, estimated by SfM. We minimize the following linear least squares cost $C_F(\boldsymbol{z})$:

$$C_F(\boldsymbol{z}) = C_{FD}(\boldsymbol{z}) + C_{FS}(\boldsymbol{z}) \quad (1)$$

where

$$C_{FD}(\boldsymbol{z}) = \sum_m (\boldsymbol{\lambda}_m^T \boldsymbol{z} - z_m)^2 \quad (2)$$

$$C_{FS} = \alpha_{FS} \sum_v \left(\boldsymbol{z}_v - \frac{1}{8} \sum_{v' \in \mathcal{N}(v)} z_{v'} \right)^2 \quad (3)$$

where C_{FD} and C_{FS} indicate the data term and the smoothness term, respectively. In the data term, $\boldsymbol{\lambda}_m^T \boldsymbol{z}$ denotes the surface height (*i.e.* z value) at $(x, y)^T = (x_m, y_m)^T$, induced by the objective parameter vector $\boldsymbol{z}$. Namely, $\boldsymbol{\lambda}_m^T \boldsymbol{z} - z_m$ denotes the difference in z between the surface point and the 3D point, where $\boldsymbol{\lambda}_m$ has only three non-zero elements, representing barycentric interpolation weights for the three vertex heights of the triangular patch that involves $(x_m, y_m)^T$. The smoothness term (3) represents the sum of the squared Laplacian convolution outputs over the mesh, where α_{FX} is a constant.

We found that the 3D points from SfM are too noisy in the 3D space to recover detailed surfaces with a fine mesh. We also know that the number of the 3D points from SfM is not always enough for recovering desirable ground surfaces. Therefore we apply this surface fitting for a rough surface mesh (*i.e.* with a large patch size).

D. Surface reconstruction by direct method minimizing the variance of pixel values

In the final step we minimize the following cost C_P for refining the initial ground surface estimated in III-C:

$$C_P(\boldsymbol{z}) = C_{PD}(\boldsymbol{z}) + C_{PS}(\boldsymbol{z}) \quad (4)$$

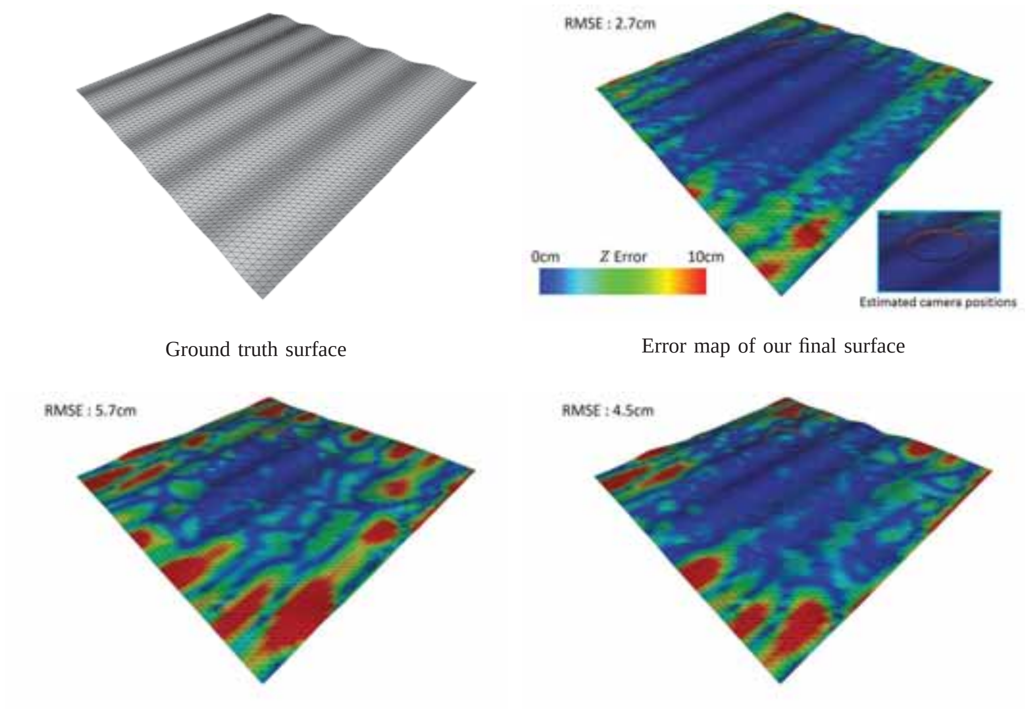
where

$$C_{PD}(\boldsymbol{z}) = \sum_{\boldsymbol{x}'_s} \left(\frac{1}{|n_s|} \sum_{n_s} (I_{n_s}[\boldsymbol{p}_{n_s}(\boldsymbol{x}_s(\boldsymbol{x}'_s, \boldsymbol{z}))] - \bar{I}_{n_s}[\boldsymbol{z}])^2 \right) \quad (5)$$

$$C_{PS} = \alpha_{PS} \sum_v \left(\boldsymbol{z}_v - \frac{1}{8} \sum_{v' \in \mathcal{N}(v)} z_{v'} \right)^2 \quad (6)$$

where C_{PD} and C_{PS} respectively indicate the data term and the smoothness term. Herein, $\boldsymbol{x}'_s = (x'_s, y'_s), (s = 1, 2, \cdots, S)$ denotes pre-defined sample points on the ground level plane. Then $\boldsymbol{x}_s = (x_s, y_s, z_s)^T$ represents the point on the surface settled by a sample point $\boldsymbol{x}'_s = (x'_s, y'_s)^T$ and the objective parameter vector $\boldsymbol{z}$. The surface point $\boldsymbol{x}_s$ can be computed using barycentric interpolation weights, similar to the case in III-C. We denote by I_{n_s} an image which observes the surface point $\boldsymbol{x}_s$, and by $|n_s|$ the number of the images. Then $\boldsymbol{p}_{n_s}$ represents the projected pixel position of $\boldsymbol{x}_s$ on the image I_{n_s}. We also denote by $\bar{I}_{n_s}[\boldsymbol{z}]$ the average of the pixel values over the images that observe $\boldsymbol{x}_s$. That is, (5) represents the sum of the variances of pixel values, where the image positions of the pixel values in a single variance are the projections from a single sample point on the surface to only the images that observe the sample point. The smoothness term (6) is the same as the initial surface estimation (3), except α_{PS} is another constant. We minimize the cost (4) by a Gauss-Newton algorithm with the initial surface parameters estimated in III-C.

The data term (5), representing the sum of the pixel value variances over the sample points on the surface [4], generally works well for indoor scenes. However, this straightforward approach generates an erroneous ground surface in the presence of outlier pixels which usually appear at the edge of the self-shadows of the rotating equipment under shiny whether conditions.



Ground truth surface

Error map of our final surface

Error map of the surface from SfM.

Error map of the surface from SfM + PMVS.

Fig. 4. Error maps of estimated surface from synthetic data. Top left: the ground truth. Top right: the error map of our final surface. Bottom left: the error map of the surface estimated using the 3D points using SfM. Bottom right: using SfM + PMVS. The RMSE shows root mean square error of all estimated vertex heights

Assuming that we have good initial surface parameters, we find outlier pixels of the surface point $\boldsymbol{x}_s$ by taking the median value $\mathrm{MED}(I_{n_s})$ and thresholding the absolute difference $|I_{n_s}[\boldsymbol{p}_{n_s}(\boldsymbol{x}_s(\boldsymbol{x}'_s, \boldsymbol{z}))] - \mathrm{MED}(I_{n_s})|$. This outlier detection process is done at the beginning of each iteration process in the optimization, followed by a single iteration process for minimizing the same cost (4), except that $|n_s|$ denotes the number of inlier pixels (images).

For enhancing the robustness and efficiency, we adopt a hierarchical meshing technique, where we first roughly optimize the ground surface using a mesh with large squares (initial surface) and the level-of-detail of the mesh is increased in stages.

We set the sample points $\boldsymbol{x}'_s$ so that their projected points cover as many pixels as possible on the image for estimating the ground surface in high precision. The sample point positions on the level planes are concentric circular, where the radial and angular intervals are defined using the given camera height, the angle between the camera optical axis and the level plane, and the physical pixel size. Here, we omit the details but, in consequence, the sample points have high density in the area near to the camera while low density in the area far to the camera.

IV. Experimental Results

Our algorithms were implemented in C++-language without explicit multi-thread implementation. We used CHOLMOD library for solving linear systems. The algorithms were run on a Windows7 PC (Xeon E3-1225 3.2GHz, 16GB).

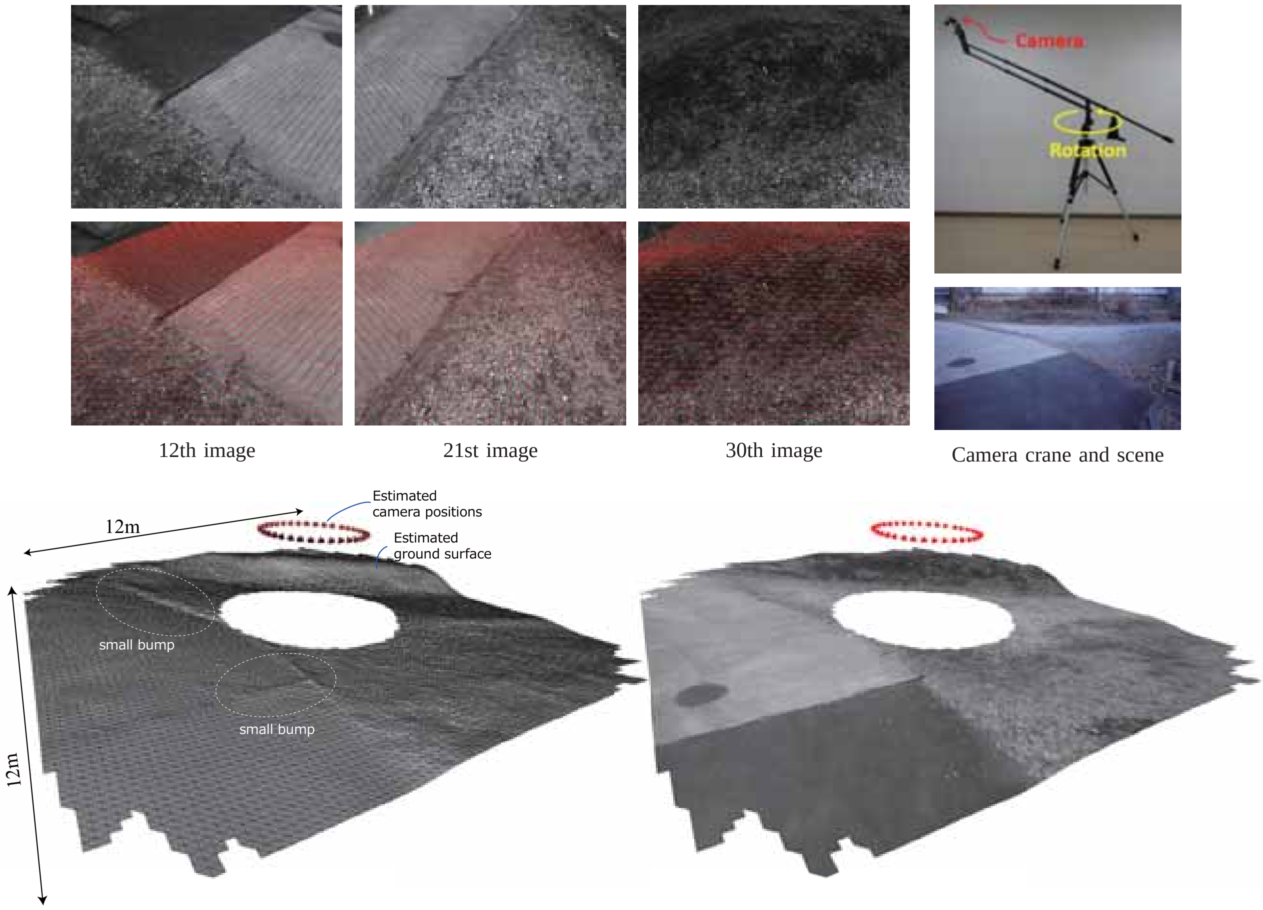
A. *Synthetic images*

We created synthetic images with a size of 640×480 pixels, horizontal field of view (FOV) of 130 degrees, position height of 2.0 meters, and rotation radius of 1 meter. The camera rotated 10 degrees per frame capturing 36 images of a textured ground surface created by a trigonometric function. We randomly added zero-mean Gaussian noise with standard deviation 3.0 graylevels to all images.

Fig. 3 shows one of the 36 input images, along with the initial surface estimated using the 3D points from SfM, the surfaces estimated at the three meshing levels and the ground truth surface, all of which are overlapped with the same input image. Fig. 4 shows the ground truth surface, and the error maps of the surface estimated by the proposed method and related methods, in the area of 16x16 meters, which are covered by the FOVs of the 36 images except around the rotation center on the ground (where only the smoothness term works). The error maps, *i.e.* textured surfaces whose vertexes are colored by error levels. The top right error map show the errors of the surface estimated by the proposed method, while the bottom error maps show the errors of the surface directly estimated by fitting the most detailed surface to the 4725 points from SfM (left), the 16107 points from SfM + PMVS (right). The result shows that our final surface is the best.

B. *Real images*

We used images with a size of 640×480 pixels, horizontal FOV of 60 degrees, position height of about 2.0 meters, and rotation radius of 1 meter. The camera was mounted



12th image 21st image 30th image Camera crane and scene

Fig. 5. Result of 3D ground surface generation in real scene (Scene 1). Top: Three input images and their overlaps with estimated surface mesh. Bottom: 3D views of the estimated surface and camera positions.

on a rotatable small camera crane and manually rotated for capturing images.

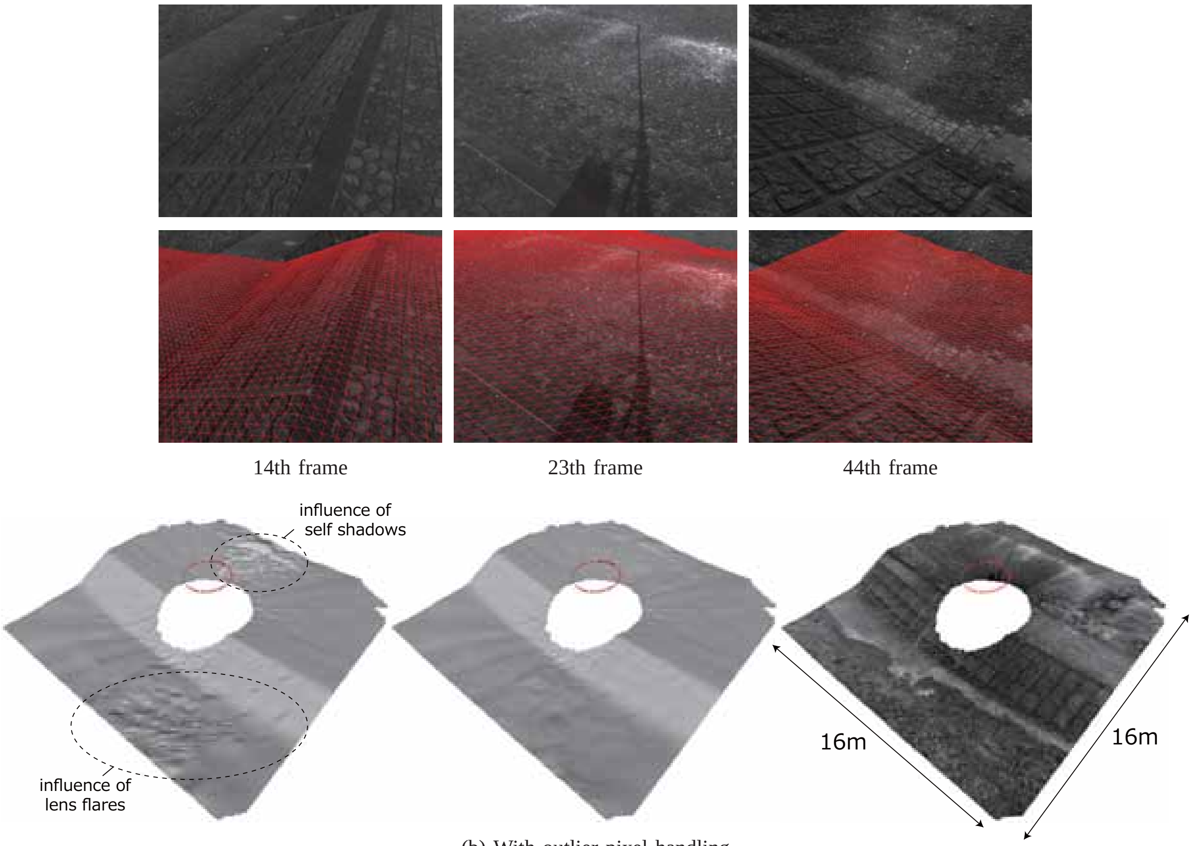
Fig. 5 shows the estimation results of Scene 1, where three of all 36 images that we took while rotating the camera about 360 degrees, their overlaps with the estimation result, and the whole view of the estimated ground surface overlapped with the estimated camera positions. We estimated the ground surface in the area of 12×12 meters, and we show in Fig. 5 only the surface's triangular patches each of which has at least one observed sample point. The surface views show our method can precisely recover even small bumps (less than 2cm in level) on the ground (see the marked parts).

Fig. 6 also shows the ground surface estimation results of Scene 2, where we estimated the surface in the area of 16×16 meters. We show three of all 50 images that we took while rotating the camera about 360 degrees. We can see not only the self-shadow of the camera crane on the ground surface as shown in the top-middle image, but also the lens flares in the opposite view directions as shown in the top-right image. These undesirable phenomena cannot be avoidable in outdoor environments. As shown in the bottom-left (a), where we simply minimize the pixel value variances using the data term (5) without outlier pixel handling, the estimated ground surface is erroneous. On the other hand, our method estimates a desiable ground surface by effectively handling the outlier pixels engendered by the self-shadows and lens flares.

V. CONCLUSIONS

We have proposed a method for estimating a regular-grid ground surface from images captured by a single rotating camera, assuming that the camera is mounted on a construction equipment. We first estimate an initial ground surface, represented by the regular grid mesh, by fitting 3D points from SfM, and then we refine the surface by minimizing the pixel value variances while handling outlier pixels. The validity of the proposed method has been demonstrated through experiments using synthetic and real images.

The total computation time under our current non-optimized implementation is too large for practical use (about 30 minutes in the case of Scene 1). In particular, the iterative Hessian computation for minimizing the variance of the pixel values takes a large amount of time. We will reduce the computational cost by keeping the barycentric interapolation weights in the memory and the parallel computation on the sample points. On the other hand, our approach can be applied to unsynchronized uncalibrated multiple cameras, which are advantageous to the 360 degree observation by a smaller angle of rotation, and be extended to the simultaneous estimation of the ground surface and camera positions.



14th frame

23th frame

44th frame

(a) Without outlier pixel handling.

(b) With outlier pixel handling (threshold=10 graylevel).

(c) Textured surface of (b).

Fig. 6. Result of 3D ground surface reconstruction in real scene (Scene 2). Top: Three input images and their overlaps with the estimated surface mesh. Bottom: Views of the surfaces estimated without/with outlier pixel handling.

Such acceleration and extensions will be studied during future research work.

References

[1] M. Bleyer, C. Rhemann, and C. Rother. PatchMatch Stereo – stereo matching with slanted support windows. In *British Machine Vision Conference (BMVC)*, pages 14.1–14.11, 2011.

[2] C. Brand, M. J. Schuster, H. Hirschmuller, and M. Suppa. Stereo-vision based obstacle mapping for indoor/outdoor SLAM. In *IEEE/RSJ International Joint Conference on Intelligent Robots and Systems (IROS)*, pages 1846–1853, 2014.

[3] R. Collins. A space-sweep approach to true multi-image matching. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 358–363, 1996.

[4] P. Fua and Y. G. Leclerc. Object-centered surface reconstruction: Combining multi-image stereo and shading. *International Journal of Computer Vision*, 16(1):35–56, 1995.

[5] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010.

[6] A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *Asian Conference on Computer Vision*, volume Part I, pages 25–38, 2010.

[7] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision (Second Edition)*. Cambridge University Press, 2003.

[8] W. Jiang, S. Sugmioto, and M. Okutomi. Panoramic 3D reconstruction using stereo multiperspective panorama. *International Journal of Pattern Recognition and Artificial Intelligence*, 24(6):867–896, 2010.

[9] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Symposium on Geometry Processing*, pages 61–70, 2006.

[10] K. Konolige, M. Agrawal, R. C. Bolles, C. Cowan, M. Fischler, and B. Gerkey. Outdoor mapping and navigation using stereo vision. In *International Symposium on Experimental Robotics (ISER)*, pages 179–190, 2006.

[11] Y. Li, H.-Y. Shum, C. K. Tang, and R. Szeliski. Stereo reconstruction from multiperspective panoramas. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):45–62, 2004.

[12] M. I. A. Lourakis and A. A. Argyros. Sba: A software package for generic sparse bundle adjustmen. *ACM Transactions on Mathematical Software*, 36(2):1–30, 2009.

[13] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2(60):91–110, 2004.

[14] T. K. Marks, A. Howard, M. Bajracharya, G. W. Cottrell, and L. Matthies. Gamma-SLAM: Using stereo vision and variance grid maps for SLAM in unstructured environments. In *IEEE International Conference on Robotics and Automation*, pages 3717–3724, 2008.

[15] D. Martinec, M. Branislav, and T. Pajdla. 3D metric reconstruction from uncalibrated omnidirectional images. In *Asian Conference on Computer Vision*, pages 545–550, 2004.

[16] K. Motooka, S. Sugimoto, M. Okutomi, and T. Shima. Robust ground surface map generation using vehicle-mounted stereo camera. In *IROS*, pages 2741–2748, 2014.

[17] P. Peer and F. Solina. Panoramic depth imaging: single standard camera approach. *International Journal of Computer Vision*, 47(1-3):149–160, 2002.

[18] M. Sch'onbein and A. Geiger. Omnidirectional 3D reconstruction in augmented Manhattan worlds. In *IEEE/RSJ International Joint Conference on Intelligent Robots and Systems (IROS)*, pages 716–723, 2014.

[19] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. *ACM SIGGRAPH*, 25(3):835–846, 2006.

[20] S. Sugimoto and M. Okutomi. Direct ground surface reconstruction from stereo images. *IPSJ Transactions on Computer Vision and Applications*, 5:60–64, 2013.

[21] C. Wu. Towards linear-time incremental structure from motion. In *International Conference on 3D Vision*, pages 127–134, 2013.

Towards Characterizing the Behavior of LiDARs in Snowy Conditions

Sébastien Michaud[1], Jean-François Lalonde[2], and Philippe Giguère[1]

[1]Computer Science and Software Engineering, Laval University
[2]Electrical and Computer Engineering, Laval University

Abstract— **Autonomous driving vehicles must be able to handle difficult weather conditions in order to gain acceptance. For example, challenging situations such as falling snow could significantly affect the performance of vision or LiDAR-based perception systems. In this paper, we are interested in characterizing the behavior of LiDARs in snowy conditions, as there seems to be little information publicly available. In particular, we present a characterization of the behavior of 4 commonly-used LiDARs (Velodyne HDL-32E, SICK LMS151, SICK LMS200 and Hokuyo UTM-30LX-EW) during the falling snow condition. Data was collected from the 4 sensors simultaneously during 6 snowfalls. Statistical analysis of these datasets indicates that these sensors can be modeled in a probabilistic manner, allowing the use of a Bayesian framework to improve robustness. Moreover, we were able to observe the temporal evolution of the impact of the falling snow during these snowstorms, and characterize the sensitivity of each device. Finally, we concluded that the falling snow had little impact beyond a range of 10 m.**

I. Introduction

The robustness of autonomous vehicles has increased prodigiously in the recent years. While long-range autonomous driving on the highway has been around for decades already [1], advances in mapping, 3D data processing and computer vision have enabled cars to drive autonomously for thousands of miles in unconstrained, city environments [2]. While this surely is an impressive feat, one quickly notes that most of these miles have been logged in California weather, which provides optimal operating conditions for sensors such as LiDARs. In order for these systems to gain acceptance worldwide, it is crucial that they could be operated in more challenging weather conditions, such as rain, fog and snow.

As we strive to make autonomous vehicles more adaptable to varying weather conditions, it is important to understand how sensors will behave in such conditions. Of particular interest, snowy conditions may cause challenging situations for sensors such as LiDARs. Indeed, the laser beams emitted may illuminate the snowflakes themselves, thus providing echoes that do not correspond to real obstacles. Consider fig. 1 for example. The same scene appears drastically different depending on whether it was captured on a clear or snowy day. While programmable lighting may help circumvent this problem [3], current LiDARs may fail under such circumstances.

In this paper, our main contribution is to provide a characterization of the behavior of four well-known LiDARs in snowy conditions. Through an extensive empirical study performed on a novel dataset captured under varying degrees of snowfall, we evaluate how much these LiDARs are sensitive—or not—to falling snow. We show that recent advances in sensor design have increased their robustness even to significant snowfall.

Fig. 1. Driving in bad weather. While autonomous vehicles have attained a great level of performance in nice weather (left), bad weather can cause significant challenges due to limited visibility (right). In this paper, we characterize the behavior in snowy conditions for oft-used sensors in autonomous cars: LiDARs. *Photo credit*: Nicole Duchesne (left), Gaetan Chevalier (right).

A. Related work

It is well-known that snow poses significant challenges to sensors mounted on-board outdoor mobile robots or other autonomous vehicles. For example, in their Antarctica exploration project, Moorehead et al. indicate that "in heavy [snow] storms, [...] the laser could not be used" [4]. Similarly, Yamauchi et al. relate that "LiDAR and stereo vision provide greater accuracy and resolution in clear weather but has difficulty with precipitation and obscurants" [5]. Common approaches for dealing with this problem include filtering 3-D data [4], or video [6], but this is often not enough to completely remove artifacts.

It is therefore important to characterize how sensors behave in such conditions. To this end, Sumi et al. [7] build a specifically designed simulated snow chamber, with white polystyrene beads flown with large fans to simulate snow. In our case, we use real world conditions to acquire a novel dataset of more than 6 days of snowfall.

Finally, we also mention the work of Servomaa et al. [8], who use LiDARs (and other sensors) to characterize snow storms for monitoring and measurement applications. In our case, we characterize the behavior of the sensors themselves for robotics applications.

Corresponding author: philippe.giguere@ift.ulaval.ca

II. Data acquisition

In this section, we first report on the relevant characteristics of the four sensors used in our dataset. We then describe the physical configuration of our test setup, then outline the weather conditions for each of the six collected snowfalls. Finally, we describe how the information from the LiDARs was preprocessed before analysis.

A. Sensors

Data acquisition was performed with the following four LiDARs: the SICK LMS200, SICK LMS151, Hokuyo UTM-30LX-EW, and the Velodyne HDL-32E. Relevant sensor information is provided in table I, but the reader is referred to the manufacturers documentation for additional information[1].

The first element that provides a qualitative overview of the sensor performance is the maximum acquisition distance. This value depends on several factors, such as lighting conditions and target remission. This value is provided directly for the HDL-32E and UTM-30LX-EW, but based on a target remission greater than $75\,\%$ for the LMS200 and LMS151. Another element to consider is the shape and area covered by the beam, which influences the probability of hitting a snowflake as well as the proportion of area it covers. A final significant element which changes from one sensor to the other is the number of echoes returned. The Hokuyo sensor can return up to three echoes, which means that it could locate two snowflakes before the beam reaches the ground. Regarding the LMS151, two echoes are evaluated by the hardware, but only one is returned. Finally, note that all LiDARs use class 1 laser with a wavelength of $905\,\mathrm{nm}$.

B. Setup configuration

Data acquisition was conducted at Pouliot Hall of Laval University, where sensors were placed close to the inner wall of a window facing N50°E. As shown in fig. 2, a wooden structure held the sensors side by side at approximately $14\,\mathrm{m}$ above the ground. The main scanning plane (i.e. XY plane in the sensor reference frame) formed a 30° angle with respect to the building wall, so as to increase the maximum distance as much as possible without having the laser beams hitting trees or a pedestrian walkway present near the building. In addition, an RGB camera was placed alongside the LiDARs to provide visual information about the scene. In this configuration, a slight opening of the window allowed to keep the instruments inside while scanning outside. To avoid direct interference between sensors, corrugated plastic layers were placed between them. Fig. 3 shows the scene as observed by the RGB camera placed with the sensors.

C. Dataset description

Data acquisition started February 12 and ended on March 2. A total of 10 episodes were collected for a total of more than 50 hours of data. Recordings were made using the Robot Operating System (ROS) [13], which provides standardized data types as well as time synchronization. Data was acquired at different times of day and in a wide variety of conditions, covering a wide range of snowflakes size, falling rate and wind speed. Table II provides an overview of our data[2]. Of these, six are used in the current study, as highlighted in this table. The dataset is publicly available upon request.

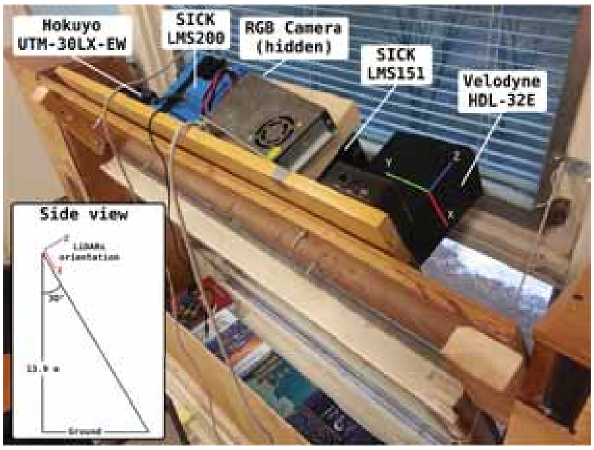


Fig. 2. The experimental setup. The 3D axis represent the orientation of the sensors and the bottom left panel represent the 2D geometry as seen from the right side of the picture.

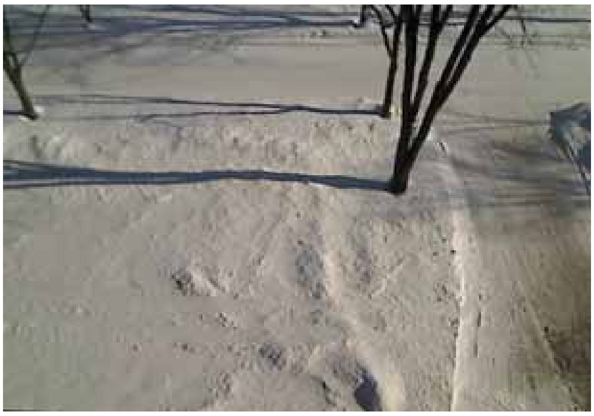


Fig. 3. View from the RGB camera.

D. Pre-selection of laser data

For each sensor, we selected a combination of angles and laser rings (for the Velodyne) or angles (for the others) that had a clear view of the snow-covered ground surface. Details for each sensor are provided in table III. The range of the ground in our scans was between $x = 15\,\mathrm{m}$ to $x = 22\,\mathrm{m}$, depending on the angle. To simplify the analysis, we considered as a snowflake echo any measurement which had a range reading of $x < 14.5\,\mathrm{m}$. As will be shown later in sec. IV, this approximation is valid as the vast majority of those events happened for $x < 10\,\mathrm{m}$.

[1] Available here: Velodyne [9], Hokuyo [10], LMS151 [11], LMS200 [12]

[2] Wind speed, daily precipitation and temperature were measured at Québec City Jean Lesage International Airport, located at a distance of $9\,\mathrm{km}$ from Laval University. Data is available here [14].

Sensor	Maximum distance	Spot area (at 30 meters)	Spot shape	Echoes
SICK LMS200	28 m	165 cm^2	Circle	1
SICK LMS151	50 m	22 cm^2	Circle	2
Hokuyo UTM-30LX-EW	30 m	196 cm^2	Ellipse	3
Velodyne HDL-32E	70 m	51 cm^2	Rectangle	1

TABLE I

OVERVIEW OF CHARACTERISTICS SPECIFIC TO EACH LIDAR.

Beginning time	Duration (HH:MM)	Snowflakes size	Falling rate	Wind speed range (km h^{-1})	Daily precipitation (cm)	Temperature (°C)
Feb 12, 9:47 am	09:21	Small	Variable	[2–13]	1.4	-14.1
Feb 14, 10:12 pm	04:12	Small	Very low	[5–13]	0.2	-21.4
Feb 19, 8:38 am	10:02	Big/small	High	[3–28]	4.5	-10.9
Mar 2, 1:06 pm	01:27	Big/small	Variable	[22–36]	1.6	-9.1
Mar 3, 10:33 pm	02:17	Big	Medium	[7–9]	5.4	-13.3
Mar 4, 11:45 am	04:12	Big/medium	Low/none	[20–30]	2.0	-4.3
Mar 17, 10:08 am	06:08	Big/medium	Low/none	[1–31]	2.0	-5.8
Mar 21, 6:44 pm	07:42	Medium/big	High	[5–33]	8.6	-5.1
Mar 30, 1:06 pm	04:45	Medium/big	High	[4–8]	8.5	-3.0
Apr 2, 1:56 pm	01:51	Small/rain	High	[2–10]	1.2	-8.4

TABLE II

OVERVIEW OF OUR SNOW DATASET. DATES IN BOLD CORRESPOND TO THE SIX DAYS USED IN THE PRESENT STUDY.

Sensor	Acquisition frequency	Selected beams/angles	Selected rings	Window size
LMS200	9.375 Hz	55–115	N/A	106 s
LMS151	25 Hz	310–220	N/A	40 s
Hokuyo	20 Hz	440–590	N/A	100 s
Velodyne	10 Hz	-0.05–0.25 rad	17–31	40 s

TABLE III

DETAILS OF MEASUREMENT SELECTION FOR THE ANALYSIS. THE WINDOW SIZE IS THE TEMPORAL WINDOW USED TO CALCULATE STATISTICS DURING THE TEMPORAL EVOLUTION OF A SNOWFALL.

III. TEMPORAL ANALYSIS

In this section, we analyze the temporal behavior of the four sensors for the duration of six complete snowstorms. In particular, we are interested in seeing how the fraction of echoes in snowflakes evolves over time, for all four sensors. First, we will discuss the highly dynamical nature of snowstorms. This will be exemplified by how consecutive scans can have significant quantitative and spatial differences in the distributions of the snowflakes echoes, which justify the use of averaging windows for our analysis. We will then present the actual temporal evolution of these statistics in the form of graphs for all four sensors, and finally briefly discuss the results for each sensor.

A. Extraction of temporal statistics

Snowstorms are highly dynamic processes, with large variation in snowfall rates over their durations. Moreover, the snow physical characteristics (size, shape or reflectance) might vary significantly during a storm, affected by ambient conditions such as humidity level and temperature. Also, wind gusts might pull snow back up in the air or drive it sideways, affecting its effective fall rate. Consequently, one expects during a snowstorm to see significant short, medium and long term variations in the fraction of LiDAR echoes corresponding to the falling snow.

Computing and reporting the temporal statistics for every scan would put too much emphasis on the very short-term statistics. Indeed, the inter-scan variation in the fraction of snowflake echoes can be significant. To better illustrate this point, we have overlaid four consecutive scans in the same plot for the LMS200 and for the first echo returned by the multi-echo Hokuyo sensor in fig. 4, for an intense snowing episode from the 02-19 dataset (see tab. II). In these figures, we can see strong variations in the fraction of snowflake echoes and their spatial distribution. One can readily see the fluctuation in these fractions as reported in the brackets of the legend in fig. 4.

To smooth out these fluctuations, statistics are extracted from a number of consecutive scans contained in a time window of around 1 minute (detailed values in tab. III). Fig. 5 shows this smoothed fraction of snowflake echoes compared to all returned laser measurements as a function of time, for the six snowiest days of our dataset. To allow for better visualization, only the LMS200 and the Hokuyo's first echo are plotted at their actual scale (1x): Others have been scaled up (from 30x to 200x), with their corresponding scaling factors reported in the legend. As will be shown below, some sensors were much more sensitive than others.

B. Detailed analysis, per sensor

1) SICK Sensors LMS200 and LMS151: Our first conclusion based on fig. 5 is that the most sensitive device was the older LMS200, first introduced in the mid-2000s. For the most intense snowstorms (fig. 5. b) 02-19, d) 03-17, e) 03-21 and f) 03-30), it peaked at around 15% of measurements triggered by the falling snowflakes, for averaging windows of 106 s. As an older-generation device, it probably uses less sophisticated algorithms and sensing, and was not directly

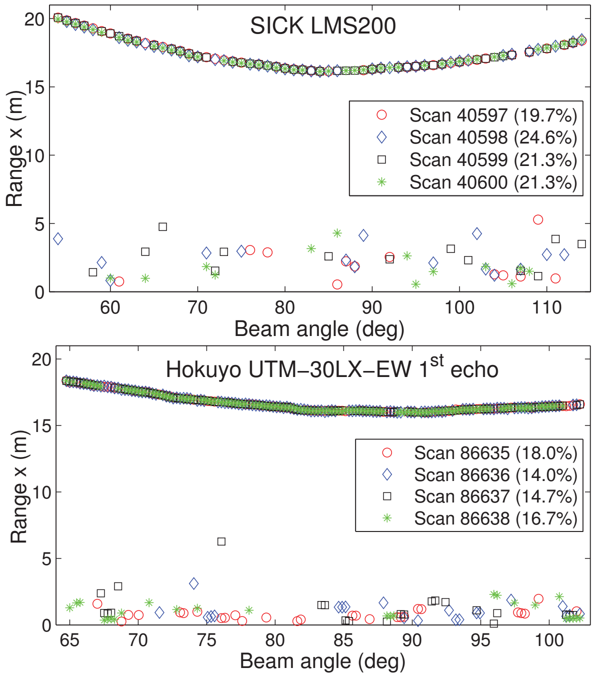


Fig. 4. Four overlaid consecutive scans for the LMS200 sensor (top), and the first echo scans for the Hokuyo sensor (bottom), taken from the 02-19 dataset. Each symbol corresponds to a particular scan. The curved line at the top corresponds to the snow surface on the ground. One can see the rapid variation of the snowflake echoes between scans, and how they are mostly limited to a range $x < 5\,\mathrm{m}$. The percentages (in brackets) are the proportion of those echoes in the snowflakes.

targeted for harsh outdoor environments. Indeed, its technical description [12] indicates that "raindrops and snow-flakes are cut out using pixel-oriented evaluation", but this seems only applicable to obstacle detection (field computation), not the actual measurements. No further details are given. On the other hand, the more recent SICK LMS151 exhibits much less sensitivity to snowflakes: The reduction factor for the fraction of snowflakes echoes is in the order of 200-300, granting this device a much higher immunity to snowstorms. Indeed, the highest peak was around 0.1 % of echoes in snowflakes during the 02-19 dataset. This seems to support the claim, obtained in the documentation from the manufacturer, that this model is targeted for "all weather conditions" [15].

2) Hokuyo UTM-30LX-EW: For this sensor, we resorted to a slightly different approach for comparison, as the device has been designed to return multiple echoes. We thus extracted statistics for the two most relevant cases: the first and last echoes. Statistics for the first echo indicate how sensitive the device is, if one wishes to detect the presence or absence of falling snow. This information could be used, for example, to adapt the driving strategy of an autonomous vehicle or inform vision algorithms of the presence of particles in the air. Using the last echo increases the probability that obstacles, such as another vehicle or the snow-covered ground, will be detected. This information would be used for localization and navigation purposes.

For the first echo, we observed that the device behaved similarly to the LMS200. Indeed, the Hokuyo first echo (blue line) closely tracks the LMS200 curves (red dashed line) almost everywhere in fig. 5, with a few exceptions. When using the last echo, the sensor behaves like the LMS151, not surprisingly as this sensor performs a 2-echo analysis and filtering. The last echo of the Hokuyo tends to reject the falling snow, but not as well as the LMS151, as it peaked at around 0.5 % in some episodes. Nevertheless, this difference might not be sufficient to impact algorithms relying on laser data. Note that tab. IV shows similar correlations between these three sensors for averages taken over the complete 02-19 dataset.

LMS200	Hokuyo first echo	Hokuyo last echo	LMS151	Velodyne HDL-32E
2.67%	3.55%	0.0113%	0.00178%	0.0100%

TABLE IV

OVERALL AVERAGE SNOWFLAKE ECHOES FOR THE COMPLETE 02-19 DATASET, PER SENSOR. THESE AVERAGES ARE SIGNIFICANTLY LOWER THAN THE INSTANTANEOUS VALUES DISPLAYED IN FIG. 5, AS SNOW WAS NOT FALLING AT ALL TIMES DURING THAT PERIOD.

3) Velodyne HDL-32E: For all purposes, the behavior of the Velodyne was similar to the last echo of the Hokuyo sensor. This is seen both in the temporal behavior in fig. 5 and in the average value displayed in tab. IV.

IV. DISTRIBUTION OF SNOWFLAKE ECHOES AS A FUNCTION OF RANGE

In the previous section, we showed how the expected fraction of snowflake echoes varied temporally during snowstorms. In some sense, it provided for a *temporal* modeling of the interaction between a snowstorm and a given LiDAR. In this section, we evacuate the temporal aspect and instead focus on how the range x affects the probability for a snowflake to trigger a measurement. To this end, we will use histograms to estimate a probability density function of those events, and show that for the weather conditions and the sensors we tested, there seems to be an upper bound on the range x beyond which falling snowflakes no longer trigger a measurement: in other words, snowflakes become invisible to the sensor past a certain range.

A. Modeling the impact of range on snowflake detection

When modeling a range sensor, one has to obtain the probability distribution of certain events (e.g. snowflakes) as a function of this range. Over the years, many researchers have proposed probabilistic models for sensors, notably [16]. In the previous section, we have estimated the probability for a given sensor S that a snowflake would generate an echo $E_{\text{snowflake}}$ given the weather condition W, or $P_S(E_{\text{snowflake}}|W)$. In this section, we take a closer look at which range x such events would be generated, that is $P_S(E_{\text{snowflake}}|x, W)$. Having such a formulation would allow for a more statistically-sound treatment of the information, such as within a Bayesian probabilistic framework. To this effect, we use histograms as approximations for the previous

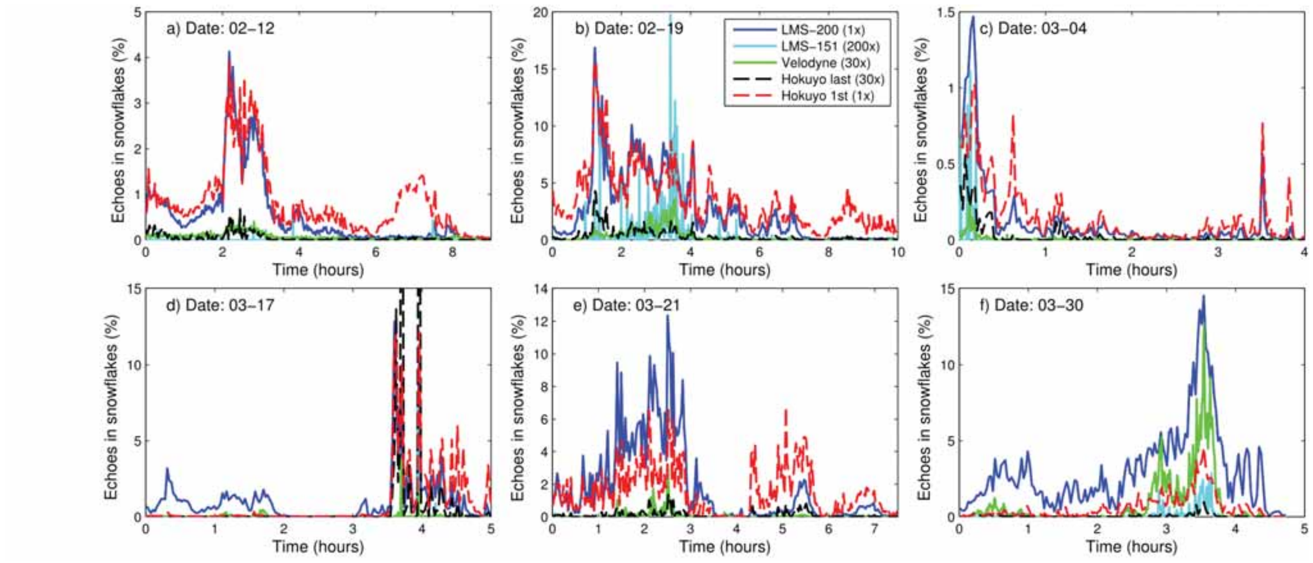


Fig. 5. Temporal evolution of the percentage of echoes coming from the falling snow (range x <5 m) during the 6 most intense episodes, for all 4 sensors. The data is smoothed by taking statistics for small time windows. Except for the LMS200 and Hokuyo first echo, all other sensors statistics have been scaled up (factor in bracket of legend b) for ease of visual comparison. Time is in hour, starting from the beginning of the data capture sequence.

distribution. In fig. 7, we have plotted these histograms for each of the four sensors. For ease of comparison, they have all been normalized by their total area in the interval $0 < x < 14\,\mathrm{m}$, as the total count varies widely between the sensors. The numbers in brackets in the legend indicate the fraction of echoes generated by snowflakes compared to the total number of data points, for a given dataset.

The general shape of these histograms is close to a log-normal distribution, with the exception of the LMS200 for a number of dates (02-12 through 03-17), which seems to follow a sum of two log-normal distributions. We attribute this log-normal shape to the interaction between two different phenomena, illustrated in a cartoon-type model in fig. 6. At short ranges $x < 3\,\mathrm{m}$, the building acts as a shield and decreases the probability of having a snowflake in the path of the laser. We recognize that this phenomenon would be most likely absent on an autonomous vehicle, thereby increasing the probability of having echoes in snowflakes at close range. However, we believe that this difference is not problematic, as close obstacles would be easily detected from *i)* the overwhelming number of LiDAR echoes on this obstacle *ii)* other sensing modalities such as vision or radar. Furthermore, if the LiDAR is to be mounted on a rooftop, one can safely ignore echoes in the first $2\,\mathrm{m}$, either in software or directly through the sensor itself (via its configuration). The other phenomenon, illustrated as the red dashed line in fig. 6, is the probability of optical detection of a snowflake by the sensor as a function of the range x. We argue that this shape is due to the rapidly decreasing light intensity of the echoes in snowflakes, as a function of x. Combining these two phenomenon yields a log-normal shaped curve (black line in fig. 6). Overall, this seems to indicate that a simple probabilistic model $P_S(E_{\text{snowflake}}|x, W)$ can be derived for these sensors.

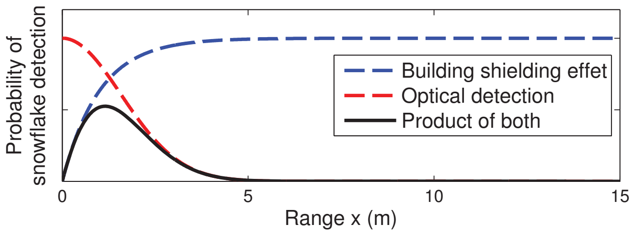


Fig. 6. Cartoon representation of the interaction between the probability of detecting a snowflake (in red) and the diminution of snowflakes due to the shielding effect of the building (in blue). The black line is the product of the two, and bear a close resemblance to the actual histograms extracted from our dataset.

B. Sensor results

As can be seen from the histograms in fig. 7, most sensors exhibit the log-normal or sum-of-log-normal distributions discussed above. We note that for certain days, the distributions are shifted to the right (greater range x). In particular, for the 03-21 and the 03-30 distributions, this shift is substantial (on the order of $1\,\mathrm{m}$). We suspect that for these days, the snowflakes were significantly larger, thus allowing for a stronger optical echo and extended range of detection.

For all sensors, we can also conclude that beyond the range $x > 10\,\mathrm{m}$, snowflakes are no longer detected, i.e. they become invisible. A small notable exception would be for the Velodyne, for which snowflakes were detected all the way to $x = 14\,\mathrm{m}$, albeit at a significantly reduced rate. Again, we do not think that this would significantly impair their use in conditions similar to our test setup.

V. Discussion and Conclusion

In this paper, we explored the impact of falling snow on the usability of 4 commonly deployed LiDARs in the context of autonomous driving vehicles. To this end, we collected

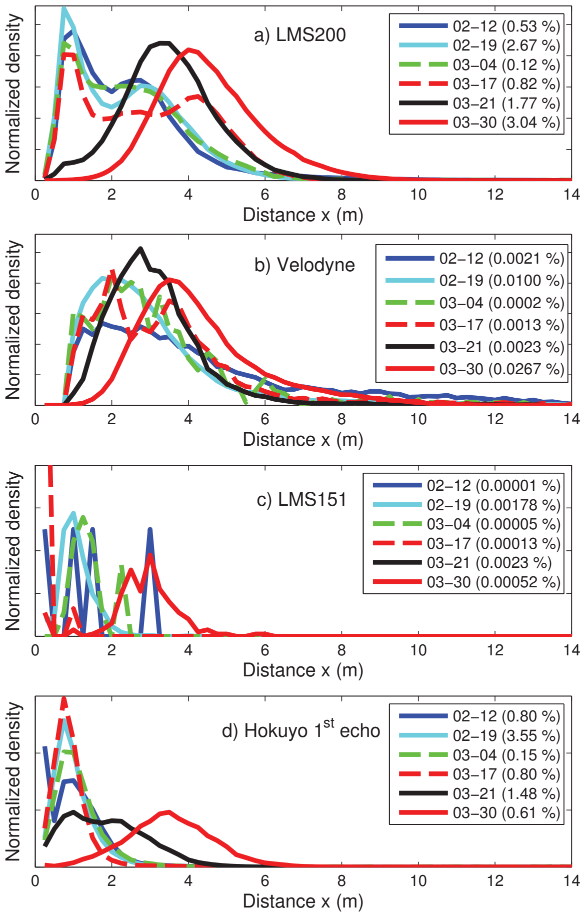


Fig. 7. Histograms of echoes in falling snow during important snowfall days, as a function of distance x reported by the sensor. Each histogram has been normalized by its area, for ease of comparison. The numbers in brackets are the fraction of data points in the complete dataset that correspond to snowflake echoes. Note that for the 03-21 dataset, the LMS151 was not working properly: thus no data is included for that day.

data during 6 snowstorms in the winter of 2015. Upon analysis, we found that the SICK LMS200 was the most sensitive LiDAR, having a peak average rate of up to 15 % of echoes coming from falling snow. Meanwhile, all 3 others never exceeded 1 %. We also presented a simple probabilistic model to take into account the effect of the range on snowflakes interference. Based on a histogram analysis, we concluded that for our experimental setup, this model can be approximated by a log-normal distribution. Most importantly, our data indicate that the impact of snowflakes on LiDAR beyond a range of $10\,\mathrm{m}$ is very limited.

A number of questions remain to explore. For example, as the LiDAR beam travels through the falling snow, its intensity will diminish. Since the maximum range of a LiDAR is heavily related to this beam intensity, we expect the maximum range to be affected during snowstorms. In our setup, we have not witnessed this issue, indicating that this effect probably happens beyond our maximum distance of $20\,\mathrm{m}$. Another aspect to be investigated is the relationship between the returned intensities and the surface type (ground or snowflakes). Also, because of the shielding effect of the building, very few snowflakes were present at close range; It might be the case that at closer range, a snowflake might be detected at more than one angle, effectively occluding small targets. Moreover, we have not investigated the impact on the measurement noise for the snowy ground surface in the presence of falling snow. Finally, it would be interesting to mount these LiDARs on a moving vehicle to investigate the impact of the vehicle velocity on the sensing behavior.

References

[1] D. Pomerleau and T. Jochem, “Rapidly adapting machine vision for automated vehicle steering,” *IEEE Expert: Special Issue on Intelligent System and their Applications*, vol. 11, no. 2, pp. 19–27, April 1996.

[2] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, *et al.*, “Autonomous driving in urban environments: Boss and the urban challenge,” *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[3] R. Tamburo, E. Nurvitadhi, A. Chugh, M. Chen, A. Rowe, T. Kanade, and S. G. Narasimhan, “Programmable automotive headlights,” in *European Conference on Computer Vision*. Springer, 2014, pp. 750–765.

[4] S. Moorehead, R. Simmons, D. Apostolopoulos, and W. L. Whittaker, “Autonomous navigation field results of a planetary analog robot in antarctica,” in *International Symposium on Artificial Intelligence, Robotics and Automation in Space*, June 1999.

[5] B. Yamauchi, “Fusing ultra-wideband radar and lidar for small ugv navigation in all-weather conditions,” in *SPIE Defense, Security, and Sensing*. International Society for Optics and Photonics, 2010, pp. 76 920O–76 920O.

[6] P. C. Barnum, S. Narasimhan, and T. Kanade, “Analysis of rain and snow in frequency space,” *International Journal of Computer Vision*, vol. 86, no. 2-3, pp. 256–274, 2010.

[7] Y. Sumi, B. Kim, Y. Matsumoto, E. Horiuchi, O. Matsumoto, and K. Ohba, “Outdoor environment simulators for vision-based safety sensors; artificial sunlight lampheads and simulated-snow chamber,” in *IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO)*, Nov 2013, pp. 125–130.

[8] H. Servomaa, K.-i. Muramoto, and T. Shiina, “Snowfall characteristics observed by weather radars, an optical lidar and a video camera,” *IEICE Transactions on Information and Systems*, vol. 85, no. 8, pp. 1314–1324, 2002.

[9] Velodyne, “Velodyne HDL-32E User’s Manual.”

[10] Hokuyo, “Scanning Laser Range Finder UTM-30LX-EW Specification.”

[11] SICK, “SICK LMS151 Datasheet.”

[12] ——, “Technical Documentation LMS200/211/221/291 Laser Measurement Systems,” Dec. 2006.

[13] Willow Garage. Robot operating system. [Online]. Available: http://www.ros.org/

[14] G. of Canada. Canadian weather. [Online]. Available: http://weather.gc.ca/canada_e.html

[15] SICK, “SICK LMS1xx Operating Instructions.”

[16] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.