

2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

IROS17

9th International workshop on Planning, Perception and Navigation for Intelligent Vehicles

Full Day Workshop
September 24th, 2017 Vancouver, Canada

<http://ppniv17.irccyn.ec-nantes.fr/>

Organizers

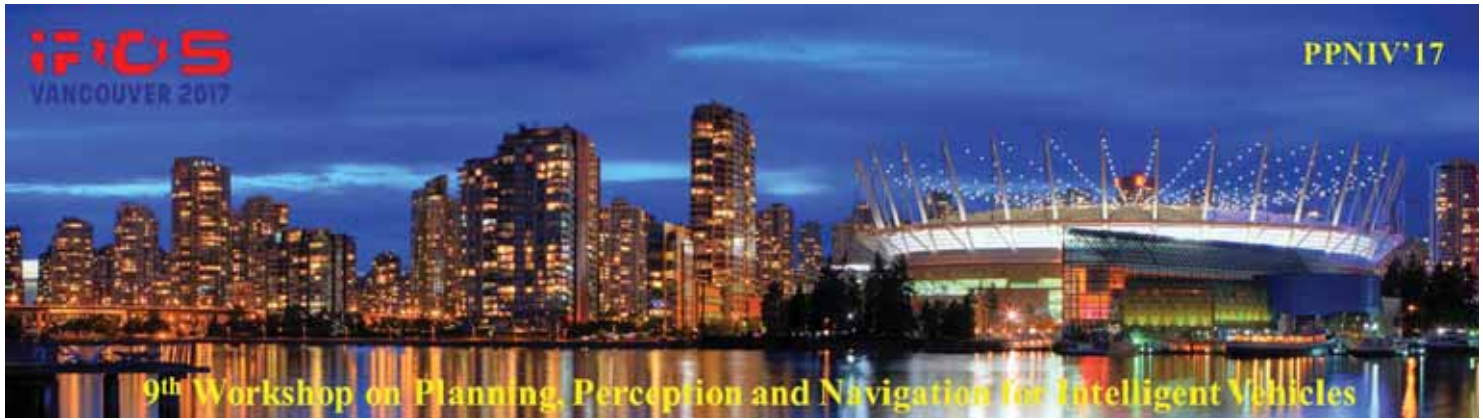
**Pr Christian Laugier (INRIA, France),
Pr Philippe Martinet (IRCCYN, France),
Pr Urbano Nunes (ISR, Portugal)
Pr Christoph Stiller (KIT, Germany)**

Contact

Professor Philippe Martinet
Ecole Centrale de Nantes,
LS2N-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@ec-nantes.fr
Home page: <http://www.irccyn.ec-nantes.fr/~martinet>



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Foreword

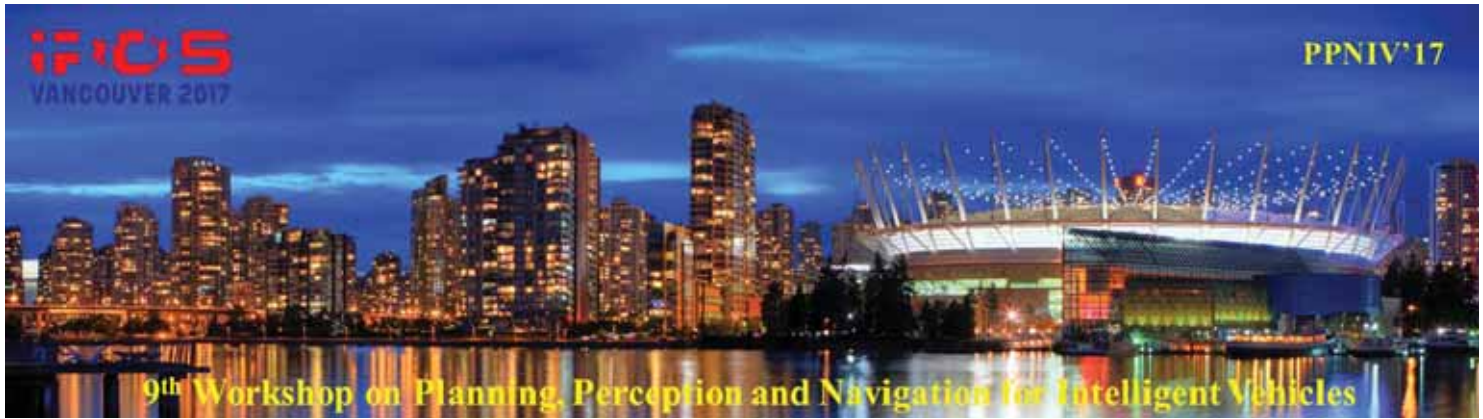
The purpose of this workshop is to discuss topics related to the challenging problems of autonomous navigation and of driving assistance in open and dynamic environments. Technologies related to application fields such as unmanned outdoor vehicles or intelligent road vehicles will be considered from 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, the sixth edition [PPNIV'14](#) was in Chicago (over 100 attendees) during IROS14, the seventh edition [PPNIV'15](#) was in Hamburg (over 150 attendees) during IROS15, and the eighth edition [PPNIV'16](#) was in Rio de Janeiro (over 100 attendees) during ITSC16.

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 Anchorage 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 16 selected papers (7 selected for oral presentation and 5 selected for interactive session. Five sessions have been organized:

- Session I: Control & Planning
- Session II: Segmentation and reconstruction
- Session III: Simulation & legal issues
- Session IV: Interactive session
- Session V: Perception

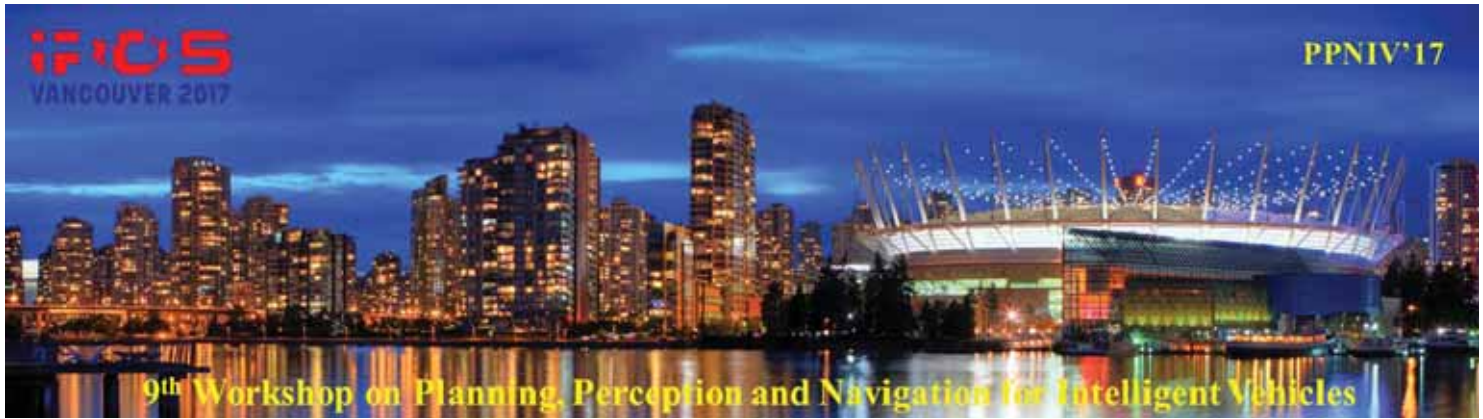


2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

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



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session I

Control & Planning

- **Title: Driving Like a Human: Imitation Learning for Path Planning using Convolutional Neural Networks**
Authors: Eike Rehder, Jannik Quehl and Christoph Stiller
- **Title: Autonomous Perpendicular And Parallel Parking Using Multi-Sensor Based Control**
Authors: David Perez-Morales, Olivier Kermorgant, Salvador Dominguez-Quijada and Philippe Martinet



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session I

Keynote speaker: **Marcelo H. Ang**
(NUS, Singapore)

Generalized Predictive Planning for Autonomous Vehicles

Abstract: We present a generalized framework for real-time predictive planning in space-time to improve autonomous driving performance in dynamic environments. Predictive planning refers to planning around predicted obstacle trajectories, where robot velocity profiles are solved in an integrated manner along with spatial paths, which is contrasted against traditional motion planning approaches which decouple the velocity and path planning problems. Autonomous vehicle deployments are still limited with respect to environmental complexity and operating speed, however the real world experimental results show that the proposed predictive planning framework can push the bounds of planning capabilities in both aspects. The planning methods are demonstrated onboard three classes of vehicles, a road car, buggy and scooter, in both unstructured pedestrian environments and on-road environments. Test scenarios include pedestrian crowd navigation, T-junction navigation, defensive driving, and overtaking.

Biography: Marcelo H. Ang, Jr. received the B.Sc. degrees (Cum Laude) in Mechanical Engineering and Industrial Management Engineering from the De La Salle University, Manila, Philippines, in 1981; the M.Sc. degree in Mechanical Engineering from the University of Hawaii at Manoa, Honolulu, Hawaii, in 1985; and the M.Sc. and Ph.D. degrees in Electrical Engineering from the University of Rochester, Rochester, New York, in 1986 and 1988, respectively. His work experience includes heading the Technical Training Division of Intel's Assembly and Test Facility in the Philippines, research positions at the East West Center in Hawaii and at the Massachusetts Institute of Technology, and a faculty position as an Assistant Professor of Electrical Engineering at the University of Rochester, New York. In 1989, Dr. Ang joined the Department of Mechanical Engineering of the National University of Singapore, where he is currently an Associate Professor, with a Joint Appointment at the Division of Engineering and Technology Management. He also is the Acting Director of the Advanced Robotics Centre. His research interests span the areas of robotics, mechatronics, and applications of intelligent systems methodologies. He teaches both at the graduate and undergraduate levels in the following areas: robotics; creativity and innovation, applied electronics and instrumentation; advanced computing; product design and realization. He is also active in consulting work in these areas. In addition to academic and research activities, he is actively involved in the Singapore Robotic Games as its founding chairman and the World Robot Olympiad as a member of the Advisory Council.



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Generalized Predictive Planning for Autonomous Vehicles

Scott Pendleton and
Marcelo H. Ang Jr.

Department of Mechanical Engineering
National University of Singapore



Massachusetts
Institute of
Technology

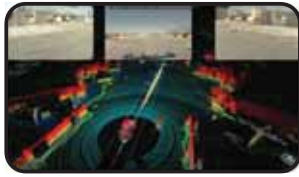
Why Autonomous Vehicles? (Singapore Perspectives)

- Reduce car ownership
 - Ride sharing, delivery, logistics
- Efficient use of resources
 - Car, road infrastructure, less parking spaces
- Public transportation
 - Last mile/first mile problem
 - Urban driving as opposed to highways
- Improved Productivity & Safety,
“greener”

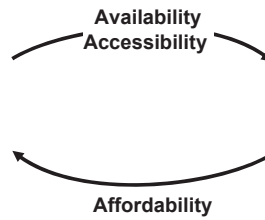
INTRODUCTION & MOTIVATION

Autonomous Mobility-on-Demand

- Vehicle sharing for first-and-last-mile transportation



Autonomy



Ride Sharing

- **Multiple vehicle classes:** Operational advantages for each vehicle class favor different environments. A combined multi-class service can extend the operational area. **True point-to-point service coverage is achievable.**
- **Disruptive technology:** Automation can enable new ways of thinking about automobiles and transportation systems in general. In particular, it can provide **affordable, convenient, on-demand mobility.**

Environments

- Road
- Pedestrian



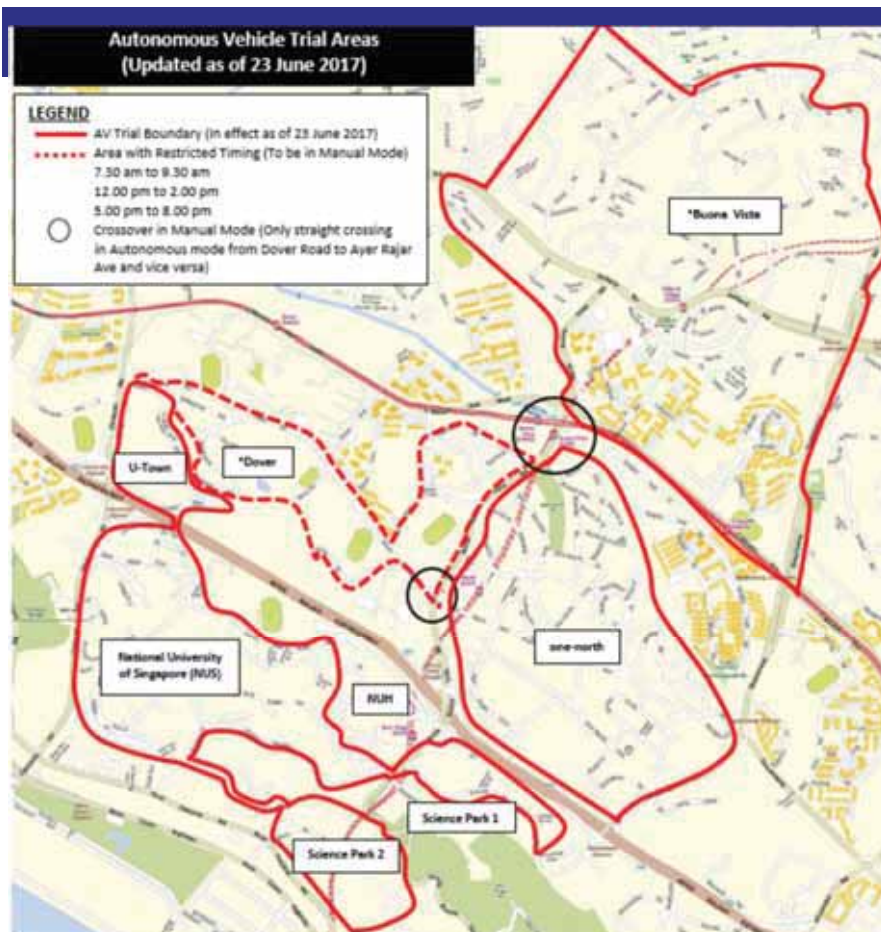
SMART=NUS Fleet



What we can confidently do?

- Reactive control with guaranteed safety (lowest layer – always on)
- Mapping and Localization
- Local planning
 - RRT* variant
 - POMDP
- Execution & Control
 - More accurate path following using kinematic constraints

Mobility on Demand using Multi-Class Autonomous Vehicles



- One North:
 - Jan 2015 – 6 km route
 - Sept 2016 – 12 km route
 - 23 June 2017 – 55 km -NUS & Science Pk

9 vehicles

- SMART-NUS: 1
- Nutonomy: 6
- Delphi : 1

A*STAR: 1

One North – Live Testing



One North – May 2017



Pedestrian crossing



Signalized Intersection



Complex intersection



Road construction



Road construction and jay walking

Public Deployment at the Chinese & Japanese Gardens (Oct 2014)



- Long Term Vehicle Testing
- To raise awareness
- To gain public acceptance

6 Days
360 km
500 Visitors
220 Trips
225 Surveys
98% "would ride again"



our autonomous mobility scooter



PREDICTIVE PLANNING FRAMEWORK

Our Planning Framework

- Interface planning modules with perception and control modules
- Incorporate acceleration constraints
- Establish replanning timing/retriggering
- Safety mechanism design for predictive planning

PREDICTIVE PLANNING FRAMEWORK

Planning Framework Overview



PREDICTIVE PLANNING FRAMEWORK

Planning Framework Overview

- Booking System & Mission Planner
 - Mobile phone access to webserver for handling mission requests as {Pickup Station, Dropoff Station}
 - Dijkstra search over directed graph of reference path segments
- Mapping/Localization
 - Vertical features extracted from 3D point cloud gathered from 2D LIDAR “rolling window” accumulation over time
- Obstacle Detection
 - SVM performed over spatio-temporal features of object clusters from 2D LIDAR

Planning Framework Overview

- Cost Map Generator
 - Obstacle avoidance cost set for grid locations in a 3D cost map layered by time dimension, up to a time horizon
- Goal Generator
 - Goal state set at constant distance ahead along route plan
- Steering Control
 - Pure-pursuit steering find constant radius arc target to forward waypoint
- Speed Control
 - Proportional Integral (PI) controller with switching mechanism for throttle vs. braking

Trajectory Planner

- Control and Path Guided RRT* (CPG-RRT*)
 - Use RG, path guided sample biasing, and min-jerk edge connection
- Same structure of RRT*, but redefine subfunctions:
 - “Nearest” is RG NN search
 - “SampleFree” uses biasing
 - “Line” uses an min-jerk profile interpolation along Dubins car paths
 - “Steer” and “CollisionFree” are built off the “Line” function

Algorithm 4 RRT*[73]

```

Input:  $x_{init}$ 
Output:  $G = (V, E)$ 
1:  $V \leftarrow x_{init}; E \leftarrow \emptyset;$ 
2: for  $i = 1, \dots, n$  do
3:    $x_{rand} \leftarrow \text{SampleFree};$ 
4:    $x_{nearest} \leftarrow \text{Nearest}(G = (V, E), x_{rand});$ 
5:    $x_{new} \leftarrow \text{Steer}(x_{nearest}, x_{rand});$ 
6:   if  $\text{CollisionFree}(x_{nearest}, x_{new})$  then
7:      $x_{near} \leftarrow \text{Near}(G, x_{new}, \eta);$ 
8:      $V \leftarrow V \cup \{x_{new}\};$ 
9:      $x_{min} \leftarrow x_{nearest}; c_{min} \leftarrow \text{Cost}(x_{nearest}) + c(\text{Line}(x_{nearest}, x_{new}));$ 
10:    for all  $x_{near} \in X_{near}$  do //Connect along minimum cost path
11:      if  $\text{CollisionFree}(x_{near}, x_{new})$ 
12:         $\wedge \text{Cost}(x_{near}) + c(\text{Line}(x_{near}, x_{new})) < c_{min}$  then
13:           $x_{min} \leftarrow x_{near}; c_{min} \leftarrow \text{Cost}(x_{near}) + c(\text{Line}(x_{near}, x_{new}));$ 
14:           $E \leftarrow E \cup \{(x_{min}, x_{new})\};$ 
15:    for all  $x_{near} \in X_{near}$  do //Rewire the tree
16:      if  $\text{CollisionFree}(x_{new}, x_{near})$ 
17:         $\wedge \text{Cost}(x_{new}) + c(\text{Line}(x_{new}, x_{near})) < \text{Cost}(x_{near})$  then
18:           $x_{parent} \leftarrow \text{Parent}(x_{near});$ 
19:           $E \leftarrow (E \setminus \{(x_{parent}, x_{near})\}) \cup \{(x_{new}, x_{near})\};$ 
return  $G = (V, E)$ 

```

Trajectory Planner: SampleFree

- Retain previous iteration knowledge by Φ_{i-1}
- Bias toward route plan by Φ_{pp}
- SampleGoal for greedy search
- RG Sample for efficient exploration

Algorithm 5 CPG-RRT* SampleFree

```

Input:  $s_g, \{M, \overrightarrow{min}, \overrightarrow{res}, R\}, \Phi_{i-1}, \Phi_{pp}, w_g, w_r, w_{ps}, w_{pp}$ 
Output:  $s$ 
1: //  $w_g, w_r, w_{ps}, w_{pp}$  should be normalized to add up to 1
2:  $w_s = \text{rand}[0, 1]$ ;
3: switch  $w_s$  do
4:   case  $< w_g$ 
5:     | assert( $s = \text{SampleGoal}(s_g)$ )
6:   case  $> w_g \wedge \leq w_g + w_r$ 
7:     | assert( $s = \text{SampleReachable}(M, \overrightarrow{min}, \overrightarrow{res}, R)$ )
8:   case  $> w_g + w_r \wedge \leq w_g + w_r + w_{ps}$ 
9:     | assert( $s = \text{SampleNearTrajectory}(\Phi_{i-1})$ )
10:  case  $> w_g + w_r + w_{ps}$ 
11:    | assert( $s = \text{SampleNearTrajectory}(\Phi_{pp})$ )
return  $s = [x, y, \theta, v, t]$ ;

```

Trajectory Planner: Line

- Controllable trajectory generation to enforce:
 - Minimum turning radius (Dubins curves)
 - Velocity bounds
 - Acceleration bounds
- Edges are min-jerk optimal for comfort
 - Minimizes $\int_{t_0}^{t_f} \ddot{p}(t)^2 dt$
 - Known to be 5th degree polynomial for position
- Trajectory defined over Dubins x Velocity x Time
 - Configuration space $\mathcal{X} = SE(2) \times \mathbb{R} \times \mathcal{T}$

Trajectory Planner: Line

- First, solve for Dubins curve in SE(2) space
- Then, solve for position, velocity, and acceleration w.r.t time by system of equations for boundary conditions:

$$p(t) = b_0 + b_1t + b_2t^2 + b_3t^3 + b_4t^4 + b_5t^5$$

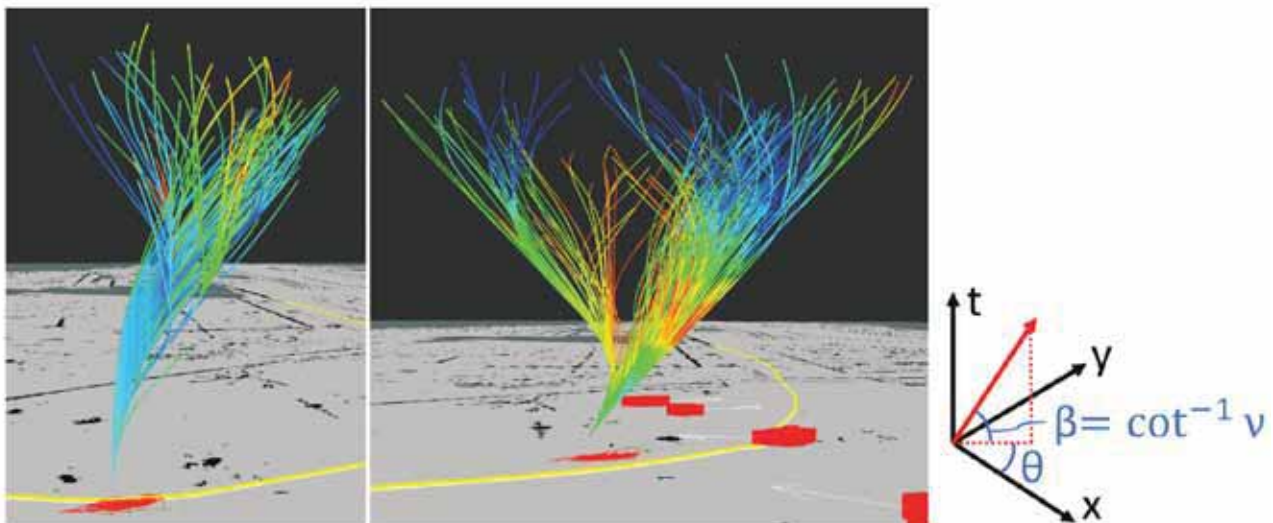
$$v(t) = b_1 + 2b_2t + 3b_3t^2 + 4b_4t^3 + 5b_5t^4$$

$$a(t) = 2b_2 + 6b_3t + 12b_4t^2 + 20b_5t^3$$

- **Known:** p_{init} , v_{init} , a_{init} , p_{final} , v_{final} . **set** $a_{final} = 0$
- Solve for constants $b_0 \dots b_5$

Trajectory Planner: Line

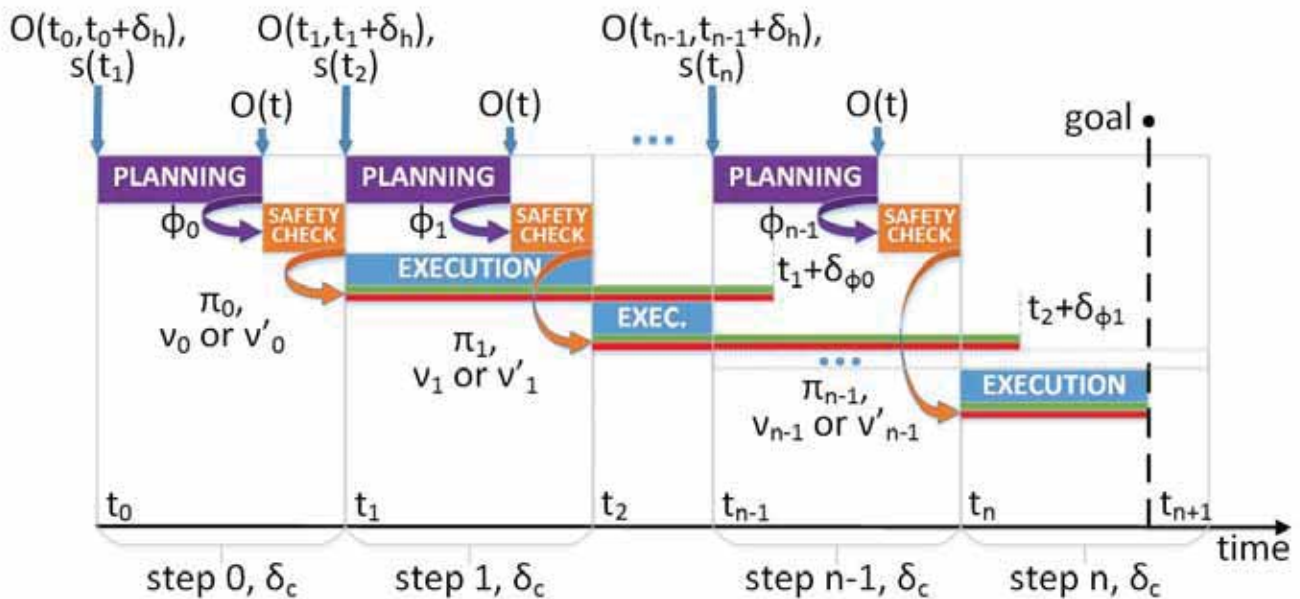
- Polynomial solutions found quickly
- Bounds checked over time interval at endpoints and roots



PREDICTIVE PLANNING FRAMEWORK

Replan Timing

- Each plans is generated while previous plan is executed



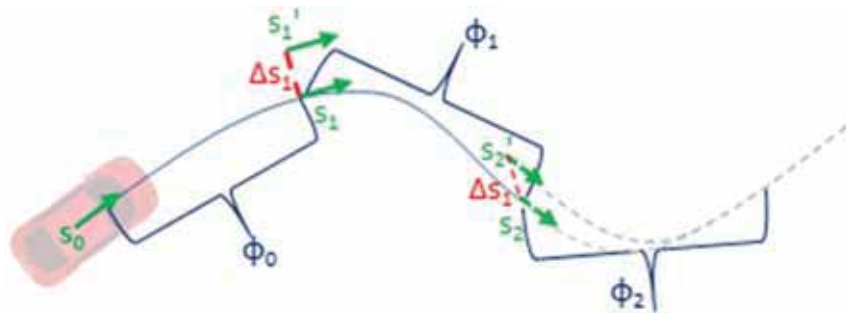
PREDICTIVE PLANNING FRAMEWORK

Safety Checking

- Each solution plan is rechecked against an updated observation before execution
- A new variant of braking Inevitable Collision State (ICS^b) is applied for passive safety:
 - A braking maneuver must exist from the commit state following the solution trajectory to satisfy dynamic minimum braking distance
 - Otherwise, velocity profile of solution is overridden by constant deceleration profile up to braking distance
- “Clear zone” applied to command stop when obstacles are very close

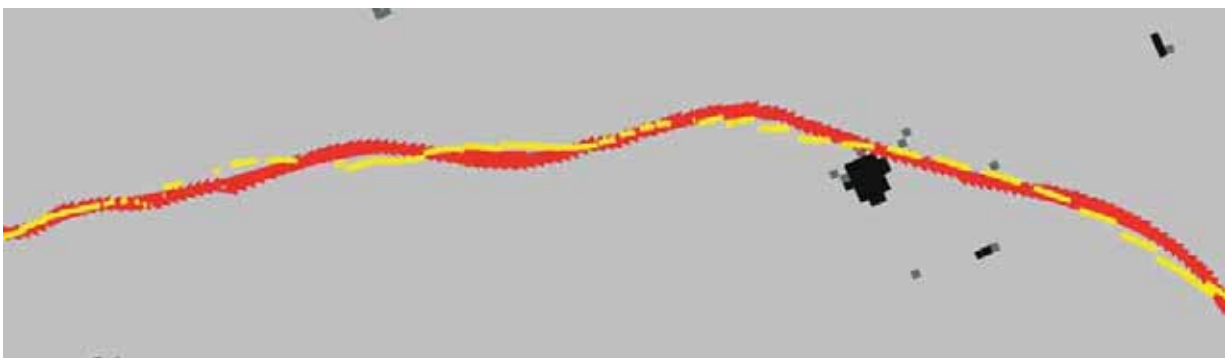
Control Interfacing

- Planner must know next commit state as root for plan tree
 - Control and/or localization error may affect true pose
 - s_1 is expected commit state at end of trajectory Φ_0 , but instead arrive at s_1'
 - Where to begin plan Φ_2 ? Introduce pose correction factor!
 - Start plan Φ_2 from state $s_2 + w \Delta s_1$ (we use $w = 0.5$)



Control Interfacing

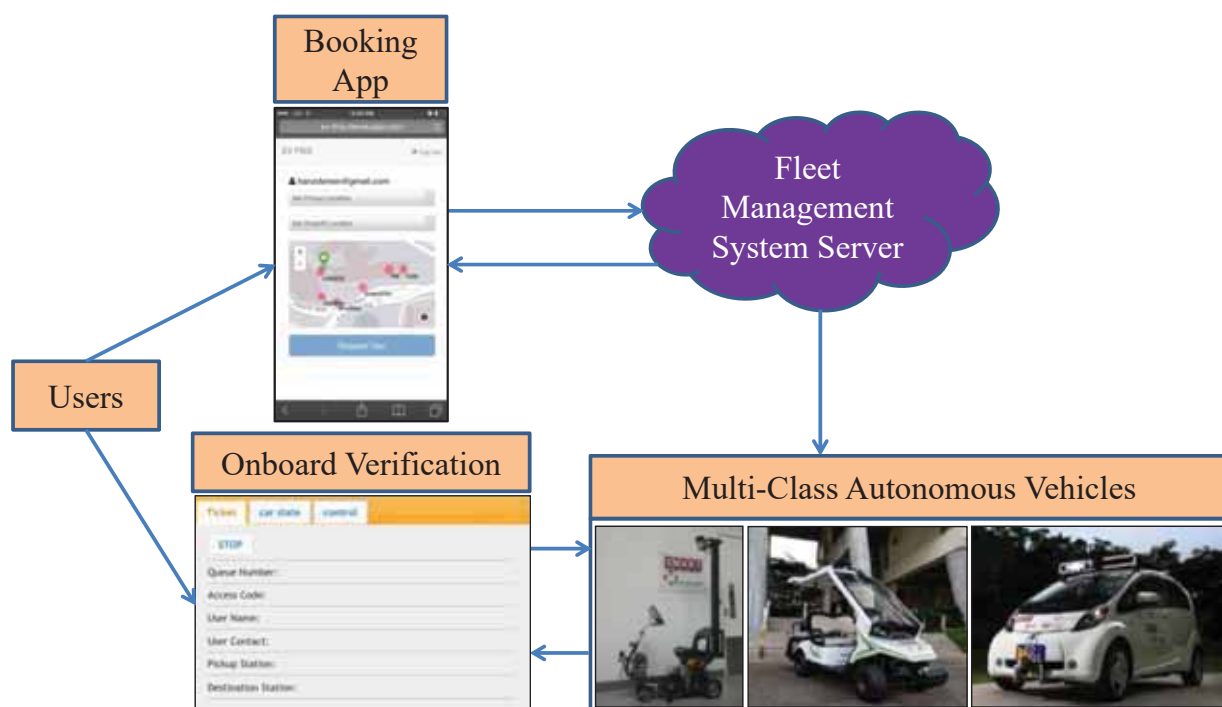
- Pose correction in practice:
 - Red is odometry trace (series of vectors)
 - Yellow is commit path
 - Overlap correlates with velocity undershoot, gap for overshoot



Summary: Planning Framework

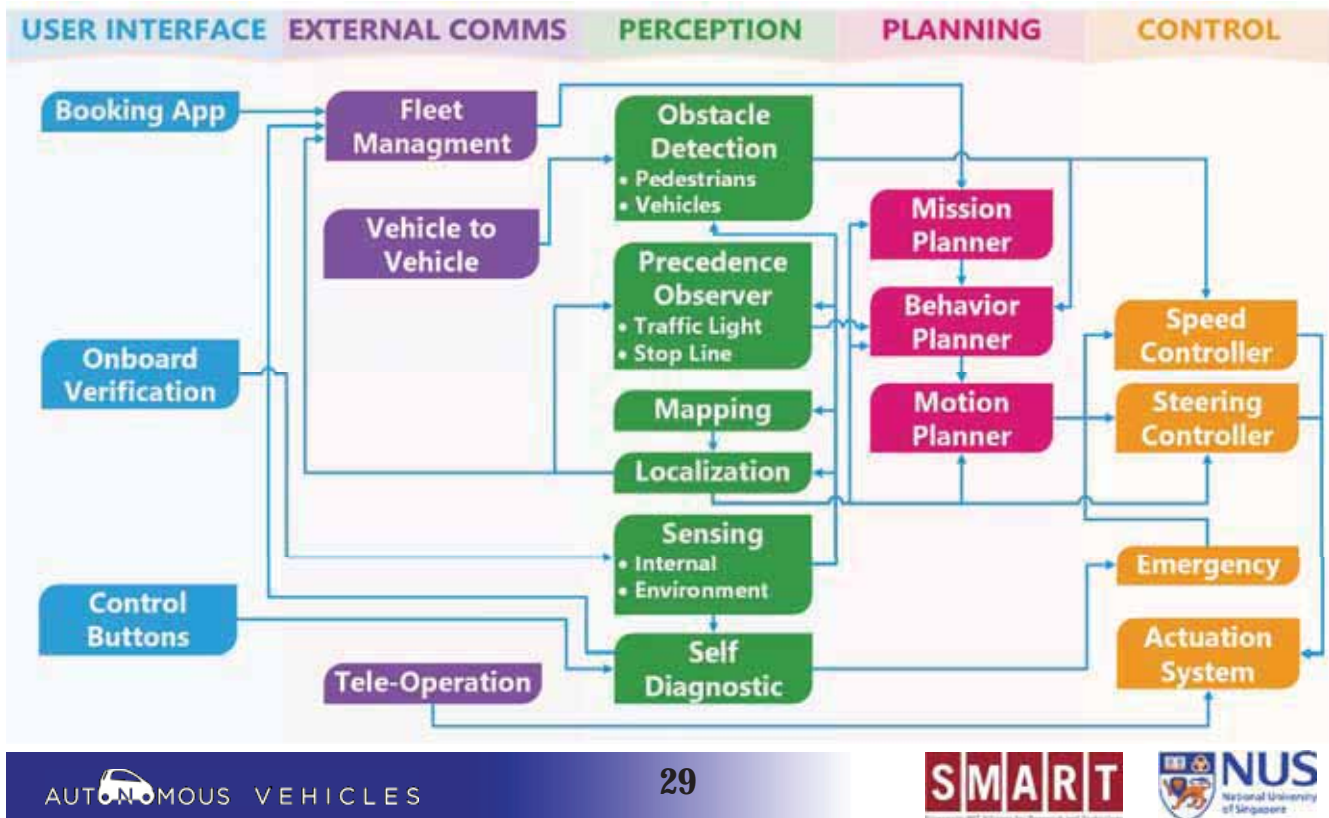
- Predictive planning framework
 - Real-time replanning in space-time
- Trajectory planning algorithm (CPG-RRT*)
 - Generates min-jerk controllable edge connections
 - Biased sampling for
 - Near previous solution trajectory
 - Near pure pursuit steering trajectory to route plan
 - Near goal
 - Reachable configuration space
- Passive safety assurances through adapted braking Inevitable Collision State Avoidance (ICS^b)

Software Overview



VEHICLE PLATFORM DEVELOPMENT

Software Overview



VEHICLE PLATFORM DEVELOPMENT

Hardware Overview

- Common Sensor Suite
 - IMU & wheel encoders for odometry
 - 1 2D LIDAR for Mapping & Localization (M&L) – fuse w/odom
 - ≥ 1 2D LIDAR for Obstacle Detection (OD)
- Similar Power Management & Off-the-shelf Computers
 - Ubuntu 14.04, ROS Indigo, i7 processor, 16GB RAM, SSD
- Differing Actuation Mechanisms to Control:
 - Steering
 - Braking/Throttle
 - Gear Selection (Forward/Reverse)

VEHICLE PLATFORM DEVELOPMENT

Hardware Overview

Start with a personal mobility scooter, then add...



VEHICLE PLATFORM DEVELOPMENT

Hardware Overview

Start with a golf car, then add...



VEHICLE PLATFORM DEVELOPMENT

Hardware Overview

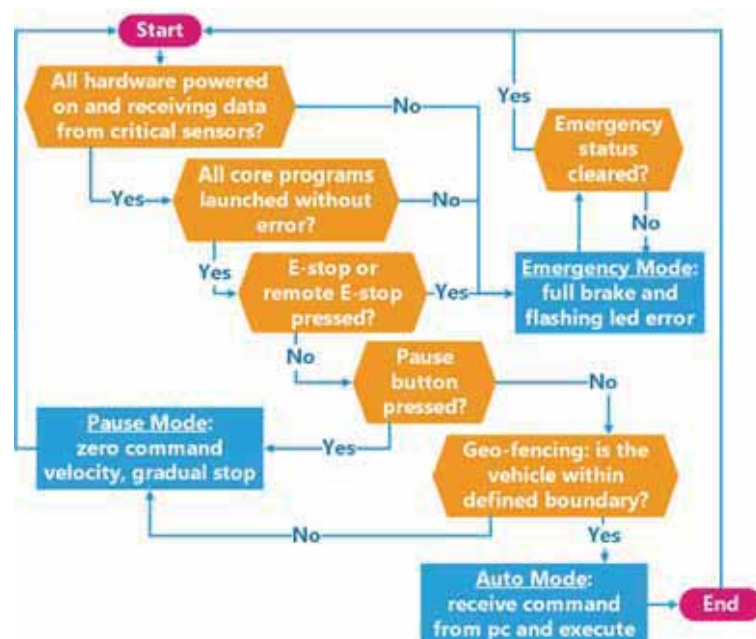
Start with a road car, then add...



VEHICLE PLATFORM DEVELOPMENT

Safety Overrides

- User Button Controls:
 - Pause
 - Auto
 - Manual
- E-stops, onboard and remote
- Visualizations onboard show perception data and planned path
- Audio cues for station arrival/departure

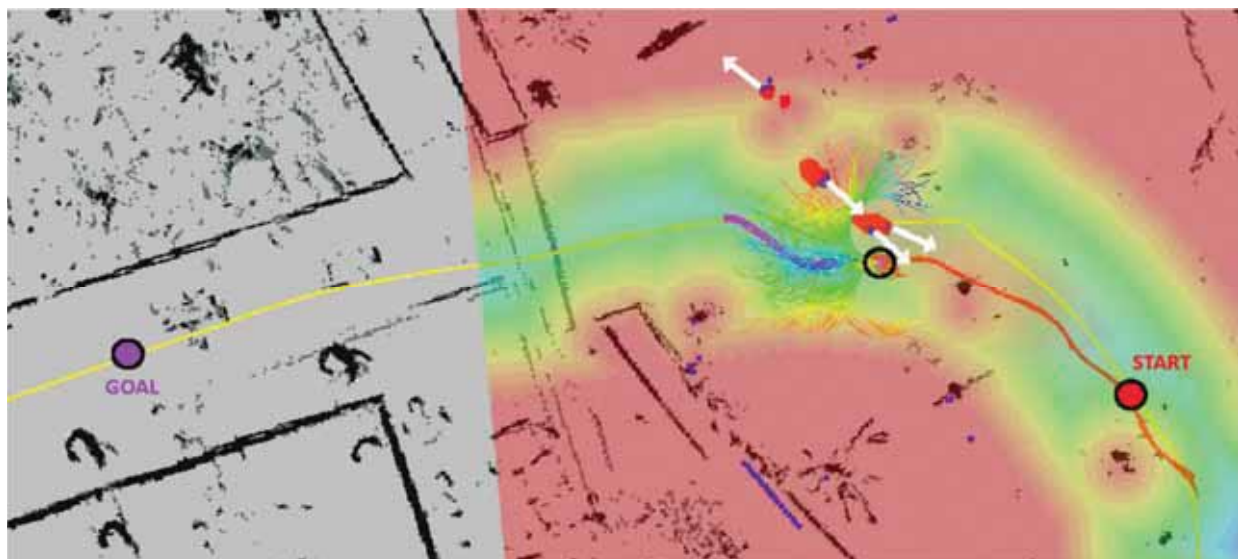


Experiment Setup

- Look for positive emergent behaviors
- Compare against baseline planning method:
 - Decoupled spatial path and velocity planning
 - Enlarge obstacle bounds forward based on velocity to treat environment as static
 - Trigger replanning only when at a stop due to blockage
- Test Scenarios:
 - Pedestrian navigation
 - T-junction
 - Defensive driving
 - Overtaking

Experiment Setup

- Planning visualization



Predictive Planning Video

- Video available on YouTube: search “FMAutonomy” channel



<https://youtu.be/eVVGZxp03Hc>

What have we achieved?

- Reactive Control – Guaranteed Safety as a Baseline
- Generalize predictive planning
 - Plans coupled spatial path and velocity
 - Demonstrated over varied vehicle types and environments in high-risk scenarios
- Reachability Guidance
 - Speed improvement by factor of 9-10
- Predictive Planning Framework
 - CPG-RRT* (biased sampling and min-jerk edges)
 - Modified ICS^b passive safety assurances

What's Next?

Towards Mapless Navigation

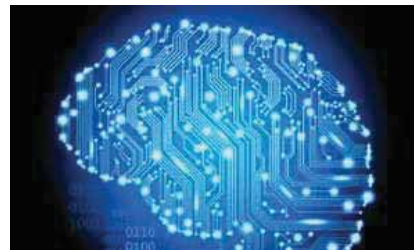


- You are “here” (blue circle)
- Go to #02-16
- Giving intelligence to robot
 - To read maps
 - Navigation to points in the map

What's Next?

Learning how to drive

- Cars and people around
- Moving directions
- Relative positions
- Speeds
- Intermediate Goal



- Steering
- Brake
- Throttle



Marcelo H ANG Jr
mpeangh@nus.edu.sg



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session I

Control & Planning

- **Title: Driving Like a Human: Imitation Learning for Path Planning using Convolutional Neural Networks**
Authors: Eike Rehder, Jannik Quehl and Christoph Stiller
- **Title: Autonomous Perpendicular And Parallel Parking Using Multi-Sensor Based Control**
Authors: David Perez-Morales, Olivier Kermorgant, Salvador Dominguez-Quijada and Philippe Martinet



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Driving Like a Human: Imitation Learning for Path Planning using Convolutional Neural Networks

Eike Rehder, Jannik Quehl and Christoph Stiller¹

Abstract—Human-like path planning is still a challenging task for automated vehicles. Imitation learning can teach these vehicles to learn planning from human demonstration. In this work, we propose to formulate the planning stage as a convolutional neural network (CNN). Thus, we can employ well established CNN techniques to learn planning from imitation. With the proposed method, we train a network for planning in complex traffic situations from both simulated and real world data. The resulting planning network exhibits human-like path generation.

I. INTRODUCTION

Motion planning is an essential component of autonomous agents navigating through the world. Especially in the context of autonomous robots and vehicles that operate in large open spaces, motion planning is a crucial task.

In research, planning an agent’s actions has been understood as an optimization problem in which the optimal actions given a cost function should be selected. Approaches only differ in the shape of both actions and cost functions.

In terms of actions, one may consider discrete or continuous actions and outcomes. For discrete state spaces, planning will take the form of graph optimization, e.g. in grid maps or state lattices [1], [2]. If the state and action space is continuous, nonlinear optimization has been proposed [3]. Also, one can think of a combination of both [4]. For a detailed study on models and their solutions, see [5].

In the end, however, all concepts for planning have in common that the function to be optimized has to be pre-specified. In most works, this is done by the careful design by the researcher, e.g. for shortest paths in presence of obstacles or minimum jerk in automated driving. This imposes demands on the perception of autonomous systems as they have to be capable of inferring the boundary conditions of the optimizer at hand.

Thus, some works have aimed to deduct the planning cost function from given sensor data [6], [7]. In this concept, an agent learns to plan its motion from imitation of observed behavior of others, thus called *Imitation Learning* (IL).

In the area of machine learning, recent advances in Deep Learning and Convolutional Neural Networks (CNN) have exceeded all expectations in a broad variety of tasks such as

image classification and segmentation, optical flow computation, etc. [8], [9], [10], [11].

With this, it is no wonder that deep learning has also found its way into planning. While some researches aim to construct an end-to-end pipeline that can create control outputs directly from sensor readings, this approach can only be reactive but not strategic [12]. Recently, long term planning has been solved using *Value Iteration Networks* (VIN) for Markov Decision Processes [13], [14]. While these works achieve great results, to our understanding, none was trained directly on observed paths. In the work of Shankar *et al.* the planning cost function was static for one network. Thus, the network could only plan for scenes it had been trained on.

In this work, we propose to model motion planning of an intelligent vehicle as Value Iteration Network. We show how the network can be trained from previously observed paths. As a training input, we rely exclusively on observed paths and not on any kind of manually annotated data. We demonstrate the performance of the network by training a cost function from aerial images to resemble human driving behavior.

II. IMITATION LEARNING FOR PLANNING

In this section, we demonstrate how to model the entire planning task as a connection of recursive neural networks. This augmentation makes the planning tasks fully differentiable and thus allows for full back propagation. This way, we can train an underlying cost function network from input data. In this work, we follow the intuition of both Tamar *et al.* as well as Shankar *et al.* [13], [14] for full Imitation Learning of path planning.

In this work, planning is executed in a state grid. For simplicity, the state grid is just a equidistant discretization of the state space. However, one could easily incorporate other state variables, such as orientation, by extending the state grid by additional dimensions. All other steps then scale accordingly.

A. State Transitions within Grids

Planning within a state grid is commonly modeled as a graph where the edges of the graph are represented by neighboring cells in the grid. However, in this work, we aim to model these transitions as components of a neural network for differentiability.

For this, we make use of the property of the Dirac Delta Function. When a Dirac Delta function shifted by a , $\delta(t-a)$,

*The research leading to these results has received funding from the German collaborative research center “SPP 1835 - Cooperative Interacting Automobiles” (CoInCar) granted by the German Research Foundation (DFG).

¹Eike Rehder, Jannik Quehl and Christoph Stiller are with the Institute of Measurement and Control Systems, Karlsruhe Institute of Technology, Karlsruhe, Germany {eike.rehder, jannik.quehl, stiller}@kit.edu

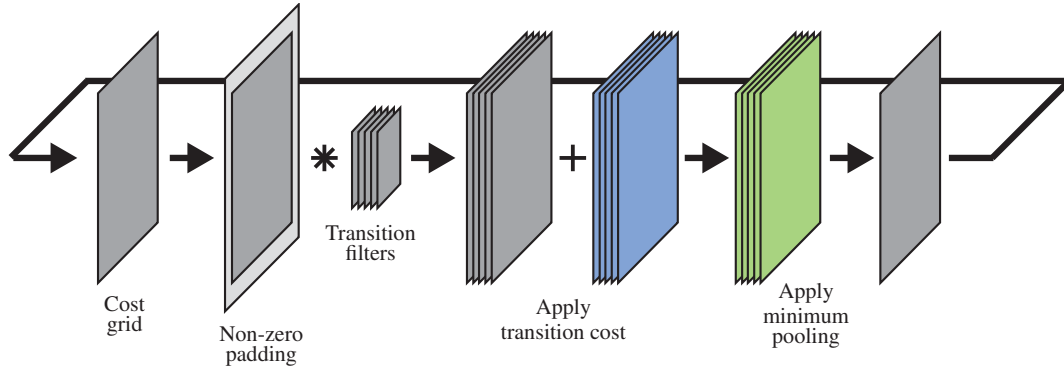


Fig. 1: Value Iteration Module as Recurrent Neural Network. The two layers that define planning behavior are the transition cost map (blue) and the cost per state and transition (green).

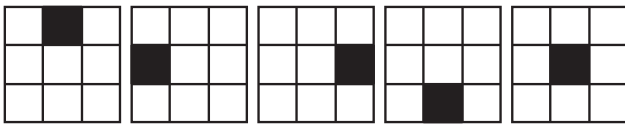


Fig. 2: Transition filters masks for four-connected neighborhood and idle (last). Black cells represent ones, white cells zeros.

is convolved with a function $g(t)$ the result is the function g shifted by a

$$\delta(t - a) * g(t) = g(t - a). \quad (1)$$

The same applies to discrete convolutions. Thus, a shift of a state in a grid can be modeled as a convolution with a two dimensional transition filter mask that resembles a discrete Dirac Delta Function.

In the context of Convolutional Neural Networks, this means that we can model transitions in a state grid as a convolutional layer with known filter masks. Exemplary filter masks for the 4-connected neighborhood are depicted in Figure 2. Note especially that the last filter mask represents the *idle element*, a centered discrete Dirac Function. Convolution with this mask has no effect on the state grid.

B. Value Iteration Module

The Value Iteration Module computes the cost to reach each state within the grid map from a given starting point. For this, it takes the following steps:

- 1) **Initialization** - Initialize cost grid with arbitrary but very large values in every state. Set the cell of the starting state to zero.
- 2) **Non-zero padding** - Pad the state grid with arbitrary but very high values. Since zero represents the starting state, zero padding would introduce faulty results in the border regions.
- 3) **Cost propagation** - Execute all possible transitions by convolving the cost grid with transition filters.
- 4) **Cost accumulation** - Add cost per state and transition as an additive layer.

- 5) **Assign minimum state cost** - Compute minimum cost per state by min-pooling over the transition direction of the cost grid.
- 6) **Recurse** - With the result of 5), restart at 2) until convergence is reached.

Figure 1 demonstrates this process. At convergence, the output of 4) (green layer in Figure) stores the transition policy. This is due to the fact that this stage represents the cost per state and possible action to end up in that specific state. An argmin operation in the direction of possible transition will result in the cheapest possible action to end up in each state.

C. Path Evaluation Module

The Value Iteration Module only computes the minimum cost per state. However, it does not compute the optimal path from starting state to goal state. For this, another recurrent network has to be introduced. Again, this network consists of several steps:

- 1) **Initialization** - Initialize state grid with all zeros. Set the cell of the destination state to one. Flip the transition filters so that they are mirrored around the centerpoint and input and output directions are exchanged. Create a one-hot representation per cell and action from the argmin of 4) in the Value Iteration Module. This is the transition selection mask. For full differentiability, this may also be a softmax operation.
- 2) **Transition selection** - Multiply the current state grid with the transition selection mask. The output will be a grid with a single one in the cheapest possible transition into the current state.
- 3) **Zero padding** - Pad the state grid zeros. Now, zeros represent non-occupied states.
- 4) **Propagate state** - Convolve with the flipped transition filters.
- 5) **Recurse** - With the result of 4), restart at 2) until starting state is reached.

This module is equivalent to the optimal predecessor backtracking in Dijkstra's Algorithm: the network traces back the entire optimal actions that led to the goal state. The

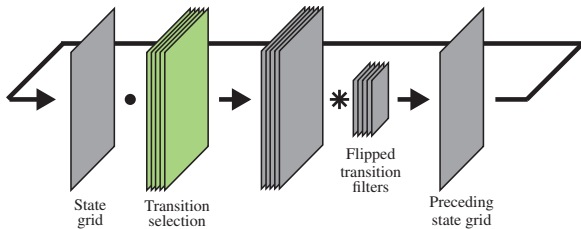


Fig. 3: Recurrent Neural Network to trace back optimal path. The transition selection layer (green) is the argmin of the cost map in Figure 1 at convergence

corresponding network is depicted in Figure 3. The transition selection policy is derived from the cost layer of the network depicted in Figure 1, both colored in green. For backtracing of the optimal path, all transitions have to be reversed. This means that for the filters, we need to flip them in spatial direction and exchange input for output. Convolutions in the forward direction mapped from a map of dimension $W \times H \times 1$ to $W \times H \times K$ for K possible transitions. For the backtracing, since we select 1-of- K possible transitions, the convolution now maps from $W \times H \times K$ to $W \times H \times 1$.

D. Transition Cost Network

To this end, the entire planning process was fully deterministic and predefined by the user. The only parameter to be changed is the cost per state and action. In this part of the network, all learning techniques may be applied. Since the planning network as explained above is fully made up from CNN components, all well-known techniques and toolboxes are at hand [15], [16].

Thus, it is left to the user to specify the input data and network architecture that will generate a cost map. This cost map then specifies the graph topology that is evaluated in the planning stage. Please also note that at time of deployment, this is the only part of the CNN that is still necessary to be a Neural Network. The planning stage may also be exchanged for any other shortest path algorithm as long as the cost map is constrained accordingly.

In this work, to demonstrate the capability of the approach, we employ a Fully Convolutional Network (FCN) operating on aerial views of roads. It is trained to predict cost per state and possible transition to imitate human behavior in traffic. See Section III for details.

E. Loss Function for Imitation Learning

Path planning can be understood as an image segmentation task. Parts of the grid may either belong to the class *path* or *not path*. Accordingly, all loss functions that are used for segmentation can be employed. Specifically, cross entropy remains available for both paths and trajectories. In the context of trajectories, one grid is evaluated against ground truth per planning step. For the path however, one single grid has to be computed for the entire planning task. Note that the output of the Path Evaluation Module is a single grid per planning step. We compute the path from N planning

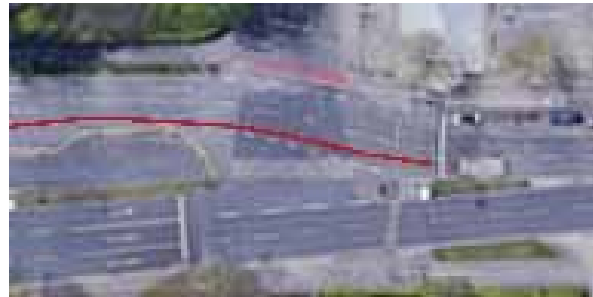


Fig. 4: Example aerial image from the dataset together with one sample path from real world trajectories. Image data from Google Maps [17]

step grids G_i with $i = 1, \dots, N$. Here, every entry in the grids G_i is in range $[0, 1]$, where the value 0 is an definitely unoccupied cell and 1 represents a definitely occupied cell. Thus, the path grid P can be computed as

$$P = 1 - \prod_{i=1}^N (1 - G_i). \quad (2)$$

Note that in the case of an argmin operation for transition selection, every grid per planning step will only feature one unique one in the entire grid. In case of the softmax operation, however, the grids can take arbitrary values in $[0, 1]$ which is necessary for differentiability.

III. EXPERIMENTS

To show the performance of the proposed approach, we design a network to plan drivable paths from aerial images. The network is trained on both, real and simulated data. We then show its capability on new aerial images that were not included in training.

A. Data

We collected a set of only six aerial images from Google Maps and created ground paths both in simulation and from real world data [17]. For the simulation, we manually annotated roughly 100 possible paths within the images. For real world data, we recorded vehicle trajectories from the road side of an intersection. The maps had a total size of $75\text{m} \times 37.5\text{m}$ and paths had varying length.

B. Network Architecture and Training

The Value Iteration Module requires three individual design aspects: the size of the map, the choice of transition filters and the computation of transition costs.

In our experiment, we discretize the input maps into 192×96 cells for planning. Thus, each cell represents a space of roughly $0.4\text{m} \times 0.4\text{m}$ in the real world. We found this sufficient for initial proof of concept, however, for more accurate planning, one might want to increase resolution.

The transition filters in our experiment represent all possible transitions from one cell to its eight neighbors as well as the idle transition. This means that the filter masks shown

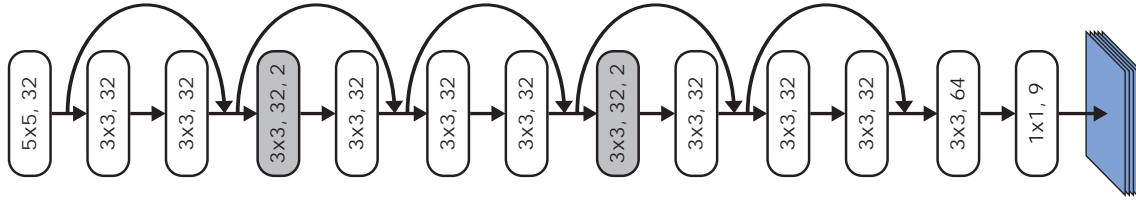


Fig. 5: ResNet for cost map computation. Non-linearities are ReLU. The shaded layers perform dilated convolutions with dilation parameter printed inside. The blue maps that are output are the same as in Figure 1

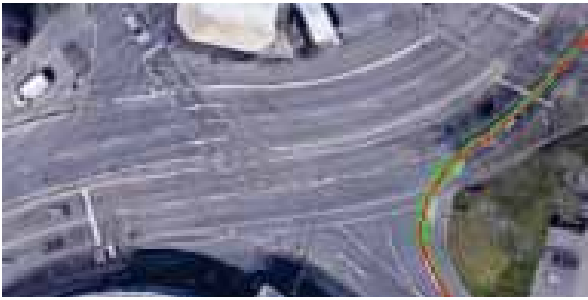


Fig. 6: Training results. Path planned by network (green) vs. human paths (red). Image data from Google Maps [17]

in Figure 2 are extended by the four filters representing the diagonal transitions.

Lastly, we computed the transition cost map from the aerial images. For this, we trained a fully convolutional residual network with dilated kernels that predicts cost per cell and transition (the blue layer in Fig. 1) [11], [18], [19]. The network architecture is depicted in Figure 5. Since our task is fairly simple and run on low resolution images, we can use a comparably shallow and narrow network. To keep the dimensions of the output same as the input dimension without the need for upsampling we use dilated kernels for every other residual block. In training, to avoid overfit due to our very limited data set, we apply dropout of 20% to the output of the first and second-to-last convolutional layer. All layers use rectified linear units (ReLU) as nonlinearity.

C. Experimental Results

We split the data into two parts, a training set of five road scenes and one as the test scene. We select the single scene for which we have human example paths for the test set. This has a very simple reason: for planning of paths, we have no means to evaluate *right* or *wrong*. We can only compare paths to see if they are *human-like* or not.

Figure 6 shows an example path from the training set together with the network’s planning result. As it can be seen, the network in general is perfectly able to replicate human path planning.

We can now look at planning results for previously unobserved scenes. For this, we run the network to re-plan paths that we have previously observed in real life.

Figure 7 shows planning results for two different maneuvers. The first test case is a simple maneuver of lane



(a) Plan for lane following on test data



(b) Plan for right turn with multiple lane changes on test data

Fig. 7: Paths planned on test data by the network (green) vs. human paths (red). Image data from Google Maps [17]

following as depicted in Fig. 7a. The network performs well and can actually generate paths very similar to the actual human behavior.

A more challenging task is depicted in Fig. 7b. Here, we asked the network to plan a path for a right turn with integrated lane change. While the human performs both tasks in one, the network first takes the right turn before taking an abrupt lane change right at the end of the path (left in Fig 7b). Also note the planning artifacts at the small drive right at the bend of the planned path.

From these results, we conclude that the network trained on such a small dataset may perform reasonably for very simple driving situations. However, it does not yet generalize well for more complex tasks.

As a final sanity check, we may look at the feature filter masks trained in the lowest layer of the network. The filter bank is depicted in Fig. 8. Note that the network was not initialized from any existing net but instead was trained from scratch on the five training images.

Judging from the filter masks, the network bases its

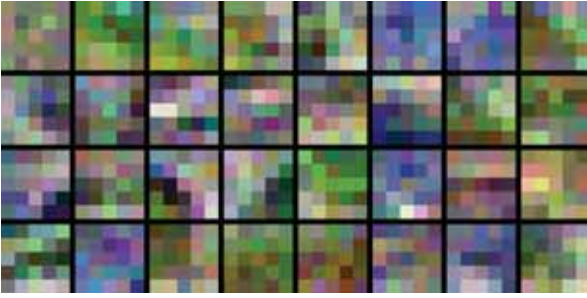


Fig. 8: Trained filter masks in lowest layer

decision in planning on two kinds of features. For once there are filter masks that represent gray-scale and blueish edges. These kinds of features are found on roads. The blueish hue of filters may stem from the blue haze of shadow areas. The other strong feature focus lies on green colors. This is due to roads being bounded by terrain and vegetation.

IV. CONCLUSION

In this work, we proposed a neural network architecture to execute planning. We trained a Value Iteration Network to imitate human motion planning behavior. A Network trained on simulated trajectories showed the capability to reproduce human actions in simple driving situations. It is especially noteworthy that this result could be achieved from simulated paths in only five different road layouts. We expect the network to generalize well if more training trajectories are provided. In general, we have shown that it is possible to replicated human planning using a unified Neural Network Architecture. This implies that intelligent vehicles might learn strategic planning while in operation in real traffic.

REFERENCES

- [1] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.
- [2] J. Ziegler and C. Stiller, "Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2009, pp. 1879–1884.
- [3] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for berthaa local, continuous method," in *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*. IEEE, 2014, pp. 450–457.
- [4] P. Bender, Ö. Ş. Taş, J. Ziegler, and C. Stiller, "The combinatorial aspect of motion planning: Maneuver variants in structured environments," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE, 2015, pp. 1386–1392.
- [5] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.
- [6] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *AAAI*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.
- [7] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa, "Planning-based prediction for pedestrians," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2009, pp. 3931–3936.
- [8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.
- [10] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," *arXiv preprint arXiv:1504.06852*, 2015.
- [11] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.
- [12] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [13] A. Tamar, Y. Wu, G. Thomas, S. Levine, and P. Abbeel, "Value iteration networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 2154–2162.
- [14] T. Shankar, S. K. Dwivedy, and P. Guha, "Reinforcement learning via recurrent convolutional neural networks," *arXiv preprint arXiv:1701.02392*, 2017.
- [15] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 675–678.
- [16] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.
- [17] Google, "Google maps," <http://maps.google.com>.
- [18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
- [19] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

Autonomous Perpendicular And Parallel Parking Using Multi-Sensor Based Control*

David Pérez-Morales¹, Olivier Kermorgant¹, Salvador Domínguez-Quijada¹ and Philippe Martinet¹

Abstract—This paper addresses the perpendicular and parallel parking problems of car-like vehicles for both forward and reverse maneuvers in one trial by improving the work presented in [1] using a multi sensor based controller with a weighted control scheme. The perception problem is discussed briefly considering a Velodyne VLP-16 and a SICK LMS151 as the sensors providing the required exteroceptive information. The results obtained from simulations and real experimentation for different parking scenarios show the validity and potential of the proposed approach.

I. INTRODUCTION

Even for experienced drivers, parking can be a difficult task, especially in big cities where the parking spots are often very narrow. The search for an increase in comfort and safety when parking has led to a quite extensive literature [2], having explored many different approaches to automate this bothersome task.

Despite the fact that the automobile industry has already started to roll out some commercial implementations of active parking assistants capable of actively controlling acceleration, braking and steering [3], the research interest in the topic remains strong.

Path planning approaches have been heavily investigated in recent years. Among the different planning techniques it is possible to distinguish between geometric approaches, with either constant turning radius [4], [5] using saturated feedback controllers, or continuous-curvature planning using clothoids [6]; heuristic approaches [7] and machine learning techniques [8]. It is worth to note that parking maneuvers with forward motions are seldom considered, with [9] for the parallel parking case and [10] for the perpendicular case being some of the few works on this regard.

A well known drawback of path planning and tracking is its dependence on the localization performance. An interesting alternative that does not rely on the localization is the use of a sensor based control approach. It has been proven to be valid for navigation [11], dynamic obstacles avoidance [12], and for parking applications [1].

*This work was supported by the Mexican National Council for Science and Technology (CONACYT). This paper describes work carried out in the framework of the Valet project, reference ANR-15-CE22-0013-02.

¹ David Pérez-Morales, Olivier Kermorgant, Salvador Domínguez-Quijada and Philippe Martinet are with LS2N, Laboratoire des Sciences du Numérique de Nantes, École Centrale de Nantes, 1 rue de la Noë, 44321 Nantes, France

¹ David.Perez-Morales@eleves.ec-nantes.fr

¹ Olivier.Kermorgant@ec-nantes.fr

¹ Salvador.DominguezQuijada@ls2n.fr

¹ Philippe.Martinet@ec-nantes.fr

The contribution of this paper is an improvement on the approach described in [1], this time considering multiple sensors, a better suited sensor feature set that allows to park in one maneuver not only in perpendicular spots but also in parallel ones with either reverse or forward motions with only some minor changes, and improved constraints for collision avoidance.

In the next section the models considered as well as the notation used are presented. In Section III the perception problem is briefly addressed showing how the sensor data is processed in order to extract the empty parking spot to latter in Section IV describe the interaction model and how to extract the required sensor features from the computed empty parking spot. Afterwards, the controller is presented in Section V and the obtained results from simulation and real experimentation for different parking scenarios are shown in Section VI. Finally, some conclusions are given in Section VII.

II. MODELING AND NOTATION

Given that parking maneuvers are low speed motions, a kinematic model can be considered as accurate enough.

A. Car-like robot model and notation

The kinematic model considered is the one used to represent a car with rear-wheel driving:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos \theta \\ \sin \theta \\ \tan \phi / l_{wb} \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \dot{\phi}, \quad (1)$$

where v and $\dot{\phi}$ are the longitudinal and steering velocities.

Table I presents the different parameters used in the paper.

TABLE I: Parameters definition

Parameters	Notation	Value
Wheelbase: Distance between the front and rear wheel axles	l_{wb}	2.588 m
Rear overhang: Distance between the rear wheel axle and the rear bumper	l_{ro}	0.657 m
Maximum steering angle	ϕ_{max}	30°
Total length of the vehicle	l_{ve}	4.084 m
Total width of the vehicle	w_{ve}	1.945 m
Maximum (desired) longitudinal velocity	$ v_{max} $	2 km/h
Maximum acceleration increment	Δ_{acc}	$\text{sign}(v) 0.2 T_s$
Maximum deceleration increment	Δ_{dec}	$\text{sign}(v) 2.5 T_s$
Maximum ϕ increment	Δ_{ϕ}	2° T_s

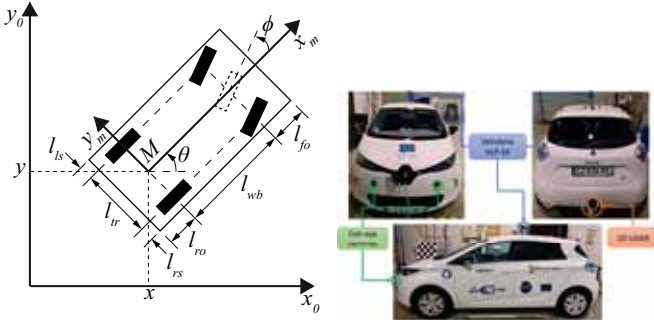
The point M is located at the mid-distance between the passive fixed wheels (rear) axle and the distance between the

rear and the front axle is described by l_{wb} . The generalized coordinates are $\mathbf{q} = [x, y, \theta, \phi]^T$ where x and y are the Cartesian coordinates of the point M , θ is the orientation of the platform with respect to the x_0 axis and the steering angle of the steerable wheel(s) is described by ϕ (Fig. 1a).

From the kinematic model it is possible to extract the following relation between ϕ and $\dot{\theta}$:

$$\phi = \text{atan}\left(\frac{\dot{\theta} l_{wb}}{v}\right) \quad (2)$$

The vehicle used for experimentation and simulation is a Renault ZOE (Fig. 1b). It is represented by its bounding rectangle.



(a) Kinematic model diagram (b) Robotized Renault ZOE

Fig. 1: Kinematic model diagram for a car-like rear-wheel driving robot and vehicle used for simulation and real experimentation

B. Multi-sensor modeling

Following our previous work [1], where a novel sensor based control technique based on the framework described in [13] was proposed, in this paper we explore a different sensor features set to park in one maneuver into perpendicular and parallel spots considering multiple sources for the sensor signals.

1) *Kinematic model*: Let us consider a robotic system equipped with k sensors (Fig. 2) that provide data about the robot pose in its environment. Each sensor S_i gives a signal (sensor feature) s_i of dimension d_i with $\sum_{i=1}^k d_i = d$.

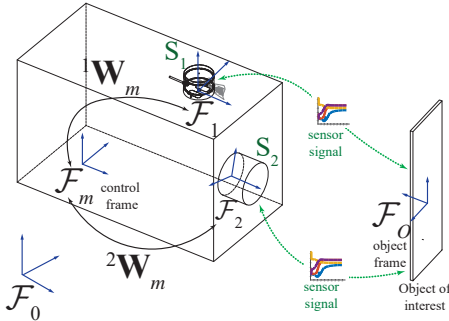


Fig. 2: Multi-sensor model

In a static environment, the sensor feature derivative can be expressed as follows:

$$\dot{s}_i = \mathbf{L}_i \mathbf{v}_i = \mathbf{L}_i {}^i \mathbf{W}_m \mathbf{v}_m \quad (3)$$

where \mathbf{L}_i is the interaction matrix of s_i and ${}^i \mathbf{W}_m$ is the screw transformation matrix that allows to express the sensor twist \mathbf{v}_i with respect to the robot twist \mathbf{v}_m .

Assuming that the vehicle to which the sensors are rigidly attached to evolves in a plane and that the sensors and vehicle have vertical parallel z axes, \mathbf{L}_i is of dimension $d_i \times 3$ and the screw transformation matrix takes the following form:

$${}^i \mathbf{W}_m = \begin{bmatrix} c({}^m \theta_i) & s({}^m \theta_i) & t_{i_x} s({}^m \theta_i) - t_{i_y} c({}^m \theta_i) \\ -s({}^m \theta_i) & c({}^m \theta_i) & t_{i_x} c({}^m \theta_i) + t_{i_y} s({}^m \theta_i) \\ 0 & 0 & 1 \end{bmatrix} \quad (4)$$

where ${}^m \mathbf{t}_i = [t_{i_x}, t_{i_y}]^T$ and ${}^m \theta_i$ are, respectively, the position and orientation of S_i with respect to \mathcal{F}_m expressed in \mathcal{F}_m , with $c({}^m \theta_i) = \cos({}^m \theta_i)$ and $s({}^m \theta_i) = \sin({}^m \theta_i)$.

Denoting $\mathbf{s} = (s_1, \dots, s_k)$ the d -dimensional signal of the multi-sensor system, the signal variation over time can be linked to the moving vehicle twist:

$$\dot{\mathbf{s}} = \mathbf{L}_s \mathbf{v}_m \quad (5)$$

with:

$$\mathbf{L}_s = \mathbf{L} \mathbf{W}_m = \begin{bmatrix} \mathbf{L}_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \mathbf{L}_k \end{bmatrix} \begin{bmatrix} {}^1 \mathbf{W}_m \\ \vdots \\ {}^k \mathbf{W}_m \end{bmatrix} \quad (6)$$

Nevertheless, since in our application the control frame \mathcal{F}_m is attached to the vehicle's rear axis with origin at the M point (Fig. 1a), it is not possible to generate a velocity along y_m on the vehicle's frame due to the nonholonomic constraint of the kinematic model (1). Assuming that there is no slipping nor skidding (i.e. $v_{y_m} = 0$), the robot twist $\mathbf{v}_m = [v_{x_m}, v_{y_m}, \dot{\theta}]^T$ can be reduced to:

$$\mathbf{v}_m = [v_{x_m}, \dot{\theta}]^T \quad (7)$$

where $v_{x_m} = v$ and considering as well the consequent reduction of \mathbf{L}_s , being now of dimension $d \times 2$.

2) *Weighted error*: We consider the weighted multi-sensor error signal, as described in [13], which is defined as:

$$\mathbf{e}_H = \mathbf{H} \mathbf{e} \quad (8)$$

where $\mathbf{e} = \mathbf{s} - \mathbf{s}^*$ is the difference between the current sensor signal \mathbf{s} and its desired value \mathbf{s}^* and \mathbf{H} is a diagonal positive semi-definite weighting matrix that depends on \mathbf{s} with its associated interaction matrix being $\mathbf{L}_H = \mathbf{H} \mathbf{L}_s$. Making a distinction between task and constraints features, $\mathbf{H} = \text{diag}(\mathbf{H}_t, \mathbf{H}_c)$ and $\mathbf{s} = [s_t, s_c]^T$. Each component h_i of \mathbf{H} may or may not vary in order to optimize the system behavior, ensure specific constraints, manage priorities or add or remove a sensor or a feature from the control law.

Task features s_t , as their name suggest, are used to perform the task by driving \mathbf{e}_t to 0. On the other hand, since the constraints features s_c are used only to ensure certain constraints, we don't care about \mathbf{e}_c as the desired value \mathbf{s}_c^* is meaningless.

III. PERCEPTION

We focus the perception on the detection of parked cars. They can be approximated by boxes considering that, when viewed from the top, have a rectangular-like shape.

The vehicle used (Fig. 1b) has been equipped with many sensors (Velodyne VLP-16, SICK LMS151, GPS, cameras in the front, etc.) to observe its environment, a computer to process the data and actuators that can be computer controlled. Since our application requires exteroceptive information from all around the vehicle at, potentially close distances, the VLP-16 and SICK LMS151 were the sensors chosen to work with.

Because both sensors provide information of a very similar nature, the data can be fused by simply converting the *LaserScan* data provided by the LMS151 to *PointCloud2* and then transforming the point cloud from LMS151's frame to the VLP-16's frame so it can be added to the point cloud provided by the latter sensor. For this, it is assumed that the time difference between the data provided by each sensor is reasonably small, i.e. the data is sufficiently synchronized.

The complete point cloud obtained from the two sensors is first filtered with a couple of crop boxes. The first crop box is keeps only the data that is close enough to the car to be relevant in a parking application and that does not represent the floor and afterwards and the second one is used to filter out the points that belong to the car's body (self-collision sensor readings). Then, an Euclidean Cluster Extraction algorithm is used to have each obstacle represented as a cluster. The orientation of each cluster is extracted by fitting a line model to the points belonging to the contour of the cluster using a RANSAC algorithm. The orientation of the bounding box will be equal to the orientation of the fitted line. After, we proceed by finding the rotated bounding box of the cluster using the previously found orientation.

The empty parking place (green rectangle in Fig. 3) is extracted using the approach described in [1]. The sensor features required for the controller are extracted from this computed parking place.

IV. INTERACTION MODEL

For the interaction model, we rely on the perception of several lines \mathcal{L}_j and points from several sensors. Since the sensor data is expressed in the Cartesian space, it can be easily transformed from one frame to another, thus allowing us to use virtual sensors placed at will.

The sensor's placement can be seen in Fig. 3. S_1 corresponds to the VLP-16 while S_2 to the LMS151. S_3 to S_5 are virtual sensors placed on the corners of the car's bounding rectangle. All the frames of the virtual sensors have the same orientation as the control frame.

To illustrate the feature extraction approach, the case of a reverse perpendicular maneuver is now detailed. As it can be seen in Fig. 3, points p_1 to p_4 correspond to the corners of the parking spot while p_5 and p_6 are, respectively, the midpoints between (p_1, p_4) and (p_2, p_3) . \mathcal{L}_1 is a line that passes through p_5 and p_6 , i.e. it passes through the center

of the parking spot. \mathcal{L}_2 is a line that passes through p_1 and p_4 thus corresponding to the depth limit of the parking spot. \mathcal{L}_3 is a line that passes through p_3 and p_4 . All the lines are parametrized using normalized Plücker coordinates.

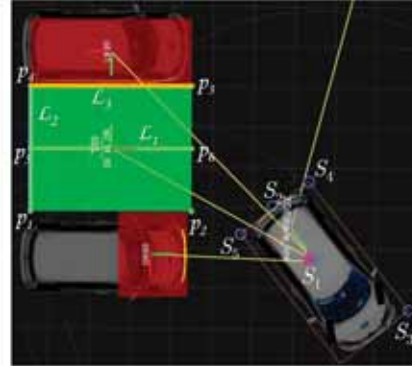


Fig. 3: Sensors' configuration and sensor features

The exact definition of the sensor features varies depending on the parking scenario although in any case, \mathcal{L}_1^* should be collinear with x_m and \mathcal{L}_2^* should be parallel to y_m and be behind the car for reverse maneuvers and in front for forward ones. In this paper we only detail the actual features used for a specific case, but deducing the features that should be used for other cases isn't complicated.

Considering the previously mentioned assumption that the vehicle to which the sensors are attached to evolves in a plane (sensors and vehicle with parallel z axes), the sensor signal $s_{i_{\mathcal{L}_j}}$ and interaction matrix $\mathbf{L}_{i_{\mathcal{L}_j}}$ for the line \mathcal{L}_j observed by S_i are defined respectively by (9) and (10)

$$\mathbf{s}_{i_{\mathcal{L}_j}} = [{}^i\mathbf{u}_j(1), {}^i\mathbf{u}_j(2), {}^i\mathbf{h}_j(3)]^T \quad (9)$$

$$\mathbf{L}_{i_{\mathcal{L}_j}} = \begin{bmatrix} 0 & 0 & {}^i\mathbf{u}_j(2) \\ 0 & 0 & -{}^i\mathbf{u}_j(1) \\ -{}^i\mathbf{u}_j(2) & {}^i\mathbf{u}_j(1) & 0 \end{bmatrix} \quad (10)$$

where ${}^i\mathbf{u}_j = {}^i\mathbf{u}_j / \|{}^i\mathbf{u}_j\|$ with ${}^i\mathbf{u}_j \neq 0$ denoting the orientation of \mathcal{L}_j and ${}^i\mathbf{h}_j = {}^i\mathbf{w}_j / \|{}^i\mathbf{u}_j\|$ with ${}^i\mathbf{w}_j$ containing the coefficients of the interpretation plane equation [14]. In the 2D configuration considered, the components of ${}^i\mathbf{u}_j$ and ${}^i\mathbf{h}_j$ that don't appear in (9) are equal to 0 thus ${}^i\mathbf{h}_j(3)$ can be interpreted as the distance to the line.

It should be noted that the weighting and constraints required to park safely change depending on the type of parking spot (parallel, perpendicular or diagonal) and on which side the parking spot is placed with respect to the car at the beginning of the maneuver.

A. Task sensor features

The control features required to perform the parking task are defined by (11), with $t = 1$ for forward maneuvers and $t = 2$ for the reverse case.

$$\mathbf{s}_t = [\mathbf{s}_{t_{\mathcal{L}_1}}, \mathbf{s}_{t_{\mathcal{L}_2}}]^T \quad (11)$$

A 2nd order approximation of the form (12) is used for the interaction matrix.

$$\mathbf{L}_t = \frac{\mathbf{L}_{\mathcal{L}_j} + \mathbf{L}_{\mathcal{L}_j}^*}{2} \quad (12)$$

The weighting matrix \mathbf{H}_t is defined by (13). The variable components h_i^t are computed using a smooth weighting function (Fig. 4) based on the one presented in [15].

$$\mathbf{H}_t = \text{diag}(h_1^t, h_2^t, h_3^{t_{const}}, h_4^t, h_5^t, h_6^{t_{const}}) \quad (13)$$

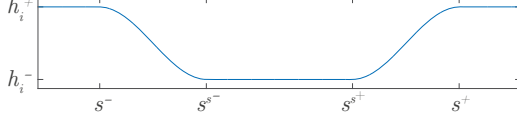


Fig. 4: Weighting function h_i^t

With this weighting function and considering the weighting parameters (31), $s_i \forall i = \{2, 4, 5\}$ could be seen as bounded constraints but in fact they are task features, given that we do care about the value of their corresponding e_i . If only s_3 and s_6 were considered as task features, the car may finish the maneuver with a bad orientation even if $e_3 \approx e_6 \approx 0$ because just 2 features are not enough to control the car's DOFs.

Due to space constraints, only the case of a reverse perpendicular parking maneuver with the spot placed on the right will be considered for the rest of this section.

B. Constraints sensor features

The constraints are defined by (14).

$$\mathbf{s}_c = [s_3, s_4, s_5]^T \quad (14)$$

For the constraints sensor features we are interested only in the components of (9) related to the distance to the feature itself, therefore:

$$s_3 = {}^3\mathbf{h}_3(3) \quad (15)$$

$$s_4 = [{}^4\mathbf{h}_2(3), {}^4\mathbf{h}_3(3)]^T \quad (16)$$

$$s_5 = [{}^5\mathbf{h}_2(3), {}^5d_{y_m}]^T \quad (17)$$

with ${}^5d_{y_m}$ being the difference $d_{y_m} = \rho_{corner} - \rho_{lat}$ measured with the sensor S_5 , expressed in the sensor frame (Fig. 5) if $\phi < 0$, defined as:

$${}^5d_{y_m} = \sqrt{({}^5x_2 + t_{5_x})^2 + ({}^5y_2 + t_{5_y} - \rho_m)^2} + \rho_m - t_{5_y} \quad (18)$$

with $\rho_m = l_{wb}/\tan \phi$ and, when $\phi \geq 0$, being simply the distance from S_5 to p_2 along y_m measured with S_5 , it is defined as:

$${}^5d_{y_m} = {}^5y_2 \quad (19)$$

with ${}^5p_2 = ({}^5x_2, {}^5y_2)$ being the point p_2 measured with S_5 .

The corresponding interaction matrices are:

$$\mathbf{L}_3 = \begin{bmatrix} -{}^3\mathbf{u}_3(2) & {}^3\mathbf{u}_3(1) & 0 \end{bmatrix} \quad (20)$$

$$\mathbf{L}_4 = \begin{bmatrix} -{}^4\mathbf{u}_2(2) & {}^4\mathbf{u}_2(1) & 0 \\ -{}^4\mathbf{u}_3(2) & {}^4\mathbf{u}_3(1) & 0 \end{bmatrix} \quad (21)$$

$$\mathbf{L}_5 = \begin{bmatrix} -{}^5\mathbf{u}_2(2) & {}^5\mathbf{u}_2(1) & 0 \\ 0 & -1 & -{}^5x_2 \end{bmatrix} \quad (22)$$

Since the constraints are used for collision avoidance, only one side of the interval $[s_c^-, s_c^+]$ (25) has to be defined for each feature.

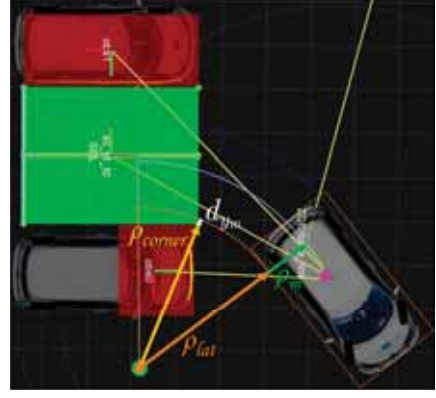


Fig. 5: Lateral constraint d

V. CONTROL

When considering the constraints presented in Sec. IV-B (particularly (18)), a chattering problem appears if the controller presented in [1] is used, even with very small weights. For this reason, that controller had to be adapted to the quadratic programming form [16] with only inequality constraints (23):

$$\begin{aligned} \mathbf{v}_m &= \text{argmin} \|\mathbf{L}_{\mathbf{H}_t} \cdot \mathbf{v}_m + \lambda \cdot \mathbf{e}_t\|^2 \\ &\text{s.t. } \mathbf{A} \mathbf{v}_m \leq \mathbf{b} \end{aligned} \quad (23)$$

with:

$$\mathbf{A} = [\mathbf{L}_{\mathbf{H}_c}, -\mathbf{L}_{\mathbf{H}_c}]^T \quad (24)$$

$$\mathbf{b} = [\alpha(\mathbf{s}_c^+ - \mathbf{s}_c), -\alpha(\mathbf{s}_c^- - \mathbf{s}_c)]^T \quad (25)$$

where α is a gain constant, λ is the control gain, \mathbf{H}_c is an identity matrix (i.e. there is no weighting on the constraints) and $[s_c^-, s_c^+]$ is the interval in which we want to keep s_c .

To limit the speed of the vehicle as it approaches to the parking spot, a deceleration profile, based on the velocity profile shown in [5], is used. It is defined by (26)

$$\begin{aligned} \text{if } \mathbf{e}(6) < \mathbf{e}(6)^{th} \\ v_{max} &= (|v_{max}| - v_{max}^0)(\mathbf{e}(6)/\mathbf{e}(6)^{th}) + v_{max}^0 \end{aligned} \quad (26)$$

with v_{max}^0 being the maximum desired velocity when the sixth component of the error vector $\mathbf{e}(6)$ tends to zero and $\mathbf{e}(6)^{th}$ is a threshold value for $\mathbf{e}(6)$. Since the low level velocity controller is not capable of reaching very small values, $v_{max}^0 = 0.2$ km/h.

The control signals v and ϕ are bounded by their respective maximum desired values as shown below:

$$|v| < |v_{max}| \quad (27)$$

$$|\phi| < \phi_{max} \quad (28)$$

To avoid large changes in the control signals at time k that may cause uncomfortable sensations for the passengers or surrounding witnesses, they are bounded by some increments (29) (30) with respect to the control signal at $k-1$.

$$(v_{k-1} - \Delta_{dec}) \geq v_k \leq (v_{k-1} + \Delta_{acc}) \quad (29)$$

$$(\phi_{k-1} - \Delta_{\phi}) \geq \phi_k \leq (\phi_{k-1} + \Delta_{\phi}) \quad (30)$$

To solve (23), a generic solver is used. To improve the stability and computation time, the optimization variables are $[v, \phi]$ and not \mathbf{v}_m , although inside the objective function $\dot{\theta}$ is computed from ϕ so (23) can be solved. When using ϕ instead of $\dot{\theta}$ one can easily impose the bounds (28) at the solving step instead of solving (23) directly with \mathbf{v}_m as optimization variables and hope for the value of ϕ computed from $\dot{\theta}$ to fall inside the bounds (28).

VI. RESULTS

To show the potential of our approach, several parking scenarios are presented below, all of them using the final form of the controller (23). The unconstrained cases were computed in MATLAB, using `fmincon` as solver. For the constrained cases, NLOpt with the SLSQP algorithm was used.

A. Unconstrained cases - MATLAB

To evaluate the performance of the proposed approach, it was first tested in unconstrained cases with a sampling time $T_s = 0.1$. As it can be seen in Figs. 6-10, the presented technique allows to perform parking maneuvers for many different scenarios (perpendicular and parallel with either reverse or forward motions) just by adjusting the weighting parameters and the specific definition of the sensor features. The final errors for all of these cases are in the order of $\times 10^{-3}$ or smaller.

As an example of the weighting approach, for the case of a reverse perpendicular parking maneuver with the parking spot placed on the right, $h_i^+ = 5$, $h_i^- = 0$, $h_3^{t_{const}} = 1$, $h_6^{t_{const}} = 0.75$ and:

$$\begin{cases} s_1^{s^+} = s_1^+ = \infty \\ s_1^{s^-} = 0.001 + s_1^* \\ s_1^- = -0.001 + s_1^* \\ s_i^{s^-} = s_i^- = -\infty \quad \forall i = \{2, 4, 5\} \\ s_i^{s^+} = -0.001 + s_i^* \quad \forall i = \{2, 4, 5\} \\ s_i^+ = 0.001 + s_i^* \quad \forall i = \{2, 4, 5\} \end{cases} \quad (31)$$

These weighting parameters allow to prioritize the error in position over the orientation for the most part of the maneuver and, when ${}^i\mathbf{u}_j^*(a)$ is almost reached, smoothly increase the corresponding weights so we can gradually switch the priority from positioning the vehicle to orientate it to avoid finishing the maneuver with a bad orientation.

It can be seen how, for all shown cases, the weights (Figs. 6d-10d) push ϕ towards 0 (Figs. 6b-10b) once ${}^i\mathbf{u}_j^*(a)$ is almost reached when close to the completion of the maneuver to keep the orientation close to the desired value.

Unlike our previous work [1], which required to perform gain-tuning for different initial conditions, this newly presented approach allows to park successfully for different (reasonable) initial positions and orientations of the same parking case using the same weighting parameters, although it should be mentioned that the stability, specially when constraints are considered, is still under study.

In Fig. 10, it can be seen how even if the car is placed considerably farther from the parking spot than in Fig. 6 and not exactly perpendicular to the spot, the vehicle is able to park correctly using the same weighting parameters, showing the stability of the presented approach.

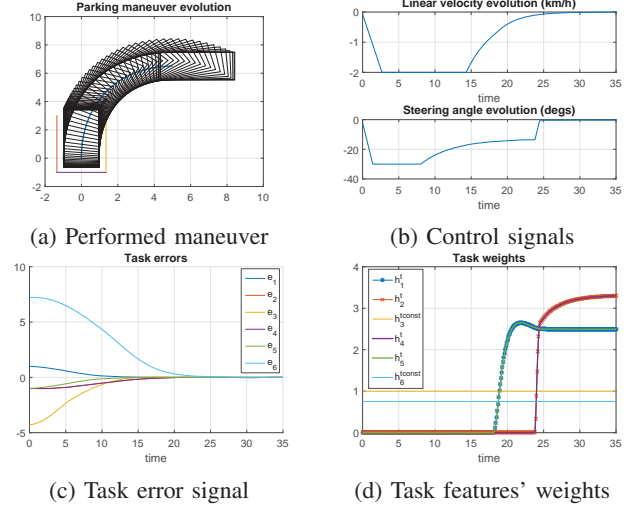


Fig. 6: Unconstrained perpendicular reverse parking maneuver

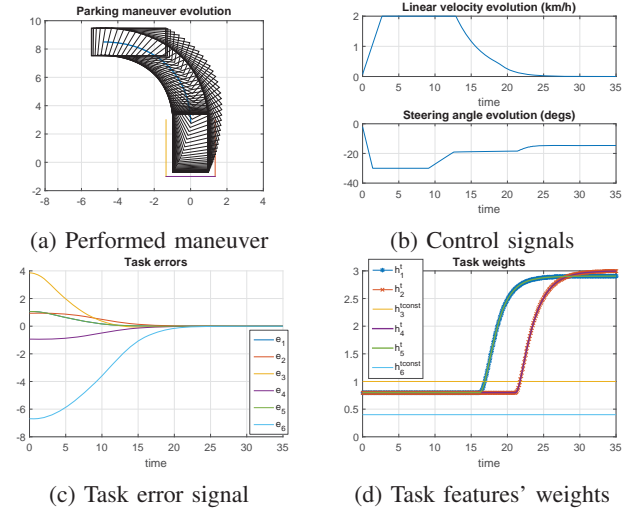


Fig. 7: Unconstrained perpendicular forward parking maneuver

B. Constrained cases

1) *Fast prototyping environment*: A homemade fast prototyping environment using the same software architecture as the one embedded inside the car is used for simulation purposes. This homemade environment is interfaced with Gazebo to simulate the exteroceptive sensors.

The case of a reverse perpendicular parking maneuver with the spot placed on the right is shown below. The weighting parameters remain the same as for the unconstrained case while the constraints are defined with $s_7^+ = -0.1$, $s_8^- = 0.15$, $s_9^+ = -0.1$, $s_{10}^- = 0.15$, $s_{11}^+ = -0.075$ and $T_s = 0.05$.

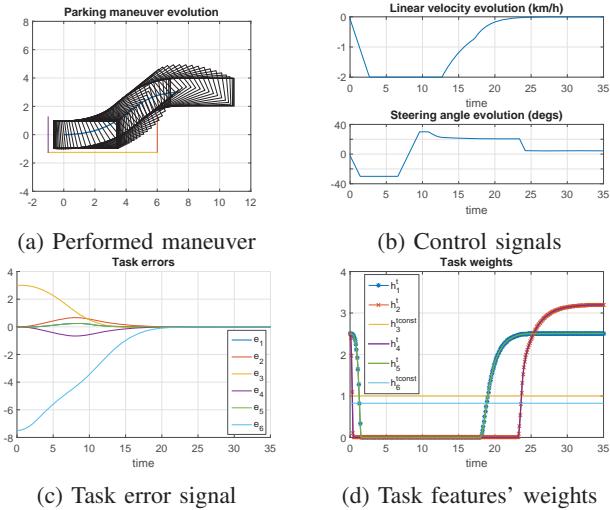


Fig. 8: Unconstrained parallel reverse parking maneuver

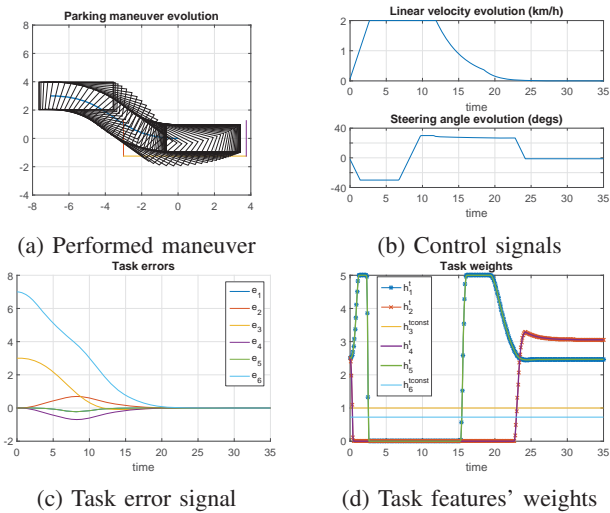


Fig. 9: Unconstrained parallel forward parking maneuver

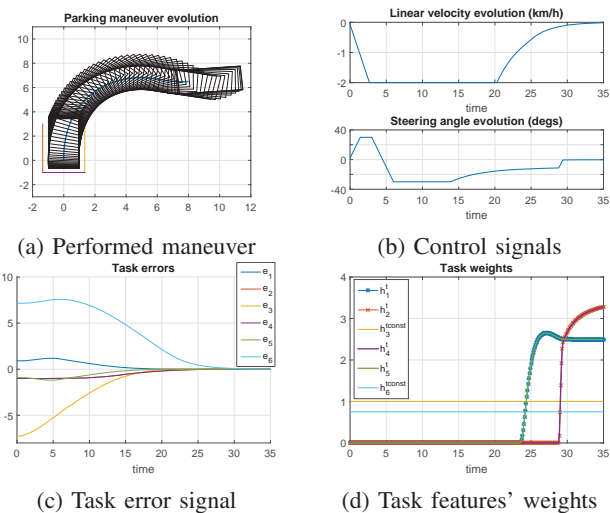


Fig. 10: Unconstrained perpendicular reverse parking maneuver from far

As it can be seen in Figs. 11-12, the car is able to park successfully while respecting the constraints in spite of the sensor noise and the less than perfect system response. The evolution of the many different signals, especially for the longitudinal velocity (Fig. 12a), is very similar to the unconstrained case (Fig. 6b). The fast deceleration at the end (Fig. 12a) is due to a stopping condition in the implementation related to e_t . Regarding the evolution of ϕ , it can be seen how, contrary to the unconstrained case, it doesn't saturate; this behavior is caused by the constraints, particularly s_{11} (Fig. 12d). The final error is $e_t = [-0.0003, -0.0096, 0.0207, -0.0096, 0.0003, 0.0569]^T$.

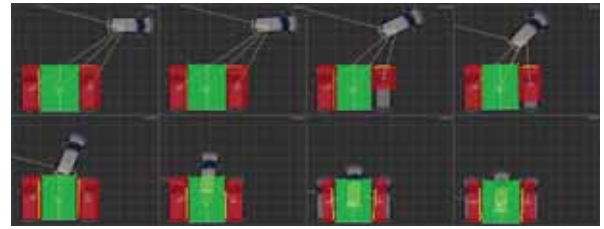


Fig. 11: Constrained perpendicular reverse parking maneuver

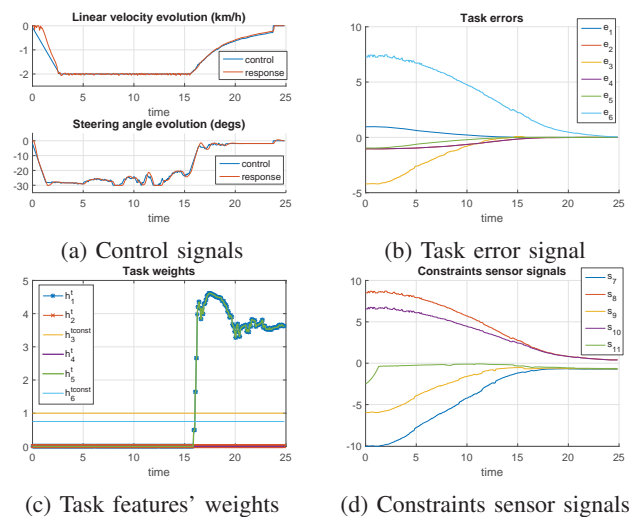


Fig. 12: Constrained perpendicular reverse parking maneuver signals

2) *Real experimentation*: Real experimentation was conducted for the same parking case (Fig. 13) as with the fast prototyping environment shown above. The weighting parameters and constraints definition remain the same.



Fig. 13: Experimental car parking in a perpendicular spot

It is obvious that the response of the system, particularly for the linear velocity (Fig. 14a), is less than ideal, reaching

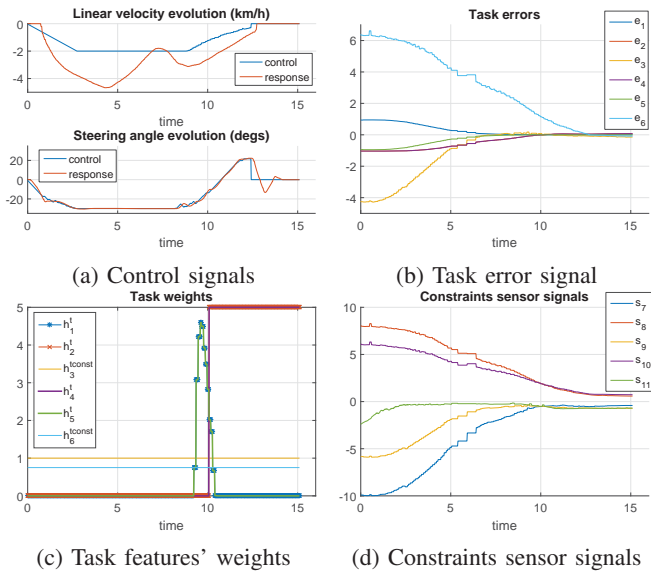


Fig. 14: Constrained perpendicular reverse parking maneuver signals

a speed more than twice as fast than what the controller indicates. This behavior can be attributed to the low-level velocity controller, which still requires some tuning to improve the performance at low velocities, therefore it has no relation to the presented technique.

Despite of the erratic response of the system in addition to the noise coming from the sensors, the constraints were respected during the whole maneuver (Fig. 14d), getting no closer than 33.88cm (s_9) to \mathcal{L}_3 .

Furthermore, the evolution of e_t (Fig. 14b) is very similar to the simulated case (Fig. 12b), although the final error is not as good, being in this case $e_t = [0.0054, 0.0743, -0.1436, 0.0743, -0.0054, -0.0833]^T$.

The smallest $\|e_t\|$ was achieved at $T = 12.6399s$, with $e_t = [0.0025, 0.0429, -0.0851, 0.0429, -0.0025, 0.0207]^T$ and $[v, \phi]^T = [0, 0]^T$ from the controller starting at $T = 12.42s$.

VII. CONCLUSIONS

Following our previous work [1], we showed how a better choice of the sensor features allows to improve the performance, stability and versatility of the presented sensor based approach, this time not only being able to deal successfully with perpendicular parking maneuvers but also with parallel ones with both reverse and forward motions with just some minor adjustments for each type of parking. The stability, specially when constraints are considered, is still under study.

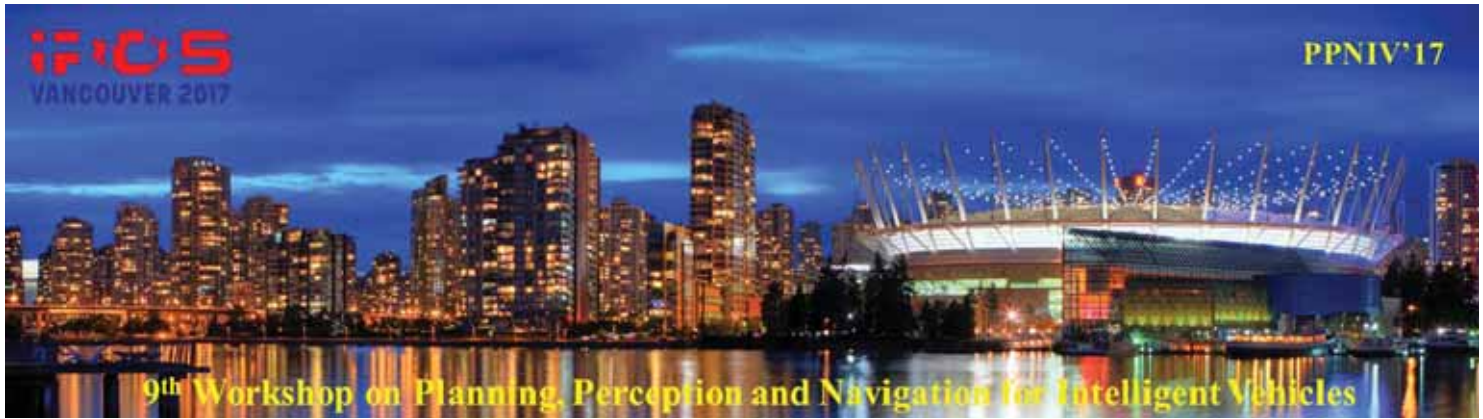
Preliminary results obtained from real experimentation validate the robustness and effectiveness of the presented approach, considering that, despite of the erratic response of the system due to the low-level velocity controller, the car parked successfully while respecting the constraints during the whole maneuver.

It is important to mention that, due to visibility constraints and in order to keep the results obtained with the fast

prototyping environment as close to the reality as possible, only reverse parking maneuvers have been tested outside MATLAB with the presented sensor feature set. Nevertheless, the multi-sensor framework gives a high expandability, allowing for future upgrades to the perception capabilities of the system.

REFERENCES

- [1] D. Pérez Morales, S. Domínguez Quijada, O. Kermorgant, and P. Martinet, "Autonomous parking using a sensor based approach," in *8th Workshop on Planning, Perception and Navigation for Intelligent Vehicles, PPNIV'16 at 19th IEEE ITSC 2016*, Rio de Janeiro, Brazil, 2016, pp. 211–216.
- [2] W. Wang, Y. Song, J. Zhang, and H. Deng, "Automatic parking of vehicles: A review of literatures," *International Journal of Automotive Technology*, vol. 15, no. 6, pp. 967–978, 2014.
- [3] Y. Song and C. Liao, "Analysis and Review of State-of-the-Art Automatic Parking Assist System," in *2016 IEEE International Conference on Vehicular Electronics and Safety*, Beijing, China, 2016, pp. 61–66.
- [4] P. Petrov, F. Nashashibi, and M. Marouf, "Path Planning and Steering control for an Automatic Perpendicular Parking Assist System," in *7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles, PPNIV'15*, Hamburg, Germany, 2015, pp. 143–148.
- [5] P. Petrov and F. Nashashibi, "Saturated Feedback Control for an Automated Parallel Parking Assist System," in *13th International Conference on Control, Automation, Robotics and Vision (ICARCV'14)*, Marina Bay Sands, Singapore, 2014, pp. 577–582.
- [6] H. Vorobieva, N. Minoiu-Enache, S. Glaser, and S. Mammar, "Geometric Continuous-Curvature Path Planning for Automatic Parallel Parking," in *2013 10th IEEE International Conference on Networking, Sensing and Control (ICNSC)*, Evry, France, 2013, pp. 418–423.
- [7] C. Chen, M. Rickert, and A. Knoll, "Path planning with orientation-aware space exploration guided heuristic search for autonomous parking and maneuvering," in *2015 IEEE Intelligent Vehicles Symposium*, Seoul, Korea, 2015, pp. 1148–1153.
- [8] G. Ntomista and M. Botsch, "Maneuver segmentation for autonomous parking based on ensemble learning," in *2015 International Joint Conference on Neural Networks (IJCNN)*, Killarney, Ireland, 2015, pp. 1–8.
- [9] H. Vorobieva, S. Glaser, N. Minoiu-Enache, and S. Mammar, "Automatic parallel parking in tiny spots: Path planning and control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, pp. 396–410, 2015.
- [10] K. Min and J. Choi, "A control system for autonomous vehicle valet parking," in *2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)*, Gwangju, South Korea, oct 2013, pp. 1714–1717.
- [11] D. A. de Lima and A. C. Victorino, "Sensor-Based Control with Digital Maps Association for Global Navigation: A Real Application for Autonomous Vehicles," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, Las Palmas, Spain, 2015, pp. 1791–1796.
- [12] Y. Kang, D. A. de Lima, and A. C. Victorino, "Dynamic obstacles avoidance based on image-based dynamic window approach for human-vehicle interaction," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, Seoul, South Korea, jun 2015, pp. 77–82.
- [13] O. Kermorgant and F. Chaumette, "Dealing with constraints in sensor-based robot control," *IEEE Transactions on Robotics*, vol. 30, no. 1, pp. 244–257, 2014.
- [14] N. Andreff, B. Espiau, and R. Horaud, "Visual Servoing from Lines," *International Journal of Robotics Research*, vol. 21, no. 8, pp. 679–699, 2002.
- [15] V. K. Narayanan, F. Pasteau, M. Marchal, A. Krupa, and M. Babel, "Vision-based adaptive assistance and haptic guidance for safe wheelchair corridor following," *Computer Vision and Image Understanding*, vol. 149, pp. 171–185, 2016.
- [16] O. Kanoun, F. Lamiroux, and P.-B. Wieber, "Kinematic control of redundant manipulators: generalizing the task priority framework to inequality tasks," *IEEE Trans. on Robotics*, vol. 27, no. 4, pp. 785–792, 2011.



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session II

Segmentation & Reconstruction

- **Keynote speaker: Miguel Ángel Sotelo (University of Alcalá, Spain)**
Title: Cooperative Autonomous Driving and Interaction with Vulnerable Road Users
- **Title: A new metric for evaluating semantic segmentation: leveraging global and contour accuracy**
Authors: Eduardo Fernandez-Moral, Denis Wolf, Renato Martins and Patrick Rives
- **Title: Multibody reconstruction of the dynamic scene surrounding a vehicle using a wide baseline and multifocal stereo system**
Authors: Laurent Mennillo, Eric Royer , Frederic Mondoty, Johann Mousainy and Michel Dhome



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

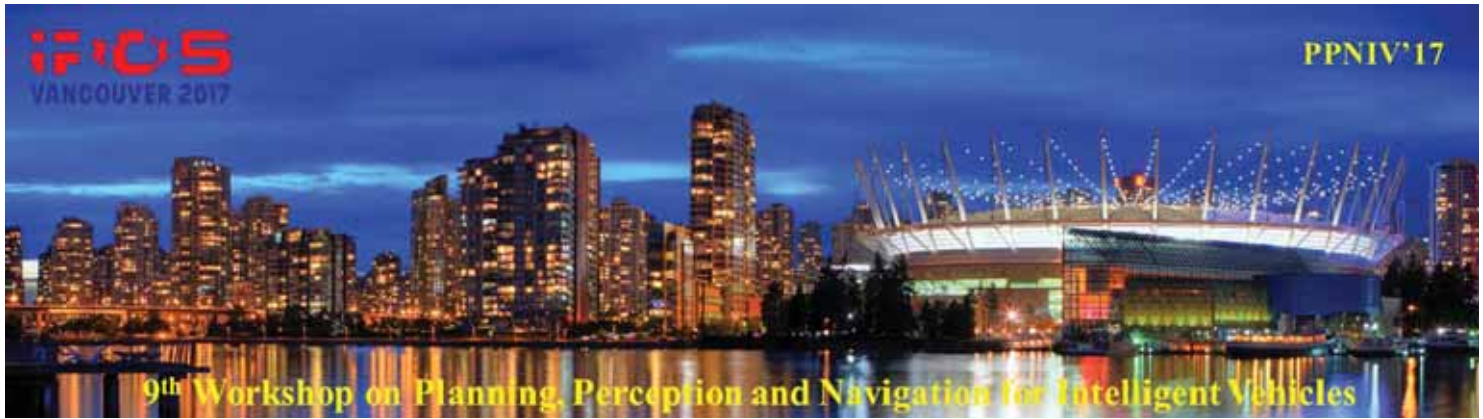
Session II

Keynote speaker: **Miguel Ángel Sotelo**
(University of Alcalá, Spain)

Cooperative Autonomous Driving and Interaction with Vulnerable Road Users

Abstract: Autonomous driving has become a blooming topic among car makers and research centers all across the globe in the past years since the announcement of Google's self-driving car in 2010. The demonstration of Google's car ability to autonomously drive on highways and urban areas changed many people's minds in the automotive industry, creating a new cohort of what could be coined as self-driving believers. Since then, the interest of car makers in self-driving has not ceased to grow and, as a matter of fact, autonomous driving developments and publications have soared worldwide. Despite rapid technological development, a number of issues, not only legal, have still to be seriously addressed before autonomous cars can robustly, safely, and efficiently circulate and mix with manually-driven vehicles in real traffic. On the one hand, experts in the field agree that autonomous vehicles will become more robust as they develop further cooperation capabilities. In other words, cooperation with traffic infrastructure, as well as with other vehicles, will make autonomous vehicles more robust and reliable, given that it is widely accepted that standalone self-driving is by far less robust than cooperative automated driving. On the other hand, self-driving cars must have the ability to predict other traffic agents' intentions, including other vehicles and Vulnerable Road Users (VRU), namely pedestrians and cyclists. This talk describes the design and development of DRIVERTIVE, a DRIVERless cooperATIVE vehicle, which aims to advance cooperative automation. DRIVERTIVE competed successfully in the Grand Driving Cooperative Challenge (GCDC) in the Netherlands in 2016. Twelve international teams participated in GCDC 2016 performing a number of cooperative manoeuvres in highways and intersections. In addition, the talk provides deep insights into the interaction with Vulnerable Road Users (VRU) by means of short-term intention recognition and accurate trajectory prediction as a means to go a step further in terms of safety and reliability, since it definitely makes the difference between effective and non-effective intervention. In contrast to trajectory-based approaches, the consideration of the whole pedestrian or cyclist body language has the potential to provide early indicators of the VRU intentions, much more powerful than those provided by the physical parameters of a trajectory. Experimental results show that accurate path prediction can be achieved at a time horizon of up to 1.0 s.

Biography: M Miguel Ángel Sotelo received the degree in Electrical Engineering in 1996 from the Technical University of Madrid, the Ph.D. degree in Electrical Engineering in 2001 from the University of Alcalá (Alcalá de Henares, Madrid), Spain, and the Master in Business Administration (MBA) from the European Business School in 2008. From 1993 to 1994, he held an Excellence Research Grant at the University of Alcalá, where he is currently a Full Professor at the Department of Computer Engineering and Vice-president for International Relations. In 1997, he was a Research Visitor at the RSISE of the Australian National University in Canberra. His research interests include Self-driving cars, Cooperative Systems, and Traffic Technologies. He is author of more than 200 publications in journals, conferences, and book chapters. He has been recipient of the Best Research



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Award in the domain of Automotive and Vehicle Applications in Spain in 2002 and 2009, and the 3M Foundation Awards in the category of eSafety in 2004 and 2009. He served as Auditor and Expert at FITSA Foundation for RTD Projects in the domain of automotive applications in 2004-2010. Miguel Ángel Sotelo has served as Project Evaluator, Rapporteur, and Reviewer for the European Commission in the field of ICT for Intelligent Vehicles and Cooperative Systems in FP6 and FP7. He was Director General of Guadalab Science & Technology Park (2011-2012) and co-founder and CEO of Vision Safety Technologies (2009-2015), a spin-off company established in 2009 to commercialize computer vision systems for road infrastructure inspection. He is member of the IEEE ITSS Board of Governors and Executive Committee. Miguel Ángel Sotelo served as Editor-in-Chief of the Intelligent Transportation Systems Society Newsletter (2013), Editor-in-Chief of the IEEE Intelligent Transportation Systems Magazine (2014-2016), Associate Editor of IEEE Transactions on Intelligent Transportation Systems (2008-2014), member of the Steering Committee of the IEEE Transactions on Intelligent Vehicles (since 2015), and a member of the Editorial Board of The Open Transportation Journal (2006-2015). He has served as General Chair of the 2012 IEEE Intelligent Vehicles Symposium (IV'2012) that was held in Alcalá de Henares (Spain) in June 2012. He was recipient of the 2010 Outstanding Editorial Service Award for the IEEE Transactions on Intelligent Transportation Systems, the IEEE ITSS Outstanding Application Award in 2013, and the Prize to the Best Team with Full Automation in GCDC 2016. At present, he is President-elect of the IEEE Intelligent Transportation Systems Society.

9th Workshop on PPNIV – Keynote

Cooperative Autonomous Driving and Interaction with Vulnerable Road Users

Miguel Ángel Sotelo

miguel.sotelo@uah.es

Full Professor


University of Alcalá (UAH)

SPAIN



9th Workshop on Planning, Perception, and Navigation for Intelligent Vehicles. Vancouver, Canada, 24th Sept. 2017

Content

- 
- ◆ **Motivation**
 - ◆ Autonomous Cooperative Driving
 - ◆ GCDC Results
 - ◆ Interaction with VRUs
 - ◆ Conclusions and Future Work

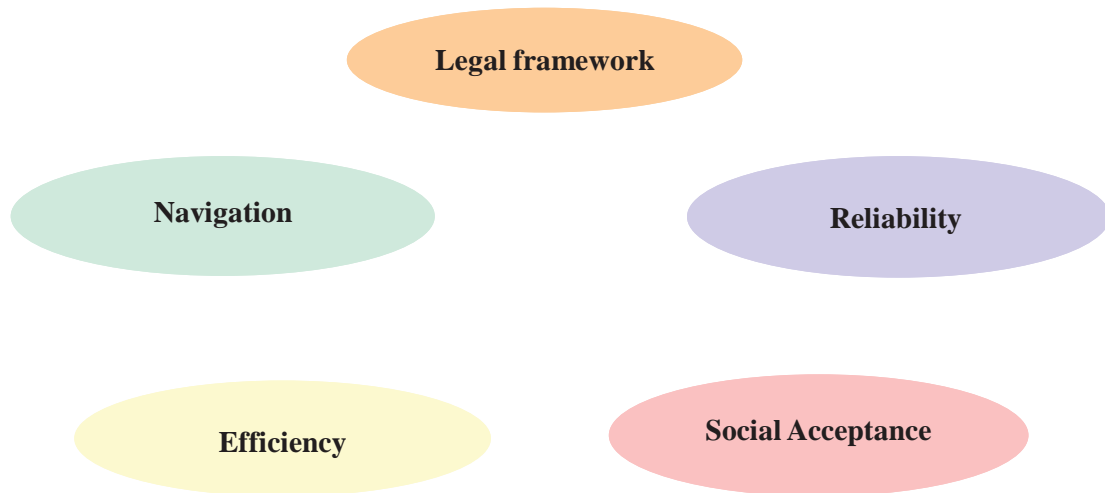


9th Workshop on Planning, Perception, and Navigation for Intelligent Vehicles. Vancouver, Canada, 24th Sept. 2017

2

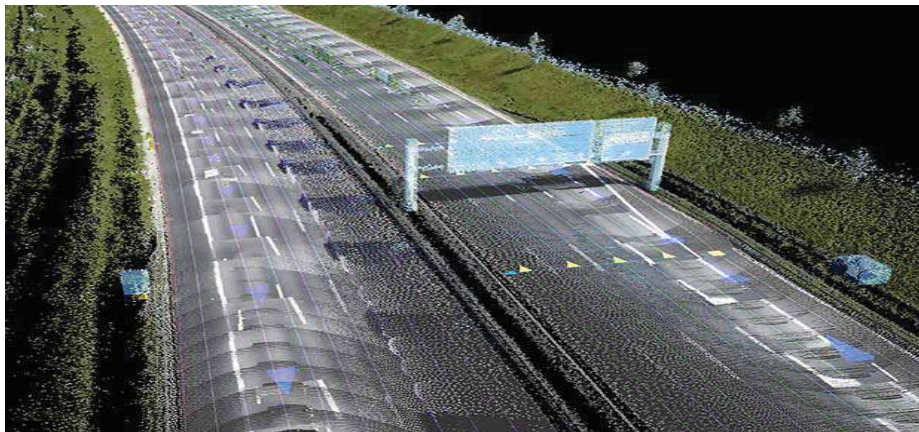
Motivation

- Despite the great development in the past years, there are still some major **limitations in Autonomous Driving**:



Limitations of Autonomous Vehicles

- **Navigation:**
 - **Enriched maps** are needed (2 Gb/Km).
 - International Consortia: BMW, Daimler, Audi (HERE).
 - Online data acquisition and map building.



Limitations of Autonomous Vehicles

- **Reliability:**

- Improvement in **sensorial capabilities (adverse weather)**.
- Development of **Cooperative Systems**.

- **Efficiency:**

- Human-like decision making and maneuvering.
- Emulation of human driving by means of **prediction of intentions** of other traffic agents, such as pedestrians and other vehicles.
 - There is a need for **enhanced cooperation and interaction capabilities**.



Content



- ◆ Motivation
- ◆ **Autonomous Cooperative Driving**
- ◆ GCDC Results
- ◆ Interaction with VRUs
- ◆ Conclusions and Future Work



Autonomous Cooperative Driving

- **Goal:**

- Increase reliability and efficiency of autonomous vehicles.

- **Techniques:**

- Cooperation with other vehicles (autonomously or manually driven) and with the infrastructure.
- Cooperation with VRUs (Vulnerable Road Users) by prediction their intentions and trajectories.

- **Limitations:**

- Strong dependency on penetration rate.



Autonomous Cooperative Driving

- **Initiatives:**

- European Commission: funding of research projects on Cooperative Systems and Autonomous Driving (FP7 and H2020).
- Grand Cooperative Driving Challenge (GCDC): International Competition on Autonomous Cooperative Driving in Helmond (The Netherlands) in 2011 and 2016.

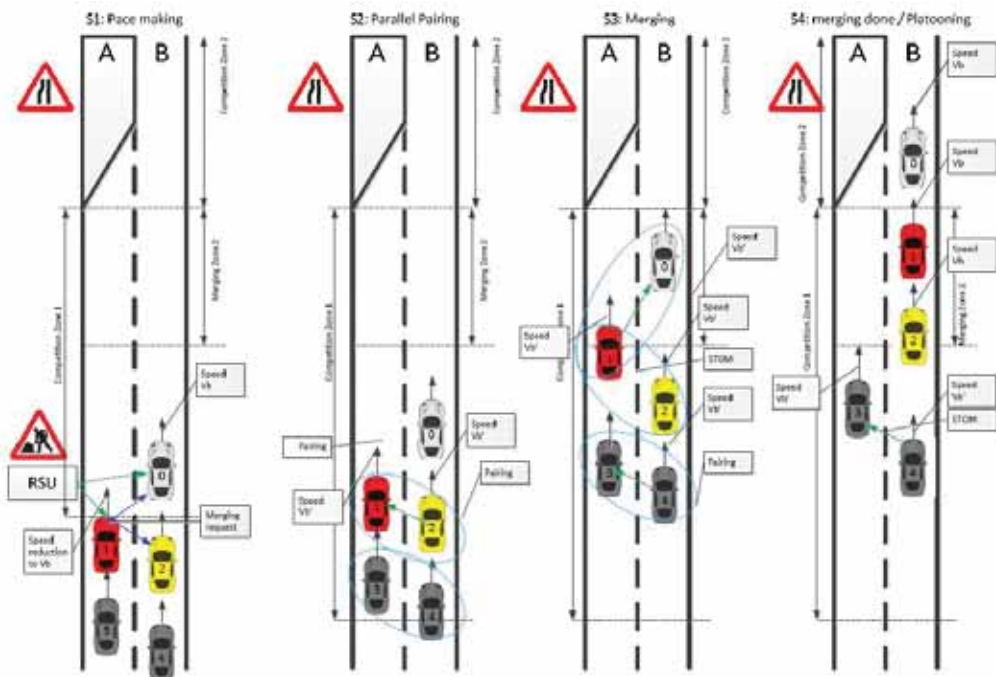
- **GCDC 2016 (three tests):**

- Platooning + Merging.
- Management of T-intersections.
- Management of emergency vehicles.



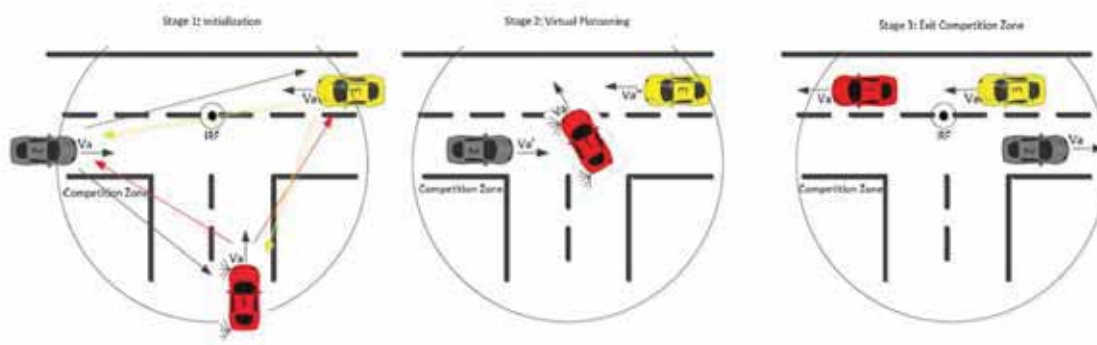
Autonomous Cooperative Driving

• GCDC 2016: Platooning + Merging



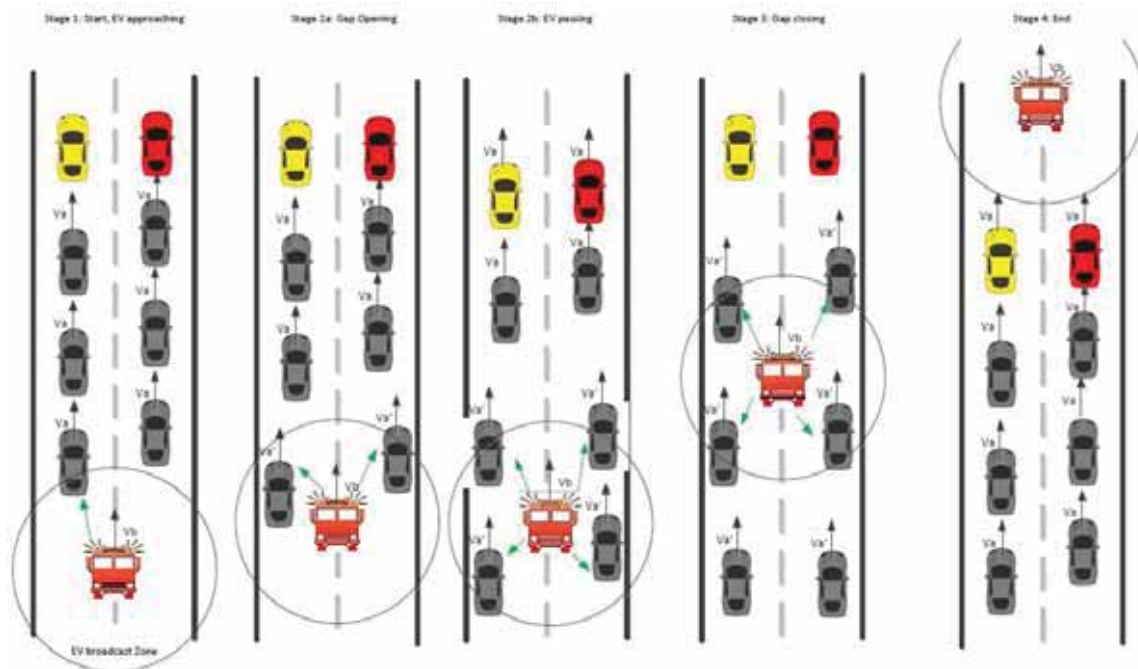
Autonomous Cooperative Driving

• GCDC 2016: Management of T intersections



Autonomous Cooperative Driving

• GCDC 2016: Management of emergency vehicles

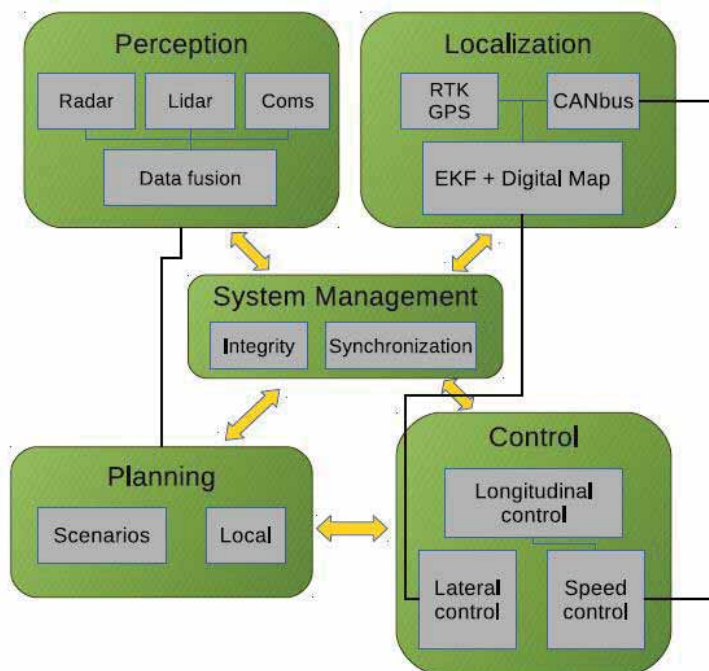


9th Workshop on Planning, Perception, and Navigation for Intelligent Vehicles, Vancouver, Canada, 24th Sept. 2017

11

Autonomous Cooperative Driving

• DRIVERTIVE – General Architecture



9th Workshop on Planning, Perception, and Navigation for Intelligent Vehicles, Vancouver, Canada, 24th Sept. 2017

12

Autonomous Cooperative Driving

• Data Fusion - Localization Example



(a) Toll plaza

(b) Under a bridge

Raw RTK GPS trajectory (red) and EKF trajectory (blue)



Autonomous Cooperative Driving

• Communications System (ITS G5 V2V standard)

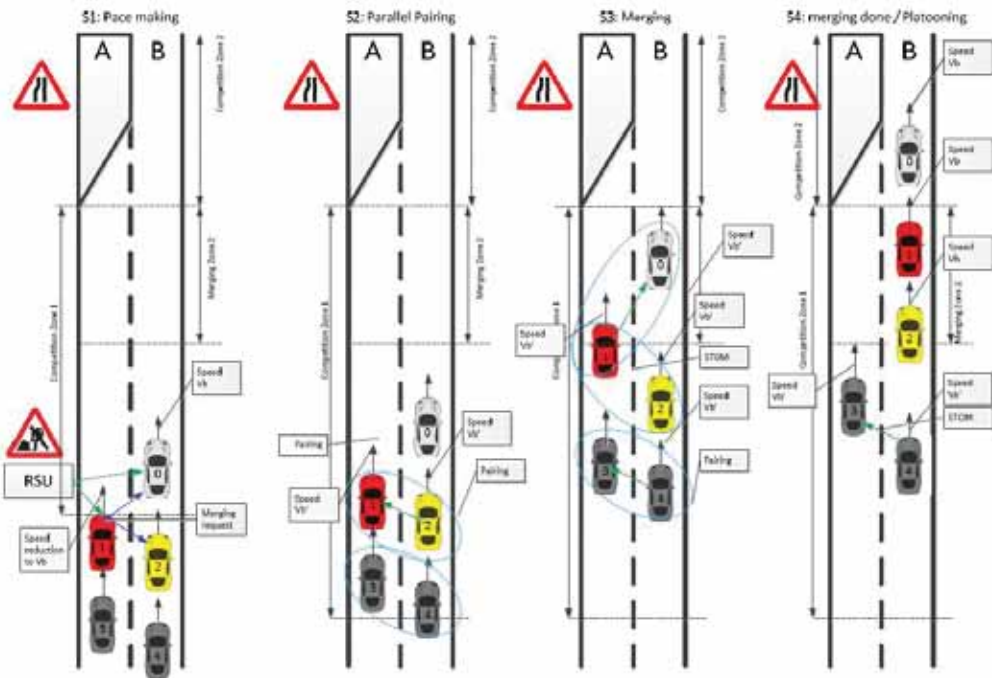
Messages types

- **CAM** (Cooperative Awareness Message): position, geometry and vehicles dynamics.
- **DENM** (Decentralized Environmental Notification Message): asynchronous messages from infrastructure or from other vehicles (e.g. emergency vehicles approaching).
- **iCLCM** (iGame Cooperative Lane Change Message): messages for interaction protocol in different scenarios.



Autonomous Cooperative Driving

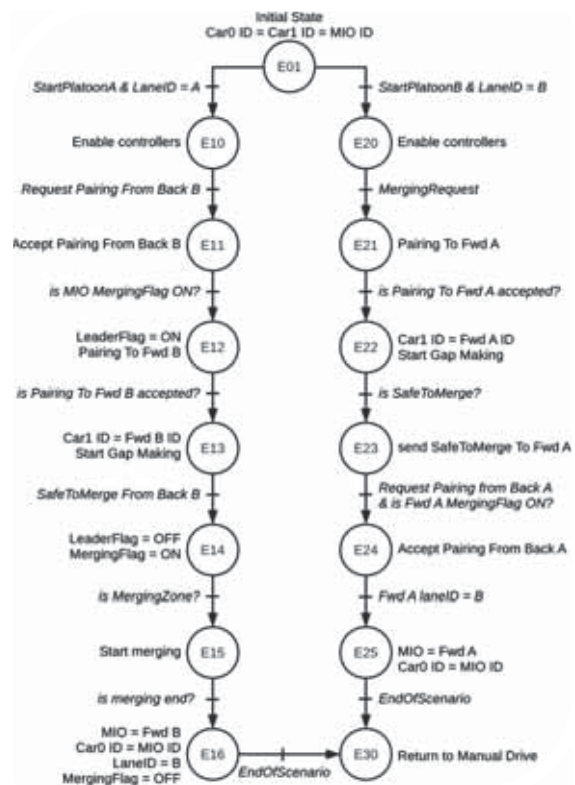
• Scenario 1: Platooning + Merging



Autonomous Cooperative Driving

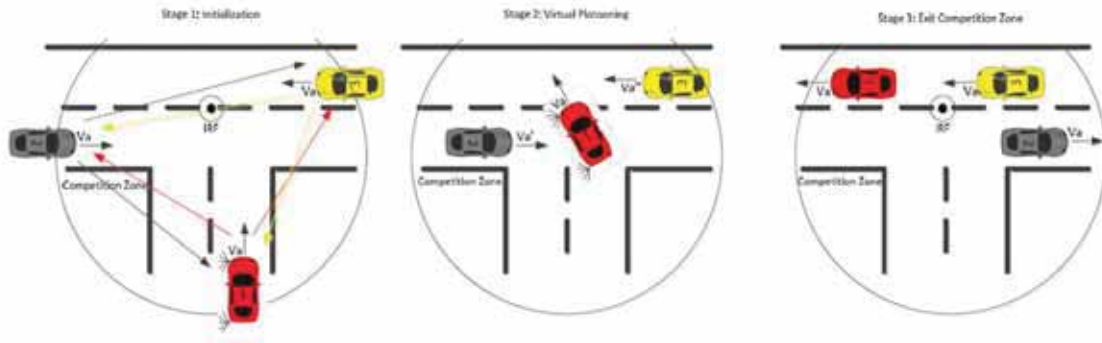
• Scenario 1: *Platooning + Merging*

Behavior on the left lane is different from that on the right lane



Autonomous Cooperative Driving

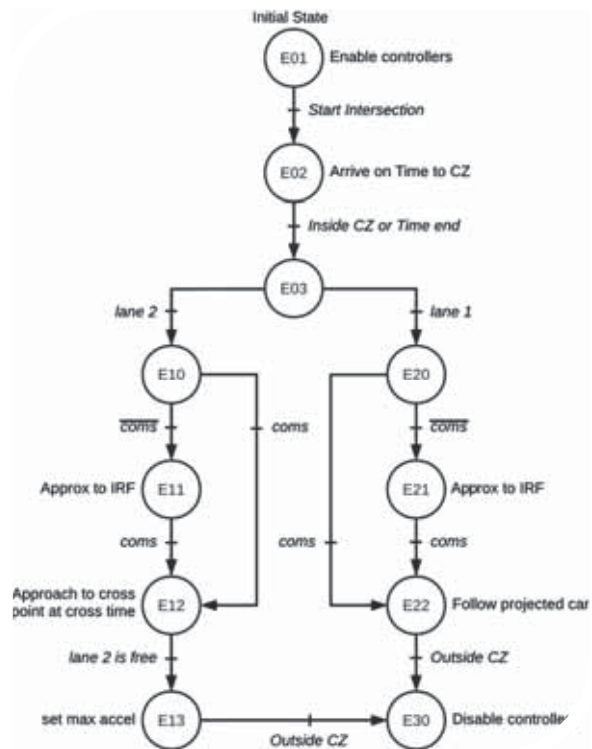
• Scenario 2: Management of T-intersections



Autonomous Cooperative Driving

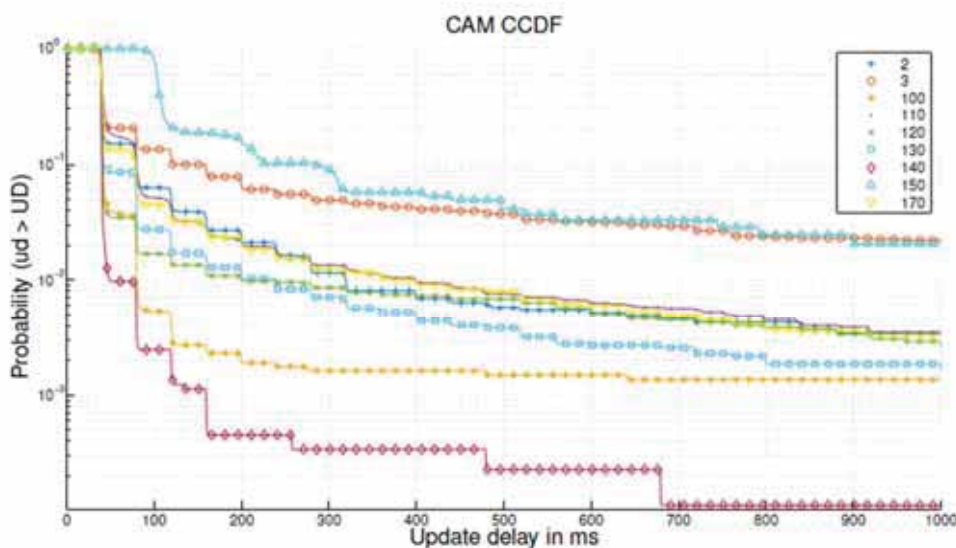
• Scenario 2: *T-Intersections*

A safety distance must be kept at all times w.r.t the preceding vehicle



Autonomous Cooperative Driving

Analysis of the communication channel (CCDF – Complementary Cumulative Distribution Function)

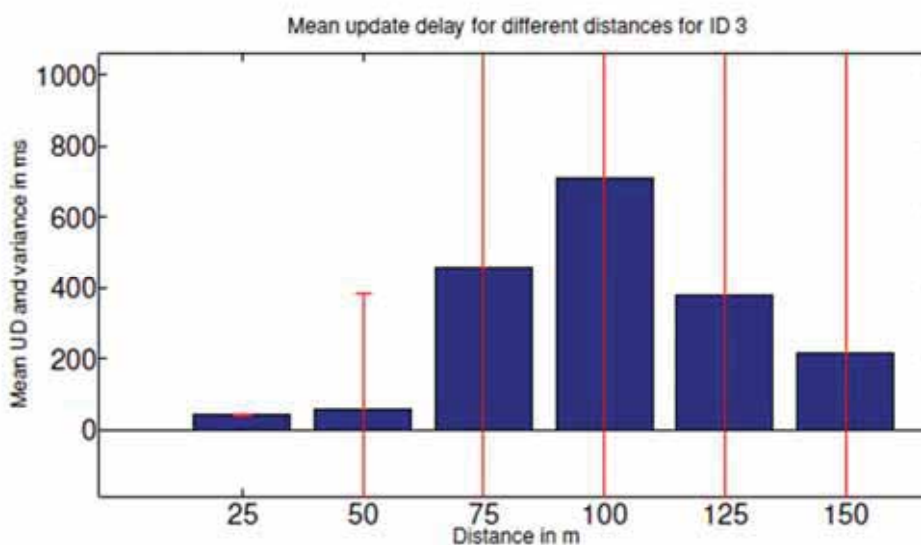


(b) CCDF for the participants in heat 2



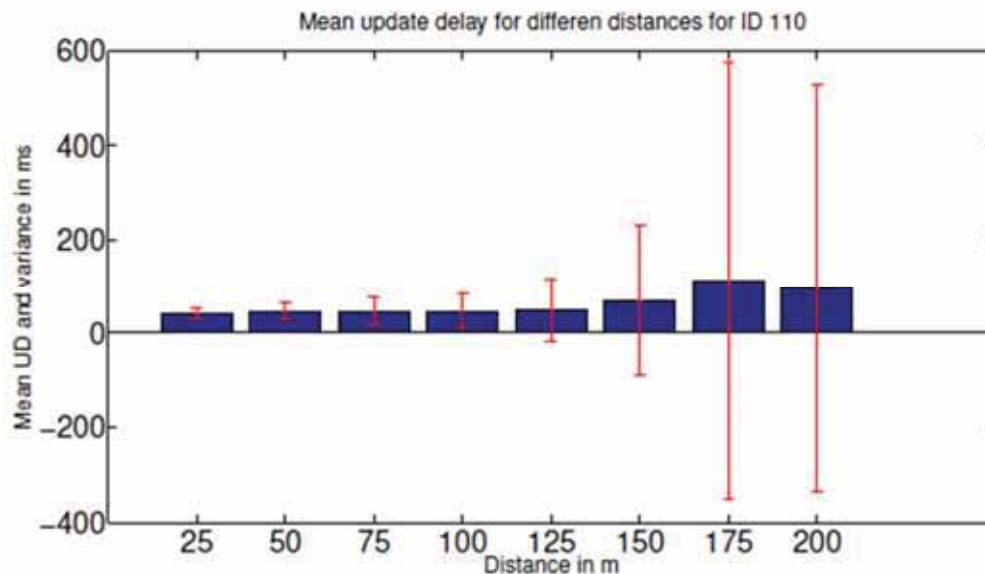
Autonomous Cooperative Driving

Mean and Variance of UD (Car)



Autonomous Cooperative Driving

Mean and Variance of UD (Truck)



Autonomous Cooperative Driving

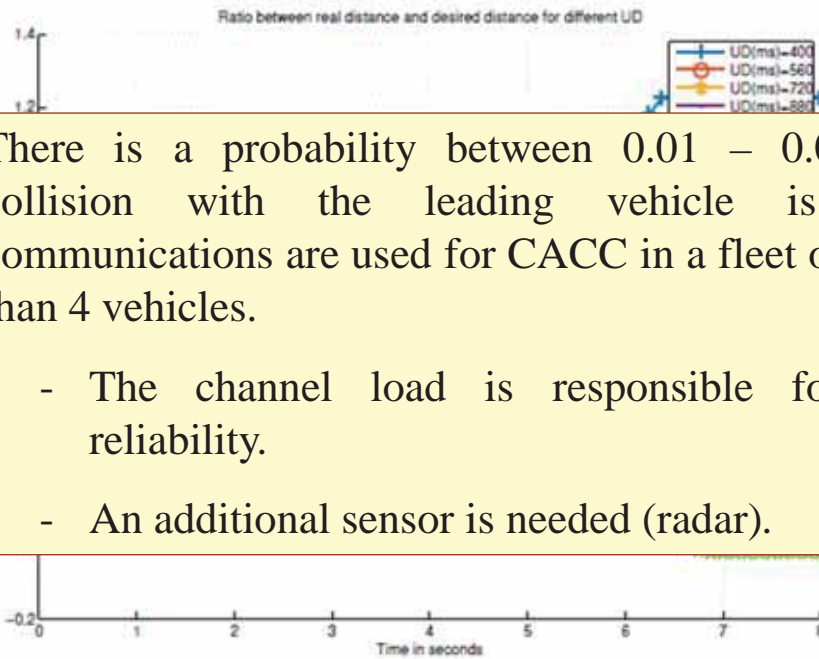
Analysis of the communication channel

- For some vehicles, the probability of large delays is significant (>10%).
- The UD degrades with distance.
- Occlusions have a strong effect on delays:
 - Trucks are less occluded given that their antennas are located at a height of 3 meters.
- **Other findings:** DCC in a highly congested channel is making some of the vehicles get stuck in Restrictive state and are not able to regularly access the channel.
 - CAM and DENM in GCDC at 25 Hz.



Autonomous Cooperative Driving

Effect of UD on Emergency Braking during CACC



- There is a probability between 0.01 – 0.001 of collision with the leading vehicle is only communications are used for CACC in a fleet of more than 4 vehicles.
 - The channel load is responsible for low reliability.
 - An additional sensor is needed (radar).

Content

- ◆ Motivation
- ◆ Autonomous Cooperative Driving
- ➔ ◆ **GCDC Results**
- ◆ Interaction with VRUs
- ◆ Conclusions and Future Work

Autonomous Cooperative Driving

- **DRIVERTIVE – University of Alcalá’s team**

- Autonomous Cooperative Vehicle (Velodyne, Radar, 3D Vision, Laser, DGPS, CANBus, Communications, fully automated)



9th Workshop on Planning, Perception, and Navigation for Intelligent Vehicles. Vancouver, Canada, 24th Sept. 2017

25



GCDC Results

DRIVERTIVE at GCDC 2016



9th Workshop on Planning, Perception, and Navigation for Intelligent Vehicles. Vancouver, Canada, 24th Sept. 2017

26



GCDC Results

DRIVERTIVE – Winner of the Prize to the Best Team with Full Automation in GCDC 2016



GCDC Results

GCDC 2016



Content

- ◆ Motivation
- ◆ Autonomous Cooperative Driving
- ◆ GCDC Results
- ➔ ◆ **Interaction with VRUs**
- ◆ Conclusions and Future Work



Motivation



The Big Problem With Self-Driving Cars Is People

And we'll go out of our way to make the problem worse

By **RODNEY BROOKS** Posted 27 Jul 2017 | 15:00 GMT



Illustration: Bryan Christie Design

Reading Body Language: A purely interpretive problem that self-driving cars cannot yet solve is that of making sense of the way people hold themselves and move. For instance, a self-driving car could not tell what any human driver could take in at a glance. The couple conversing animatedly near the curb [left] are not about to wander into traffic. If, however, one person turns away from the other and in the direction of the street, it means she's about to cross [right].



Motivation

• Pedestrian Path Prediction in the Automotive:

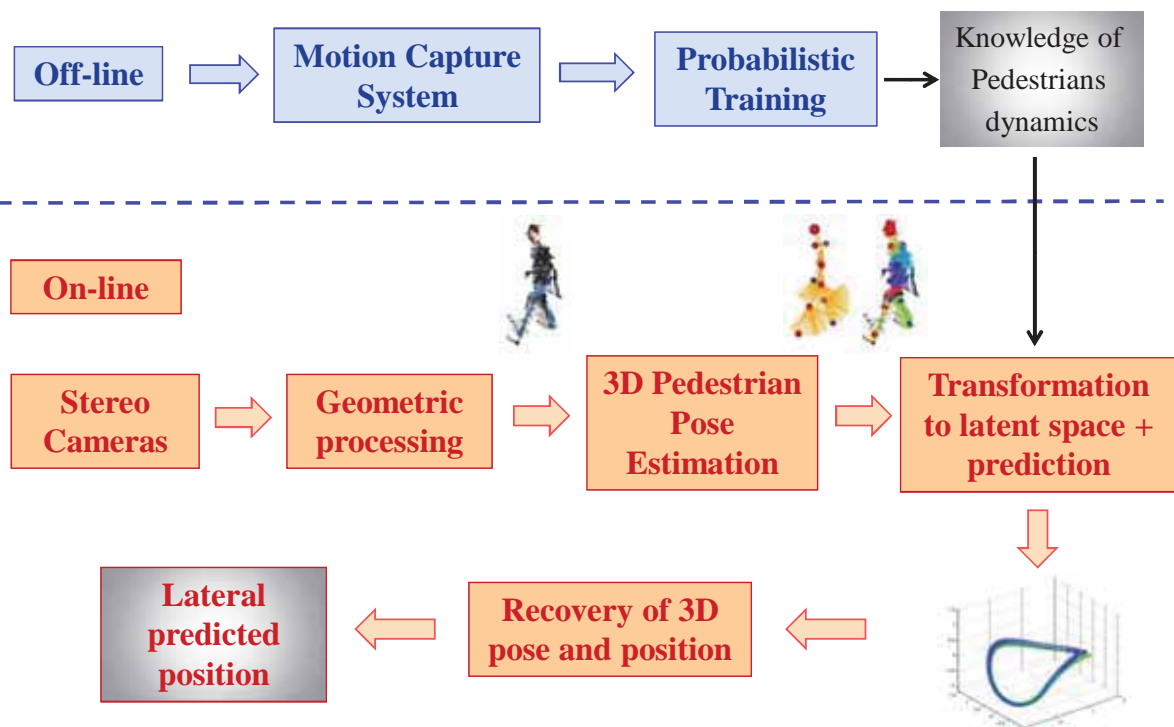
- Further improvement in state-of-the-art ADAS by means of action classification
 - Walking, Stopping, Running-in
- J... effective interventions
- Initiation of emergency braking 0.16 s in advance can potentially reduce severity of accidents injuries by 50%
- Early recognition of pedestrians stopping actions can provide more accurate last-second active interventions

Strong gains are expected in the performance and reliability of active pedestrian protection systems



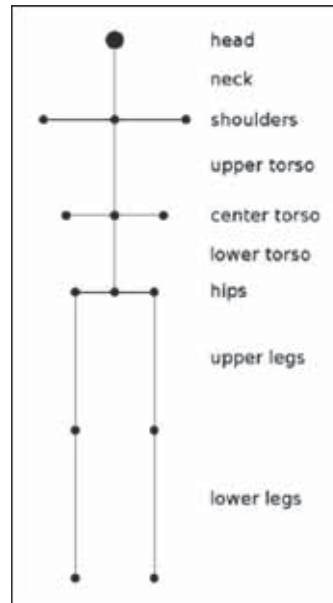
Proposed Approach

Global Scheme



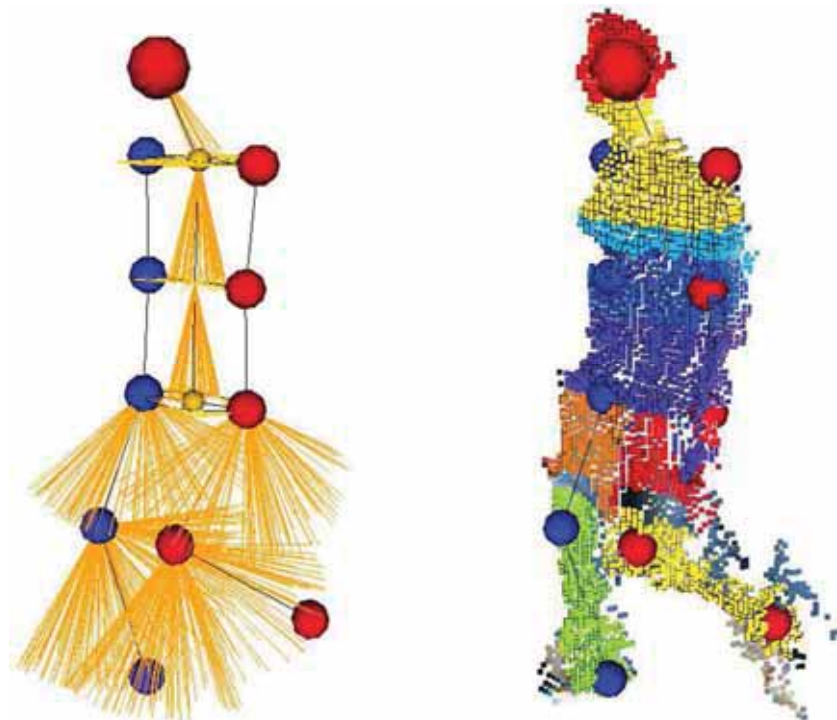
Pedestrian Pose Measurement

Pedestrian Skeleton considered in this research



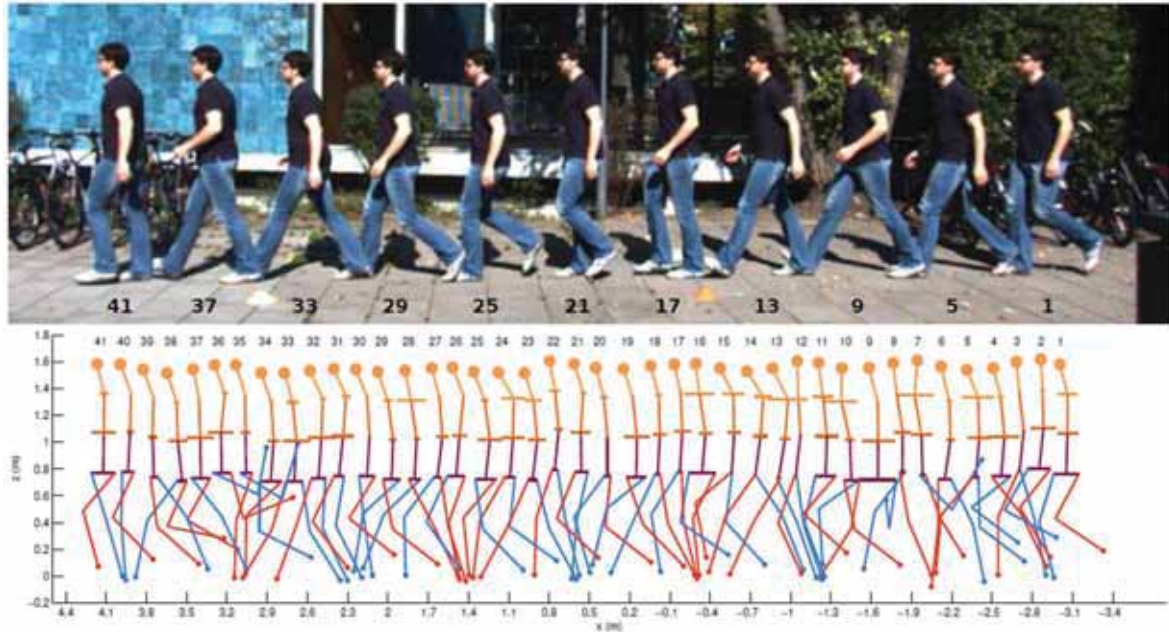
Pedestrian Pose Measurement

Method for Joints Extraction - Example



Pedestrian Pose Measurement

Method for Joints Extraction – Results

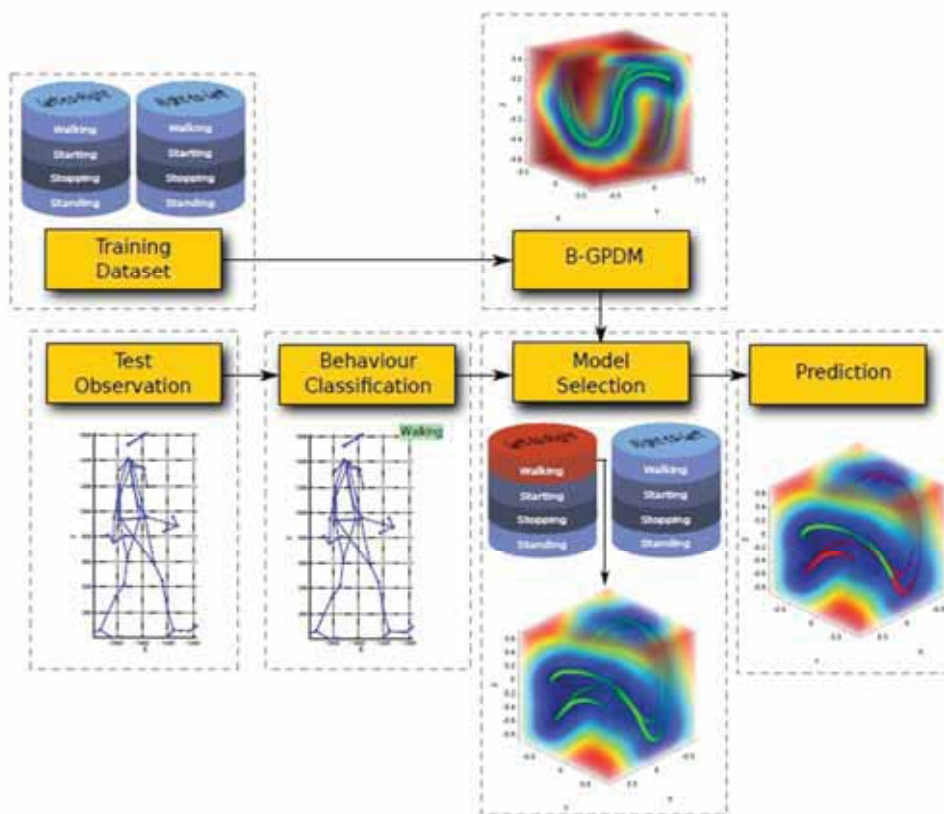


Pedestrian Pose Measurement

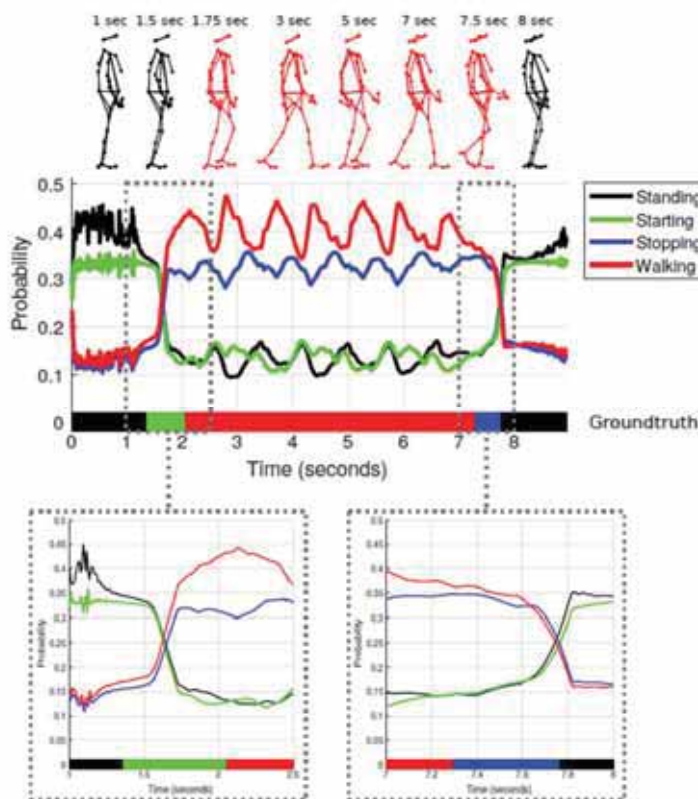
Body parts detection using Deep Learning



General Method - Overview



Activity Recognition - Example



Experimental Results

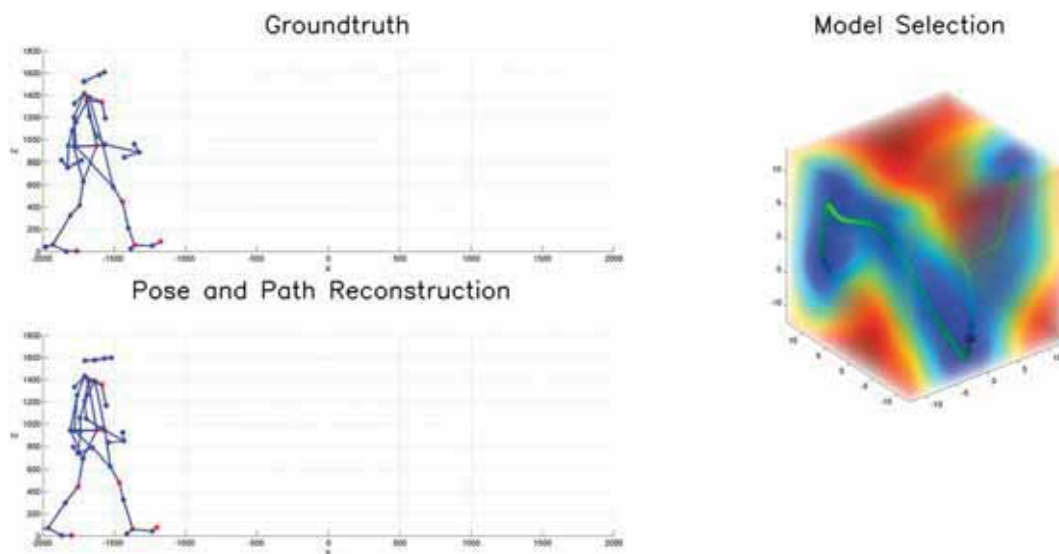
Detection Delay - Summary

Transition	Mean	Std	Median	Max	Min
Standing - Starting	57.98 ms	120.87 ms	50.00 ms	525.00 ms	-441.67 ms
Starting - Walking	-154.30 ms	183.66 ms	-208.33 ms	341.67 ms	-446.67 ms
Walking - Stopping	102.05 ms	157.86 ms	66.67 ms	416.67 ms	-450.00 ms
Stopping - Standing	89.84 ms	131.48 ms	58.33 ms	450.00 ms	-466.67 ms



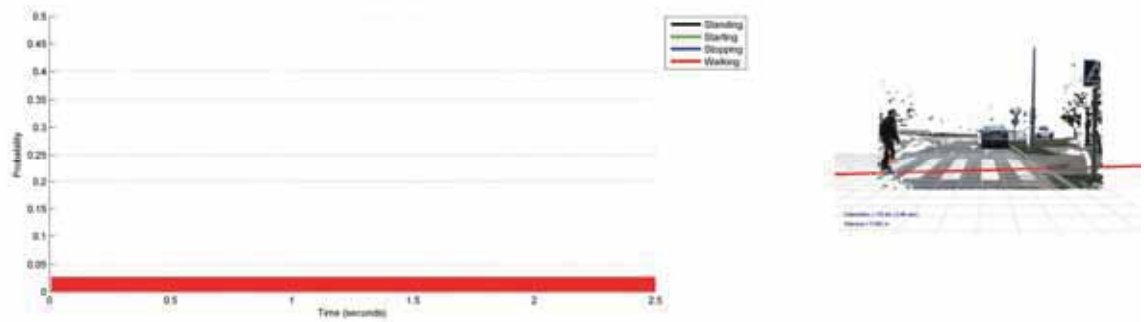
Experimental Results

Probabilistic Action Classification



Experimental Results

Probabilistic Action Classification



Experimental Results

Video sequence showing prediction results



Intelligent Interface with VRUs

GRAIL – GReen Assistant Interfacing Light



Intelligent Interface with VRUs

GRAIL – GReen Assistant Interfacing Light



Content

- ◆ Motivation
- ◆ Autonomous Cooperative Driving
- ◆ GCDC Results
- ◆ Interaction with VRUs
- ◆ **Conclusions and Future Work**



Conclusions and Future Work

Conclusions

- Autonomous Cooperative Systems will pave the way to the massive and robust deployment of self-driving cars.
- The V2V communication link is still a weakness that needs further attention from the scientific community in order to provide real-time and robust communication capability among large fleets of vehicles.
- Anticipating the intentions of other traffic participants, such as VRUs and vehicles, is essential for mimicking human drivers behavior.

Future Work

- Enhancement of V2V communication channel for large fleets of vehicles (antenna placement, frequency of data, etc.).
- Context-based action prediction using Probabilistic Graphical Models (Bayesian Networks) is under development for VRUs and vehicles.
 - Gaze direction, group behavior.



9th Workshop on PPNIV – Keynote

Cooperative Autonomous Driving and Interaction with Vulnerable Road Users

Thanks for your kind attention!

Miguel Ángel Sotelo

miguel.sotelo@uah.es

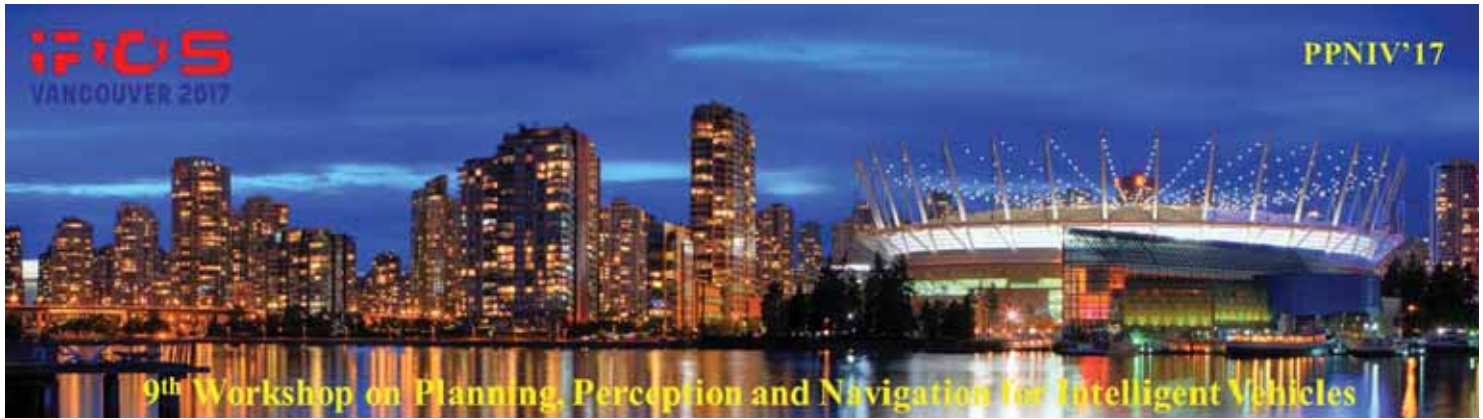
Full Professor

University of Alcalá (UAH)

SPAIN



9th Workshop on Planning, Perception, and Navigation for Intelligent Vehicles. Vancouver, Canada, 24th Sept. 2017



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session II

Segmentation & Reconstruction

- **Title: A new metric for evaluating semantic segmentation: leveraging global and contour accuracy**
Authors: Eduardo Fernandez-Moral, Denis Wolf, Renato Martins and Patrick Rives
- **Title: Multibody reconstruction of the dynamic scene surrounding a vehicle using a wide baseline and multifocal stereo system**
Authors: Laurent Mennillo, Eric Royer , Frederic Mondoty, Johann Mousainy and Michel Dhome



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

A new metric for evaluating semantic segmentation: leveraging global and contour accuracy

Eduardo Fernandez-Moral¹, Renato Martins¹, Denis Wolf², and Patrick Rives¹

Abstract—Semantic segmentation of images is an important problem for mobile robotics and autonomous driving because it offers basic information which can be used for complex reasoning and safe navigation. Different solutions have been proposed for this problem along the last two decades, and a relevant increment on accuracy has been achieved recently with the application of deep neural networks for image segmentation. One of the main issues when comparing different neural networks architectures is how to select an appropriate metric to evaluate their accuracy. Furthermore, commonly employed evaluation metrics can display divergent outcomes, and thus it is not clear how to rank different image segmentation solutions. This paper proposes a new metric which accounts for both global and contour accuracy in a simple formulation to overcome the weaknesses of previous metrics. We show with several examples the suitability of our approach and present a comparative analysis of several commonly used metrics for semantic segmentation together with a statistical analysis of their correlation. Several network segmentation models are used for validation with virtual and real benchmark image sequences, showing that our metric captures information of the most commonly used metrics in a single scalar value.

I. INTRODUCTION

The problem of semantic segmentation consists of associating a class label to each pixel of a given image, resulting in another image of semantic labels, as shown in figs. 1a and 1b. This problem of image understanding is highly relevant in the context of mobile robotics and autonomous vehicles, for which accurate information of the objects in the scene may be applied for decision making or safe and robust navigation among others [1].

Semantic segmentation has seen a rapid progress over the past decade. Recent advances achieved by training different types of Convolutional Neural Networks (CNN) have improved notably the accuracy of state-of-the-art techniques [2], [3], [4], [5], [6], [7], [8]. Among the many CNN architectures available, convolutional encoder-decoder networks are particularly well adapted to the problem of pixel labeling. The encoder part of the network creates a rich feature map representing the image content and the decoder transforms the feature map into a map of class probabilities for every pixel of the input image. Such operation takes into account the pooling indices to upsample low resolution features into the original image resolution. The advantages of this class of network were presented in [5], [6]. The approach in [6] was later extended to a Bayesian framework in [7] to provide the

¹Lagadic team. INRIA Sophia Antipolis - Méditerranée. 2004 Route des Lucioles - BP 93, 06902 Sophia Antipolis, France. Email: eduardo.fernandez-moral@inria.fr, renato-jose.martins@inria.fr, patrick.rives@inria.fr

²University of Sao Paulo - ICMC/USP, Brazil. denis@icmc.usp.br

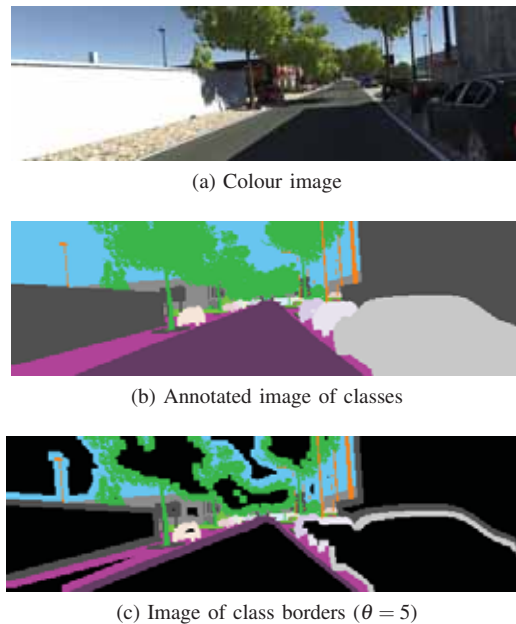


Fig. 1: Extraction of class borders from an annotated image of labels from the Virtual KITTI dataset [12].

probabilities associated to the pixel labels. Apart from end-to-end CNNs, Conditional Random Fields (CRFs) have also been used for scene semantic segmentation [9], [3], [10]. In [11], a CNN model is used to extract features which are feed to a Support Vector Machine-based CRF to increase the accuracy of image segmentation.

The recent availability of 3D range sensors and RGB-D cameras has also been exploited for semantic segmentation [13], [2], [14], [8]. An initial exploration of adding geometric information besides color (e.g., depth images) was addressed in [13], but the global accuracy improvement was marginal. Later, [2] presented an approach where depth information is encoded into images containing horizontal disparity, height above the ground and angle with gravity, which outperforms previous solutions using raw depth for indoor scenes. A different strategy for the same problem is presented in [8], which proposes to fuse depth features and color features in the encoder part of an encoder-decoder network. Another CNN-based approach for joint pixel-wise prediction of semantic labels, depth and surface normals was presented in [15].

The appearance of public datasets and benchmarks for semantic segmentation, both from virtual and real scenarios

[16], [12], [17], facilitates the comparison of solutions, and promotes the standardization of comparison metrics. Still, the choice of the most appropriate metrics to evaluate semantic segmentation is a problem itself, which gains relevance with the increase of performance and complexity of semantic segmentation techniques.

A. Contribution

In this paper, we investigate the problem of finding a single accuracy metric that accounts for both global pixel classification and good contour segmentation. We propose a new metric based on [18] and [19] which makes use of the Jaccard index to account for boundary points with a candidate match belonging to the same class in the target image. As we show in our experiments, this metric blends the characteristics of the Jaccard index (which is the *de facto* standard in semantic segmentation) and the border metric BF in a simple formulation, thus allowing to compare easily the outputs of different segmentation solutions.

B. Outline

The remainder of the paper is organized as follows. Section II-A reviews related works. In section II-B, we introduce the traditionally used segmentation evaluation metrics and their limitations. Section III describes our proposed metric. We present the different CNN architectures and the experimental results in section IV, considering simulated and real benchmark image sequences, such as the virtual KITTI and KITTI. Finally, in section VI, we draw conclusions and highlight future improvements and perspectives.

II. SEMANTIC SEGMENTATION METRICS

In this section, we review some recent related works and the background on commonly used evaluation metrics for semantic segmentation.

A. Related works

Comparing the accuracy of different semantic segmentation approaches is commonly carried out through different global and class-wise statistics, such as, global precision, class-wise precision, confusion matrix, F-measure or the Jaccard index (also called “intersection over union”). These metrics are described in more detail in section II-B. Global metrics like the precision may be a good indicator to evaluate different solutions when the different semantic categories have a similar relevance (both in terms of frequency of appearance and practical importance). But this is not the case in most applications, where objects which have fewer pixels may be significantly more relevant than others (e.g., “traffic light” or “cyclist” classes versus the “sky” in the context of autonomous vehicles). On the other hand, class-wise metrics (e.g., [6], [8]) avoid the previous limitation, but computing accuracies for each class individually means that we cannot compare different segmentation solutions directly (without specifying quantitatively the relevance of each class). An alternative metric is to average the chosen class-wise metric m according to the total number of classes

n (e.g., $\bar{m} = \sum_{i=1}^n m_i/n$). This class-wise average is less affected by imbalanced class frequencies than global metrics.

Another relevant aspect when evaluating segmentation approaches is to measure the quality of the segmentation around class contours. [20] proposes to measure the ratio between correct and wrong classified pixels in a region surrounding the class boundaries, instead of considering all image pixels. Other contour-based metrics include the Berkeley contour matching score [18], the boundary-based evaluation [21] and the contour-based score [19]. All these measures are based on the matching between the class boundaries in the ground truth and the segmented images. [21] computes the mean and standard deviation of a boundary distance distribution between pairs of boundary images. [18] computes the F1-measure from precision and recall values using a distance error tolerance θ to decide whether a boundary point has a match or not. [19] proposes an adaptation of [18] to multi-class segmentation, where the score (BF) is computed as the average of F_1 scores over the classes present in the segmented image.

The trade-off between global and contour segmentation is an important issue since both: a high rate of correctly labeled pixels and a good contour segmentation are desirable. For instance, in the context of autonomous navigation, we are interested in segmenting accurately the borders of the road and sidewalk in order to delimit the navigable space for each agent. In [19], the authors suggest to use both the Jaccard index and BF as accuracy metrics to capture different aspects of the segmentation quality (global and contour). However, when the problem consists in ranking different segmentation approaches based on their results, it is required to rely on a single measure so that different solutions can be directly compared. This problem is highly relevant, for instance, while using CNNs for semantic segmentation, because we are often interested in finding the set of hyperparameters which produce the best accuracy. This requires the comparison of multiple models using a single score. Besides, accuracy metrics which are also influenced by the quality of boundaries are interesting as loss functions to train the segmentation models.

B. Standard accuracy metrics

This section describes the most common metrics used for semantic segmentation. For reference, a general analysis of accuracy metrics for classification tasks can be found in [22].

The “accuracy”, or the ratio of the correctly classified elements over all available elements can be calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \quad (1)$$

whose notation is detailed in table I.

The “precision”, or positive predictive value (PPV), is the relation between true positives and all elements classified as positives:

$$Precision = \frac{TP}{TP + FP}. \quad (2)$$

TABLE I: Class confusion matrix and notation.

		Predicted class	
		Positive	Negative
True class	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

The ‘‘Recall’’, or true positive value (TPV), is the relation between true positives and all positive elements:

$$Recall = \frac{TP}{TP + FN}. \quad (3)$$

The F-measure [23] is a widely used metric to evaluate classification results, which consists of the harmonic mean of precision (2) and recall (3) metrics:

$$F_\beta = \frac{(\beta^2 + 1)TP}{(\beta^2 + 1)TP + \beta^2 FN + FP} \quad (4)$$

where β is scaling between the precision and recall. Considering $\beta = 1$, leads to the widely used F1-measure:

$$F_1 = \frac{2TP}{2TP + FN + FP}. \quad (5)$$

Another common metric to evaluate the results of classification is the Jaccard index (JI):

$$JI = \frac{TP}{TP + FN + FP}. \quad (6)$$

Global accuracy metrics are not appropriate evaluation measures when class frequencies are unbalanced, which is the case in most scenarios both in real indoor and outdoor scenes, since they are biased by the dominant classes. To avoid this, the metrics above are usually evaluated per-class, and their result is averaged over the total amount of classes.

The confusion matrix (C), is a squared matrix where each column represents the instances in a predicted class while each row represents the instances in an actual class. Thus, a value C_{ij} represents the elements of the class i which are classified as the class j :

$$C_{ij} = |S_{gt}^i \circ S_{ps}^j| \quad (7)$$

where S_{gt}^i and S_{ps}^j are the binarized maps of the ground truth class i and predicted class j respectively, (\circ) represents the element-wise product and ($|\cdot|$) is the L1 norm. Note that the confusion matrix is also useful to compute the above metrics in a class-wise manner, e.g.:

$$JI^k = \frac{C_{kk}}{\sum_{i=1}^n C_{ik} + \sum_{j=1}^n C_{kj} - C_{kk}}. \quad (8)$$

III. A NEW METRIC FOR SUPERVISED SEGMENTATION

This section describes a new metric for supervised segmentation which measures jointly the quality of the segmented regions and their boundaries. Our metric is inspired by the BF score presented in [19], which is defined as follows. Let’s call B_{gt}^c the boundary of the binary map of the S_{gt}^c of class c in the ground truth and likewise, B_{ps}^c for

its predicted segmentation. For a given distance threshold θ , the precision for class c is defined as:

$$P^c = \frac{1}{|B_{ps}^c|} \sum_{x \in B_{ps}^c} [[d(x, B_{gt}^c) < \theta]] \quad (9)$$

and the recall

$$R^c = \frac{1}{|B_{gt}^c|} \sum_{x \in B_{gt}^c} [[d(x, B_{ps}^c) < \theta]] \quad (10)$$

with $[[\cdot]]$ the Iversons bracket notation, where $[[z]] = 1$ if $z = \text{true}$ and 0 otherwise, and $d(\cdot)$ the Euclidean distance measured in pixels. The F_1^c measure for class c is given by:

$$BF^c = F_1^c = \frac{2 \cdot P^c \cdot R^c}{P^c + R^c}. \quad (11)$$

The BF in (11) has two main drawbacks. Firstly, it disregards the content of the segmentation beyond the threshold distance θ under which boundaries are matched. Secondly, the results of this metric depends on a discrete filtering of the distribution of boundary distances, so that the same score is obtained for different segmentations (with different perceptual quality) as far as the same amount of boundary pixels are within the distance θ . This is shown in table II, which shows different infra and over-segmentations with their corresponding scores.

In order to handle these shortcomings, we compute the distances from the boundary binary map to the binary map of the predicted segmentation $B_{gt}^c \rightarrow S_{ps}^c$ for a given class c to obtain the amount of true positives ($TP_{B_{gt}}^c$) and false negatives (FN^c). Similarly, we compute the distance from the boundary of the predicted segmentation to the binary map of the ground truth $B_{ps}^c \rightarrow S_{gt}^c$ for class c to obtain the amount of true positives ($TP_{B_{ps}}^c$) and false positives (FP^c). The total number of true positives is defined as ($TP^c = TP_{B_{gt}}^c + TP_{B_{ps}}^c$). Note that while the BF measure is based on boundary-to-boundary matches, our proposed BJ score is boundary-to-object. To avoid the second shortcoming, we propose to measure the values above with a continuous measure of the boundary distances, so that the following values are defined:

$$TP_{B_{gt}}^c = \sum_{x \in B_{gt}^c} z \text{ with } z = \begin{cases} 1 - (d(x, S_{ps}^c)/\theta)^2 & \text{if } d(x, S_{ps}^c) < \theta \\ 0 & \text{otherwise.} \end{cases} \quad (12)$$

$$FN^c = |B_{gt}^c| - TP_{B_{gt}}^c \quad (13)$$

$$TP_{B_{ps}}^c = \sum_{x \in B_{ps}^c} z \text{ with } z = \begin{cases} 1 - (d(x, S_{gt}^c)/\theta)^2 & \text{if } d(x, S_{gt}^c) < \theta \\ 0 & \text{otherwise.} \end{cases} \quad (14)$$

$$FP^c = |B_{ps}^c| - TP_{B_{ps}}^c \quad (15)$$

Then, the score for class c , which we call *Boundary Jaccard* (BJ^c) is defined according to the Jaccard index:

$$BJ^c = \frac{TP^c}{TP^c + FP^c + FN^c}. \quad (16)$$

This new score is not zero when the ground truth and the predicted segmentation for a given class have some

TABLE II: Examples of infra-segmentation and over-segmentation of a pedestrian from the Cityscapes dataset. The ground truth corresponds to figure in the center.

0	.12	.45	.64	.86	← JI →	.88	.77	.66	.54	.30
0	0	0	0	.99	← BF →	.99	0	0	0	0
0	.20	.46	.47	.77	← BJ →	.79	.64	.50	.50	.48

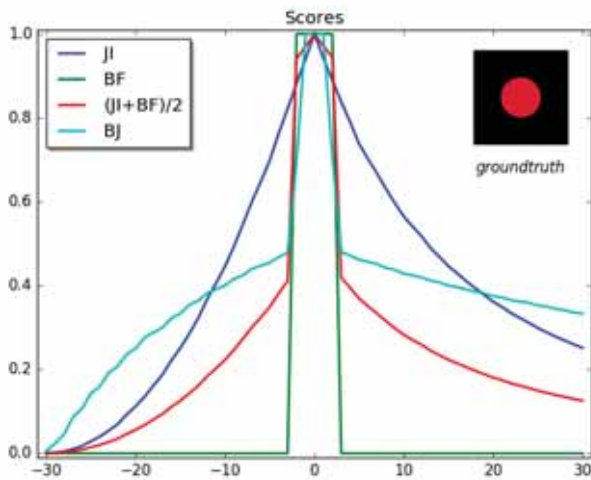


Fig. 2: Per-class scores of the segmented circle (top-right) for different levels of infra/over segmentation. The parameter θ is set to 4 pixels for both BF and BJ, which corresponds to 0.0075 of the image diagonal.

overlapping ($|S_{gr}^c \cup S_{gr}^c| > 0 \Rightarrow BJ^c > 0$). This behavior is similar for the metric JI^c but not for BF^c . On the other hand, the BJ^c score increases when the boundaries of ground truth and predicted segmentation get closer, like for BF^c , but with a more continuous behavior than the latter. Figure 2 shows an example to illustrate the behavior of the metrics BJ^c , BF^c and JI^c for different levels of infra/over segmentation, as showed in table II.

Finally, in order to compute the per-image BJ score, we average the BJ^c scores over all the classes present either in the ground truth or in the predicted segmentation. The score for a given image sequence is obtained as the average of per-image BJ's over the number of images contained in the sequence. It is worth to mention that per-image scores are more interesting than scores obtained over the full dataset (i.e., where a single BJ^c score is computed) for several reasons, as discussed in [19]. To mention some of these: *i*) per-image scores reduce the bias *wrt.* very large objects, and *ii*) they allow the statistical analysis about the performance of different segmentation frameworks in different parts of the dataset.

IV. EXPERIMENTAL ANALYSIS OF ACCURACY METRICS

This section presents a number of qualitative and quantitative results showing the accuracy of different types of CNN trained and tested on the Virtual KITTI [12] and KITTI [17] datasets and the comparison of the different evaluation metrics. The results confirm that measuring accuracy in the neighborhood of class borders is useful to compare different solutions without the need to provide class weights. Furthermore, the proposed metric BJ is correlated with both JI and BF, i.e., it captures the performance of these two scores. Note that in the following experiments, we focus our attention to the point of evaluating different accuracy metrics as it's the aim of this paper, rather than evaluating the suitability of different network architectures to the problem of semantic segmentation in urban scenes.

A. CNN architectures for semantic segmentation from RGB-D data

Using color and depth information has proven to be useful for semantic segmentation [2], [14], [8]. However, it's not clear yet how these two types of data should be fed into the CNN, and which network architecture is optimal for the problem. Without trying to solve this problem, we just describe here several solutions in order to compare later the suitability of different accuracy statistics. The network models analyzed in the next section are FuseNet [8], SegNet [6], and some modified versions of the latter that we describe here.

We introduce a modification of the VGG16 topology [24] employed by SegNet (see fig. 3a) to obtain a more compact network which we call Compact Encoder Decoder Convolutional Neural Network (CEDCNN), which is illustrated in fig. 3b. This network model increases the number of parameters of the filters in each resolution to produce higher dimensional feature maps, and reduces the number of consecutive convolution filters (convolution+batch normalization+ReLU) to reduce the complexity and non-linearity of the model. We also employ a modification of SegNet which is similar to [14], called SegNet2, with two separate networks for color and geometric information, whose result is concatenated and filtered by an additional convolution layer as shown in figure 3c. In the same way as for SegNet2, we also modify our model CEDCNN to obtain a new network, called

CEDCNN2, with two different pipelines to extract feature maps from color and geometric information separately.

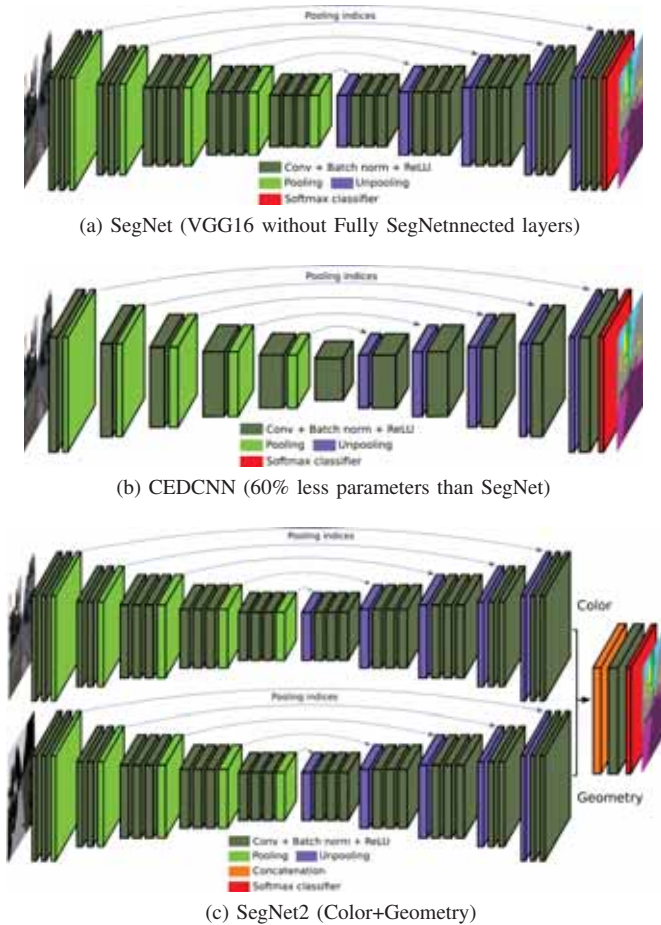


Fig. 3: CNN topologies employed in our experiments: a) SegNet, b) CEDCNN, c) SegNet2.

B. Comparison of different metrics

Firstly, we provide a qualitative analysis of the behavior of different metrics with infra-segmented and over-segmented objects, as shown in fig. 2. We produce synthetic segmentations of the ground truth of different object classes of interest, e.g., “traffic sign” or “pedestrian”. For instance, using the “pedestrian” class shown in table II, we produce infra-segmented objects by removing layers of labeled pixels of its boundary, such as the segmentations at the left of table II. Conversely, we produce over-segmented objects by adding layers of labeled pixels beyond the boundary, see the images at the right of table II. Figure 2 shows the score of different per-class metrics: JI, BF, the average of JI and BF, and BJ. The horizontal axis represents the amount of infra-segmentation (negative values) and over-segmentation (positive values) according to the number of 1-pixel layers removed or added to the ground truth, which is represented at the center of this graph, where all scores are 1.

We see that the Jaccard index has the most gradual behavior, since it depends only on the amount of pixels correctly

and wrongly classified. The BF score measures the quality of the segmented boundaries, it shows a discontinuous trend according to the threshold parameter used to distinguish inliers from outliers. The previous measures may be averaged to obtain a score that accounts for both: the number of pixels correctly labeled and the quality of contours of the segmentation. While the discontinuity of this metric is less severe than for BF, it is still something undesirable because the score depends highly on the threshold value θ . Finally, the BJ score shows a continuous behavior because its value depends on the distance, instead of a filter, so that the θ parameter has less influence on its value. The BJ score is higher than JI for infra-segmented objects, which is interesting because to avoid miss-classifications. The BJ score is close to 0.5 for over-segmented objects with bad contour segmentation. This effect is reasonable, since an over-segmentation is always preferable to a miss-classification. Besides, the effect of over-segmentations penalizes the BJ scores of the surrounding objects in the image.

C. Semantic segmentation of RGB and Depth on Virtual KITTI

This experiment makes use of the Virtual KITTI dataset [12] for training and testing different models for semantic segmentation. This dataset contains RGB, depth and labeled images with 13 classes: *sky, sidewalk, tree, vegetation, building, road, guard rail, traffic sign, traffic light, pole, car, van and truck*. It is composed of 5 virtual scenarios resembling those from the KITTI dataset [17], generated by simulating different illumination and weather conditions. Our training data is composed of 3846 observations chosen along different parts of the 5 scenarios contained in the dataset, scattering the selected images through the different sequences with different conditions (clone, fog, morning, overcast, rain and sunset). Each model is trained independently from scratch from the same training data. The test data used to produce the results shown in the following tables is composed of 1266 images selected from different sections of the same dataset.

First, we evaluate different ways to feed geometric information into SegNet, which is trained from images of different types: color (RGB), raw depth (D) encoded in one channel with 16 bits for centimeter precision, normal vectors plus depth (ND), and normal vectors plus elevation from the ground (NE). The images ND and NE are encoded as 3-channel images with 8 bits per channel, with 2 channels containing the first two components of the normal directions and the third channel containing depth or elevation, accordingly scaled to 8 bits [2].

Table III presents the accuracy measured as the recall (R), the mean recall of all classes, the mean JI and considering the metric BJ. The best scores are highlighted in bold. The first 5 rows of the table (white background) correspond to SegNet for different inputs. We observe that the combination of surface normal directions plus depth or elevation achieve the best results, with slightly better accuracy for ND. These outperform the accuracy obtained using RGB, raw depth, and the case of 4-channel RGBD input which concatenates RGB

with raw depth (with 8 bits for each color channel and 16 bits for depth)¹. Regarding the accuracy of the model SegNet2 (see fig. 3c), the use of input data from RGB-ND achieves the best results, for which all the global accuracy metrics indicate that it is the best model. Note that recall measured on the class borders are very close to the mean recall. In fact, both measures are quite similar because computing the recall only on class borders leverages the effect of unbalanced frequencies of the different classes, while being more stable to the presence of low-frequency (“rare”) classes with lower class-wise accuracy.

TABLE III: Semantic segmentation accuracy of SegNet and SegNet2 using color and geometric information (in %).

Model \ Metric	recall	m. R	m. JI	BJ
SegNet (RGB)	81.7	61.9	41.2	61.7
SegNet (D)	85.8	65.2	47.0	67.1
SegNet (ND)	88.6	70.2	51.1	69.8
SegNet (NE)	88.5	71.5	48.9	69.5
SegNet (RGBD)	78.1	64.1	41.8	60.7
SegNet2 (RGB-D)	88.5	71.0	49.4	70.5
SegNet2 (RGB-ND)	90.3	71.8	52.9	71.7

We analyze next other network architectures like FuseNet [8], together with the network topologies introduced in section IV-A: SegNet2, CEDCNN and CEDCNN2. Table IV shows the accuracy measured with the same global statistics of the previous table. For easier reference, this table also shows the results of SegNet for RGB and SegNet2 for RGB-ND in the two first rows. The results show that FuseNet, which was designed for semantic segmentation of indoor images from RGB-D data, achieves a performance comparable with SegNet. The authors of FuseNet argued in [8] that the relevant geometric features can be learned from raw depth by the CNN without the need of previous transformations. However, we observe a relevant improvement by comparing the results of FuseNet using RGB-D vs. RGB-ND, for which the surface directions contribute to improve the accuracy. For this case, the images are “virtually” acquired from a forward facing camera mounted in a car. Therefore, the surface directions have some invariants, such as the angle with gravity, that constitute a relevant source of information.

TABLE IV: Global accuracy of different types of networks using color and geometric information (in %).

Model \ Metric	recall	m. R	m. JI	BJ
SegNet (RGB)	81.7	61.9	41.2	61.7
SegNet2 (RGB-ND)	88.6	70.2	51.1	69.8
FuseNet (RGB-D)	85.2	65.9	45.8	64.9
FuseNet (RGB-ND)	88.1	64.6	47.2	68.9
CEDCNN (RGB)	88.8	72.8	48.6	70.5
CEDCNN2 (RGB-D)	90.1	79.7	60.0	77.5
CEDCNN2 (RGB-ND)	92.6	81.7	64.7	80.0

We remark that the different models achieve the best semantic segmentation depending on the class, while the best

¹Note that the virtual dataset has “perfect” geometry, which explains the high accuracy rates using only geometric information.

model overall (according to BJ) is CEDCNN2 with RGB-ND, which has a considerable better performance segmenting classes with lower frequencies, such as “traffic light” or “truck”, while the scores of large frequency classes like “sky”, “tree” or “road” are generally more stable across the different models. This fact is depicted in fig. 4 with confusion matrices for three different architectures. Note that if we need to choose between one of the FuseNet models, we need to consider the metric for all classes. Having unbalanced class frequencies has a great influence on the final score, because multi-resolution CNN are well suited by design to segment large homogeneous classes, but they are harder to train in order to achieve similar scores on low frequency classes, which sometimes are more important for many practical applications like for the case of autonomous driving.

Regarding the different accuracy metrics, we observe that the mean recall and the mean JI are less stable across the different experiments. This occurs because the accuracy of low frequency classes have a large variability even for similar models, and this variability is also reflected in their mean values. This effect is also observed in the normalized confusion matrices, see fig. 4, where the diagonal elements correspond to recall of each class, and where the JI for the i -th class is related to the values contained the i -th row and i -th column. On the other hand, BJ presents a more stable behavior for similar models, where even little changes on its value seem to be a good indicator to choose the best model according to the visualization of the predicted segmentation.

V. CORRELATION OF DIFFERENT METRICS

This section measures the correlation of the different metrics evaluated in the previous experiment. We compute the per-image score on the segmented test sequence of Virtual KITTI (RGB-ND) obtained with the model CEDCNN2, and measure the correlation of the different metrics for ranking the quality of each segmented image. We employ the Spearman’s rank correlation (ρ), which is a nonparametric measure of rank correlation, defined as the Pearson correlation coefficient between the ranked variables. It is used here to measure the statistical dependence between the ranking of different accuracy metrics. For a sample of size n , with the n raw scores X_i, Y_i , the Spearman’s rank correlation is defined as

$$\rho = \frac{cov(r_{gX}, r_{gY})}{\sigma_{r_{gX}} \sigma_{r_{gY}}} \quad (17)$$

where r_{gX}, r_{gY} are the ranks of the score distributions X, Y . Since we choose integer values for the rank, the formula is simplified to

$$\rho = 1 - \frac{6 \sum_{i=1}^n (r_{gX_i} - r_{gY_i})^2}{n(n^2 - 1)} \quad (18)$$

Table V shows the ranking correlations among metrics, where we can see that the BJ score is correlated to both JI and BF, showing that they capture similar information. Notice that the correlations with BJ are higher than the correlations among other pairs of scores.

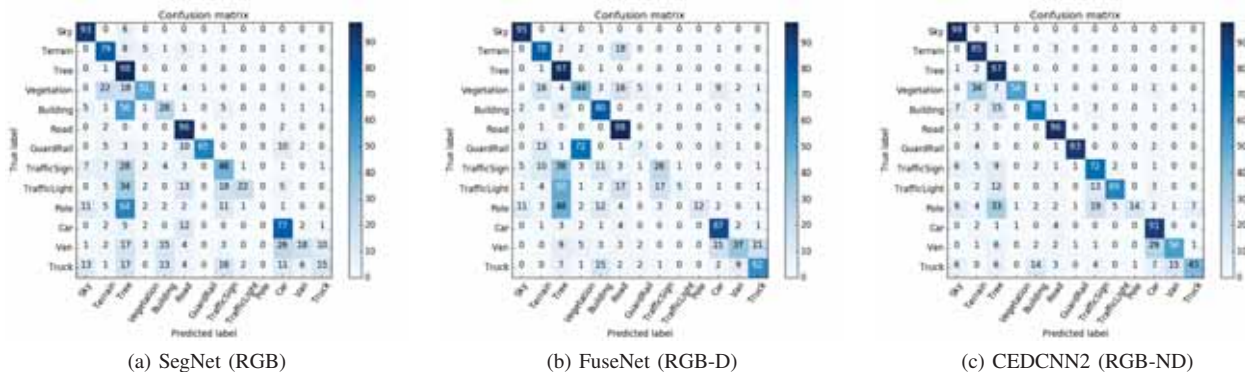


Fig. 4: Normalized confusion matrices (in %) of semantic segmentation in the real KITTI dataset with: a) SegNet (RGB), b) FuseNet (RGB-D) and c) CEDCNN2 (RGB-ND).

TABLE V: Spearman's rank correlation of different segmentation scores.

metric	JI	BF	(JI+BF)/2	BJ
JI	-	0.48	0.59	0.63
BF	-	-	0.68	0.65
(JI+BF)/2	-	-	-	0.73

VI. CONCLUSIONS

This paper addresses the problem of measuring the accuracy of semantic segmentation of images, which is an essential aspect when comparing different segmentation approaches. The global recall, mean recall and mean JI statistics have been traditionally employed to evaluate different image segmentation results, however, these metrics are not satisfactory enough when the classes frequencies are very unbalanced. We present a simple and efficient strategy to compute the recall on border regions of the different classes which leverages unbalanced frequencies, and is a good indicator to measure class segmentation. Our proposed metric encodes jointly the rate of correctly labeled pixels and how homeomorphic is the segmentation to the real object boundaries. We also present results for several different CNN architectures using two state-of-the-art benchmark datasets. Though we address this problem in the context of urban images segmentation, our results can also be extended to other contexts, like for indoor scenarios.

The research in this paper was partly motivated by the need of segmentation solutions with better segmentation of contours, for which traditional metrics were not suitable. In our future research, we plan to study how to give more importance to the segmentation of such contours during the training phase of the CNN and on obtaining optimal CNN designs for semantic segmentation of complex dynamic outdoor scenes.

REFERENCES

- [1] R. Drouilly, P. Rives, and B. Morisset, "Semantic representation for navigation in large-scale environments," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1106–1111.
- [2] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in *European Conference on Computer Vision*. Springer, 2014, pp. 345–360.
- [3] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.
- [4] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.
- [5] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1520–1528.
- [6] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *arXiv preprint arXiv:1511.00561*, 2015.
- [7] A. Kendall, V. Badrinarayanan, and R. Cipolla, "Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," *arXiv preprint arXiv:1511.02680*, 2015.
- [8] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, "Fusenet: Incorporating depth into semantic segmentation via fusion-based cnn architecture," in *Proc. ACCV*, vol. 2, 2016.
- [9] L. Ladický, P. Sturgess, K. Alahari, C. Russell, and P. H. Torr, "What, where and how many? combining object detectors and crfs," in *European conference on computer vision*. Springer, 2010, pp. 424–437.
- [10] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, "Conditional random fields as recurrent neural networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1529–1537.
- [11] F. Liu, G. Lin, and C. Shen, "Crf learning with cnn features for image segmentation," *Pattern Recognition*, vol. 48, no. 10, pp. 2983–2992, 2015.
- [12] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4340–4349.
- [13] C. Couprie, C. Farabet, L. Najman, and Y. LeCun, "Indoor semantic segmentation using depth information," *arXiv preprint arXiv:1301.3572*, 2013.
- [14] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, "Multimodal deep learning for robust rgb-d object recognition," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 681–687.
- [15] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2650–2658.
- [16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3213–3223.

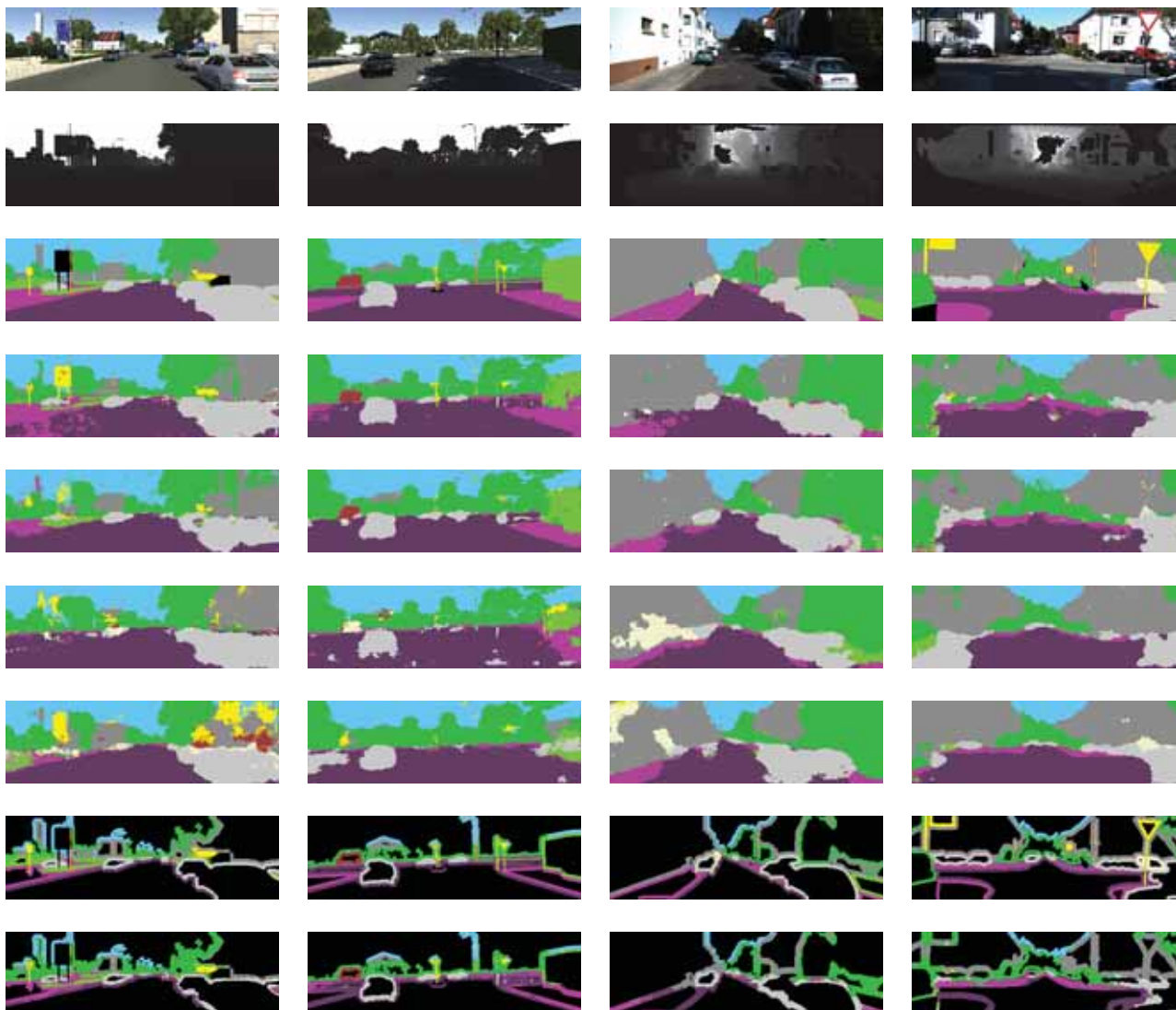


Fig. 5: Semantic segmentation produced by the different models. The first 2 columns correspond to test images from Virtual KITTI, while and the last 2 columns correspond to images from our KITTI test. The first row shows the input RGB image, followed by depth, groundtruth labels. Rows 4th to 7th show the segmentation produced by: CEDCNN2 (RGB-D), CEDCNN (RGB), SegNet2 (RGB-D) and SegNet (RGB) respectively. The 8th row shows the border regions from the ground truth, which are used to evaluate border recall and our metric in eq. (16). The 9th row shows the border precision of CEDCNN2 (RGB-D).

- [17] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.
- [18] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 5, pp. 530–549, 2004.
- [19] G. Csurka, D. Larlus, F. Perronnin, and F. Meylan, "What is a good evaluation measure for semantic segmentation?," in *BMVC*, vol. 27, 2013, p. 2013.
- [20] P. Kohli, L. Ladicky, and P. H. Torr, "Robust higher order potentials for enforcing label consistency," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.
- [21] J. Freixenet, X. Muñoz, D. Raba, J. Martí, and X. Cufí, "Yet another survey on image segmentation: Region and boundary information integration," *Computer VisionECCV 2002*, pp. 21–25, 2002.
- [22] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.
- [23] C. J. Van Rijsbergen, *Information Retrieval*. Butterworths, 1979.
- [24] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

Multibody reconstruction of the dynamic scene surrounding a vehicle using a wide baseline and multifocal stereo system

Laurent Mennillo^{*†}, Éric Royer^{*}, Frédéric Mondot[†], Johann Mousain[†] and Michel Dhome^{*}

^{*}Pascal Institute, Clermont Auvergne University, UMR 6602 - UCA / CNRS / SIGMA

63178 Aubière, France

[†]Technocentre RENAULT

78280 Guyancourt, France

Abstract—Multibody Visual SLAM has become increasingly popular in the field of Computer Vision during the past decades. Its implementation in robotic systems can benefit numerous applications, ranging from personal assistants to military surveillance to autonomous vehicles. While several practical methods use multibody enhanced SfM techniques and monocular vision to achieve scene flow reconstruction, most rely on short baseline stereo systems. In this article, we explore the alternative case of wide baseline and multi-focal stereo vision to perform incremental multibody reconstruction, taking inspiration from the increasingly popular implementation of heterogeneous camera systems in current vehicles, such as frontal and surround cameras. A new dataset acquired from such heterogeneous camera setup mounted on an experimental vehicle is introduced in this article, along with a purely geometrical method performing incremental multibody reconstruction.

I. INTRODUCTION

This article is related to the automotive industry and focuses on driving aid systems and autonomous navigation. Multibody SLAM techniques often rely on expensive and difficult-to-integrate sensors, such as lidar systems [23]. By contrast, digital video cameras have been extensively developed during the last decades, rapidly becoming small, efficient and inexpensive products. Most of the vehicles currently available dedicate these sensors to provide the driver a convenient visualization of the vehicle surroundings. Recently, more specific tasks involving video cameras (road sign and pedestrian detection, automatic emergency braking, line departure warning, blind spot monitoring, etc.) have been introduced. Besides, precisely calibrated cameras allow for the tridimensional reconstruction of an observed scene, which extends the potential applications of these systems. Some are straightforward, like visual odometry or visual simultaneous localization and mapping, but it is also possible to dynamically evaluate the road context related to this information to further enable the autonomous capabilities of a vehicle, which have been explored in applications such as scene understanding, obstacle avoidance or path planning [2], [4]. Moreover, behavior modeling of mobile objects could further improve the detection of dangerous situations (pedestrian crossing, brutal stop of another vehicle, excessive speed, bad road positioning, right of way violation,

etc.). While several practical methods use multibody enhanced SfM techniques and monocular vision to achieve scene flow reconstruction, most rely on short baseline stereo systems. In this article, we explore the alternative case of wide baseline and multifocal stereo vision to perform incremental multibody reconstruction, taking inspiration from the increasingly popular implementation of heterogeneous camera systems in current vehicles, such as frontal and surround cameras.

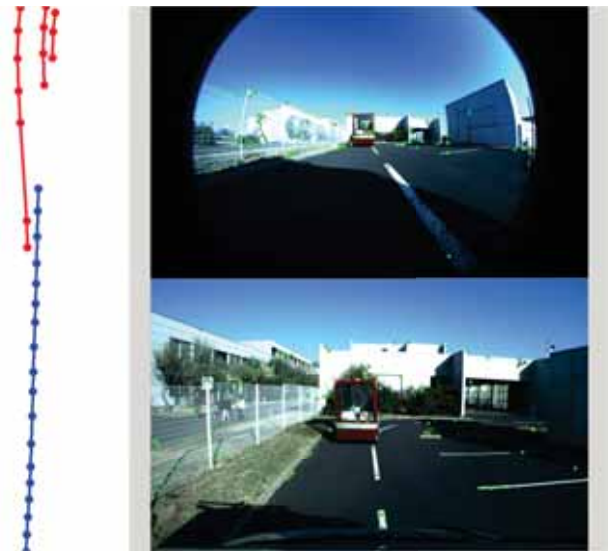


Figure 1. Example of trajectories reconstruction. In this sequence, the acquisition vehicle is following another moving vehicle. The blue trajectory is from the acquisition vehicle, while the red trajectories correspond to the red points on the moving vehicle visible in the two views on the right.

II. RELATED WORK

Intelligent vehicles today can be considered as the practice field of many computer vision algorithms. Indeed, research on the subject has led to several applications such as visual odometry [16] or visual simultaneous localization and mapping [3]. However, most methods focus on the reconstruction of static, rigid environments. Dynamic parts of the scene are often considered as outliers and filtered out using robust statistical

methods like RANSAC. Such approach could seem inappropriate in the context of driving aid systems and autonomous vehicles, as most hazardous traffic situations involve mobile objects, but one can then consider that the high complexity and computational cost of such dynamic reconstruction algorithms have been the limiting factors of their practical expansion.

Multibody VSLAM refers to the ensemble of techniques used to reconstruct and track the static and mobile objects of a dynamic scene in three dimensions with vision. However, while some techniques rely on global data optimization, this article focuses on incremental reconstruction, which allows its online use in actual moving vehicles. These incremental techniques can further be divided into two categories, mainly depending on the number of cameras used for reconstruction.

Monocular methods are the most challenging ones, in that they have to compensate the camera ego-motion parameters to retrieve the independent motion of each mobile object of the scene. Many incremental monocular methods [17], [8], [19] extend classical Structure-from-Motion theory [7] to the challenging case of dynamic scenes involving multiple rigid-body motions. The different elements to consider for such frameworks involve features matching and clustering based on their estimated motion, also known as subspace clustering [22], the tracking and independent reconstruction of these feature clusters with respect to their relative camera pose and finally the aggregation of all the reconstructed elements to scale.

The second category of methods used to perform multibody VSLAM involve multiple camera systems, generally under the form of identical stereo camera pairs which allow for dense reconstruction and segmentation of mobile objects using depth maps from optical flows [13], [18], [1], [24]. While short baseline stereo has been well studied in the context of autonomous navigation, it is not the case of multifocal and wide baseline stereo cameras pairs. The method presented in this article is intended to address this case on a heterogeneous multi-camera system.

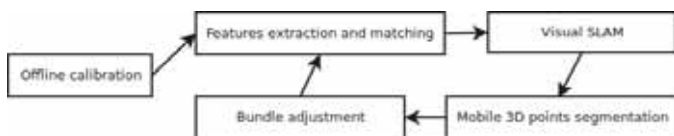


Figure 2. Overview of the framework used in our method.

III. FRAMEWORK

An overview of the framework presented in this article is shown in figure 2. After an initial offline step of intrinsic and extrinsic camera calibration following the method introduced in [9], feature points are then extracted, matched and undistorted for each frame using the unified camera model presented in [5] and then fed to the visual SLAM module, which estimates the ego-motion parameters of the multi-camera system. These parameters are then used to compute the multi-view geometric constraints of the segmentation process

which filters and reconstructs the static and mobile features. Following the SLAM and segmentation procedures, the reconstructed points and camera poses are further refined by two dedicated optimization steps minimizing the reprojection error with bundle adjustment.

A. Sparse feature extraction and matching

Dense feature matching from stereo camera pairs has been well studied for the case of dynamic scene reconstruction. These techniques often involve the use of dense flow fields to detect and segment the rigid-body motions of the scene [13]. By contrast, while obtaining disparity maps from wide baseline systems has proven achievable [21], the current methods are not appropriate for time-constrained scenarios. The approach used in this article, while conventional, produces accurate extraction and matching of sparse features in the case of wide baseline and multifocal stereo.

To account for the heterogeneous focal lengths of our system, the frames obtained from cameras with longer focal lenses are downsampled and slightly blurred to adjust for the different size (pixelwise) of the elements in the scene that are simultaneously seen by cameras with shorter focal lenses. These downsampled frames are then used for feature extraction.

The SIFT feature detector and descriptor [11] has been chosen for feature detection and description as it produces a large number of relatively stable points.

The feature extraction process is divided into three parts for each frame. SIFT feature detection is first performed on the entire frame to get an initial feature set. The frame is then divided into blocks of an n by n grid, while the features belonging to each block are grouped into clusters. The best features are finally retained for each cluster. This first part allows for a good feature repartition on the frame. The second part is designed to enhance the temporal detection of previously triangulated features. We used the Lucas Kanade method as introduced in [12] to track these features on consecutive frames and thus increase the chances to detect the same 3D point for a longer period of time. The last part finally merges the two sets of features, eliminating duplicates based on their respective euclidean distance. The result of this extraction process is a feature set $f_{i,t}$ for each frame, where $i \in 0 \dots m$ and $t \in 0 \dots n$ correspond respectively to the camera and time of observation.

The feature matching process between two sets $f_{i,t}$ and $f_{i',t'}$ then rely on two geometric constraints. A locality constraint L_c and the epipolar constraint E_c .

The locality constraint L_c is used for the *temporal matching* of features seen in frames acquired with the same camera at different times. This constraint allow for a feature $x \in f_{i,t}$ to be matched with a feature $x' \in f_{i',t'}$ if the euclidean distance d_E between x and x' is inferior to a threshold d_{L_c} . Each potential match $p(x, x')$ must then satisfy the following equation

$$p(x, x') \iff d_E(x, x') < d_{L_c}$$

The epipolar constraint E_C is used for the *stereo matching* of features seen in frames acquired simultaneously by different cameras with overlapping fields of view and whose extrinsic parameters are known beforehand. This constraint allow for a feature $x \in f_{i,t}$ to be matched with a feature $x' \in f_{i',t}$ if the euclidean distance to their respective epipolar lines l' and l is inferior to a threshold d_{E_C} . Each potential match $p(x, x')$ must then satisfy the following equation

$$p(x, x') \iff \begin{cases} d_E(x, l') < d_{E_C} \\ d_E(x', l) < d_{E_C} \end{cases}$$

where $l = F_{i,i'}^T x'$, $l' = F_{i,i'} x$ and $F_{i,i'}$ is the fundamental matrix between cameras i and i' .

Following these two constraints, when more than one potential match $p(x, x')$ exists for either feature in their respective sets, the best match $m(x, x')$ retained is the one for which the euclidean distance d_E , or L^2 , between each feature descriptor (not the distance in pixels) is minimal $\min(d_E(x, x'))$.

Finally, the multi-camera matching scheme allows each camera to be stereo matched with the other ones for which there is an overlapping field of view and temporally matched at consecutive times of observation. That last point is of crucial importance for the tracking of features, meaning that a feature must at least be matched once temporally at the current time of observation to be tracked in subsequent frames.



Figure 3. Reconstruction of the rigid environment generated by the visual SLAM module and its corresponding trajectory.

B. Visual SLAM

The visual SLAM module is independent of the following segmentation process proposed in this article. Its main purpose is to estimate the ego-motion parameters of the multi-camera system in order to efficiently compute the geometric constraints used in the segmentation process. The approach chosen is a bundle adjustment visual SLAM, as presented in [14], in opposition to filter based approaches such as [3] for its higher accuracy [20]. Briefly, the initial epipolar geometry is computed by the 5-point algorithm [15] with RANSAC for the first three frames and the subsequent poses are determined by camera resection [6], [10]. During this incremental process, the 3D points are reconstructed with the mid-point algorithm

and some sets of frames, referred as key frames, are selected for local optimization by bundle adjustment to further refine their respective camera poses and associated 3D points. A full sequence reconstruction and its associated trajectory, generated by the visual SLAM module, are shown in figure 3.

C. Mobile 3D points segmentation and tracking

A 3D point X must at least be associated with a couple of observations $(o_{i,t}^X, o_{i',t'}^X)$, each from a specific camera $i, i' \in 0 \dots m$ at a specific time $t, t' \in 0 \dots n$, for its reconstruction. These observations can either be *temporal* ($i = i' \wedge t \neq t'$) or *stereo* ($i \neq i' \wedge t = t'$) and correspond to feature matches $m(x, x')$ obtained from the feature matching module. A 3D point can also be associated with more than two observations, all of which form the set o^X of the observations associated with the 3D point X . One should note that at this point, all observations are retained from the feature matching module to allow for mobile object detection, contrary to most SLAM methods which eliminate the outliers that do not satisfy the main epipolar geometry of the scene. The objective of the mobile segmentation module is then to determine from these observations the class of their associated 3D point, which can either be *static* ($X \in S$), *mobile* ($X \in M$) or into the *outlier* class ($X \in O$).

1) *3D point consistency*: A 3D point is considered as *consistent* when it satisfies the consistency constraint C_c . This constraint specifies that the reprojection error of this point for all its observations is inferior to a certain threshold t_{C_c} , which translates as

$$C_c(o^X) \iff \forall o_{i,t}^X \in o^X, (o_{i,t}^X - P_{i,t}X) < t_{C_c}$$

where $P_{i,t}$ is the projection matrix of the i^{th} camera at time t . Incidentally, a *static* 3D point ($X \in S$) must be consistent for all of its associated observations.

2) *3D point mobility*: On the opposite, a *mobile* 3D point might not be consistent for all its temporal observations. However, each 3D point can be split temporally and must remain consistent for each of its temporalities, which allow for different positions of the point at different times. Then, the first mobility constraint M_{c1} specifies that

$$M_{c1}(o^X) \iff \forall t, \forall o_{i,t}^X \in o_t^X, (o_{i,t}^X - P_{i,t}X) < t_{M_{c1}}$$

where o_t^X is the set of observations associated to the point X at time t . Considering that the point X is moving, only *stereo* observations allow for its reconstruction at time t . There must then be at least two stereo observations $(o_{i,t}^X, o_{i',t}^X)$ for each temporality t . This leads to the second mobility constraint M_{c2} , which states that

$$M_{c2}(o^X) \iff \forall t, |o_t^X| \geq 2$$

Finally, the detection of a mobile point being only possible from several *temporal* observations, there must at least be two

temporal observations $(o_{i,t}^X, o_{i,t'}^X)$ in the set o^X . Hence the third mobility constraint which states that

$$Mc3(o^X) \iff \exists(o_{i,t}^X, o_{i,t'}^X) \in (o^X)^2, t \neq t'$$

In practice, while the minimum of temporal observations is two, a minimum of three has been used to mitigate false positives by ensuring that the trajectory of these observations is consistent (see III-C5), meaning that a mobile 3D point has to be tracked in at least three consecutive frames.

3) *Segmentation algorithm*: Using the consistency and mobility constraints, the segmentation process then proceeds with the following algorithm for each 3D point X to determine its class C (*static*, *mobile* or *outlier*).

Algorithm 1 Segmentation algorithm

Input: o^X

Output: class C of X : $(C = S) \vee (C = M) \vee (C = O)$

```

1: if  $(Cc(o^X))$  then
2:    $C = S$ 
3: else
4:   if  $(Mc1(o^X) \wedge Mc2(o^X) \wedge Mc3(o^X))$  then
5:      $C = M$ 
6:   else
7:      $C = O$ 
8:   end if
9: end if
10: return  $C$ 

```

Each 3D point is first checked for consistency and considered as *static* if consistent. If not, the point is further tested for mobility, in which case it is considered as *mobile* if all mobility constraints are satisfied and as an *outlier* if not. The outliers are then discarded at this point.

4) *3D point splitting for optimization*: Following the segmentation algorithm, each mobile point is then split temporally as a set of individual points X_t which correspond to the different positions of the point X at each temporality t . This step allows for a generic optimization of all 3D points regardless of their class ($C = S$ or $C = M$), which is performed on all mobile 3D points X_t by minimizing their reprojection error for all their observations o_t^X with bundle adjustment.

5) *Trajectory consistency*: As a final step and to further refine the segmentation, the trajectory of each mobile point X composed of the individual points X_t is checked for its consistency. Several constraints of smoothness for speed and changes in direction are used. The constraint for speed specifies that the euclidean distance allowed between each pair of consecutive points $(X_t, X_{t'})$ is comprised between $d_{min} < d_E(X_t, X_{t'}) < d_{max}$. Similarly, as each mobile object is assumed to rest on the ground plane, the change in elevation allowed between each pair of consecutive points $(X_t, X_{t'})$ must not exceed a threshold $d_{Elevation}$. As for the changes of direction, the angle formed by each triplet of consecutive

points $(X_t, X_{t'}, X_{t''})$ projected on the ground plane must not exceed a threshold α . All these constraints on the trajectory of each mobile point X allow for the detection and dismissal of erratic movements generated by false matches occurring in the feature matching module.

D. Parameter tuning

Each step of the described framework rely on various parameters affecting the overall performance of the proposed method. While some of the values used for these parameters directly come from the literature, an empirical tuning approach has been adopted to retain the best value for the other parameters in regard to the results obtained on our associated dataset. More precisely, parameters in section III-C, which is the main contribution of our method, use the following values for the consistency and first mobility constraints $t_{Cc} = 3.0$, $t_{Mc1} = 3.0$, while the values used to ensure the trajectory consistency in section III-C5 are $d_{min} = 0.1$, $d_{max} = 10.0$, $d_{Elevation} = 1.0$ and $\alpha = 60.0$. One should although consider that these values are specifically intended to work well on our associated dataset and are thus given on an indicative basis only.

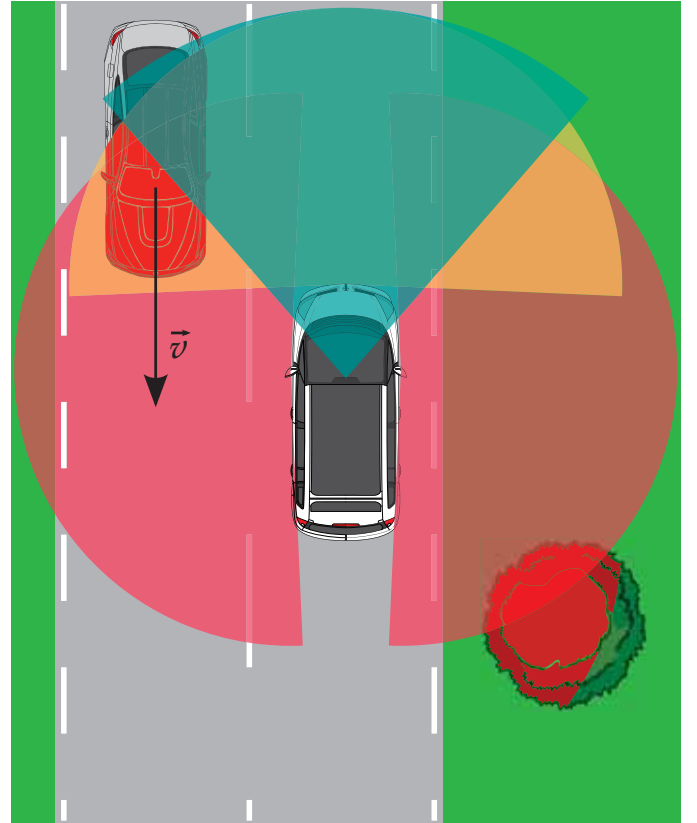


Figure 4. Top-down view of the vehicle with the four cameras and overlapping fields of view. According to our feature matching scheme, temporal matching is performed for each camera, while stereo matching is performed between the windscreen camera (in blue) and the three others (in orange and red) and between the front grille camera (in orange) and the side cameras (in red).

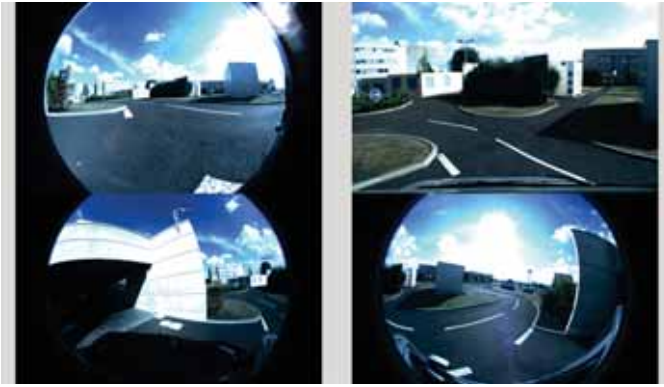


Figure 5. Views acquired with the four cameras. Top left is the front grille camera, top right the windscreen camera and bottom left and right are the side-left and side-right cameras.

IV. DATASET

While several datasets allowing for the evaluation of scene flow have recently been introduced, notably the KITTI dataset for scene flow estimation [13], no publicly available dataset to our knowledge uses an array of wide baseline and multifocal stereo cameras, which is the reason behind the creation of our own. The dataset presented in this article has been acquired in a realistic but controlled environment, which is composed of static and mobile elements such as cars and pedestrians. A total of eight different sequences corresponding to different road traffic scenarios at low speed have been acquired in order to assess the robustness of the proposed algorithm and the quality of the reconstructions.

The experimental vehicle has been equipped with four rigidly mounted digital cameras and a D-GPS. Spec-wise, the cameras use identical 2.3 MP, global shutter and synchronized sensors which record at 25 frames per second. The synchronization part of the acquisition process, being of utmost importance to ensure the geometrical correctness of our method, has been performed by hardware triggering. Three cameras are equipped with fisheye lenses equivalent to a 185 degrees horizontal FOV, while the fourth is equipped with a longer focal lens equivalent to an 80 degrees horizontal FOV. The fisheye cameras have been respectively placed on the front grille, pointing front in the longitudinal axle of the vehicle and on each side of the roof above the driver and passenger doors, pointing to the left and right perpendicular to the front camera. Finally, the last camera has been placed on the roof, above the windscreen, pointing front in the longitudinal axle of the vehicle. A top-down view of the vehicle with the four cameras and overlapping fields of view is shown in figure 4, while actual views acquired with the four cameras are shown in figure 5.

The choices of position, specs and optical characteristics of the cameras have been motivated by the increasingly popular implementation of heterogeneous camera systems in current vehicles, such as frontal (*e.g.*, Mobileye cameras) and surround cameras (*e.g.*, Around View Monitoring systems). These

experimental conditions should then help demonstrate the potential uses of such multi-camera systems in the challenging task of autonomous navigation.

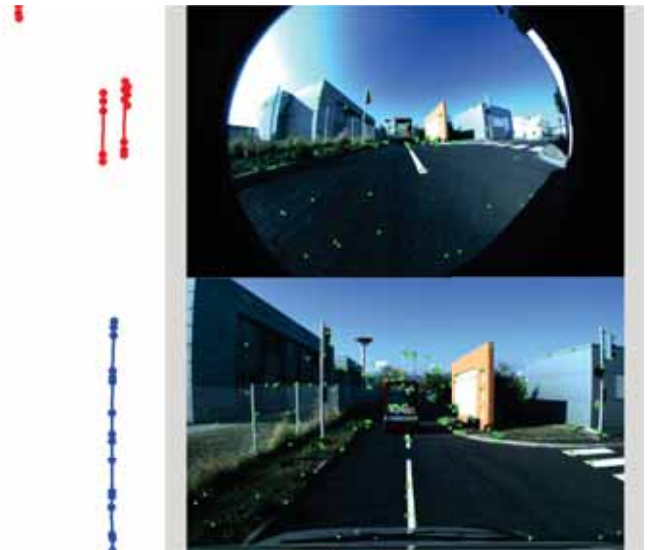


Figure 6. Example of false positive. The red trajectory in the top left corner is due to false matches misinterpretation on the traffic light.

V. EXPERIMENTAL RESULTS

Qualitative results have been obtained from the algorithm on various sequences from our dataset. One example of trajectory reconstruction is shown in figure 1. In this particular case, the acquisition vehicle whose trajectory is shown in blue is following another moving vehicle in front. Three mobile points are tracked and reconstructed simultaneously on the moving vehicle, which corresponds to the three red trajectories on the top left of the figure. These red points are also visible on the moving vehicle in the two provided views. One can note that some of the green points, which are static 3D points, also lie on the moving vehicle. These green points are not considered as mobile as they have not been tracked and stereo matched for three consecutive frames. While this also explains the relatively few number of mobile points detected, this is a limitation of the current state of the method, as the segmentation constraints used do not allow for a false observation to be associated with a moving 3D point, in which case the point becomes an outlier and is then dismissed. Views of the windscreen and front grille cameras corresponding to this particular trajectory can be seen in figure 7.

While several similar valid occurrences of tracking and reconstruction appear in the different sequences, some false positives are also to be noted. In figure 6, one can see the false red trajectory in the top left corner. This false positive is caused by false matches of features on the traffic light visible on the left in the views. These matches are correct according to the epipolar matching constraint E_c , as they lie on the epipolar lines, but are not pointing at the same static 3D point (one is at the top, the other at the bottom of the light) on the object and are thus misinterpreted as a moving point. These false

positives occur more frequently in repeating patterns areas such as the ground, grass, or the grilling visible on the left of the views in figure 6. Finally, although being semantic errors rather than geometric ones, shadows of moving objects can also be misinterpreted as moving areas and thus be considered as false positives.



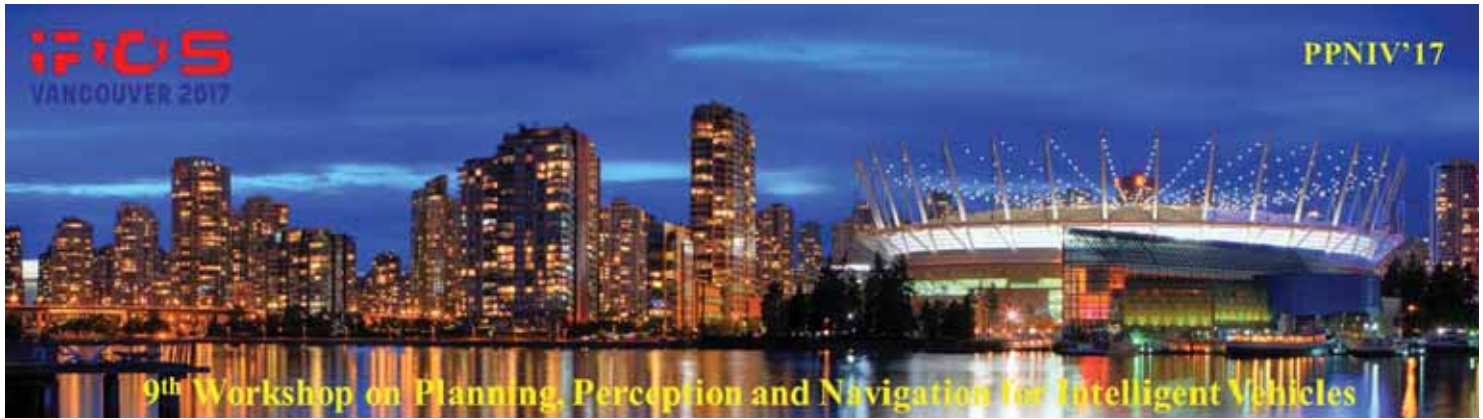
Figure 7. Views of the windscreen and front grille cameras corresponding to the trajectories shown in figure 1.

VI. CONCLUSION AND FUTURE WORKS

The observed qualitative results show that the purely geometrical method presented in this article works as intended on our dataset. However, several improvements can be made regarding the results. The reconstructed moving trajectories can indeed be considered as the starting point of a denser matching surrounding the area near the moving features. This could, in turn, allow to work with more points per moving object, which would enable their tracking and reconstruction to scale in the non-overlapped field of views of the multi-camera system. Also, some flexibility during the segmentation process, allowing to work on a larger number of points, could be achieved by pondering each observation and scoring the motion potential of each point instead of considering it as an outlier. These perspectives will eventually be explored in future works.

REFERENCES

- [1] M. Bleyer, C. Rhemann, and C. Rother. Extracting 3d scene-consistent object proposals and depth from stereo images. pages 467–481. Springer, 2012.
- [2] M. Buehler, K. Iagnemma, and S. Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. Springer, 2009.
- [3] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, 2007.
- [4] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun. 3d traffic scene understanding from movable platforms. *Pattern Analysis and Machine Intelligence*, 36(5):1012–1025, 2014.
- [5] C. Geyer and K. Daniilidis. A unifying theory for central panoramic systems and practical implications. In *European Conference on Computer Vision*, pages 445–461. Springer, 2000.
- [6] B. M. Haralick, C.-N. Lee, K. Ottenberg, and M. Nölle. Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 13(3):331–356, 1994.
- [7] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision, Second Edition*. Cambridge Univ. Press, 2003.
- [8] A. Kundu, K. M. Krishna, and C. Jawahar. Realtime multibody visual slam with a smoothly moving monocular camera. In *International Conference on Computer Vision*, pages 2080–2087. IEEE, 2011.
- [9] P. Lébraly, E. Royer, O. Ait-Aider, C. Deymier, and M. Dhome. Fast calibration of embedded non-overlapping cameras. In *International Conference on Robotics and Automation*, pages 221–227. IEEE, 2011.
- [10] V. Lepetit, F. Moreno-Noguer, and P. Fua. Epnnp: An accurate $o(n)$ solution to the pnp problem. *International Journal of Computer Vision*, 81(2):155–166, 2009.
- [11] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [12] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2, IJCAI'81*, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.
- [13] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *Computer Vision and Pattern Recognition*, pages 3061–3070. IEEE, 2015.
- [14] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3d reconstruction. In *Computer Vision and Pattern Recognition*, volume 1, pages 363–370. IEEE, 2006.
- [15] D. Nistér. An efficient solution to the five-point relative pose problem. *Pattern Analysis and Machine Intelligence*, 26(6):756–770, 2004.
- [16] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *Computer Vision and Pattern Recognition*, pages I–652. IEEE, 2004.
- [17] K. E. Ozden, K. Schindler, and L. Van Gool. Multibody structure-from-motion in practice. *Pattern Analysis and Machine Intelligence*, 32(6):1134–1141, 2010.
- [18] N. D. Reddy, I. Abbasnejad, S. Reddy, A. K. Mondal, and V. Devalla. Incremental real-time multibody vslam with trajectory optimization using stereo camera. In *Intelligent Robots and Systems*, pages 4505–4510. IEEE, 2016.
- [19] R. Sabzevari and D. Scaramuzza. Monocular simultaneous multi-body motion segmentation and reconstruction from perspective views. In *International Conference on Robotics and Automation*, pages 23–30. IEEE, 2014.
- [20] H. Strasdat, J. Montiel, and A. J. Davison. Real-time monocular slam: Why filter? In *International Conference on Robotics and Automation*, pages 2657–2664. IEEE, 2010.
- [21] E. Tola, V. Lepetit, and P. Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *Pattern Analysis and Machine Intelligence*, 32(5):815–830, 2010.
- [22] R. Vidal. Subspace clustering. *Signal Processing Magazine*, 28(2):52–68, 2011.
- [23] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte. Simultaneous localization, mapping and moving object tracking. *The International Journal of Robotics Research*, 26(9):889–916, 2007.
- [24] K. Yamaguchi, D. A. McAllester, and R. Urtasun. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In *European Conference on Computer Vision*, pages 756–771. Springer, 2014.



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

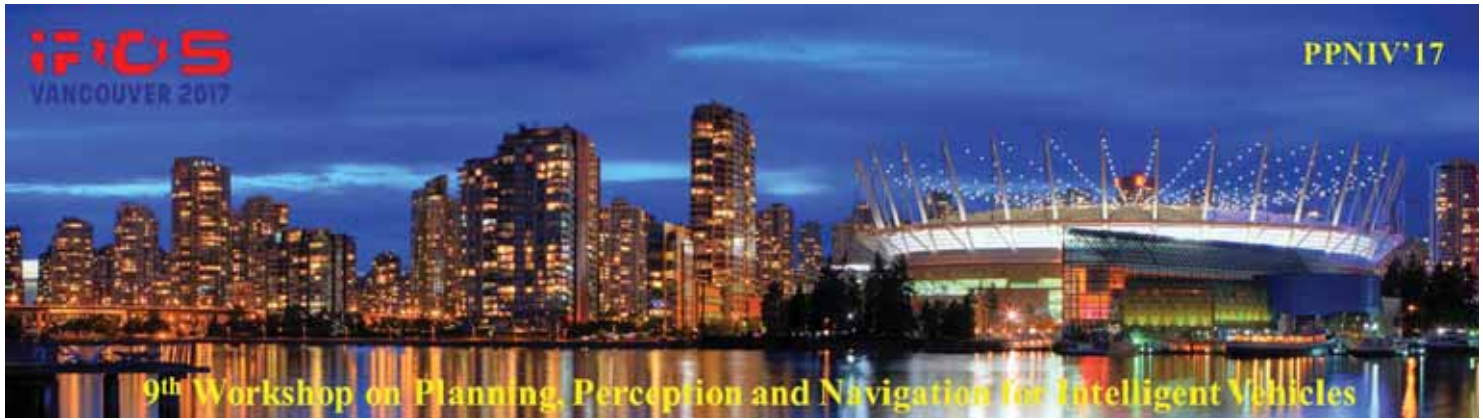
Session III

Simulation & Legal issues

- **Keynote speaker: Seung-Woo Seo (SNU, South Korea)**
Title: Technical and legal challenges for urban automated driving
- **Title: AutonoVi-Sim: Autonomous Vehicle Simulation Platform with Weather, Sensing, and Traffic control**
Authors: Andrew Best, Sahil Narang, Lucas Pasqualin, Daniel Barber and Dinesh Manocha



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session III

Keynote speaker: **Pr Seung-Woo Seo**
(SNU, South Korea)

Technical and legal challenges for urban automated driving

Abstract: Over the decade, researchers have been investigating viable technologies to realize autonomous driving in urban environments. Some key components of urban autonomous driving are the abilities to detect and track multiple objects, understand the situations, and decide optimal action policy. Most basically, autonomous vehicles in urban environments should be robust to uncertainties including unpredictable movement of moving objects, varying road situations, and uncertain traffic regulations. While there has been large progress in the technologies, there still remain many challenges that make the autonomous driving hard in urban environments. They include diverse dilemmas that we frequently encounter in daily driving. In this talk, we will discuss several key issues related to dilemma situations in urban autonomous driving. We will briefly introduce our approaches to these problems, and additionally, present some results of our research activities including the SNU Automated Drive, called SNUver.

Biography: Seung-Woo Seo is the professor of Electrical Engineering in Seoul National University, Seoul, and Director of Intelligent Vehicle IT (IVIT) Research Center funded by Korean government and automotive industries. He received his Ph.D. from the Pennsylvania State University, University Park, USA, and B.S. and M.S. degrees from Seoul National University, Seoul, Korea, all in Electrical Engineering. He was with the Faculty of the Department of Computer Science and Engineering, Pennsylvania State University, and served as a Member of the Research Staff in the Department of Electrical Engineering in Princeton University, Princeton, NJ. In 1996, he joined the Faculty of Seoul National University. He has served as Chair or a Committee Member in numerous international conferences and workshops. He was General Co-chair of 2015 IEEE Intelligent Vehicle Symposium held in Seoul, and has been serving as a Member of Steering Committee of IEEE Transactions on Intelligent Vehicles. He has also served as an Organizing Committee Chair of the International Unmanned Solar Vehicle Challenge in 2012, and served for five years as a Director of the Information Security Center in Seoul National University. He has been very active in the research on autonomous driving and has successfully developed an autonomous call-taxi called SNUver in 2015. In 2017, he demonstrated autonomous driving in the city road of Seoul for the first time in Korea. His research areas include autonomous driving, information security, and system optimization.



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Technical and Legal Challenges for Urban Autonomous Driving

Seung-Woo Seo, Prof.
Vehicle Intelligence Lab.
Seoul National University
sseo@snu.ac.kr



Contents

- I. Main Challenges for Urban Autonomous Driving**
 - I. Dilemma in Autonomous Driving
- II. Approach to Human-like Driving**
 - I. Intention-Aware Decision Making
 - II. Imitation Learning
- III. Autonomous Driving Research in SNU**
 - I. Demonstration of SNUver
- IV. Conclusion**

Challenges for Urban Autonomous Driving

Considerations for Urban Autonomous Driving

- Moving & static objects
 - Pedestrians
 - Other vehicles
 - Traffic light & signs
 - Unforeseen events
- Crossing intersection
- Turning
- Lane changes
- Parking
- Entering and exiting drop off stations
- Etc.



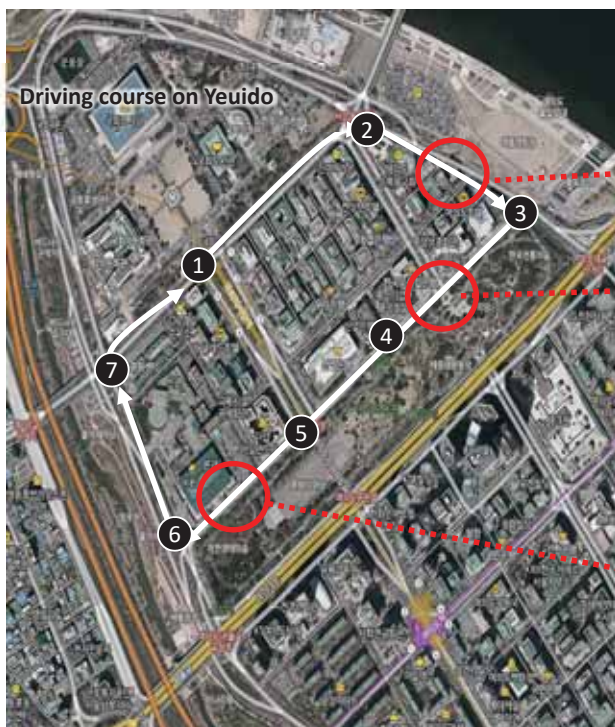
First Self-driving in City Road in Korea(2017. 6. 22)



Yeouido Area in Seoul



Demonstration at Yeouido Area in Seoul



7

Dilemma in Autonomous Driving

In urban environments, **dilemma situations frequently occur**



Crossing a double-yellow line to pass by an illegally parked car



Lane-change in heavy traffic



Decisions at a yellow traffic light

8

Dilemma in Autonomous Driving

3 Different Aspects

- I. Legal aspect
- II. Interactivity aspect
- III. Technology aspect

9

Legal Aspect

Crossing a double-yellow line to pass by an illegally parked car

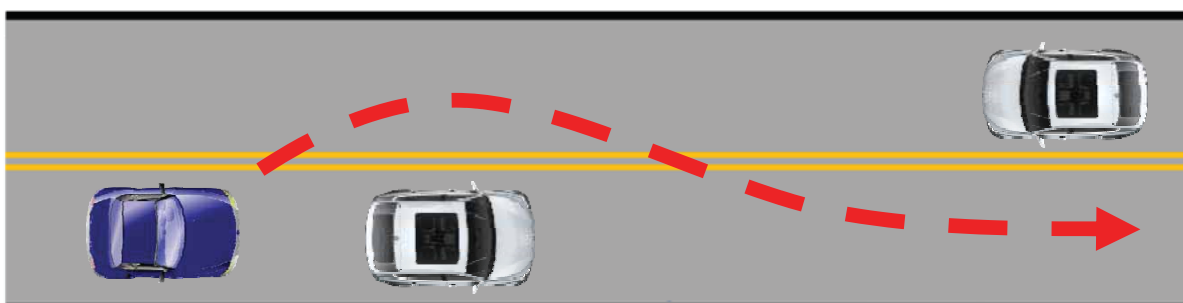
Crossing a double-yellow line

illegal & socially compliant decision

vs.

Waiting until an illegally parked car leaves

legal & impractical decision

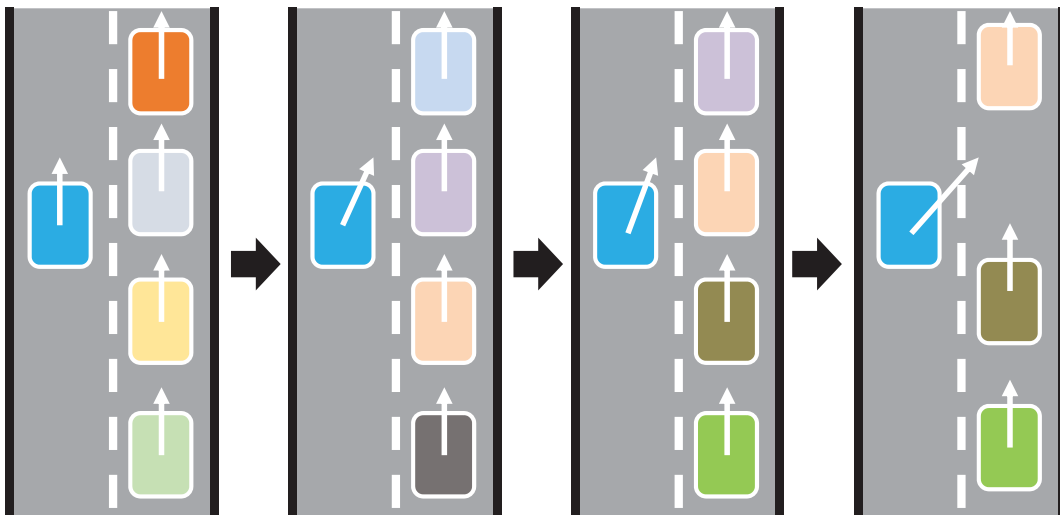


“AV violating the traffic law”



Interactivity Aspect

- Interactive driving (ex. Lane cut-in)



Dilemma in Autonomous Driving

3 Aspects

I. Legal aspect

EX) Crossing a double-yellow line to pass an illegally parked car

II. Interactivity aspect

EX) Lane-change in heavy traffic unsignalized intersection

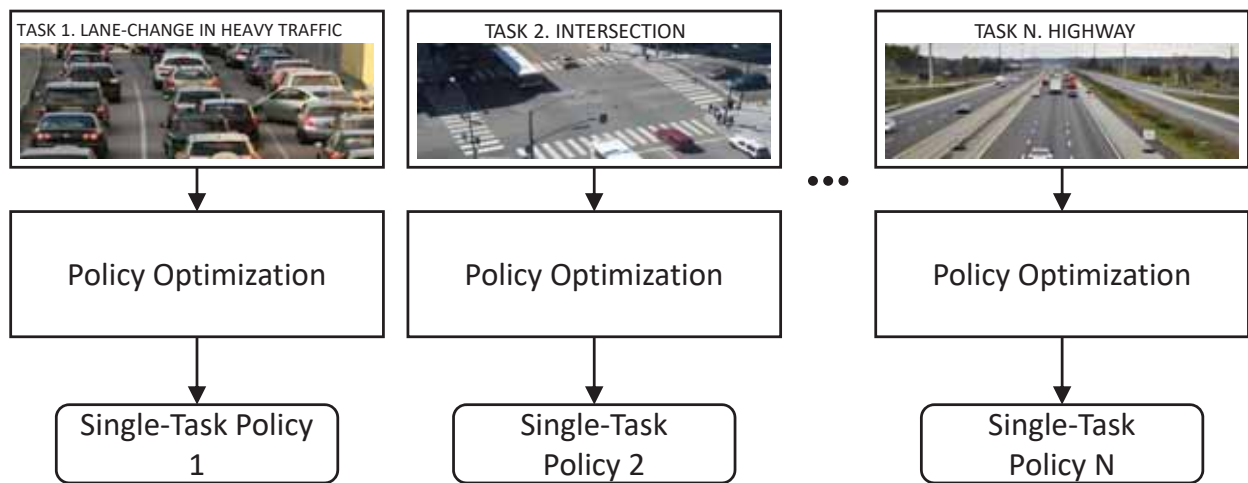
Human-Like Driving

III. Technology aspect

13

Approach to Human-Like Driving

Overall Framework

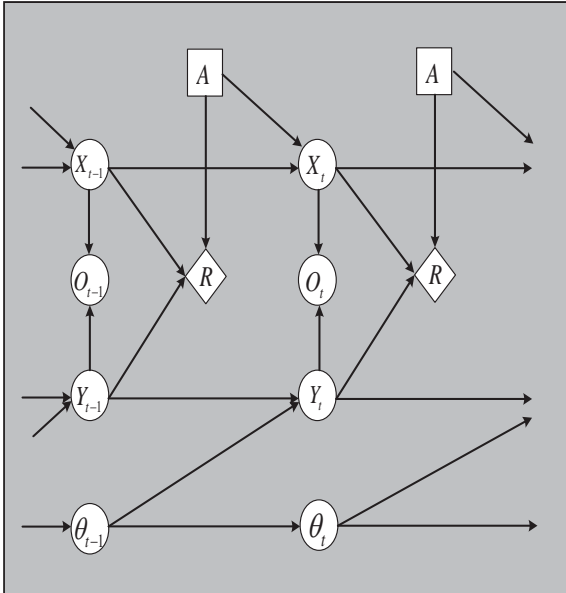


Unsignalized Intersection



Intention-Aware Decision Making

Model for Decision Making



- The state space "S" is a joint space $X \times Y \times \theta$
 - X : Ego-vehicle's state space

$$X_t^{ego} = [x_t^{ego}, y_t^{ego}, \theta_t^{ego}]$$
 - Y : Other vehicles' state space

$$Y_t^n = [x_t^n, y_t^n, \theta_t^n]$$
 - θ : Other vehicles' driving intention

$$\theta = [i_{long}^n, i_{lat}^n]$$
 - The action space "A" : $A = [Acc., Dec., Const.]$
 - The reward model
 - Very high penalty when vehicle is predicted to collide.
 - Very high reward when vehicle arrives at its goal.
 - Low penalty when vehicle moves at each step
- ➔ Passing through intersection as fast as possible without any collision

17

Intention-Aware Decision Making

Experimental Environment

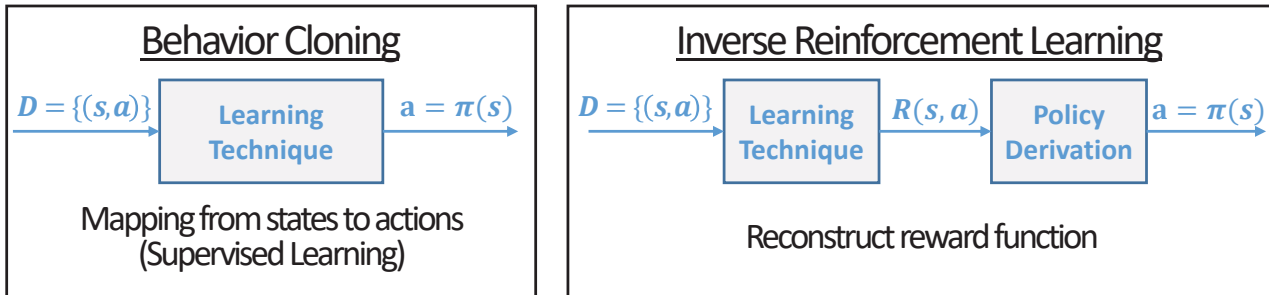


18

Imitation Learning

▪ Learning from Expert Drivers

- Expert drivers understand human interactions on the road and comply with **mutually accepted rules**, which are learned from countless experience



Brenna D. Argall, et al. "A survey of robot learning from demonstration", Robotics and Autonomous Systems 57 (2009): 469-483

19

Imitation Learning

▪ Driving dilemma in single lane road

- Crossing a double-yellow line to pass by an illegally parked car

Demonstration of expert drivers



Sang-Hyun Lee and Seung-Woo Seo, "A Learning-Based Framework for Handling Dilemmas in Urban Automated Driving", IEEE International Conference on Robotics and Automation(ICRA), 2017

20

Imitation Learning

- **Experimental Environments**



Autonomous Driving Research in SNU

Automated Driving Research in SNU

SNUver

SNU Automated Drive



[November 19, 2013]
Grand Prize in unmanned
self-driving car contest

[June 22, 2017]
Automated Driving in
Urban Environments



[November 4, 2015]
Driverless taxi on
SNU Campus

[November 15, 2016]
Door-to-Door Automated
Driving on **SNU Campus**

SNUver 1 (2015)



SNUver 2 (2016)

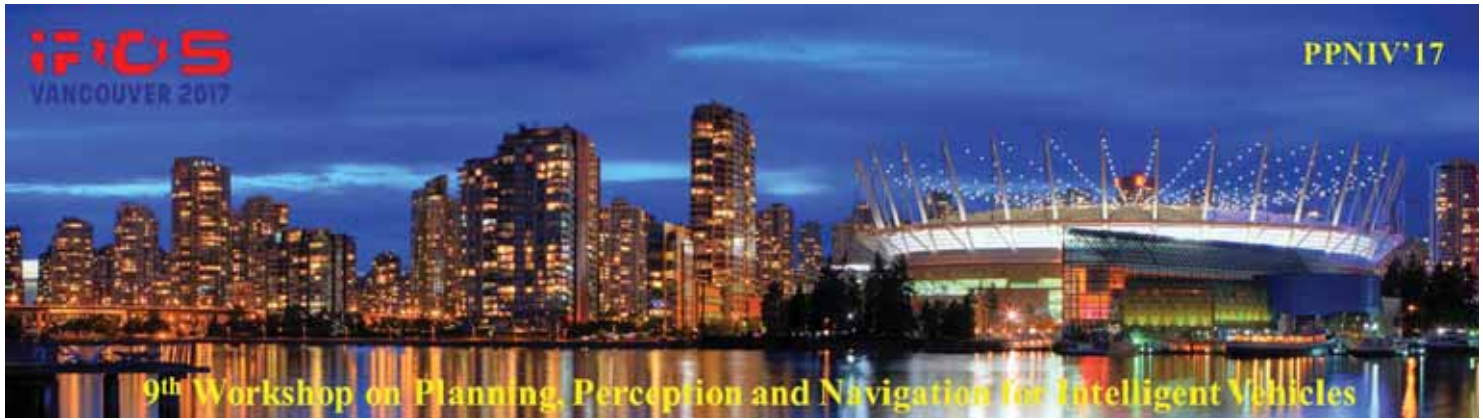


SNUvi (2017)



Conclusion

- Discussed several key issues related to dilemma in urban autonomous driving
 - Briefly introduced our learning-based approaches to human-like driving
 - There still remain many challenges that make the urban autonomous driving very hard
-
- Future Work



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session III

Simulation & Legal issues

- **Title: AutonoVi-Sim: Autonomous Vehicle Simulation Platform with Weather, Sensing, and Traffic control**
Authors: Andrew Best, Sahil Narang, Lucas Pasqualin, Daniel Barber and Dinesh Manocha



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

AutonoVi-Sim: Autonomous Vehicle Simulation Platform with Weather, Sensing, and Traffic control

Andrew Best¹ and Sahil Narang¹ and Lucas Pasqualin² and Daniel Barber² and Dinesh Manocha¹

Abstract—We present AutonoVi-Sim, a novel high-fidelity simulation platform for testing autonomous driving algorithms. AutonoVi-Sim is a collection of high-level extensible modules which allows the rapid development and testing of vehicle configurations and facilitates construction of complex road networks. AutonoVi-Sim supports multiple vehicles with unique steering or acceleration limits, as well as unique tire parameters and dynamics profiles. Engineers can specify the specific vehicle sensor systems and vary time of day and weather conditions to gain insight into how conditions affect the performance of a particular algorithm. In addition, AutonoVi-Sim supports navigation for non-vehicle traffic participants such as cyclists and pedestrians, allowing engineers to specify routes for these actors, or to create scripted scenarios which place the vehicle in dangerous reactive situations. AutonoVi-Sim also facilitates data analysis, allowing for capturing video from the vehicle’s perspective and exporting sensor data such as relative positions of other traffic participants, camera data for a specific sensor, and detection and classification results. Thus, AutonoVi-Sim allows for the rapid prototyping, development and testing of autonomous driving algorithms under varying vehicle, road, traffic, and weather conditions.

I. INTRODUCTION

Autonomous driving represents an imminent challenge encompassing a number of domains including robotics, computer vision, motion planning, civil engineering, and simulation. Central to this challenge is the safety considerations autonomous vehicles navigating the roads surrounded by unpredictable actors. Humans, whether drivers, pedestrians, or cyclists, often behave erratically, inconsistently, or dangerously, forcing other vehicles (including autonomous vehicles) to react quickly to avoid hazards. In order to facilitate acceptance and guarantee safety, vehicles must be tested not only in typical, relatively safe scenarios, but also in these dangerous, less frequent scenarios.

Aside from safety concerns, costs pose an additional challenge to the testing of autonomous driving algorithms. Each new configuration of a vehicle or new sensor requires re-calibration of a physical vehicle, which is labor intensive. Furthermore, the vehicle can only be tested under condition limited either by a testing track, or the current traffic conditions if a road test is being performed. This means the vehicle can be tested no faster than real-time and without any speedups or parallel testing.

The ability to test a driving algorithm in a high-fidelity, physics driven simulation would allow for testing novel

approaches without incurring intense labor costs. New vehicles or novel sensor configurations could be explored on many scenarios at once under a wide array of traffic and weather conditions. Engineers could also test the driving algorithm under conditions which are too dangerous to utilize the real vehicle. For example, engineers could test how the vehicle would respond to erratic behavior from other drivers, pedestrians, or cyclists without endangering the vehicle’s passengers or other traffic participants. Insights gained from simulation would provide critical information on algorithmic inefficiencies before actual vehicle testing.

In an effort to provide such a testing platform, we present AutonoVi-Sim, a simulation framework for testing autonomous driving algorithms and sensors. AutonoVi-Sim is a collection of high-level, extensible modules designed to allow researchers and engineers to rapidly configure novel road networks, driving scenarios, and vehicle configurations, and to test these in a variety of weather and lighting conditions. AutonoVi-Sim captures a variety of autonomous driving phenomena and testing requirements including:

- **Varying vehicle types and traffic density:** AutonoVi-Sim includes various vehicle models allowing for training classification on differing vehicle shapes, sizes, and colors. In addition, AutonoVi-Sim provides high fidelity traffic simulation, supporting dynamic changes in traffic density and the capacity to model the surrounding vehicles in high-fidelity.
- **Rapid Scenario Construction:** Typical road networks can be easily laid out using spline painting and are automatically connected for routing and navigation purposes. AutonoVi-Sim supports many lane configurations and atypical road geometry such as cloverleaf overpasses.
- **Cyclists and Pedestrians:** Non-vehicle traffic can be included in a scenario as part of the larger simulation or given specific scripted parameters. Cyclists and pedestrians are supported and can be given navigation destinations like vehicles or specific scenario behaviors to test the ego-vehicle’s response time (e.g. walking into the road in front of the ego-vehicle).
- **Varied Sensor Configurations:** Sensor placement can be varied per-vehicle to determine how a particular approach responds to differing environmental information. At runtime, sensor failure or loss of fidelity can be simulated as well.

The rest of the paper is organized as follows. In section II,

¹ Andrew Best, Sahil Narang and Dinesh Manocha are at the University of North Carolina, Chapel Hill

² Daniel Barber and Lucas Pasqualin are at the University of Central Florida

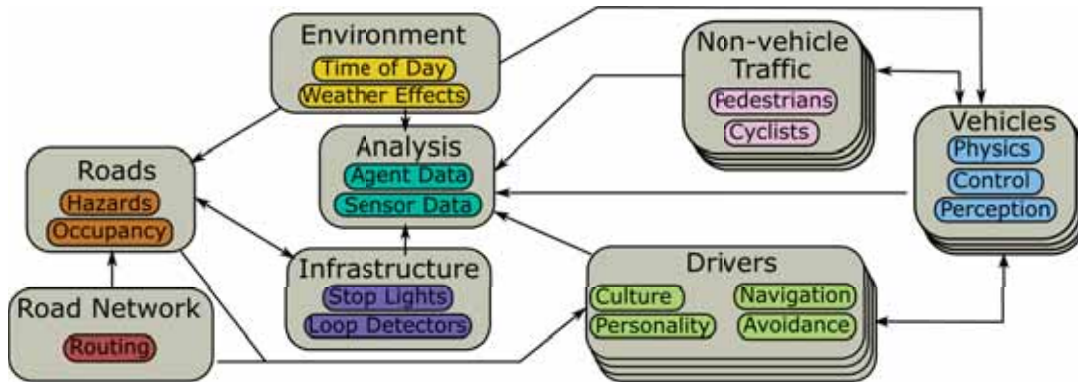


Fig. 1. **AutonoVi-Sim Platform Overview:** The eight modules composing AutonoVi-Sim encompass varying aspects of autonomous driving. The Road, Road Network, and Infrastructure modules define the driving environment. The Environment module allows engineers to specify specific environment conditions including time of day and weather. The Non-Vehicle traffic module allows engineers to specify navigation goals for pedestrians and cyclists, or setup specific triggered behaviors. The Drivers and Vehicles modules work as a pair to define current traffic conditions and specific driving destinations and decisions for the vehicles in the simulation. Each vehicle in the simulation has a unique set of sensing capabilities and a single driver which operates the vehicle during the simulation. Finally, the Analysis module is used to catalog and export data, including agent positions and sensor readings, for analysis.

we motivate simulation as a tool for advancing autonomous driving and detailed related work in the field. In section III, we detail the core modules provided by AutonoVi-Sim. We reserve discussion of the two vehicle related modules for section IV and offer demonstrations of the simulator.

II. RELATED WORK

Simulation has been an integral tool in the development of controllers for autonomous vehicles. [1], [2], and [3] offer in-depth surveys of the current state of the art and the role simulation has played. Many successful vehicle demonstrations of autonomy were first tested in simulation [4], [5], [6]. Recent work in traffic modelling has sought to increase the fidelity of the modelled drivers and vehicles; a survey is provided in [7].

Recent studies support the use of high-fidelity microscopic simulation for data-gathering and training of vision systems. [8] and [9] and leveraged Grand Theft Auto 5 to train a deep-learning classifier at comparable performance to manually annotated real-world images. Several recent projects seek to enable video games to train end-to-end driving systems, including ChosenTruck and DeepDrive-Universe which leverages the OpenAI Universe system. Using video game data provides benefits in the fidelity of the vehicle models but limits the ability to implement sensing systems and access data beyond visual data. A fully dedicated high-fidelity simulator can address these limitations and provide access to point-cloud data, visual data, and other vehicle sensors without the limitations imposed by adapting gaming software.

III. SIMULATION MODULES

Drawing from recent work in crowd simulation, [10], AutonoVi-Sim is divided into eight extensible modules, each with various sub-components. The modules are Environment, Road Network, Road, Drivers, Infrastructure, Vehicles, Non-vehicle Traffic, and Analysis. Each module captures some aspect of autonomous driving simulation and can be extended

and modified to suit the specific needs of a particular algorithm. Figure 1 shows the connection between components in AutonoVi-Sim. In this section, we will detail the modules which make up the basic simulation system, reserving discussion of the vehicle and driving strategy modules for section IV.

A. Roads

Roads in AutonoVi-Sim are represented by their center line, a number of lanes and directions thereof, and the surface friction of the road. Roads are placed interactively by drawing splines on a landscape which allows quick construction. Each road maintains occupancy information, average flow, and can maintain hazard information. The road module also maintains the set of hazards such as potholes or debris, which can be specified by density (number of hazards per km) or interactively by placing them on the road.

Alternately, roads can be specific pieces of geometry as in the case of intersections. This provides the flexibility to place specific intersections and model atypical road constructions for modelling specific environments. Figure 2(A) shows an example of road placement in AutonoVi-Sim.

B. Infrastructure

Infrastructure controllers represent traffic lights, signage, and any other entity which modifies the behaviors of vehicles on the road. These controllers can be added specifically to roads, as in the case of intersections, or placed independently as in signage or loop detectors. Vehicles implement their own detection of these entities as is described in section IV-A.2.

C. Road Network

The road network in AutonoVi-Sim provides the basic connectivity information for the traffic infrastructure to the vehicles in the simulation. At run-time, the network is automatically constructed by connecting roads into a directed graph. The road network provides GPS style routing to vehicles and localization for mapping purposes. Coupled with the road and infrastructure modules, the Road Network

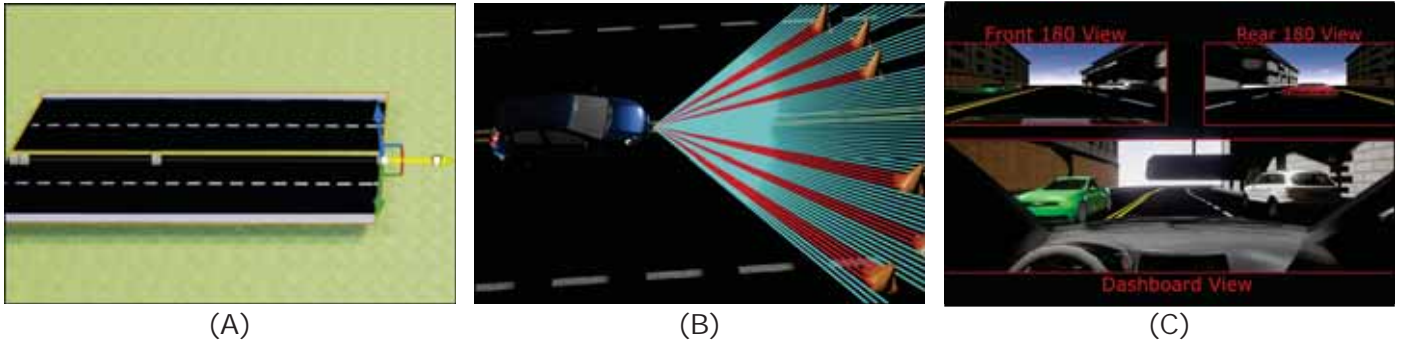


Fig. 2. **AutonoVi-Sim Environment and Sensor Setup:** (A): Roads in AutonoVi-Sim are constructed by dragging spline points along the road's center line. This allows for complex roads to be created quickly and efficiently. (B): An example configuration of a hatchback with a laser rangefinder navigating around traffic cones. Returned beams are illustrated in red. Beams which do not return data are illustrated in cyan for debugging. (C): An example camera configuration for a test vehicle. A 180 degree forward facing camera, a 180 degree rear-facing camera, and a dashboard camera are illustrated.

also provides information about upcoming traffic and current road conditions.

D. Environment

The environment module allows engineers to specify the specific environmental conditions for a given driving scenario. This currently includes time of day and weather. The system implements varying levels of fog and rain conditions. Environmental effects such as road friction reduction are controlled by the environment module.

E. Non-Vehicle Traffic

AutonoVi-Sim implements two non-vehicle traffic participants: pedestrians and cyclists. Pedestrians operate separately from the road network and can be given specific destinations. By default, pedestrians follow safe traffic rules to navigate to their goal. They can also be setup to trigger specific occurrences. For example, as the ego-vehicle nears, a pedestrian can be triggered to walk into the street in front of the vehicle to test its reaction time.

Cyclists operate similarly to vehicles in AutonoVi-Sim. Cyclists are given destinations and route over the road network. Similarly to pedestrians, cyclists can be programmed to trigger erratic behavior under specified conditions. For example, as the ego-vehicle approaches, a cyclist can be triggered to stop in the road, suddenly change direction, or enter the road in an unsafe fashion.

F. Analysis and Data Capture

AutonoVi-Sim implements a module for logging positions, velocities, and behaviors of the various traffic participants. It also supports logging egocentric data from the vehicle, such as relative positions of nearby entities at varying times during simulation. Camera-based sensors can record out the video data captured during simulation as can LIDAR based sensors Section IV-A.2 describes sensors in more detail.

IV. AUTONOMOUS DRIVING MODULES

The simulation modules described in section III serve as the basis for AutonoVi-Sim. This section describes the two core modules which allow for testing autonomous driving

and sensing algorithms under varying conditions, the Drivers and Vehicles modules.

A. Vehicles

The vehicle in AutonoVi-Sim is represented as a physics-driven entity with specific tire, steering, and sensor parameters. Physics parameters include the base tire coefficient of friction, the mass of the vehicle, engine properties such as gear ratios, and the physical model for the vehicle. Each of these parameters can vary between vehicles and relevant properties such as tire friction or mass can vary at runtime as needed.

1) *Control and Dynamics:* Vehicle control is provided on three axes: steering, throttle, and brake inputs. The specific inputs are chosen each simulation step by the driver model, described in section IV-B. The vehicle's dynamics are implemented in the NVidia PhysX engine. This allows the simulator to model the vehicle's dynamics and communicate relevant features such as slipping as needed by the driving algorithm.

2) *Perception:* The perception module of a vehicle provides the interface for information about surroundings to be gathered and stored in the vehicle. The basic sensing module in AutonoVi-Sim employs a ray-cast with configurable uncertainty, detection time, classification error rate, and sensor angle / range. This module is sufficient to test scenarios such as late detection or misclassification of pedestrians with minimal intervention. A vehicle can be equipped with multiple sensors with varying angles and fidelity, allowing the vehicle to simulate high-fidelity sensors in the longitudinal directions and broader, less accurate sensors in lateral directions. In addition, interaction with environmental conditions can be specified for the basic sensors, including performance impacts and uncertainty caused by weather effects.

In addition, the perception module provides interfaces to a generic camera interface and Monte-Carlo scanning ray-casts to simulate various sensor types. These interfaces can be extended to implement LIDAR or camera-based neural network classifiers in simulation. The LIDAR can be configured to change the scanning range, angle, and resolution. Similarly, the camera resolution, color parameters, and refresh rate can

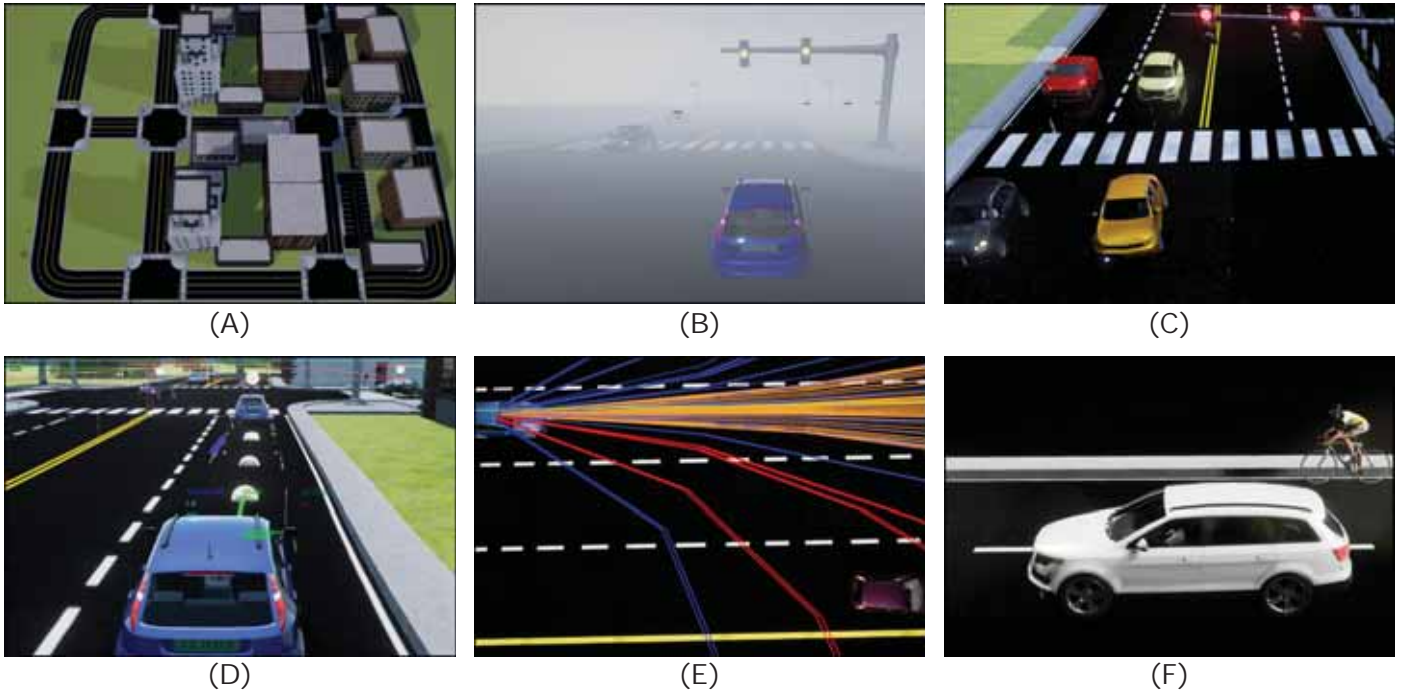


Fig. 3. **Simulated scenarios and conditions in Autonomi-Sim:** (A): A simulated city modelled in Autonomi-Sim. Closed circuit road networks allow engineers to test driving algorithms over long timescales by assigning new navigation goals periodically. (B): Heavy fog obstructs the view of the vehicle. (C): Vehicles pass through a slick intersection during rainy conditions. (D): The simple lane-keeping navigation approach projects the position of the vehicle forward in time and adjusts the speed accordingly. Each white orb represents a future predicted position of the vehicle given the current controls. (E): The Autonomi driving algorithm projecting sampled controls along its current path. Red control paths indicate predicted collisions with the nearby vehicle. Orange control paths represent the controls sampled by adaptive sampling around the prior best control set. (F): An SUV navigating with Autonomi changes lanes and passes a cyclist safely.

be configured for each camera sensor. Figure 2 shows an example of a camera-based sensor and simple LIDAR.

B. Drivers

Driving decisions in Autonomi-Sim, including routing and control inputs, are made by driver models. A driver model fuses information from the road network and the vehicle's sensors to make appropriate decisions for the vehicle. The specific update rate of the driver model can be configured as well as what sensors the model supports and prefers. Each model can implement any necessary parameters needed for the specific approach.

Autonomi-Sim currently implements three driver models. The first is a simple lane-following approach which employs control methods similar to a driver assistance lane-keeping system. This driver is used to generate passive vehicles travelling along their destinations without aberrant or egocentric behavior. These vehicles are capable of lane-changes and turns, but follow simple rules for these maneuvers and rely on perfect sensing models to accomplish them.

The more extensive driving model, Autonomi, is described in detail in [11]. This model uses optimization-based maneuvering with traffic constraints to generate behaviors such as overtaking and combines steering and braking maneuvers through a data-driven vehicle dynamics prediction model.

Finally, the simulator implements a manual driving mode, which can be activated from any autonomous driver. Manual mode allows an engineer to drive the vehicle using a

keyboard, game-pad, or steering wheel and pedal combination. As described in [12], this manual operation is being employed to test vehicle signalling and connected vehicle operation. It can also be used to collect data for neural-network methods.

Figures 2 and 3 detail several example scenarios and configurations we have tested in Autonomi-Sim. Additional details on Autonomi and additional simulations and testing environments can be found in [11].

V. CONCLUSION

We have presented Autonomi-Sim, a platform for autonomous vehicle simulation with the capacity to represent various vehicles, sensor configurations, and traffic conditions. We have demonstrated Autonomi-Sim's applicability to a number of challenging autonomous-driving situations and detailed the ways in which Autonomi-Sim can enhance the state of the art in testing autonomous-driving algorithms. Autonomi-Sim is a modular, extensible framework. While many modules currently represent preliminary implementations of advanced functionality, the extensible nature of the framework provides the basis for progress in the various disciplines which define autonomous driving.

Our work is in active development and still faces a number of limitations. Autonomi-Sim contains basic implementations of the various modules such as sensors for perception, a physics engine to simulate dynamics etc. However, each of these modules can be extended to more accurately reflect

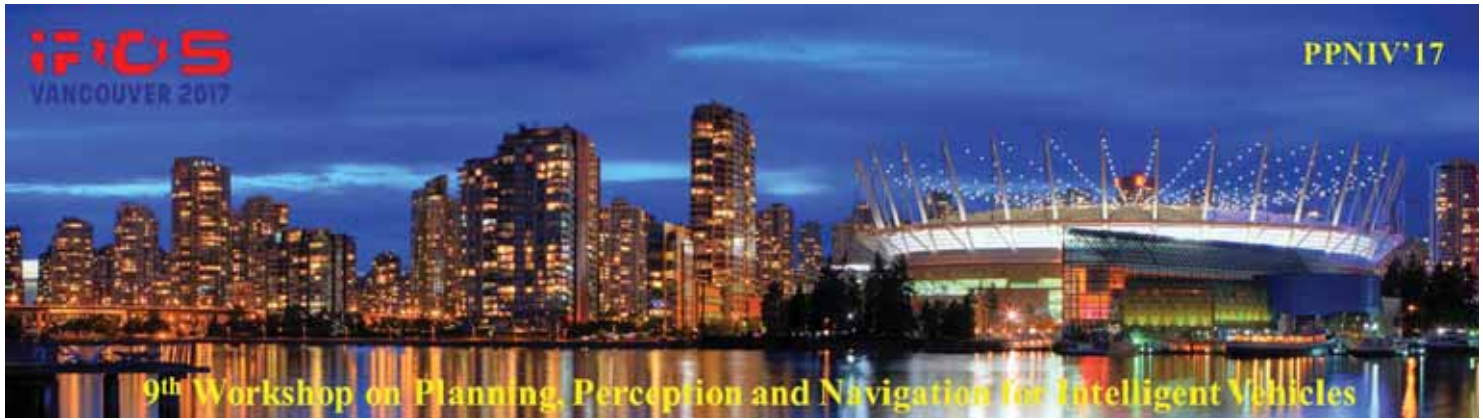
real world conditions. For example, AutonoVi-Sim currently lacks calibration information to replicate specific sensors and sensor configurations. In the future we hope to model specific sensing packages and algorithms to test specific real-world configurations. In addition, we seek to explore the transfer between algorithms trained on AutonoVi-Sim and actual test vehicles.

REFERENCES

- [1] Pendleton et al. Perception, Planning, Control, and Coordination for Autonomous Vehicles. *Machines*, 5(1):6, 2017.
- [2] Christos Katrakazas, Mohammed Quddus, Wen-Hua Chen, and Lipika Deka. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 60:416–442, 2015.
- [3] Mohammad Saifuzzaman and Zuduo Zheng. Incorporating human-factors in car-following models: A review of recent developments and research needs. *Transportation Research Part C: Emerging Technologies*, 48:379–403, 2014.
- [4] Sterling J. Anderson, Steven C. Peters, Tom E. Pilutti, and Karl Iagnemma. Design and development of an optimal-control-based framework for trajectory planning, threat assessment, and semi-autonomous control of passenger vehicles in hazard avoidance scenarios. *Springer Tracts in Advanced Robotics*, 70(STAR):39–54, 2011.
- [5] Maxim Likhachev and Dave Ferguson. Planning Long Dynamically Feasible Maneuvers for Autonomous Vehicles. *The International Journal of Robotics Research*, 28(8):933–945, 2009.
- [6] M. Aeberhard, S. Rauch, M. Bahram, G. Tanzmeister, J. Thomas, Y. Pilat, F. Homm, W. Huber, and N. Kaempchen. Experience, Results and Lessons Learned from Automated Driving on Germany’s Highways. *IEEE Intelligent Transportation Systems Magazine*, 7(1):42–57, 2015.
- [7] Bo Chen and Harry H. Cheng. A review of the applications of agent technology in traffic and transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, 11(2):485–497, 2010.
- [8] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. *Playing for Data: Ground Truth from Computer Games*, pages 102–118. Springer International Publishing, Cham, 2016.
- [9] M. Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? In *IEEE International Conference on Robotics and Automation*, pages 1–8, 2017.
- [10] Sean Curtis, Andrew Best, and Dinesh Manocha. Menge: A modular framework for simulating crowd movement. *Collective Dynamics*, 1(0):1–40, 2016.
- [11] Andrew Best, Sahil Narang, Daniel Barber, and Dinesh Manocha. AutonoVi: Autonomous vehicle planning with dynamic maneuvers and traffic constraints. In *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, In Press.
- [12] Daniel Barber and Andrew Best. Connected and automated vehicle simulation to enhance vehicle message delivery. In *8rd International Conference on Applied Human Factors and Ergonomics AHFE, Los Angeles, USA*, In Press.



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session IV

Interactive session

- **Title: Lateral Controllers using Neuro-Fuzzy Systems for Automated Vehicles: A Comparative Study**
Authors: Sarouthan Sriranjana, Ray Lattarulo, Joshue Perez-Rastelli, Javier Ibanez-Guzman, Alberto Pena
- **Title: Asynchronous Multi-Sensor Fusion for 3D Mapping and Localization**
Authors: Patrick Geneva, Kevin Eckenhoff, and Guoquan Huang
- **Title: Towards Cooperative Motion Planning for Automated Vehicles in Mixed Traffic**
Authors: Maximilian Naumann and Christoph Stiller
- **Title: Prediction of Urban Pedestrian Behaviour using Natural Vision and Potential Fields**
Authors: Pavan Vasishta, Dominique Vaufreydaz and Anne Spalanzani
- **Title: Constant Space Complexity Environment Representation for Vision-based Navigation**
Authors: Jeffrey Kane Johnson
- **Title: Relocalization under Substantial Appearance Changes using Hashing**
Authors: Olga Vysotska, Cyrill Stachniss
- **Title: Experimental study of the precision of a multi-map AMCL-based localization system**
Authors: Gaetan Garcia, Salvador Dominguez Quijada, Jean-Marc Blosseville, Arnaud Hamon, Xavier Koreki and Philippe Martinet
- **Title: PedLearn: Realtime Pedestrian Tracking, Behavior Learning, and Navigation for Autonomous Vehicles**
Authors: Aniket Bera and Dinesh Manocha
- **Title: Safe Navigation in Dynamic, Unknown, Continuous, and Cluttered Environments**
Authors: Mike D'Arcy, Pooyan Fazli and Dan Simon



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Lateral Controllers using Neuro-Fuzzy Systems for Automated Vehicles: A Comparative Study

Sarouthan Sriranjana¹, Ray Lattarulo², Joshué Pérez-Rastelli², Javier Ibañez-Guzman¹, Alberto Peña²

Abstract—Different implementations on automated vehicles are being introduced by researchers and manufacturers, particularly for longitudinal control. Some applications include traffic jam assistance, emergency assisted braking, Cruise Control, among others. However, lateral control applications are less common due to the complexities of the dynamic. In this paper, an Artificial Intelligence approach to control the steering wheel of an automated vehicle is presented. Two new lateral controllers are developed. One is based on human expertise (Fuzzy Logic), and the other is based on an Adaptive Network based Fuzzy Inference System (ANFIS) using expert driver data. Those controllers have been tested in a simulation environment, called Dynacar, and they were compared with a classical PID controller, giving promising results.

I. INTRODUCTION

In recent years, the research and development in automated driving is intensively increased in automotive industry. Indeed, automated vehicles are opening a new road-map to many new applications and benefits for the society, i.e.: new mobility alternative, increasing safety and reducing parking areas, assistance to drivers and intelligent and connected infrastructures. Moreover, this is topic where many research groups, universities and manufactures are working around the world.

Based on most of the previous real automated vehicle implementations in the literature, a general control architecture is divided in six main stages: acquisition, perception, communication, decision, control and actuation ([1]), where the perception, decision and control are the most critical. An automated vehicle uses sensors to get information on the external environment, i.e.: for the ego positioning (GPS, radars) and the detection of others obstacles around (Lidars and cameras). This information is processed in order to find a safe trajectory to be followed by the vehicle. This process is called the Global and local planner module, some authors call it navigation. Then, this trajectory is sent to the control module to maintain the vehicles on the road. Finally, the actuators receive and execute this command by the steering wheel and pedals.

The Original Equipment Manufacturers (OEMs) have been started to work in this field fifteen years ago with the first uses of Advanced Driver Assistance Systems (ADAS), where the most known is the Anti-lock braking system (ABS). The

main goal of ADAS is to improve the safety of the passengers in the vehicles and the Vulnerable Road Users. Studies say that ninety five percent of the road accidents are caused because of a human factor ([2]). Most of the accidents are caused by a human limitation, as well as reaction time or distractions. The next step is the implementation of more safety and robust functionalities for automated vehicle.

So far, lateral control applications have special attention, where complex models have been used mainly in automated longitudinal system [3] and [4]. Other researches have demonstrated that some Artificial intelligence (AI) techniques, as fuzzy logic, offers a good solutions to control complex system, like automated vehicles [5]. Fuzzy sets do not need an exact mathematical model of the plan, as in [6]. These controllers can imitate the human driver behavior with a knowledge base (or rules). However, in most of the cases, there aren't standard methodologies defined these rules bases and the membership functions. Saifizul et al. [7] has simulated an ANFIS controller for the lateral applications. However, this approach only considers the constant curvature in the circuit (not path planning), with some significant overshoot and oscillation on the steering angle at the moment the curvature changes [7].

In this paper, we will focus on the lateral control of the vehicle applying three different controllers. Different maneuvers have been tested, as urban intersections, lane keeping and lane changes, based on a real time path planning presented on [8]. The goal is to reduce the lateral error to reference given by the path planner. In this work, a study of different control techniques is presented, where a PID, a Fuzzy Logic and an ANFIS controllers are considered for the lateral dynamic of an automated vehicle. The modelled vehicle used to validate our approach is an automated Renault Twizy.

The rest of the paper is organized as follow: a brief state of the art of the control in the automotive field in Section 2. Next there is a description of the simulation environment, Dynacar, used for our validation. Then we will see the functioning of tree different controllers for the lateral control: a PID controller, the fuzzy logic and the Neuro-Fuzzy control in Section 4. Finally, we will compare the performances of these three controllers in Section 5. We concluded on the performances of these controllers and the future works.

II. WORK MOTIVATION

In automated driving field, the control of the vehicle is divided in lateral and longitudinal. The first one concerns the action on the steering wheel. There are different kinds

¹(e-mail: sarouthan.sriranjan@u-psud.fr and javier.ibanez-guzman@renault.com)

²Tecnalia Research and Innovation, Derio, Vizcaya, Spain, 48160. (e-mail: {rayalejandro.lattarulo, joshue.perez, alberto.peña}@tecnalia.com).

* Authors wants to thank to the ECSEL project ENABLE-S3 for its support in the development of this work, and the Non-Disclosure Agreement between Tecnalia and Renault s.a.s

of controllers that can be used for the case of a lateral or longitudinal actions ([9]).

The steering of a vehicle is considered as a nonlinear dynamic system, especially at high speed. It is possible to control this system through techniques that allows quick, smooth and high quality control ([5]). The action on the lateral control depends mainly of 2 modules of the global architecture, the path planner (trajectory) and controller itself (tracking) ([1]).

Sei-Bum Choi has developed an adaptive control law for the lateral control of automated vehicle using magnetic sensors in the vehicle's front wheels. Another big change in the low level steering wheel system has been the incorporation of electric-power-assisted steering (EPS) as a substitution for the traditional hydraulic power steering (HPS) systems in the new generation of vehicles ([10]).

However, the most classical lateral controller found is the Proportional Integral Derivative (PID) controller ([11]). Indeed, this controller is present in many applications in the industry because of the reduced number of parameters to be tuned.

Methods using fuzzy logic, linear matrix inequality optimization and yaw rate control have been used in order to make the vehicle following the reference trajectory ([12], [13]).

The main advantage of a fuzzy controller is that it is not necessary to have an exact mathematical model of the system to control it. The control problem is reduced to estimate the input, set up a rule base and assign the output values. Besides, this controller can emulate the behavior from drivers due to knowledge base, using the human experience, and if-then rules. Many contributions related with Fuzzy Logic controller have been published ([6], [9]). However, the problem to tune and to estimate the rule bases for lateral applications remains still an open issue.

Some automated methods to tune the Fuzzy controller have been developed for longitudinal controllers [14]. This Neuro-Fuzzy system combines the semantic rules and learning capability of neural networks. Some applications have used neuro-fuzzy systems to control nonlinear systems or to adjust controllers [15]. In the Intelligent Transportation System (ITS) field, neuro-fuzzy systems had been used in the traffic modelling, as in [16]. However, this kind of controller has not been tested in lateral dynamic for automated vehicles.

Previous works on neuro-fuzzy controller ([14]) presents a real time implementation of a neuro-fuzzy controller for the longitudinal control of a vehicle. This neuro-fuzzy system improves the performances of previous controller by including the experience of expert drivers. In light of the size of the data base used, this controller gives good results. We are on the next step, to study the use of fuzzy logic and neuro-fuzzy controller for the lateral control of the vehicle.

III. SIMULATION PLATFORM

Dynacar (figure 1) is a simulation tool developed by Tecnelia which provides a real-time vehicle model. It focuses on two main domains the dynamics of the vehicle and the



Fig. 1: Dynacar by Tecnelia

electronics architecture of the vehicle. It provides a complete architecture to simulate and validate the automated capabilities of a vehicle [17]. This architecture is particularly adapted for urban scenarios. Different lateral control algorithms were tested with this platform.

It provides a high-fidelity vehicle physics simulation. This is combined with a Pacejka tire model, and submodels for elements like the engine, transmission, steering system, braking system, aerodynamics, etc ([18]). For this work, we use the equations of the lateral dynamic of the bicycle model as shown in (1) and (2), as are explained in [19]:

$$\ddot{y} = -\frac{2C_{\alpha f} + 2C_{\alpha r}}{mV_x} \dot{y} + \left(-\frac{2\alpha C_{\alpha f} - 2bC_{\alpha r} - V_x}{mV_x} \right) \dot{\psi} + \frac{2\alpha C_{\alpha f}}{m} \delta \quad (1)$$

$$\ddot{\psi} = -\frac{2\alpha C_{\alpha f} + 2C_{\alpha r}}{I_z V_x} \dot{y} + \left(-\frac{2a^2 C_{\alpha f} - 2b^2 C_{\alpha r}}{I_z V_x} \right) \dot{\psi} + \frac{2\alpha C_{\alpha r}}{I_z} \delta, \quad (2)$$

where m is the vehicle mass, V_x is the longitudinal speed, I_z is the yaw inertia, a and b are the distance between the front/rear wheels and the center of gravity, respectively, and δ represents the front wheel steering angle (which is the control signal in the automated control approach).

The values used in model for the twizy described before are resumed in table I:

TABLE I

Parameter	Value	Unit
m	582.5	[kg]
I_z	314.28	[kg.m ²]
Wheelbase	1.686	m
Dist. to COG (b)	0.4231	m
$C_{\alpha f}$	5.784	[1/rad]
$C_{\alpha r}$	17.163	[1/rad]

The real-time capability is very valuable, as, combined with its notable modularity and interfacing options, it permits to execute tests with driver-in-the-loop (DiL) and hardware-in-the-loop (HiL) setups, for instance for ECU (Electronic Control Unit) development or also motor test bench testing ([20]).

For this application, a reference trajectory is generated using parametric curves as in [8]. This path planner is used

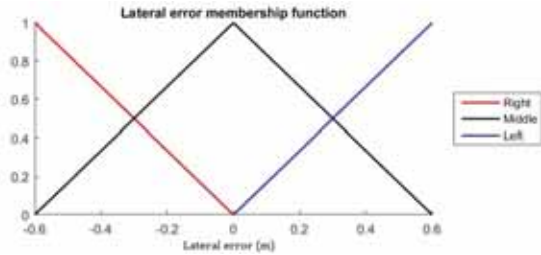


Fig. 2: Membership function for the Lateral Error.

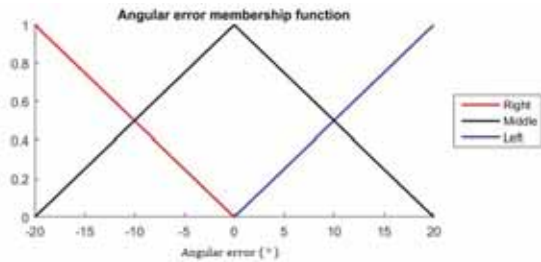


Fig. 3: Membership function for the Angular Error.

as the reference for the lateral control module, using different controllers.

IV. LATERAL CONTROL

On the current section, three control techniques will be explain to gather some concepts related to these ones, including the designing process. These ones were tested on an automated driving simulator to verify the differences between them and, advantages and disadvantages that each presents.

A. PID controller

The first controller implemented on the current approach was a PID controller associated to the lateral error (this is calculated as a reference of the frontal point of the vehicle and the path). The equation associated to the steering is:

$$C_v = K_p e_{lat} + K_i \int_t e_{lat} dt + K_d \frac{d}{dt} e_{lat} \quad (3)$$

K_p , K_i and K_d are the gains fixed manually on the vehicle using classic techniques of tuning.

B. Fuzzy logic controller

For the fuzzy controller, two input variables were used: the lateral and the angular error. Each variable is defined by a membership function affecting its corresponding linguistic labels, which is represented in the rule base. For the current approach, a triangular shape membership function was implemented.

The lateral and angular errors use three labels in each case: Left, Middle and Right. The membership functions are symmetric considering the ideal symmetry of the steering wheel. The range used for the lateral error is [-0.6; 0.6] (m) as it is illustrated on figure 2 and the one for the angular error is [-20; 20] (degrees) as it is illustrated on figure 3.

The defuzzification operation uses a method called “center-of-area” [5]. This is one of the most prevalent and

physically appealing of all defuzzification methods. Here, W_i are the weights of the linguistic label i for each membership function. O_i are the assigned values of the singleton output for the label i . Finally, X_i is the crisp value of each rule condition. Each crisp value is calculated by the Mamdani inference method:

$$X_i = \sum \frac{W_i O_i}{W_i} \quad (4)$$

To define the output, nine singletons were established with various weights: $Right_{P4}$, $Right_{P3}$, $Right_{P2}$, $Right_{P1}$, $Middle$, $Left_{P1}$, $Left_{P2}$, $Left_{P3}$, $Left_{P4}$, which were defined between [-1 and 1] and spaced each 0.25.

The rule base interprets the input variables, based on the IF . . . THEN form. The target output is given by the steering position. The rules are shown as follows:

IF $Lateral_{Error}$ Right AND $Angular_{Error}$ Right THEN Steering $Right_{P4}$
 IF $Lateral_{Error}$ Right AND $Angular_{Error}$ Middle THEN Steering $Right_{P3}$
 IF $Lateral_{Error}$ Right AND $Angular_{Error}$ Left THEN Steering $Right_{P2}$
 IF $Lateral_{Error}$ Middle AND $Angular_{Error}$ Right THEN Steering $Right_{P1}$
 IF $Lateral_{Error}$ Middle AND $Angular_{Error}$ Middle THEN Steering $Middle$
 IF $Lateral_{Error}$ Middle AND $Angular_{Error}$ Left THEN Steering $Left_{P1}$
 IF $Lateral_{Error}$ Left AND $Angular_{Error}$ Right THEN Steering $Left_{P2}$
 IF $Lateral_{Error}$ Left AND $Angular_{Error}$ Middle THEN Steering $Left_{P3}$
 IF $Lateral_{Error}$ Left AND $Angular_{Error}$ Left THEN Steering $Left_{P4}$

C. Neuro-Fuzzy Controller

Pursuing the development of fuzzy-logic-based controller, many industrial processes are now controlled using the knowledge of expert operators. Thus, the fusion of Artificial Neural Networks and Fuzzy Inference Systems has grown interest among researchers. There are some limitations in the IF . . . THEN systems. There are not standard methods for transforming experience from the driver into the rule base.

The learning process of the Neural Networks is based on the adjustment of the weight of the connections between the nodes net. Neuro-Fuzzy systems combine the easy handling of the IF . . . THEN rules of the fuzzy logic with the learning capacity of neural networks.

The Adaptive-Network-Based-Fuzzy Inference System (ANFIS) was one of the first neuro-fuzzy systems developed. The principle is to extract fuzzy rules at each level of a neural network. When the rules have been obtained, they provide the information on the overall behavior of the process. ANFIS implements the Takagi-Sugeno model for the IF . . . THEN rules of the fuzzy logic.

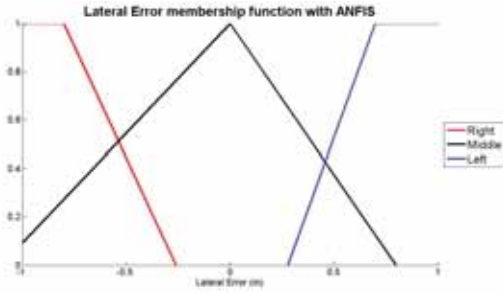


Fig. 4: Membership function for the Lateral Error of the ANFIS.

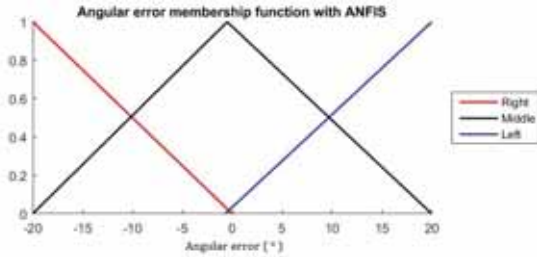


Fig. 5: Membership function for the Angular Error of the ANFIS.

Neuro Fuzzy is based on the creation of a controller by using a learning process from a data base. We want to train our ANFIS system using as reference the expert knowledge of a driver and for this purposes, it was recorded the behavior of this driver trying to follow the road. The steering position, the lateral and the angular error were recorded during the rolling session.

The parameters chosen for the learning process of the ANFIS systems were:

Algorithm	ANFIS
System	2 Inputs and 1 Output
Membership function shape	Triangular
Number of membership func.	3
Inference system	Takagi-Sugeno
Number of rules	9
Training algorithm	Back propagation
Training data set	25000
Validation data set	25000

The proposed neuro-fuzzy controller has two input variable like the previous fuzzy controller studied (the lateral and the angular error) and one output variable (the steering position). Each input has three membership functions, bringing to nine rules.

The two inputs have still the same membership function shape: three triangular shapes. The range used for the lateral error is $[-1; 1]$ meters (figure 4) and the one for the angular error is $[-20; 20]$ degrees (figure 5). These ranges depend on the data base used for the learning process and, of course, a human driver has a driving way that contains more imprecision than a driving from a controller which is tuned to have the lowest error.

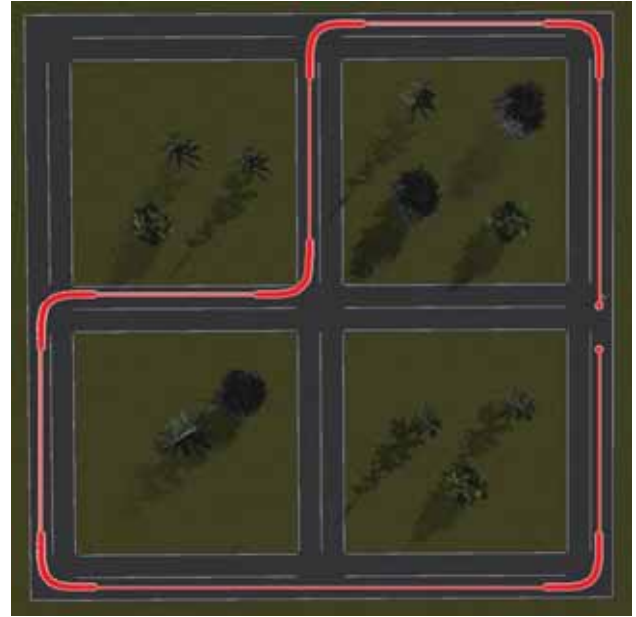


Fig. 6: Circuit of the driving session for the Simulation on Dynacar.

The nine singletons of the ANFIS system are:

Singleton	Value
Right P4	-0.9535
Right P3	-0.538
Right P2	-0.485
Right P1	-0.235
Middle	0.02
Left P1	0.24
Left P2	0.46
Left P3	0.67
Left P4	0.95

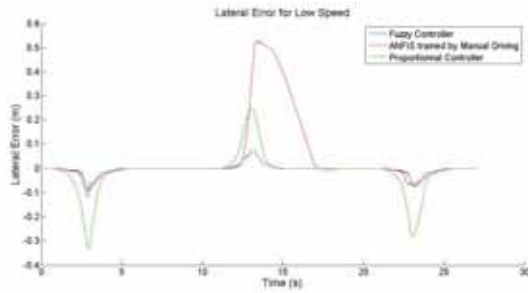
In previous works [7], the dataset was tuned by a Fuzzy Logic model. In our approach with ANFIS, the controller is trained by human knowledge. The fuzzy controller has a close behavior to the human driver.

V. TESTS AND RESULTS

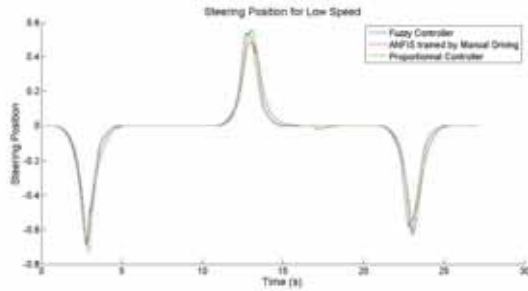
In order to compare the performances of these three controllers, they have been tested in simulation using a predefined circuit. It contains six intersections as it is illustrated on figure 6.

Additionally, the use cases defined to verify the behavior of the controllers were four. A segment of the circuit with three consecutive intersections and a lane change, both at low speed, are the first and the second scenario. In the case of the third and fourth use cases are considered the same intersections and lane changing but at high speed.

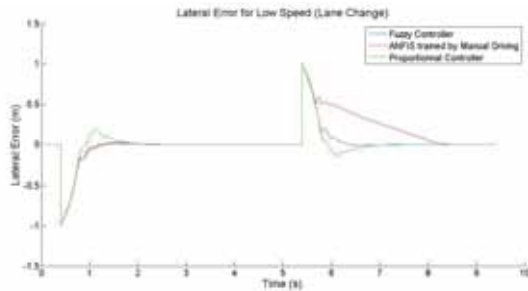
For the purposes of the current work, low speed is considered as a constant speed of 20 km/h and high speed is based on a speed profile given by comfort and curvature constraints with a maximum value of 50 km/h. And the controllers will be compared using the lateral error and steering responses.



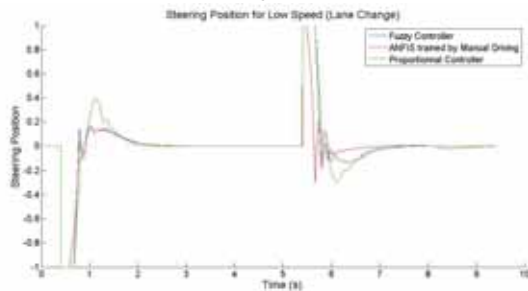
(a) Intersections (lateral error)



(b) Intersections (steering)



(c) Lane change (lateral error)



(d) Lane change (steering)

Fig. 7: Lateral error and Steering at low speed.

A. Test at low speed

Figure 7 shows the lateral error (figures 7a and 7c) and steering (figures 7b and 7d) at constant speed of 20 km/h. In such a way, the figure 7a depicts how the fuzzy controller has a better response reducing the lateral error on the intersections scenario. In the case of the fuzzy trained with ANFIS has a better response reducing the lateral error, but in the other direction is less effective than the PID controller (directly related with the data used for training). In other hand, the figure 7b illustrates the steering for these three controllers and they depict a similar response (in some cases

the fuzzy controllers is a little less noisy).

On figures 7c and 7d are illustrated the lateral error and steering (respectively) on the lane change scenario (at low speed) but there are not big differences between the controllers. However, the fuzzy and ANFIS controllers are more effective to correct the lateral error (faster than the PID).

B. Test at high speed (using speed profile)

Figure 8 shows the lateral error (figures 8a and 8c) and steering (figures 8b and 8d) using a speed profile associated with the curvature of the path and lateral acceleration concepts (related with comfort on driving) with a maximum speed of 50 km/h. The figures 8a and 8b depict the same behavior that in the case of lateral error behavior associated to the intersections scenario at low speed.

Figures 7c and 7d are illustrated the lateral error and steering (respectively) on the lane change maneuver. In this case, the behaviors of the controllers have changed considerably, compared to the low speed scenario. The lateral error has less overshoots in the case of the fuzzy controller trained with ANFIS and it has less abrupt changes on the steering and less overshoots.

VI. CONCLUSION

On the current approach, three controller techniques were tested for the lateral control of a automated vehicle to verify the advantages and disadvantages that each presents.

The PID controller illustrates a good response at low speed. However, PID controller are difficult to be tuned and the gains are not adapted to the speed. Fuzzy logic is useful compared to PID controller because a complex mathematic model is not needed. Fuzzy Logic is intuitive by thanks to using Labels and semantic rules, as humans do.

Besides, using neural methods are powerful because we just need Driver experience to train the system (no mathematical model or label based model). Our controller is generated by the data set from human drivers' experiences, giving a better result.

In the case of the fuzzy controller, it shows a good response with some overshoots on the cases of abrupt changes as on lane change. For the current approach the fuzzy controller was designed just using lateral parameters (lateral error and angular error) but in future works will consider variables of the longitudinal domain as speed (MISO system) to improve the response of the controller (the PID is just presented for SISO systems).

The ANFIS techniques used to train a fuzzy controller could be considered as the best alternative. Admittedly, some results are a bit less effective than the fuzzy controller in our tests. But these performances can be improved due to they used the data gather from an expert driver to emulate the driving behaviors. The time used to develop and tune this controller is really short compared to the time for the other controllers thanks to the use of learning process. In future works, this controller can be improved with a great amount of data. This lateral longitudinal control can also be

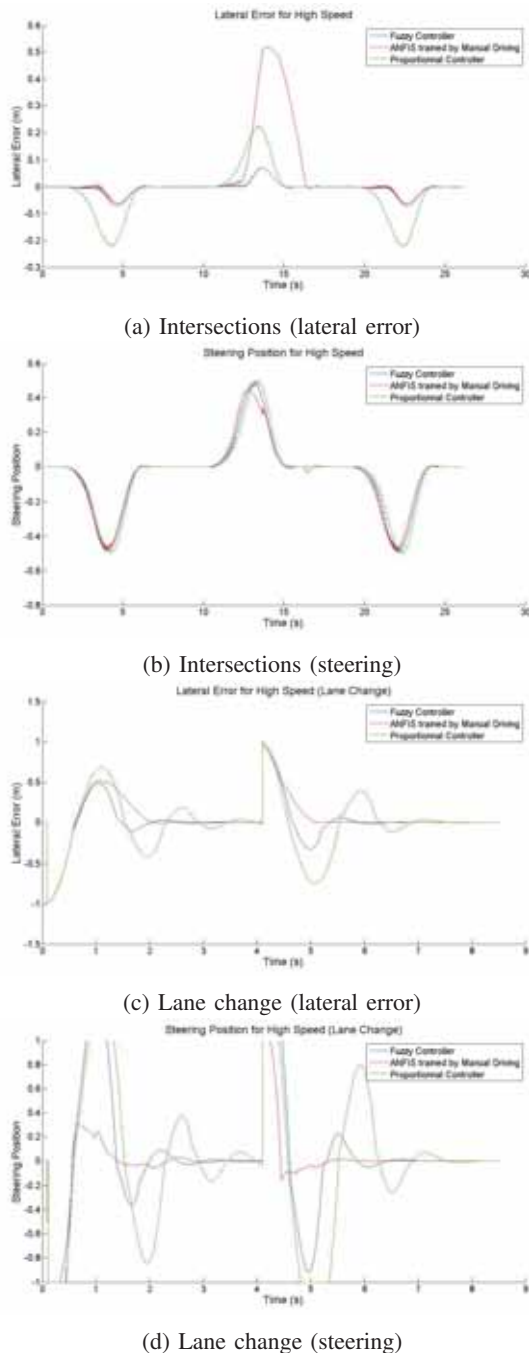


Fig. 8: Lateral error and Steering at high speed.

improved if other variables are taking into account. Finally, a multi-variable system, considering longitudinal variables, and more driving situations may improve our approach.

REFERENCES

- [1] D. Gonzalez and J. Perez, "Control architecture for cybernetic transportation systems in urban environments," *IEEE Intelligent Vehicles Symposium Proceedings (IV)*, 2013.
- [2] L. Li and X. Zhu, "Design concept and method of advanced driver assistance systems," *Fifth conference on measuring technology and mechatronics automation*, pp. 434 – 437, 2013.
- [3] A. Katriniok, J. P. Maschuw, F. Christen, L. Eckstein, and D. Abel, "Optimal vehicle dynamics control for combined longitudinal and lateral autonomous vehicle guidance," in *Control Conference (ECC), 2013 European*, pp. 974–979, July 2013.
- [4] P. Song, C. Zong, and M. Tomizuka, "Combined longitudinal and lateral control for automated lane guidance of full drive-by-wire vehicles," *SAE Int. J. Passeng. Cars Electron. Electr. Syst.*, vol. 8, pp. 419–424, 04 2015.
- [5] M. Sugeno and M. Nishida, "Fuzzy control of model car," *Fuzzy Sets Systems*, vol. 16, no. 2, pp. 103 – 113, 1985.
- [6] J. Perez, V. Milanés, and E. Onieva, "Cascade architecture for lateral control in autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 1, pp. 73 – 82, 2011.
- [7] A. A. Saifizul, M. Z. Zainon, and N. A. A. Osman, "An anfis controller for vision-based lateral vehicle control system," in *2006 9th International Conference on Control, Automation, Robotics and Vision*, pp. 1–4, Dec 2006.
- [8] J. Perez, R. Lattarulo, and F. Nashashibi, "Dynamic trajectory generation using continuous-curvature algorithms for door to door assistance vehicles," *IEEE Intelligent Vehicles Symposium Proceedings (IV)*, 2014.
- [9] V. Milanés, J. Villagra, J. Perez, and C. Gonzalez, "Low-speed longitudinal controllers for mass-produced cars: A comparative study," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 1, pp. 620 – 628, 2012.
- [10] S. B. Choi, "The design of a look-down feedback adaptive controller for the lateral control of front-wheel-steering autonomous highway vehicles," *IEEE Transactions On Vehicular Technology*, vol. 49, no. 6, pp. 2257 – 2269, 2000.
- [11] S. Bennett, "The past of pid controllers," *IFAC Workshop on Digital Control: Past, Present and Future of PID Control*, pp. 1 – 11, 2000.
- [12] H.-S. Tan, F. Bu, and B. Bougler, "A real-world application of laneguidance technologiesautomated snowblower," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 3, pp. 538 – 548, 2007.
- [13] R. White and M. Tomizuka, "Autonomous following lateral control of heavy vehicles using laser scanning radar," *Proceedings of the American Control Conference*, vol. 3, pp. 2333 – 2338, 2001.
- [14] J. Perez, A. Gajate, V. Milanés, E. Onieva, and M. Santos, "Design and implementation of a neuro-fuzzy system for longitudinal control of autonomous vehicles," *IEEE International Conference on Fuzzy Systems*, 2010.
- [15] C. Quek, M. Pasquier, and B. Lim, "A novel self-organizing fuzzy rule-based system for modelling traffic flow behaviour," *Expert Systems with Applications*, vol. 36, no. 10, pp. 12167 – 12178, 2009.
- [16] K. M. Udofia and J. O. Emagbeter, "Development of multi-agent anfis-based model for urban traffic signal control," in *2013 International Conference on Connected Vehicles and Expo (ICCVE)*, pp. 662–669, Dec 2013.
- [17] R. Lattarulo, J. Perez, and M. Dendaluce, "A complete framework for developing and testing automated driving controllers," *Preprints of the 20th World Congress The International Federation of Automatic Control*, 2017.
- [18] I. Iglesias, "Herramienta para acelerar el desarrollo de nuevos sistemas y controles para automocin," *Revista Automtica e Instrumentacin*, vol. 478, pp. 45 – 49, 2015.
- [19] R. Rajamani, *Vehicle Dynamics and Control*. 2012.
- [20] I. I. Aguinaga, A. M. Sandi, and A. P. Rodriguez, "Vehicle modelling for real time systems application. the virtual rolling chassis," *Revista Dyna*, vol. 88, pp. 206 – 215, 2013.

Asynchronous Multi-Sensor Fusion for 3D Mapping and Localization

Patrick Geneva, Kevin Eickenhoff, and Guoquan Huang

Abstract—In this paper, we address the problem of 3D mapping and localization of autonomous vehicles while focusing on optimally fusing multiple heterogeneous and asynchronous sensors. To this end, based on the factor graph-based optimization framework, we design a modular sensor-fusion system that allows for efficient and accurate incorporation of any navigation sensor of different sampling rates. In particular, we develop a general method of out-of-sequence (asynchronous) measurement alignment to incorporate heterogeneous sensors into a factor graph for mapping and localization in 3D, without requiring the addition of new graph nodes, thus allowing the graph to have an overall reduced complexity. The proposed sensor-fusion system is validated on a collected dataset, in which the asynchronous-measurement alignment is shown to have an improved performance when compared to a naive approach without alignment.

I. INTRODUCTION

Autonomous driving is an emerging technology that enables the reduction of traffic accidents and allows for those who are unable to drive for various medical conditions to regain their independence, by performing intelligent perception and planning based on multimodal sensors such as LIDARs, cameras, IMUs and GPS. It is critical for an autonomous vehicle to perform precise, robust localization for decision making as it is a sub-system that cannot fail during online autonomous operation. There have been a large amount of research efforts focused on multi-sensor fusion for state estimation for localization [20], which has reached a certain level of maturity, yielding a bounded problem given the well structured environment a vehicle operates in. In particular, graph-based optimization has recently prevailed for robot mapping and localization [2]. Due to the different sampling rates of the heterogeneous sensors, measurements arrive at different times. Accurate alignment of such out-of-sequence (i.e., asynchronous) measurements before optimally fusing them through graph optimization, while essential, has not been sufficiently investigated in the literature.¹

Factor graph-based formulation [6] is desirable due to its ability to allow for the delayed incorporation of asynchronous measurements. Indelman et al. [11] address the

problem of the inclusion of asynchronous measurements by taking advantage of IMU preintegrated terms. This allows them to incorporate any set of asynchronous sensors whose rates are longer than that of the IMU. Sünderhauf et al. [18] looked to address the incorporation of measurements with unknown time delays. Using high frequency odometry measurements, they create a state for *each* incoming odometry measurement so that delayed factors can be directly connected to its closest state. While both of these can be used to address arbitrary amounts of delay between sensors, they add a large amount of additional factors and edges to the graph. In contrast, the proposed approach incorporates measurements of different frequencies without significant increase of the overall graph complexity. It should be noted that while this does reduce the computational cost of optimization, reductions in graph size are always welcomed as a robot’s physical memory becomes less of an issue.

Specifically, as the main contribution of this paper, we accurately align both asynchronous unary and binary graph factors based on our analytically derived linear 3D pose interpolation. This interpolation allows for the direct addition of asynchronous measurements into the graph, without the need for extra nodes to be added or for the naive ignoring of the measurement delay. Patron-Perez et al. [1] first proposed a spline-based trajectory method that allows for the fusion of delayed measurements with the consequence of an increase of overall system complexity and deviation from a pure pose graph. Outside of graph-based optimization, interpolation has been used to correct time offsets of continuous measurements such as LIDAR point clouds and rolling shutter cameras [3, 10]. In particular, Guo et al. [10] introduce the idea of linear interpolation between past camera poses, which allow for the use of extracted features from rolling shutter cameras. Ceriani et al. [3] use a linear interpolation between two poses in $SE(3)$ to unwarp lidar point measurements. In this work, we focus on the use of such linear interpolation in the graph-based optimization framework to allow for the efficient alignment of asynchronous measurements.

From the system perspective, we design and implement a modular framework for fusing a variety of sensors, where we separate the sensor fusion and pose estimation to allow for any sensor to be incorporated. This system design allows for the easy incorporation of additional sensors, while allowing for active sensor pose estimation modules to be changed without affecting the multi-sensor fusion. This is achieved by fusing emitted 3D pose estimates from sensor odometry (ego-motion) modules. The proposed sensor framework can then leverage these 3D poses, emitted in their own local frame of

This work was partially supported by the University of Delaware College of Engineering, UD Cybersecurity Initiative, the Delaware NASA/EPSCoR Seed Grant, the NSF (IIS-1566129), and the DTRA (HDTRA1-16-1-0039).

The authors are with the Department of Mechanical Engineering, University of Delaware, Newark, DE 19716, USA. Email: {pgeneva, keck, ghuang}@udel.edu

¹It should be noted that the asynchronous measurement alignment under consideration is different from the time synchronization (or temporal sensor calibration) [19]; that is, even if sensors are well synchronized, their observations still arrive asynchronously.

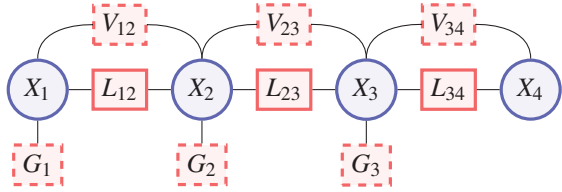


Fig. 1: Example factor graph that our system created. States that will be estimated are denoted in circles and measurements are denoted in squares. Note: we differentiate interpolated factors through the dashed outlines.

reference, in the global estimates of the robot.

II. GRAPH-BASED ESTIMATION

As the vehicle moves through the environment, a set of measurements, \mathbf{z} , is collected from its sensors, such as LIDAR scans, images, GPS, etc. These measurements relate to the underlying state to be estimated, \mathbf{x} . This process can be represented by a graph, where nodes correspond to parameters to be estimated (i.e., historical vehicle poses). Incoming measurements are represented as edges connecting their involved nodes (see Figure 1). Under the assumption of independent Gaussian noise corruption of our measurements, we formulate the Maximum Likelihood Estimation (MLE) problem as the following nonlinear least-squares problem [13]:

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x}} \sum_i \|\mathbf{r}_i(\mathbf{x})\|_{\mathbf{P}_i}^2 \quad (1)$$

Here, \mathbf{r}_i is the zero-mean residual associated with measurement i , \mathbf{P}_i is the measurement covariance, and $\|\mathbf{v}\|_{\mathbf{P}}^2 = \mathbf{v}^\top \mathbf{P}^{-1} \mathbf{v}$ is the energy norm. This problem can be solved iteratively by linearizing about the current linearization point, $\hat{\mathbf{x}}^-$, and defining a new optimization problem in terms of the error state, $\Delta \mathbf{x}$:

$$\Delta \mathbf{x}^- = \arg \min_{\Delta \mathbf{x}} \sum_i \|\mathbf{r}_i(\hat{\mathbf{x}}^-) + \mathbf{H}_i \Delta \mathbf{x}\|_{\mathbf{P}_i}^2 \quad (2)$$

Where $\mathbf{H}_i = \frac{\partial \mathbf{r}_i(\hat{\mathbf{x}}^- \boxplus \Delta \mathbf{x})}{\partial \Delta \mathbf{x}}$ is the Jacobian of i -th residual with respect to the error state. We define the generalized update operation, \boxplus , which maps a change in the error state to one in the full state. Given the error state $\{^i_G \boldsymbol{\theta}, ^G \tilde{\mathbf{p}}_i\}$, this update operation can be written as $\{\text{Expv}(-^i_G \boldsymbol{\theta}) ^i_G \mathbf{R}, ^G \mathbf{p}_i + ^G \tilde{\mathbf{p}}_i\}$.² After solving the linearized system, the current linearization point is updated as $\hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- \boxplus \Delta \mathbf{x}^-$. In this work, we parameterize the pose of each time step as $\{^i_G \mathbf{R}, ^G \mathbf{p}_i\}$, which describes the rotation from the global frame $\{G\}$ to the local frame $\{i\}$ and the position of the frame $\{i\}$ seen from the global frame $\{G\}$ of reference. This linearization process is then repeated until convergence. While there are openly available solvers [5, 12, 13], the computational complexity of the graph based optimization can reach $O(n^3)$ in the worst case.

²Throughout the paper we denote the vector form of the matrix exponential as $\text{Expv}(\cdot)$ which maps $\mathbb{R}^3 \rightarrow SO(3)$, (e.g. $\text{Expv}(^G \boldsymbol{\theta}) = ^G \mathbf{R}$). Similarly, we define the matrix logarithm $\text{Logv}(\cdot)$ to map between $SO(3) \rightarrow \mathbb{R}^3$, (e.g. $\text{Logv}(^G \mathbf{R}) = ^G \boldsymbol{\theta}$).

A reduction in the number of states being estimated can both help with the overall computational complexity and the physical size of a graph during long term SLAM. Naively, if a new node is to be created at each sensor measurement time instance, the overall graph optimization frequency can suffer. To prolong high frequency graph optimization, we present our novel method of measurement alignment which allows for the estimation of the poses of a *single* sensor's measurements.

III. ASYNCHRONOUS MEASUREMENT ALIGNMENT

A. Unary Factors

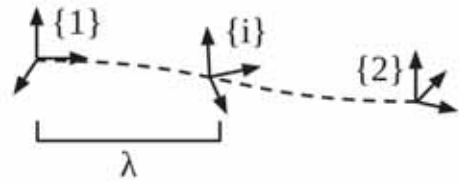


Fig. 2: Given two measurements in the global frame of reference $\{1\}$ and $\{2\}$, we interpolate to a new pose $\{i\}$. The above λ is the time-distance fraction that defines how much to interpolate the pose.

Unary factors can appear when sensors measure information in respect to a single node. For example, GPS can provide global position measurements indirectly through latitude, longitude, and altitude readings, while LIDAR scan-matching to known maps can provide a direct reading of the global pose. Motivated to not add new graph nodes when receiving asynchronous data, we add a “corrected” measurement to an existing node by performing pose interpolation between two sequential sensor measurements. Note: that for GPS measurements we only need to perform 3D position interpolation, however for completeness we have derived the following interpolation for a 3D pose. We define a time-distance fraction between two consecutive poses as follows:

$$\lambda = \frac{(t_i - t_1)}{(t_2 - t_1)} \quad (3)$$

where t_1 and t_2 are the timestamps of the bounding measurements, and t_i is the desired interpolation time (i.e. the timestamp of the existing node). Under the assumption of a constant velocity motion model, we interpolate between the two pose readings:

$$^i_G \mathbf{R} = \text{Expv}\left(\lambda \text{Logv}(^2_G \mathbf{R}_1^G \mathbf{R}_2^G \mathbf{R}_1^\top)\right) ^1_G \mathbf{R} \quad (4)$$

$$^G \mathbf{p}_i = (1 - \lambda) ^G \mathbf{p}_1 + \lambda ^G \mathbf{p}_2 \quad (5)$$

where $\{^i_G \mathbf{R}, ^G \mathbf{p}_i\}$ is the interpolated measurement 3D pose and $\{^1_G \mathbf{R}, ^G \mathbf{p}_1\}$ and $\{^2_G \mathbf{R}, ^G \mathbf{p}_2\}$ are the bounding poses. While this interpolated measurement can now be directly added to the graph, the last step is to correctly compute the corresponding covariance needed in graph-based optimization.

Hence, we perform the following covariance propagation:

$$\mathbf{P}_i = \mathbf{H}_u \mathbf{P}_{1,2} \mathbf{H}_u^T \quad (6)$$

$$\mathbf{H}_u = \begin{bmatrix} \frac{\partial^i \tilde{\boldsymbol{\theta}}}{\partial^1 \tilde{\boldsymbol{\theta}}} & \mathbf{0}_{3 \times 3} & \frac{\partial^i \tilde{\boldsymbol{\theta}}}{\partial^2 \tilde{\boldsymbol{\theta}}} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \frac{\partial^G \tilde{\mathbf{p}}_i}{\partial^G \tilde{\mathbf{p}}_1} & \mathbf{0}_{3 \times 3} & \frac{\partial^G \tilde{\mathbf{p}}_i}{\partial^G \tilde{\mathbf{p}}_2} \end{bmatrix} \quad (7)$$

where $\mathbf{P}_{1,2}$ is the joint covariance matrix from the bounding poses, and $\tilde{\boldsymbol{\theta}}$ and $\tilde{\mathbf{p}}$ are the error states of each angle and position measurement, respectively. For detailed calculations of all Jacobians derived in this paper, we refer the reader to the companion tech report [9]. The resulting non-zero Jacobian matrix entries are defined as:

$$\frac{\partial^i \tilde{\boldsymbol{\theta}}}{\partial^1 \tilde{\boldsymbol{\theta}}} = -{}^i \mathbf{R} \left(J_r(\lambda \text{Logv}({}^2 \mathbf{R})) \lambda J_r^{-1}(\text{Logv}({}^2 \mathbf{R})) - \mathbf{I} \right) \quad (8)$$

$$\frac{\partial^i \tilde{\boldsymbol{\theta}}}{\partial^2 \tilde{\boldsymbol{\theta}}} = {}^i \mathbf{R} J_r(-\lambda \text{Logv}({}^2 \mathbf{R}^T)) \lambda J_r^{-1}(\text{Logv}({}^2 \mathbf{R}^T)) \quad (9)$$

$$\frac{\partial^G \tilde{\mathbf{p}}_i}{\partial^G \tilde{\mathbf{p}}_1} = (1 - \lambda) \mathbf{I}, \quad \frac{\partial^G \tilde{\mathbf{p}}_i}{\partial^G \tilde{\mathbf{p}}_2} = \lambda \mathbf{I} \quad (10)$$

where the Right Jacobian of $SO(3)$ denoted as $J_r(\phi)$ and its inverse $J_r^{-1}(\phi)$ is defined as the following [4, 8]:

$$J_r(\phi) = \mathbf{I} - \frac{1 - \cos(\|\phi\|)}{\|\phi\|^2} [\phi \times] + \frac{\|\phi\| - \sin(\|\phi\|)}{\|\phi\|^3} [\phi \times]^2 \quad (11)$$

$$J_r^{-1}(\phi) = \mathbf{I} + \frac{1}{2} [\phi \times] + \left(\frac{1}{\|\phi\|^2} - \frac{1 + \cos(\|\phi\|)}{2\|\phi\| \sin(\|\phi\|)} \right) [\phi \times]^2 \quad (12)$$

B. Binary Factors

Designing multi-sensor systems for estimation often requires fusing asynchronous odometry readings from different sensor modules (e.g., ORB-SLAM2 [14] or LOAM [21]). A difficulty that arises is the unknown transformation between the global frame of references of each module. This unknown comes from both the rigid transformation between sensors (which can be found through extrinsic calibration) and each module *initializes* its global frame of reference independently. Rather than directly modifying the codebase of each module, we combine the sequential odometry measurements into *relative* transforms; thereby, we remove the ambiguity of each module-to-module transformation.

In particular, given two poses in the second sensor's world frame, $\{{}^1 \mathbf{R}, {}^o \mathbf{p}_1\}$ and $\{{}^2 \mathbf{R}, {}^o \mathbf{p}_2\}$ with the joint covariance $\mathbf{P}_{1,2}$, we calculate the relative transformation as follows:

$${}^2 \mathbf{R} = {}^2 \mathbf{R}_o \mathbf{R}^T \quad (13)$$

$${}^1 \mathbf{p}_2 = {}^o \mathbf{R} ({}^o \mathbf{p}_2 - {}^o \mathbf{p}_1) \quad (14)$$

where we define the unknown global frame of these 3D pose measurements as $\{o\}$ and their corresponding reference frames as $\{1\}$ and $\{2\}$. To calculate the relative covariance matrix, we perform the following covariance propagation based on the above measurement transformation:

$$\mathbf{P}_{12} = \mathbf{H}_r \mathbf{P}_{1,2} \mathbf{H}_r^T \quad (15)$$

where $\mathbf{P}_{1,2}$ is the joint covariance matrix of each pose in the $\{o\}$ frame of reference. The resulting Jacobian matrix \mathbf{H}_r is defined as the following:

$$\mathbf{H}_r = \begin{bmatrix} -{}^2 \mathbf{R} & \mathbf{0}_{3 \times 3} & \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ [{}^1 \mathbf{R} ({}^o \mathbf{p}_2 - {}^o \mathbf{p}_1) \times] & -{}^o \mathbf{R} & \mathbf{0}_{3 \times 3} & {}^o \mathbf{R} \end{bmatrix} \quad (16)$$

We now have the $\{{}^2 \mathbf{R}, {}^1 \mathbf{p}_2\}$ relative transform between two poses and corresponding covariance \mathbf{P}_{12} . If this transformation is not in the same sensor frame of reference (e.g., relative transform is between camera to camera and the state is LIDAR to LIDAR), one can use the method described in Appendix A to convert the measurement into the frame of reference of the state.

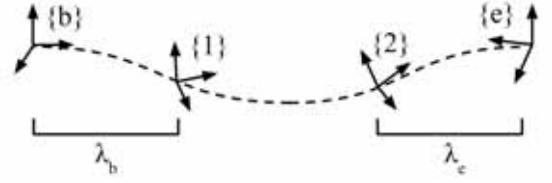


Fig. 3: Given a relative transformation, calculated using (13) and (14), between the $\{1\}$ and $\{2\}$ frame of reference, we extrapolate this relative transformation to the desired beginning $\{b\}$ and end $\{e\}$ poses. The above λ s are the time-distance fractions that we use to extrapolate the relative transformation.

Due to the asynchronous nature of the measurements from two different sensors, the times corresponding to the beginning and end of the relative transformation will not align with matched existing state poses. Therefore, under the assumption of a constant velocity motion, we *extrapolate* the relative transformation across the desired interval. This intuitively corresponds to a “stretching” of the relative pose measurement in time. We define two time-distance fractions that determine how much the relative transformation needs to be extended (see Figure 3):

$$\lambda_b = \frac{t_1 - t_b}{t_2 - t_1}, \quad \lambda_e = \frac{t_e - t_2}{t_2 - t_1} \quad (17)$$

The λ 's describe the magnitude that the relative transformation is to be “stretched” in each direction, with the subscripts b and e denoting the beginning and end state poses. These time-distance fractions can also be negative, corresponding to the “shrinking” of the relative transformation. Given the relative transform and the time-distance fractions, we define the following extrapolation equations:

$${}^e \mathbf{R} = \text{Expv}[(1 + \lambda_b + \lambda_e) \text{Logv}({}^2 \mathbf{R})] \quad (18)$$

$${}^b \mathbf{p}_e = (1 + \lambda_b + \lambda_e) \text{Expv}[-\lambda_b \text{Logv}({}^2 \mathbf{R})] {}^1 \mathbf{p}_2 \quad (19)$$

The covariance propagation is then given by:

$$\mathbf{P}_{be} = \mathbf{H}_i \mathbf{P}_{12} \mathbf{H}_i^T, \quad \mathbf{H}_i = \begin{bmatrix} \frac{\partial^e \tilde{\boldsymbol{\theta}}}{\partial^2 \tilde{\boldsymbol{\theta}}} & \mathbf{0}_{3 \times 3} \\ \frac{\partial^b \tilde{\mathbf{p}}_e}{\partial^1 \tilde{\boldsymbol{\theta}}} & \frac{\partial^b \tilde{\mathbf{p}}_e}{\partial^1 \tilde{\mathbf{p}}_2} \end{bmatrix} \quad (20)$$

where \mathbf{P}_{12} is the relative factor covariance calculated above. The resulting non-zero Jacobian matrix entries are defined as:

$$\frac{\partial^e \hat{\boldsymbol{\theta}}}{\partial^2 \hat{\boldsymbol{\theta}}} = \mathbf{J}_r \left[(1 + \lambda_b + \lambda_e) \text{Logv}(\hat{\mathbf{R}}^\top) \right] (1 + \lambda_b + \lambda_e) \mathbf{J}_r^{-1} \left[\text{Logv}(\hat{\mathbf{R}}^\top) \right] \quad (21)$$

$$\frac{\partial^b \hat{\mathbf{p}}_e}{\partial^1 \hat{\boldsymbol{\theta}}} = \left(- (1 + \lambda_b + \lambda_e) \text{Expv} \left[\lambda_b \text{Logv}(\hat{\mathbf{R}}^\top) \right] \left[\mathbf{1} \hat{\mathbf{p}}_2 \times \right] \right. \\ \left. \mathbf{J}_r(\lambda_b \text{Logv}(\hat{\mathbf{R}}^\top)) \lambda_b \mathbf{J}_r^{-1}(\text{Logv}(\hat{\mathbf{R}}^\top)) \right) \quad (22)$$

$$\frac{\partial^b \hat{\mathbf{p}}_e}{\partial^1 \hat{\mathbf{p}}_2} = (1 + \lambda_b + \lambda_e) \text{Expv} \left[-\lambda_b \text{Logv}(\hat{\mathbf{R}}) \right] \quad (23)$$

IV. SYSTEM DESIGN

A. Design Motivations

The proposed method allows for the reduction of the overall graph complexity during asynchronous sensor fusion. We now propose a system that leverages the use of asynchronous sensors in the application of autonomous driving. To both facilitate the flexibility of the vehicle design and reduce cost, we aim to run the system on a vehicle *without* access to a GPS unit and with low cost asynchronous sensors (i.e., without the use of electronic triggering). This design constraint presents the unique challenge of still needing to localize the vehicle in the GPS frame of reference without the use of a traditional GPS sensor. By publishing the vehicle state estimate in the GPS frame of reference, we allow for existing global path planning and routing modules to continue to work as expected. To overcome this challenge, we present a unique prior LIDAR map that allows for the vehicle to both initialize and localize in the GPS frame of reference. Specifically we design a framework with two separate sub-systems as follows:

- Creation of an accurate prior map using a vehicle that has an additional Real Time Kinematic (RTK) GPS sensor unit.
- Leverage the prior map in GPS denied localization to determine the 3D pose in the GPS frame of reference.

This framework is flexible and cost effective as only a single “collection” vehicle is needed to build the prior map that multiple lower cost vehicles can leverage. Specifically, this prior map allows for localization in the GPS frame of reference without the use of GPS measurements during runtime and can support localization in GPS denied environments (e.g., tunnels or parking garages). Both sub-systems can leverage the proposed asynchronous factor interpolation to enable the use of low cost asynchronous sensors while ensuring a reduction of overall graph complexity.

B. System Overview - Prior Map

The first sub-system we propose is one that generates an accurate prior map that can be leveraged by the second sub-system to localize in the GPS frame of reference. Shown in Figure 4, we fuse odometry measurements from openly available stereo and LIDAR modules, ORB-SLAM2 [14] and LOAM [21], respectively, with a RTK GPS unit. Both of these modules provide six degree of freedom pose

estimation.³⁴ We estimate LIDAR states connected with consecutive non-interpolated binary factors from LOAM LIDAR odometry. To provide additional robustness and information into the graph, we connect consecutive states with interpolated binary factors (Section III-B) from ORB-SLAM2 visual odometry. To ensure that the estimated states are in the GPS frame of reference, we attach interpolated unary factors (Section III-A) from the RTK GPS sensor. Both ORB-SLAM2 visual binary factors and RTK GPS unary factors need to be interpolated because both sensors are asynchronous to the LIDAR sensor.

The graph can be solved in real-time using an incremental solver such as iSAM2 [12] or offline with a full batch solver. It is then simple to construct a prior map using the estimated states and their corresponding LIDAR point clouds. To evaluate the overall quality of the generated prior map point cloud, the cloud is visually inspected for misalignment on environmental planes such as walls or exterior of buildings. The generated prior map from the experimental dataset can be see in Figure 5.

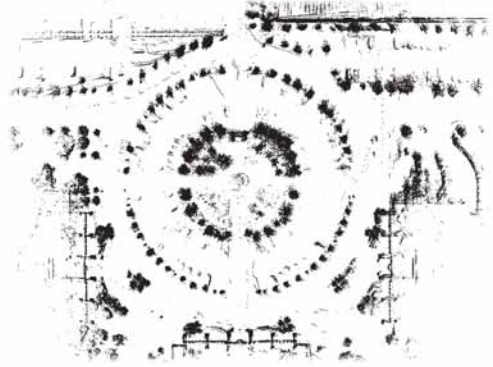


Fig. 5: Prior map generated from the experimental dataset

C. System Overview - GPS Denied Localization

Using the generated prior map, localization in the GPS frame can be preformed *without* the use of a GPS sensor. As seen in Figure 4, we estimate LIDAR states that are connected with non-interpolated and interpolated binary factors (Section III-B) from LOAM and ORB-SLAM2 odometry modules, respectively. In addition to these two binary factors, we preform Iterative Closest Point (ICP) matching between the newest LIDAR point cloud to the generated prior map. This ICP transform can then be added as a non-interpolated unary factor into the factor graph. These unary factors constrain the graph to be in the GPS frame of reference during 3D pose estimation.

To provide real-time localization capabilities, we leverage the iSAM2 solver during GPS denied state estimation. The

³Note: both modules do *not* normally provide a corresponding covariance needed for batch optimization. We reference the reader to the appendices in the companion tech report [9].

⁴We run LOAM in the default mode, while ORB-SLAM2 is run first in “mapping” mode to generate a map of the environment. We then use ORB-SLAM2 in “localization” mode to ensure that the estimate does not jump due to loop closures when fusing its output with the proposed systems.

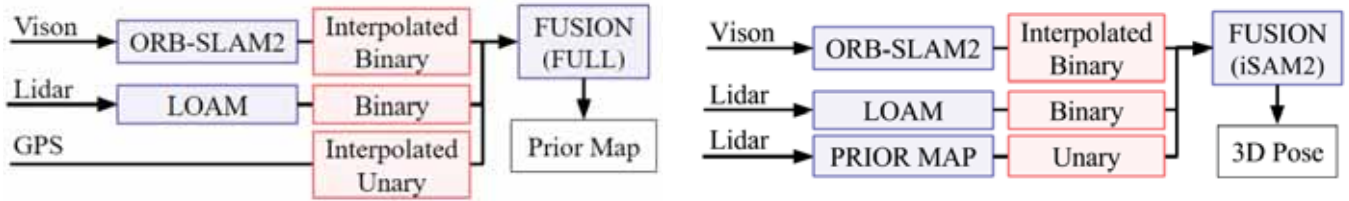


Fig. 4: Overview of the flow of data through the system, where all incoming measurements are denoted on the far left of each figure. These measurements are first processed through an odometry module if needed (seen in blue) and then converted into factors (seen in red) that can be added to factor graph. The prior map system (left) leverages RTK GPS measurements to create a prior map in the GPS frame of reference. The GPS denied estimation system (right) uses the generated LIDAR maps to ensure that the pose estimation is in the GPS frame of reference.

estimation operates at the frequency of the LIDAR sensor limited only by the speed the LOAM module can process measurements. It was found that when creating a unary factor using ICP matching to the prior map took upwards of 1-2 seconds. To overcome this long computation time, incoming LIDAR clouds are processed at a lower frequency in a secondary thread, and then added to the graph after successful ICP matching.

V. EXPERIMENTAL RESULTS

A. System Validation

To assess the overall performance of the GPS denied system, we constructed a data collection vehicle with both a combination of low cost sensors and a RTK GPS sensor. The vehicle is equipped with a 8 channel Quanergy M8 LIDAR [16], ZED stereo camera [17], and RTK enabled NovAtel Propak6 GPS sensor [15]. The Quanergy M8 LIDAR was run at 10Hz, while the ZED stereo camera was run at 30Hz with a resolution of 672 by 376. The RTK enabled NovAtel Propak6 GPS sensor operated at 20Hz with an average accuracy of ± 15 centimeters. The GPS solution accuracy allows for the creation of a high quality prior map (see Figure 5). To facilitate the GPS denied system, a dataset was first collected on the vehicle and then processed using a full batch solver. Following the proposed procedure in Section IV-B, LIDAR factors are added to the factor graph, while both stereo and GPS factors are interpolated and then directly connected to corresponding LIDAR states. The resulting LIDAR point cloud, created in the GPS frame of reference, can then be used during GPS denied navigation.

To represent the real world, the GPS denied system was tested on the day following the data collection for the prior map. This was to introduce changes in the environment, such as changes in car placement and shrubbery, while also showing that the prior map can still be leveraged. The same vehicle was used with the only difference being that the RTK GPS was not used in the GPS denied localization. This RTK GPS was instead used to provide an accurate ground truth comparison. Following the proposed procedure in Section IV-C, incoming LIDAR point clouds are matched to the map generated the previous day and then added to the factor graph after successful ICP alignment.

The estimated vehicle state is compared to the corresponding output of the RTK GPS. As seen in Figure 6, when performing GPS denied localization, the system was able to remain within a stable 2 meter accuracy.

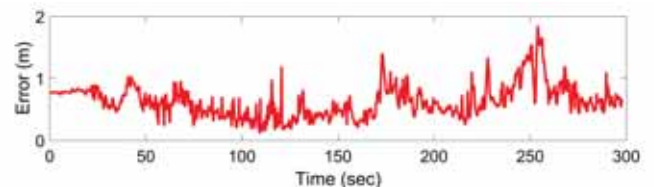


Fig. 6: Average position error magnitude over 10 runs. GPS denied estimation compared at each time instance, of the 840 meter long run, with the RTK GPS position. Average vehicle speed of 6mph.

B. Evaluating the Asynchronous Measurement Alignment

Having shown that the system is able to accurately localize in real-time without the use of GPS, we next evaluated how the interpolation impacts the estimation. To do so, we did not use the ICP matching to the LIDAR prior cloud and instead only used the pure odometry from LOAM and ORB-SLAM2. We compared the proposed factor interpolation method against a naive approach of factor addition into the graph which ignores the issue of time delay and directly attaches incoming factors to the closest nodes without interpolation.

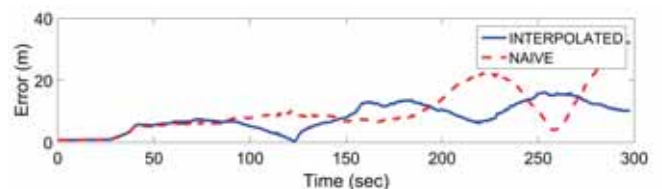


Fig. 7: Comparison of the proposed method and a naive approach of adding incoming factors to the closest nodes, denoted as “interpolation” and “naive” respectively.

Seen in Figure 7, the proposed factor interpolation outperformed the estimation accuracy of the naive approach. The *average* error of the naive approach was 10.24 meters and the proposed method’s average error is 8.00 meters (overall

21.8% decrease). This shows that the use of interpolation on incoming binary factors can greatly increase the estimation accuracy, without increasing graph complexity.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we have developed a general approach of asynchronous measurement alignment within the graph-based optimization framework of mapping and localization in order for optimal fusion of multimodal sensors. The designed framework provides a modular system with the ability to replace individual modules and allow for any sensor to be incorporated. The system has been tested on a experimental dataset and compared to a naive approach to show the improvement due to the proposed asynchronous measurement alignment. Looking forward, we will incorporate other sensors, such as Inertial Measurement Units (IMUs), through the use of IMU preintegration developed in our prior work [7] to improve the system fault tolerance, if the main sensor fails. We will also investigate how to improve the current mapping and localization, in particular, when autonomously driving in dynamic urban environments.

APPENDIX

A. Static Transformations

Another issue that commonly arises in the application of multi-sensor fusion is the ability to convert from one frame of reference to another. For example, in this paper, binary factors from the external ORB-SLAM2 visual odometry library. Given this relative transform $\{C_1^2 \mathbf{R}, C_1^1 \mathbf{p}_{C2}\}$ in the camera frame and a corresponding covariance \mathbf{P}_{C12} , we would like to transform from the camera to the LIDAR sensor frame. This can be done as follows:

$${}^{L2}_{L1} \mathbf{R} = {}^L_C \mathbf{R} {}^{C2}_{C1} \mathbf{R} {}^L_C \mathbf{R}^\top \quad (24)$$

$${}^{L1}_{L2} \mathbf{p}_{L2} = {}^L_C \mathbf{R} \left({}^{C2}_{C1} \mathbf{R}^\top {}^C \mathbf{p}_L + {}^{C1}_{C2} \mathbf{p}_{C2} - {}^C \mathbf{p}_L \right) \quad (25)$$

where we define the LIDAR frame of reference as $\{Li\}$, $i \in \{1, 2\}$ and the camera frame of reference as $\{Ci\}$, $i \in \{1, 2\}$. It is assumed that the static transform, $\{{}^L_C \mathbf{R}, {}^C \mathbf{p}_L\}$, from the LIDAR to camera frame of reference are known from offline calibration. Given the above transform, special care needs to be taken to calculate the relative covariance matrix in the LIDAR frame of reference as follows:

$$\mathbf{P}_{L12} = \mathbf{H}_s \mathbf{P}_{C12} \mathbf{H}_s^\top \quad (26)$$

where \mathbf{P}_{C12} is the relative camera covariance. For detailed calculations of this Jacobian, please see the companion tech report [9]. The resulting Jacobian matrix \mathbf{H}_s is defined as the following:

$$\mathbf{H}_s = \begin{bmatrix} {}^L_C \mathbf{R} & \mathbf{0}_{3 \times 3} \\ -{}^L_C \mathbf{R} {}^{C2}_{C1} \mathbf{R}^\top [{}^C \mathbf{p}_L \times] & {}^L_C \mathbf{R} \end{bmatrix} \quad (27)$$

REFERENCES

- [1] Steven Lovegrove, Alonso Patron-Perez, and Gabe Sibley. "A Spline-Based Trajectory Representation for Sensor Fusion and Rolling Shutter Cameras". In: *International Journal of Computer Vision* 113.3 (2015), pp. 208–219. ISSN: 1573-1405. DOI: 10.1007/s11263-015-0811-3.
- [2] Cesar Cadena et al. "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age". In: *IEEE Transactions on Robotics* 32.6 (2016), pp. 1309–1332.
- [3] Simone Ceriani et al. "Pose interpolation SLAM for large maps using moving 3D sensors". In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 750–757.
- [4] Gregory S Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*. Vol. 2. Springer Science & Business Media, 2011.
- [5] Frank Dellaert. "Factor graphs and GTSAM: A hands-on introduction". In: (2012).
- [6] Frank Dellaert and M. Kaess. "Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing". In: *International Journal of Robotics Research* 25.12 (Dec. 2006), pp. 1181–1203.
- [7] Kevin Eickenhoff, Patrick Geneva, and Guoquan Huang. "High-Accuracy Preintegration for Visual-Inertial Navigation". In: *Proc. of International Workshop on the Algorithmic Foundations of Robotics*. San Francisco, CA, Dec. 2016.
- [8] Christian Forster et al. "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation". In: *Robotics: Science and Systems*. Georgia Institute of Technology, 2015.
- [9] Patrick Geneva, Kevin Eickenhoff, and Guoquan Huang. *Asynchronous Multi-Sensor Fusion for 3D Mapping and Localization*. Tech. rep. RPNG-2017-002. Available: http://udel.edu/~ghuang/papers/tr_async.pdf. University of Delaware, 2017.
- [10] Chao X Guo et al. "Efficient Visual-Inertial Navigation using a Rolling-Shutter Camera with Inaccurate Timestamps." In: *Robotics: Science and Systems*. Citeseer, 2014.
- [11] Vadim Indelman et al. "Information fusion in navigation systems via factor graph based incremental smoothing". In: *Robotics and Autonomous Systems* 61.8 (2013), pp. 721–738.
- [12] M. Kaess et al. "iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering". In: *2011 IEEE International Conference on Robotics and Automation*. May 2011, pp. 3281–3288. DOI: 10.1109/ICRA.2011.5979641.
- [13] R. Kummerle et al. "g2o: A general framework for graph optimization". In: *Proc. of the IEEE International Conference on Robotics and Automation*. Shanghai, China, May 2011, pp. 3607–3613.
- [14] Raul Mur-Artal and Juan D Tardos. "ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras". In: *arXiv preprint arXiv:1610.06475* (2016).
- [15] Novatel. *ProPak6 Triple-Frequency GNSS Receiver*. <https://www.novatel.com/assets/Documents/Papers/ProPak6-PS-D18297.pdf>.
- [16] Inc. Quanergy Systems. *Quanergy M8 LIDAR*. <http://quanergy.com/m8/>.
- [17] Stereolabs. *ZED Stereo Camera*. <https://www.stereolabs.com/zed/specs/>.
- [18] Niko Sünderhauf, Sven Lange, and Peter Protzel. "Incremental Sensor Fusion in Factor Graphs with Unknown Delays". In: *Advanced Space Technologies in Robotics and Automation (ASTRA)*. 2013.
- [19] Z. Taylor and J. Nieto. "Motion-Based Calibration of Multimodal Sensor Extrinsic and Timing Offset Estimation". In: *IEEE Transactions on Robotics* 32.5 (Oct. 2016), pp. 1215–1229.
- [20] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, 2005.
- [21] Ji Zhang and Sanjiv Singh. "LOAM: Lidar Odometry and Mapping in Real-time." In: *Robotics: Science and Systems*. Vol. 2. 2014.

Towards Cooperative Motion Planning for Automated Vehicles in Mixed Traffic

Maximilian Naumann¹ and Christoph Stiller^{1,2}

Abstract—While motion planning techniques for automated vehicles in a reactive and anticipatory manner are already widely presented, approaches to cooperative motion planning are still remaining. In this paper, we present an approach to enhance common motion planning algorithms, that allows for cooperation with human-driven vehicles. Unlike previous approaches, we integrate the prediction of other traffic participants into the motion planning, such that the influence of the ego vehicle’s behavior on the other traffic participants can be taken into account. For this purpose, a new cost functional is presented, containing the cost for all relevant traffic participants in the scene. Finally, we propose a path-velocity-decomposing sampling-based implementation of our approach for selected scenarios, which is evaluated in a simulation.

I. INTRODUCTION

In the field of intelligent vehicles, tremendous progress has been achieved in the last decades [1]. With the first successful experiments of close-to-production cars in real traffic [2], automated driving has gained more and more attention in public.

In order to improve the reliability and thus the safety of automated vehicles, but also to increase their efficiency, *cooperation* is focused on in recent research. Here, cooperation through explicit communication of (fused) sensor information and desired driving behaviour [3] as well as negotiation of possible solutions [4], [5], [6] or centralized approaches [7] are frequently addressed.

However, as reported in [8], cooperative behavior does not require V2X-communication. Furthermore, as automated vehicles will share the road with human-driven cars at least at the beginning, cooperation with human drivers in non-V2X-equipped cars is essential. Also, a natural, cooperative, human-like behavior of automated vehicles potentially increases their social acceptance.

To the best of the authors’ knowledge, previous motion planning approaches treated other traffic participants as obstacles which are to be avoided, similar to static obstacles like parked cars [2], [9]. While such approaches can deal with many everyday situations, such as driving autonomously or following other vehicles, some maneuvers, such as overtaking with oncoming traffic or

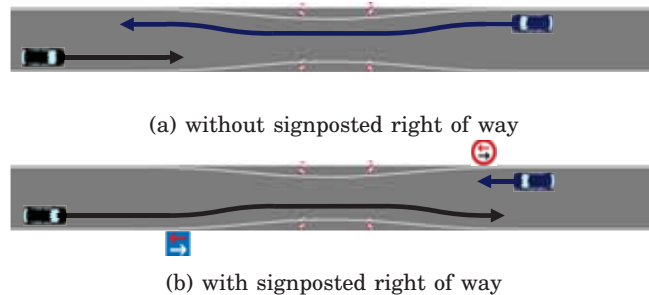


Fig. 1: Narrowing, with and without signposted right of way.

passing a narrowing (cf. Figure 1), require combinatorial approaches, as already reported by [2]. Still, even with combinatorial considerations as proposed by [10], [11], cooperative behavior cannot be implemented: If the motion prediction of other traffic participants is done isolated from the motion planning for the ego-vehicle, the behavior can be foresighted, but not cooperative in a bidirectional manner [8]. According to a study about German road traffic, cooperative behavior on average only occurs in the scale of one cooperative action per hour per traffic participant [12]. Thus, their treatment by a separate method, besides the conventional motion planning, is reasonable.

This paper addresses the problem of cooperative motion planning without V2X-communication. We propose a cost functional for trajectory ensembles, consisting of one trajectory per vehicle. Thereby, we acknowledge the fact that not only the behavior of other traffic participants affects us, but also our behavior affects the others in a closed loop. We consider the motion planning problem as the problem to find a globally optimal solution for a specific situation, knowing that every traffic participant has a different viewpoint considering optimality. The costs depend on vehicle dynamics, passenger comfort, driving intention and trajectory clearance, as well as the traffic regulations, as further outlined in Section II. In this approach, the prediction of other traffic participants is integrated into the motion planning. As the assumption of cooperative behavior might be violated by some traffic participants, this risk is assessed and the trajectory is only driven if a safe "plan B" [13] trajectory is still possible in case of unexpected behavior. An implementation of this approach is presented in Section III. The proposed algorithm is finally evaluated in Section IV.

*We gratefully acknowledge support of this work by the Tech Center a-drive

¹The authors are with FZI Research Center for Information Technology, Mobile Perception Systems, 76131 Karlsruhe, Germany naumann@fzi.de

²The author is also with Karlsruhe Institute of Technology (KIT), Institute of Measurement and Control, 76131 Karlsruhe, Germany stiller@kit.edu

II. GLOBAL OPTIMUM APPROACH

This section introduces the main building blocks of our approach to cooperative motion planning. Central to this approach is the assumption that all traffic participants are aware of each other and therefore react on each other's behavior in a closed loop. Subsequently, the trajectories for all relevant traffic participants are considered as one trajectory ensemble, and the quality of the solution depends on the trajectory of every participant separately as well as on the pairwise relation of the trajectories among each other.

This section is structured as follows: First, the representation of one trajectory in the ensemble is introduced. Subsequently, the cost functional is introduced. Next, before a solution is selected, the limitations to this approach are treated by a "plan B".

A. Behavior Policy

Cooperative motion planning is aware of the interaction of traffic participants. Therefore, wrong assumptions concerning the behavior of other traffic participants might cause undesired behavior. Even though, theoretically, any feasible behavior is possible, the authors make the following assumption: *Every traffic participant follows the traffic regulations, as long as this compliant behavior is physically feasible.*

Consequently, assuming perfect perception, a collision involving our vehicle can only be caused by violating the traffic regulation without foreseeable reason while our reaction at the time of violation is insufficient to avoid the collision.

Arising from this assumption, we pursue the following policies:

- If we have to give way, we can exclude a collision independent of others' behavior.
- If we have the right of way or the situation is not clearly regulated, we can exclude a collision if others behave rule compliant.

B. Trajectory Representation

For the representation of a single, deterministic trajectory, the established method of [14] is chosen: The trajectory $\mathbf{x}(t) = (x(t), y(t))^T$ is a mapping $\mathbb{R} \rightarrow \mathbb{R}^2$, with tangent angle ψ and curvature κ . A trajectory ensemble consists of one trajectory per traffic participant: $\mathbf{X} = (\mathbf{x}^1, \mathbf{x}^2, \dots)$, where the superscript describes the participants identifier.

C. Cost Functional

As proposed in [8], the quality of a solution, given by a trajectory ensemble, is determined by a cost functional. The lowest costs denote the best solution. Costs exceeding a certain value represent an infeasible solution. The cost functional is the sum of the costs of every traffic participant i

$$G_{\text{total}} = \sum_i G_i.$$

The costs G_i pursue two main goals: They ensure the feasibility of the trajectory but also rate its comfort and effectiveness for a single car. For this reason, the properties of the trajectory, such as velocity and acceleration, are rated with multiple evaluation functionals:

The feasibility costs exceed a certain bound if a trajectory is physically not feasible. The pleasantness costs reflect the wish of the passenger to travel steady and comfortable, including the perceived safety of the journey. Furthermore, the costs should motivate compliance with the traffic regulations. Not yielding is avoided by upscaling the costs of the vehicle that has the right of way in the pairwise trajectory costs.

In this approach, the ability to cooperate is associated with the ability to estimate the cost or quality of a solution for other traffic participants.

With the above information, the costs G_i per participant can be split into costs $G_{i,0}$ that only concern the own trajectory and costs $G_{i,j}$ that consider the relation to other trajectories:

$$G_i = G_{i,0} + \sum_j G_{i,j}$$

1) *Formulation of the trajectory properties:* Analog to [14] the properties of the trajectory that are examined by the evaluation functionals are

- the velocity $\mathbf{v}(t) = \dot{\mathbf{x}}(t)$
- the acceleration $\mathbf{a}(t) = \ddot{\mathbf{x}}(t)$
- the jerk $\mathbf{j}(t) = \dddot{\mathbf{x}}(t)$
- the distance to the left and right driving corridor bound $d_{\text{left}}(\mathbf{x}(t))$ and $d_{\text{right}}(\mathbf{x}(t))$
- the yaw rate $\omega(t) = \dot{\psi}(t)$ and
- the curvature $\kappa(t)$.

Additionally, properties of trajectory pairs describe their distance to each other. The shortest spatial distance is described by

$$d_{\min}(\mathbf{x}^1(t), \mathbf{x}^2(t)) = \min_t (d(\mathbf{x}^1(t), \mathbf{x}^2(t), t)),$$

where d denotes a distance measure between states of different vehicles.

To account for the perceived safety, but also to obey the traffic regulations, another property is introduced. Here, we can make use of time-referenced measures, as they equal a velocity-referenced spatial distance measure. In general, a collision is only possible if paths overlap. When determining the criticality, respectively the collision risk, of two trajectories, their closest point in time and space is crucial. Regarding a violation of the right of way, Cooper investigated the *post encroachment time (PET)* for specific scenarios [15]. Based on the latter, also regarding the potential collision zone, we propose the *time of zone clearance (TZC)* as a measure for the criticality of two trajectories with overlapping paths: The TZC is the time that elapses between the first vehicle leaving potential collision zone and the second vehicle entering this area, independent of the right of way (cf. Figure 2).



(a) blue vehicle drove first (b) black vehicle drove first

Fig. 2: The TZC is the time that the second vehicle takes to enter the red potential collision zone, assuming constant velocity.

Given the paths are overlapping and given the trajectories are not colliding, the TZC is calculated as follows:

$$\begin{aligned} \text{TZC} &= \text{TZC}(\mathbf{x}^{\text{first}}(t), \mathbf{x}^{\text{second}}(t)) \\ &= \frac{\text{gap along path}}{\text{velocity of the second vehicle}} \\ &= \frac{s^{\text{second}}(t_{\text{second,in}}) - s^{\text{second}}(t_{\text{first,out}})}{v^{\text{second}}(t_{\text{first,out}})} \end{aligned}$$

with s^{second} being the path of the vehicle that passes the collision zone second, v^{second} being the scalar velocity along this path, $t_{\text{first,out}}$ being the time at which the first vehicle clears the collision zone and $t_{\text{second,in}}$ being the time at which the second vehicle enters the collision zone. Constant velocity is chosen as passengers cannot foresee the planned trajectory and as it reflects possible actions (maximum deceleration or acceleration) best.

If the paths do not overlap, the TZC is defined to be infinite, if the trajectories collide, it is less or equal zero.

2) *Formulation of the evaluation functionals:* As in this work the costs are also calculated for human-driven cars in order to predict their behavioral decisions, they should reflect humans' understanding of the quality of a trajectory. Therefore, the previously introduced scalar trajectory properties $f(\mathbf{X})$ are investigated. Vectorial properties, such as the acceleration, are therefore split into their longitudinal and lateral part, using a motion model.

The costs of a trajectory are subdivided into three zones:

- comfort zone $\mathcal{Z}_{\text{comf}}$
- discomfort zone $\mathcal{Z}_{\text{disc}}$
- infeasibility zone \mathcal{Z}_{inf}

each for positive (+) and negative (-) deviation from the optimum f_{opt} . The functionals $G(f)$ expressing the costs induced by a trajectory property f are called *evaluation functionals*.

For the sake of steadiness and piecewise differentiability, all costs are starting from zero at their lower bound but do not vanish at the start of the next zone.

Accordingly, the total costs G are defined as

$$G(f) = \begin{cases} G_{\text{comf}} & , f \in \mathcal{Z}_{\text{comf}} \\ G_{\text{comf}} + G_{\text{disc}} & , f \in \mathcal{Z}_{\text{disc}} \\ G_{\text{comf}} + G_{\text{disc}} + G_{\text{inf}} & , f \in \mathcal{Z}_{\text{inf}}. \end{cases}$$

The comfort component induces only little costs

$$G_{\text{comf}}^+(f) = a_+ \cdot (\Delta f_{\text{comf}}^+)^2$$

depending on the distance of f to the optimal value

$$\Delta f_{\text{comf}}^+ = \Delta f_{\text{comf}}^+(\mathbf{X}) = f(\mathbf{X}) - f_{\text{opt}}.$$

Consequently, given a comfort threshold T_{comf} and assuming a comfortable deviation $\Delta f_{\text{cmargin}}^+$, the parameter a_+ is to be set to

$$a_+ = \frac{T_{\text{comf}}}{\left(\Delta f_{\text{cmargin}}^+\right)^2}.$$

The costs G_{comf}^- for comfortable negative deviation are calculated correspondingly with the parameter a_- .

The discomfort costs rise quadratic, but direction-dependent:

$$G_{\text{disc}}^+(f) = b_+ \cdot (\Delta f_{\text{disc}}^+)^2$$

depending on the distance of f to the upper start of the discomfort zone $f_{\text{disc,start}}^+$

$$\Delta f_{\text{disc}}^+ = \Delta f_{\text{disc}}^+(\mathbf{X}) = f(\mathbf{X}) - f_{\text{disc}}^+.$$

For logical reasons, the parameter b_+ should be notably higher than a_+ . Negative deviations are treated correspondingly with the parameter b_- .

Before the property represents the infeasibility of a trajectory, the infeasibility costs rise exponentially

$$G_{\text{inf}}^+(f) = c_+ \cdot (\Delta f_{\text{inf}}^+)^2 \cdot e^{|\Delta f_{\text{inf}}^+|}$$

depending on the distance of f to the upper infeasible value f_{inf}^+ minus a margin f_{margin}^+ from which the costs start rising

$$\Delta f_{\text{inf}}^+ = \Delta f_{\text{inf}}^+(\mathbf{X}) = f(\mathbf{X}) - \left(f_{\text{inf}}^+ - \Delta f_{\text{margin}}^+\right).$$

Consequently, given an infeasibility threshold T_{inf} and assuming a margin $\Delta f_{\text{imargin}}^+$, the parameter c_+ is to be set to

$$c_+ = \frac{T_{\text{inf}}}{\left(\Delta f_{\text{imargin}}^+\right)^2 \cdot e^{|\Delta f_{\text{imargin}}^+|}}.$$

Further, the infeasibility zone \mathcal{Z}_{inf} includes the margin in this notation. Again, negative deviations are treated correspondingly with the parameter c_- .

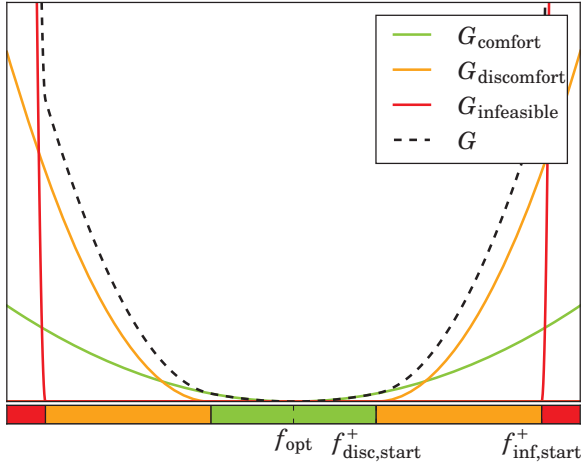


Fig. 3: Composition of the cost function G for a single trajectory property f : Very low costs around the optimum value f_{opt} , increasing rapidly in close vicinity of f_{inf} .

3) *Formulation of the cost functional*: With the evaluation functionals, the cost functional for a single property f is composed as follows (cf. Figure 3):

$$G(f) = G_{\text{comf}}^+ \cdot \sigma(\Delta f_{\text{comf}}^+) + G_{\text{comf}}^- \cdot \sigma(-\Delta f_{\text{comf}}^-) \\ + G_{\text{disc}}^+ \cdot \sigma(\Delta f_{\text{disc}}^+) + G_{\text{disc}}^- \cdot \sigma(-\Delta f_{\text{disc}}^-) \\ + G_{\text{inf}}^+ \cdot \sigma(\Delta f_{\text{inf}}^+) + G_{\text{inf}}^- \cdot \sigma(-\Delta f_{\text{inf}}^-),$$

where σ denotes the step function. The right of way of i over j is acknowledged by adding the comfort-related costs of vehicle i , upscaled with factor u , to the pairwise trajectory costs, if i has the right of way:

$$G_{i,j,\text{row}} = u (G_{i,0,\text{comf}} + G_{i,0,\text{disc}}).$$

A suitable choice of u ensures that the right of way is heeded, but its violation is still feasible, as stated in Section II-A.

The full cost functional is composed as follows:

$$G_{\text{total}}(\mathbf{X}) = \sum_i \left(G_{i,0}(\mathbf{x}^i) + \sum_j G_{i,j}(\mathbf{x}^i, \mathbf{x}^j) \right)$$

with singleton trajectory costs for vehicle i

$$G_{i,0}(\mathbf{x}^i) = G_{\mathbf{v}} + G_{\mathbf{a}} + G_{\mathbf{j}} + G_{\omega} + G_{\kappa} + G_{\text{offset}}$$

and pairwise trajectory costs for vehicle i due to vehicle j

$$G_{i,j}(\mathbf{x}^i, \mathbf{x}^j) = G_{\text{TZC}} + G_{d_{\text{min}}} + G_{\text{row}}.$$

D. Plan B

In order to obey our previously introduced policy, plan B trajectories are to be checked, as proposed in [13]. By doing so, we avoid maneuvering into situations that lead to collisions, if we made wrong assumptions concerning the behavior of other traffic participants. As their execution is unlikely, we accept discomfortable but feasible trajectories. This corresponds to a neglect of the comfort terms in the upper cost functional. As with

the previous trajectories, plan B trajectories can be calculated via a local continuous method [14], a sampling-based method such as RRT* [16] or other approaches.

E. Selection of Solution

As for passenger comfort, the evaluation of the TZC should already cause high discomfort costs at around 2s, a security margin is induced intrinsically by this approach. Thus, even a very small optimum, represented by a small range of minimal costs, does not equal a physically optimal trajectory, that would pass objects as close as possible in space-time. Rather, it already contains those security margins that are considered comfortable by humans and that consequently should be feasible with measurement uncertainties in the range of human perception errors. Hence, the optimum point can be chosen independent of its wideness, as long as a valid plan B protects the approach against consequences of wrong assumptions.

III. IMPLEMENTATION

In the following, a first approach for cooperative motion planning in specific situations, based on the previously introduced cost functional, is presented.

A. Path-Velocity Decomposition

Several potentially cooperative situations have highly constraint driving corridors for the traffic participants, independent of the order and number of traffic participants. Consequently, we make use of the *path-velocity decomposition (PVD)*, as introduced by [17]. The calculation of paths in static environments has already been widely investigated. Hence, valid paths are considered predefined (cf. Figure 4) and the implementation focuses on the velocity profiles along the paths.

B. Sampling

As the optimization problem is non-convex, but the control variable for the velocity of each vehicle is only one-dimensional, a classical sampling approach is chosen. Therefore, the trajectories $\mathbf{x}(t)$ are approximated by discretization in equidistant time steps:

$$\mathbf{x}_i = \mathbf{x}(t_i), \quad t_i = t_0 + i\Delta t.$$

For each car, multiple trajectories are sampled: Starting with an initial position and velocity, a random jerk sequence determines the velocity profiles and thus the trajectory. Next, the overall costs of each trajectory ensemble are calculated. For the solutions with the lowest costs, the plan B trajectory is checked until a valid plan B is found.

C. Plan B

Instead of using a different planning method with the assumption or classical prediction of disadvantageous behavior of others, we again make use of the PVD: Given the paths, a collision is only possible in particular areas that can be determined a priori. Thus, unlike in [13],

no trajectory has to be planned. Rather, the plan B-consideration can be seen as a “what could I do if”-consideration. The key questions are: In every time step, what could the other vehicle do that leads to a collision with us? And what could we do to avoid this? This consideration can be split into the following cases:

1) *Other vehicle drives first*: If the other vehicle drives first, it can only cause a collision by deceleration. In reaction, we can decelerate as well. If we can manage to stop before the collision zone, we have a valid plan B.

2) *Ego vehicle drives first*: If the ego vehicle drives first, the other vehicle can only cause a collision by acceleration. In reaction, we can also accelerate, to still drive first, or decelerate to stop before the other vehicle collides with us. As the path is regarded as predefined, changing the path is not considered.

D. Implications on the cost functional

Since in this implementation, trajectories are discretized in time, derivatives are approximated by finite differences. Thus, the functionals of section II-C turn into functions. Consequently, trajectory properties that depend on a single minimum, such as TZC, can be largely affected if this minimum is not sampled. In order to avoid this, either the sampling rate must be sufficiently high, or the point of the exact minimum has to be interpolated. As this implementation is not based on linear optimization but on sampling, we interpolate the crucial points.

The jerk is not considered to avoid high order derivatives. Also, the curvature itself is not considered as the predefined path guarantees the compliance with the steering geometry. However, it is used to calculate the lateral acceleration values. Furthermore, the shortest spatial distance d_{\min} either lies in the collision zone and is considered by the TZC, or it is not relevant. Hence, it is neglected as well.

E. Selection of Solutions

As explained in section II-E, criticality protection is ensured via the costs of the TZC and the check for a plan B. Consequently, the solution with the lowest costs and a valid plan B is selected and its ego trajectory is executed, as long as its costs do not exceed the feasibility-threshold. In case no solution has a valid plan B, an emergency braking maneuver is triggered. Note: In in-car applications, the parallel running classical, reactive motion-planner would have to take over control in this case.

IV. EVALUATION

In this section, the method outlined in section III is evaluated for two scenarios, a left turn at a T-intersection and passing through a narrowing of the road (cf. Figures 1 and 4).

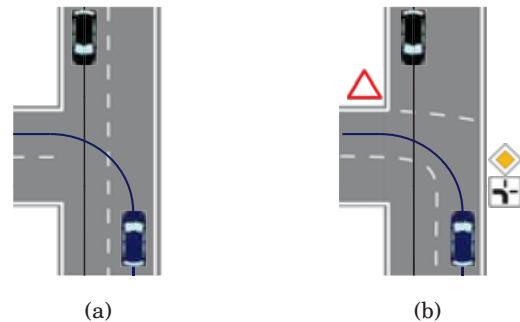


Fig. 4: Left turn at T-junction, with and without signposted right of way and predefined paths.

A. Simulation

For both scenarios, each with and without signposted right of way, but sharing the same paths, velocity profiles were sampled. From the resulting trajectories, ensembles with one trajectory per vehicle were generated. In order to reduce computational cost, trajectories that did not reach the end of the collision zone were excluded from the cost calculation. Furthermore, colliding trajectory ensembles were excluded. The remaining ensembles were analyzed with respect to

- comfort costs
- discomfort costs
- infeasibility costs
- traffic regulation costs.

B. Analysis

As depicted in Figure 5 and 6, the initial states were chosen in a way that the optima of both vehicles overlap in the collision zone. In the T-junction scenario, the right of way is regulated with and without traffic signs. A violation of the right of way causes high costs so the optimal solution is following the rules. The trajectory of the vehicle that has right of way is not interfered (cf. Figure 5 (2) and (3)).

In the narrowing scenario, the right of way is not regulated without traffic signs. Here, due to equal cost parameters, the vehicle that is closer to the narrowing passes first. Still, traffic signs can overrule this globally most comfortable solution and shift the optimum (cf. Figure 1 and 6 (3)).

If a collision can only be avoided by one of the vehicles, as the other is too close to the collision zone, the optimal solution is the collision avoidance. Even though this violates the traffic regulations, the infeasibility costs overrule discomfort costs and traffic regulation costs.

Further, if we do not interfere a vehicle that has the right of way, its costs G_{row} are constantly high, but not raised by our behavior. In this case, the optimal solution is that we pass first, without violation of traffic rules.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a new approach to cooperative motion planning, able to cooperate with human

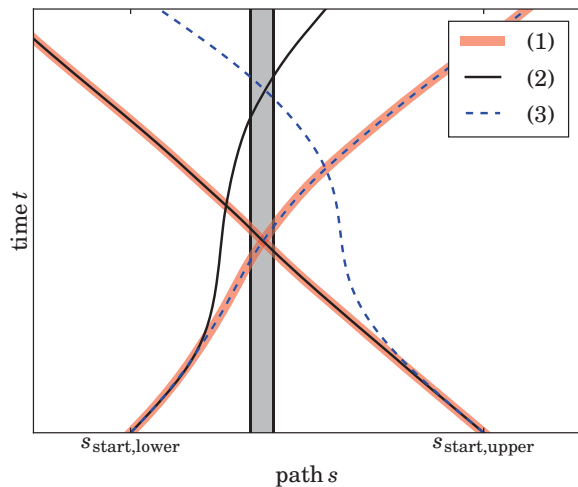


Fig. 5: Minimum cost trajectories in the T-junction scenario with the collision zone marked in grey: (1) for each vehicle solely on the road, (2) when upper vehicle has right of way (Fig. 4a), (3) when lower vehicle has right of way (Fig. 4b).

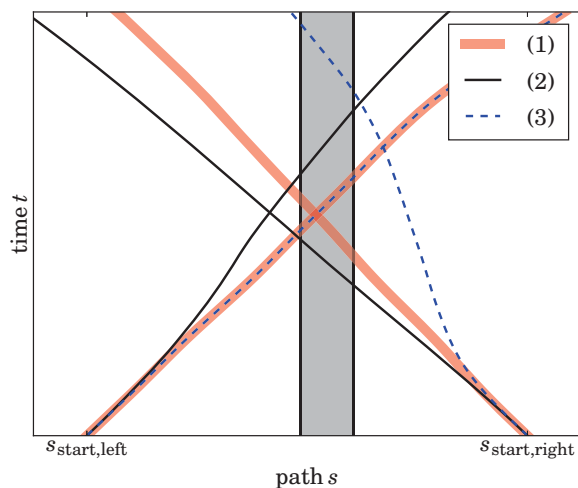


Fig. 6: Minimum cost trajectories in the narrowing scenario with the collision zone marked in grey: (1) for each vehicle solely on the road, (2) when no right of way predefined (Fig. 1a), (3) when left vehicle has right of way (Fig. 1b).

drivers and automated vehicles without requiring V2X-communication. While the approach is valid for two-dimensional motion planning, our first implementation covers several scenarios deploying PVD.

The preliminary results for the simulated scenarios demonstrate that the method produces safe and comfortable cooperative trajectories in a narrowing and a typical intersection scenario. Individual trajectory costs have been extended by costs accounting for mutual comfort and safety of any pair of trajectories. Other traffic participants have been taken into account by incorporating their individual costs. The total trajectory costs for each participant have been segmented into three areas representing comfortable driving, uncomfortable

driving and collision/infeasibility.

Future work includes real time implementation and on-road experiments with our vehicle "BerthaOne". Several parametrizations will be used for the cost functional, considering different vehicle types and driver behaviors. Furthermore, probabilistic trajectories will be accommodated to account for inherent uncertainties in perception and behavior.

REFERENCES

- [1] K. Bengler, K. Dietmayer, B. Färber, M. Maurer *et al.*, "Three Decades of Driver Assistance Systems - Review and Future Perspectives," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 4, pp. 6–22, 2014.
- [2] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn *et al.*, "Making Bertha Drive - An Autonomous Journey on a Historic Route," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 2, pp. 8–20, 2014.
- [3] C. Englund, L. Chen, J. Ploeg, E. Semsar-Kazerouni *et al.*, "The Grand Cooperative Driving Challenge 2016: boosting the introduction of cooperative automated vehicles," *IEEE Wireless Communications*, vol. 23, no. 4, pp. 146–152, August 2016.
- [4] D. Carlino, S. D. Boyles, and P. Stone, "Auction-based autonomous intersection management," in *IEEE Int. Conf. on Intell. Transp. Syst.*, Oct 2013, pp. 529–534.
- [5] M. Elhenawy, A. A. Elbery, A. A. Hassan, and H. A. Rakha, "An Intersection Game-Theory-Based Traffic Control Algorithm in a Connected Vehicle Environment," in *IEEE Int. Conf. on Intell. Transp. Syst.*, Sept 2015, pp. 343–347.
- [6] H. Rewald and O. Stursberg, "Cooperation of autonomous vehicles using a hierarchy of auction-based and model-predictive control," in *Proc. of the IEEE Intell. Vehicles Symposium*, June 2016, pp. 1078–1084.
- [7] S. Manzingler, M. Leibold, and M. Althoff, "Driving Strategy Selection for Cooperative Vehicles using Maneuver Templates," in *Proc. of the IEEE Intell. Vehicles Symposium*, 2017.
- [8] M. Naumann, P. Orzechowski, C. Burger, O. S. Tas, and C. Stiller, "Herausforderungen für die Verhaltensplanung kooperativer automatischer Fahrzeuge," in *AAET Automatisiertes und vernetztes Fahren*. Braunschweig, Germany: ITS automotive nord e.V., Feb 2017, pp. 287–307. [Online]. Available: <http://www.mrt.kit.edu/z/publ/download/2017/Naumann2017AAET.pdf>
- [9] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A Review of Motion Planning Techniques for Automated Vehicles," *IEEE Transactions on Intell. Transp. Syst.*, vol. 17, no. 4, pp. 1135–1145, April 2016.
- [10] P. Bender, Ö. Ş. Taş, J. Ziegler, and C. Stiller, "The combinatorial aspect of motion planning: Maneuver variants in structured environments," in *IEEE Intell. Vehicles Symposium (IV)*, June 2015, pp. 1386–1392.
- [11] X. Qian, F. Alché, P. Bender, C. Stiller, and A. de La Fortelle, "Optimal trajectory planning for autonomous driving integrating logical constraints: An MIQP perspective," in *IEEE Int. Conf. on Intell. Transp. Syst.*, Nov 2016, pp. 205–210.
- [12] A. Benmimoun, D. Neunzig, and C. Maag, "Effizienzsteigerung durch professionelles/partnerschaftliches Verhalten im Strassenverkehr," *FAT-Schriftenreihe*, no. 181, 2004.
- [13] F. Damerow and J. Eggert, "Risk-Aversive Behavior Planning under Multiple Situations with Uncertainty," in *IEEE Int. Conf. on Intell. Transp. Syst.*, Sept 2015, pp. 656–663.
- [14] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for Bertha - A local, continuous method," in *IEEE Intell. Vehicles Symposium Proc.*, June 2014, pp. 450–457.
- [15] P. Cooper, "Experience with traffic conflicts in Canada with emphasis on "post encroachment time" techniques," in *International calibration study of traffic conflict techniques*. Springer, 1984, pp. 75–96.
- [16] S. Karaman and E. Frazzoli, "Sampling-based Algorithms for Optimal Motion Planning," *Int. J. Rob. Res.*, vol. 30, no. 7, pp. 846–894, Jun. 2011.
- [17] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition," *Int. J. Rob. Res.*, vol. 5, no. 3, pp. 72–89, 1986.

Urban Pedestrian Behaviour Modelling using Natural Vision and Potential Fields

Pavan Vasishta¹, Dominique Vaufreydaz² and Anne Spalanzani³

Abstract—This paper proposes to model pedestrian behaviour in urban scenes by combining the principles of urban planning and the sociological concept of *Natural Vision*. This model assumes that the environment perceived by pedestrians is composed of multiple potential fields that influence their behaviour. These fields are derived from static scene elements like side-walks, cross-walks, buildings, shops entrances and dynamic obstacles like cars and buses for instance. Using this model, autonomous cars increase their level of situational awareness in the local urban space, with the ability to infer probable pedestrian paths in the scene to predict, for example, legal and illegal crossings.

I. INTRODUCTION

As the race to attain and deploy fully autonomous vehicles on urban roads heats up, the concept of Situational Awareness (SA) takes centre stage. Situational awareness is the natural human ability to understand, react and predict the environment based on previously learnt parameters, the utilisation of which is most frequently seen while driving. Human drivers need to balance different variables – speed, route selection, positions of pedestrians, cyclists, other cars, etc. – while trying to predict their future states. In fact, errors in maintaining situation awareness are the most frequent cause of errors in real-time tasks such as driving [1] and can be attributed to many accidents.

Situational Awareness can be described in three incremental abstract levels [2] as *Perception*, *Comprehension* and *Projection*. A human driver in an urban street cycles through these levels continuously. Objects and elements– pedestrians, obstacles, other cars, cross-walks, interesting areas– in the environment are identified. These elements are contextually understood with regard to the environment that they are in– the answer to the question “Why is that element there?” For example, the answer to “Why is there a cross-walk on the street?” is to facilitate a crossing from one side of the road to another. Finally, the objects and elements are understood together and their future interactions are projected with a certain probability: a cross-walk *may* be used by a pedestrian if he/she is close to it. A human driver’s specific course of action is decided by these continuously evolving projections.

The main motivation of this work is to increase the situational awareness of an autonomous car in the context of driving in urban streets. A major driving force is the adoption

of sociological ideas for understanding pedestrian behaviour in inner city areas. Pedestrian behaviour has been postulated to be a function of the built environment; i.e. their movement is a consequence of the presence of certain positive and negative *attractors* [3]. This behaviour, called *Natural Motion*, is an extension of Gibson’s *Natural Vision* which envisages human behaviour as wanting to move in a direction that interests them the most in their field of view [4]. These positive attractors, here called “Points of Interest (POI)”, may be present as an element in the scene. They can be monuments, places of public interest, public transportation... Other, more common, POIs are areas of commercial interest - stores, restaurants, etc., that are seen very frequently in an urban centre [5]. The presence of these POIs in any scene influences the behaviour of pedestrians within it. Understanding these influences allows the autonomous car to perform actions that are instinctive in a human driver - project pedestrian future states and intuit areas of legal and illegal crossings.

The major contribution of this work is the creation of a new framework for quick comprehension of urban streets. It also forms a base to project future states of pedestrians without the need of their presence in the scene, analogous to a human driver’s intuition. We model the scene as attractive and repulsive potential fields. The novelty of our approach is the introduction of POIs whose attractiveness influences pedestrian behaviour.

The paper is divided into five sections. Section II deals with related work in the field of pedestrian motion prediction, followed by section III, the theoretical basis of the framework and the methods used to project future states. Section IV discusses the implementation of this framework, its results and validation of a conducted experiment. Section V concludes this paper with a discussion on the current work and envisaged future work.

II. RELATED WORK

As far as the authors know, there has been little to no work done in the field in accounting for Point of Interest influences in urban pedestrian prediction. Much work, however, has been done in modelling and prediction of pedestrian routes and route-choice behaviour [6]. Most pedestrian behaviour prediction algorithms depend on learning frameworks based off of observed data. Modelling the inherent pedestrian variables is one such technique. A data driven approach on this, minimising an energy function that accounts for many personal factors like speed, grouping etc., can be found in [7]. A similar approach, based on the *Social Force Model* can be seen in [8].

¹ Pavan Vasishta is a PhD student in the CHROMA team (Inria, email: Pavan.Vasishta@inria.fr)

² Dominique Vaufreydaz is an associate professor in the Pervasive Interaction team (Univ. Grenoble Alpes, CNRS, Inria, LIG, F-38000 Grenoble France, email: Dominique.Vaufreydaz@inria.fr)

³ Anne Spalanzani is an associate professor in the CHROMA team (Univ. Grenoble Alpes, Inria, F-38000 Grenoble France, email: Anne.Spalanzani@inria.fr)

Others use MDPs and its variants to predict the beliefs of pedestrian crossings in a scene [9]. Destinations are assumed to be known. In close spirit to our work are [10], [11] and [12]. These works build a cost function based on the environment. [10] learns the cost function of the environment via observed trajectories. An MDP is solved with rewards based on observed trajectories as well with known destinations. Working on static scenes, it requires previously observed trajectories to model the cost function and thus is infeasible for rapidly changing scenes on autonomous vehicles. [11] continues this work by semantically segmenting the observed environment to construct a cost function fed to an hMDP to predict pedestrian positions, even without pedestrian observations. While this knowledge is transferable, it is computationally expensive. The computation problems are solved in [12], yet being applied only for static scenes with learned trajectories. Last, [13] looks at the distance of the pedestrian from the kerb, position and velocity of the car from the cross-walk to learn an “*Inner city model*” to predict pedestrian crossings.

These works require learning from observed data to predict pedestrian positions. Our framework envisages to solve the issues of computational complexity and dynamicity while considering environmental and social factors. It also does *not* use learning for building environmental models which makes it very useful to deploy in unknown environments, with very few dependencies.

III. THEORETICAL FRAMEWORK

Pedestrian crossing behaviour in urban areas can be classified into two broad categories - legal crossings and illegal crossings. Legal crossings are such movements of a pedestrian that account for the safest path from one side of the street to the other. These generally happen on a cross-walk. An illegal crossing is an abnormal behaviour wherein the pedestrian decides to not take the cross-walk to cross the street.

In a structured urban environment, for legal crossings to occur, certain assumptions are made:

- The edges of the road repel pedestrians such that their paths are restricted to the side-walk.
- A cross-walk acts as a conduit between the two sides of the street and offers no resistance to crossing
- The road acts as a barrier for crossing, repelling pedestrians towards the side-walks.
- Static and Dynamic obstacles on the road are repulsive in nature, increasing the resistance of the road and pushing back pedestrians towards side-walks.
- Side-walks offer no resistance to pedestrian movement.
- Points of Interest are a reason for pedestrians to cross from one side of the street to another.

An illegal crossing occurs when at least one of these assumptions is violated. Predicting these areas of illegal crossings leads to a higher level of SA. Looking at these assumptions, it can be seen that a system of potential fields [14] can be a good fit as a model for explaining urban behaviour. Each of the assumptions made earlier can be represented as a function of a potential field.

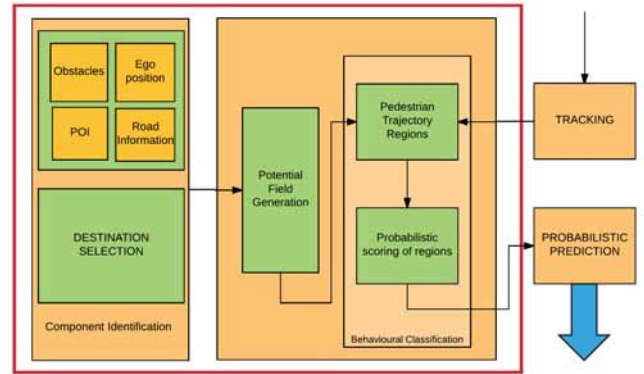


Fig. 1: Architecture of the Framework. Only the red block has been implemented in this work

The architecture of the proposed framework can be found in Fig. 1. From the observed scene, certain informations need to be extracted. These informations are road width, number of lanes, the closest Points of Interest (POI) from the observer, their orientation, the position of the closest cross-walk, etc. Static and Dynamic obstacles on the road also need to be identified.

Destination points in the scene are chosen and fed forward along with the scene information to the Potential field generator. This potential field generation step generates a grid with a potential “Map” based on the extracted data. The generated “Map” is used by the Behaviour classification module, first to generate pedestrian trajectory regions and to score these regions with a probability.

Firstly, to demonstrate the model, it is assumed that the scene under consideration is well-structured. This implies that there is an observable demarcation between the road and the side-walk, the road and the cross-walk and that the lanes on the road are easily observable. It is also assumed that the width of the road (L_{road}), the width of the lane (L_{Lane}), and the POI positions are known, can be computed using sensors embedded into the autonomous car or can be retrieved from a global map. Given the position of the ego vehicle, the distance to the cross-walk and the side-walk can be extrapolated from known information. Considering the pedestrian as a self-driven particle under the influence of attractive and repulsive forces, a potential field can be constructed which produces certain motion behaviours [5].

A destination, by definition, draws a self driven particle towards it. Thus, a POI can be a destination. Conversely, all destinations in a scene are points of interests. Making this assumption, the viable ends of the observed scene are designated as POI and the potential field is recalculated.

A grid is defined for the observed area with its origin at the top left corner and extending to (X_m, Y_m) , the maximum grid values with a specified grid resolution. Each cell on the grid can take the attributes *road*, *cross-walk*, *empty*, *POI*, *obstacle* and *edge* and a range of values between 0 and 1. The resultant

model is a linear function of all component potential values at each cell in the grid. Thus,

$$U_{\text{total}} = U_{\text{Edge}} + U_{\text{Road}} + U_{\text{Obs}} + U_{\text{CW}} + U_{\text{POI}} \quad (1)$$

where U_{Edge} , U_{Road} , U_{Obs} , U_{CW} and U_{POI} are potentials associated with the road edges, the road, obstacles on the road, the cross-walk and the POIs.

A. Computing Potentials

For each cell in the grid, the different potentials can be calculated as follows.

1) *Edge Potential*: The edge potential must repel pedestrians towards the center of the side-walk. An illegal crossing occurs when the self driven particle can exert enough force to overcome this potential. For each cell with a center (x, y) , the value of the potential is defined by

$$U_{\text{Edge}}^{ij} = \frac{1}{2}\eta \left(\frac{1}{\rho(x^{ij}, y^{ij})} \right) \quad (2)$$

where $\rho(x, y)$ is the distance of the i^{th} and j^{th} edge cell from all other cells in the grid. η is a scaling factor dependent on L_{Road} . The calculated values are ceiled to an appropriate value. The total edge potential is a summation of potential values of all edge-containing cells.

2) *Road potential*: Based on a sociological study conducted in France [15], it can be inferred that the propensity of illegally crossing a road is linearly dependent on its width. Thus, a narrow road entices a pedestrian to cross illegally while a wider one does not.

For cell (C^{ij}) with an attribute *road*, the calculated potential value is:

$$U_{\text{Road}}^{ij} = \beta_{\text{Road}} \exp \left(- \left[\left(\frac{x_{ij} - x_{\text{road}}}{\sigma_x} \right)^2 + \left(\frac{y_{ij} - y_{\text{road}}}{\sigma_y} \right)^2 \right] \right) \quad (3)$$

β_{Road} , σ_x and σ_y are dependent on the width of the road as explained earlier.

The total road potential is the summation of potential values of all road-containing cells.

3) *Obstacle Potential*: Obstacles in the scene can be distinguished as static and dynamic obstacles. For either classification, the response of the self-driven particle under the effect of the obstacle remains the same. The self driven particle cannot cross through the obstacle and the approach to the obstacle is slow. A Yukawa potential [16] is considered a fit for the expected behaviour.

A static obstacle takes the shape that it is perceived to be. A dynamic obstacle (for example, other cars in the scene), is described as a rectangular shape with a triangular shape extending forward in the direction of motion. Thus, the potential is described by,

$$U_O^n = \Lambda \frac{\exp(-\alpha \mathbf{K})}{\mathbf{K}} \quad (4)$$

Where Λ and α decide the behaviour of U_O^n . Larger the values, sharper the drop off of the potential near the obstacle.

\mathbf{K} is the distance of the obstacle from every point on the workspace, i.e.,

$$\mathbf{K} = \|C^{ij} - C^{\text{Obs}}\| \quad (5)$$

The total effect of all the obstacles in the workspace is given as

$$U_{\text{Obs}} = \sum_{n=0}^N U_O^n \quad (6)$$

Where N is the total number of obstacles observed. The extremely large values that are generated are truncated to a maximum viable value.

4) *POI Potential*: A Point of Interest (an inexhaustive list of what may be considered as a POI can be found in [5]) generates an attractive pull in the scene. With sufficient motivation, the self-driven particle can escape the influence of a POI. A POI is also a terminal point in the scene - the implication being that all exits in the scene are POIs. The potential of a POI situated at a cell defined by $(x_{\text{poi}}, y_{\text{poi}})$ is a Gaussian function centered at $(x_{\text{poi}}, y_{\text{poi}})$. β_{poi} , σ_x , σ_y depend on the global importance of the specific Point of Interest.

5) *Cross-walk Potential*: The cross-walk connects the two side-walks of the street and acts as a resistance-less conduit for the self driven particle. Thus, the potential of the cross-walk is the smallest value in the area.

B. Behavioural Classification

1) *Trajectory Regions*: Pedestrian route choice behaviour, in inner city limits, can be described in terms of optimisations. A pedestrian either tries to perform a distance optimisation to a destination at one extreme or optimises for safety at the other. Thus, all possible pedestrian paths can be captured between these two behaviours. From each destination in the scene to all the others, an A* search is performed with the heuristic:

$$h(s) = \alpha C(s, s+1) + (1 - \alpha)\rho(g, s+1) \quad (7)$$

Where $C(s, s+1)$ is the cost of moving from the current state s to the next state $(s+1)$. $\rho(g, s+1)$ is the normalised distance between the goal (destinations) and the next state. α is the parameter contributing to the integration of safe trajectory optimisation and shortest distance optimisation. Varying this parameter allows for simulating the different trajectories pedestrians might take, like partially optimising for distance and partly for safety.

2) *Probability map of pedestrian trajectories*: Trajectories generated for each entrance and exit are collated to create regions of probable pedestrian trajectories. These are then analysed to find regions of overlap. The larger the value of the region of overlap, higher the probability that a pedestrian might be present there.

IV. IMPLEMENTATION AND RESULTS

In the current work, the red block in Fig. 1 has been implemented on the dataset provided in [17]. This is a dataset generated in Martigny, Switzerland from a static camera overlooking a city square. The camera captures a scene containing 2 POIs and a crosswalk. It also captures pedestrian and traffic circulation. POI positions have been extracted from OpenStreetMaps. The scene at a specific instant is shown in Fig. 2

This scene is segmented into a grid containing features of the component potential fields - road, POI, crosswalk and sidewalks. Road parameters are taken from Swiss national standards and a potential field resulting from the constituent components are created as explained in III. This potential field “Map” is then used to determine areas of probable trajectories and pedestrian probability.

A. Validation

To show that this approach follows *Natural Vision* [4] is sufficient validation to use it to efficiently model the environment. Naive and sufficient validations are described for the approach, based on chosen destinations.



Fig. 2: Scene at a specific instant from the dataset. No dynamic obstacles on the road are observed at this instant.

Fig. 4 forms the crux of our work. This scene, taken from the acquired dataset, is the representation of the observation at a specific instant (leftmost image in Fig. 4). The width of the road and lanes were extracted manually based on Swiss national standards. With this data, the scene is reconstructed by placing its constituent elements at relative positions on a grid. The obstacle has been identified and tracked using the YOLO2 framework. Attributes of each cell on the grid is manually defined as cross-walk, side-walk, road, edge or obstacle, as well as for the POI. With these data populated on the grid, a resultant potential field “Map” has been generated as defined in section III

1) *Primary Validation*: A first naive validation of this method is to show that observed pedestrian presence matches those areas predicted by the framework. In this scene (Fig. 2), there are 4 potential destinations, marked (1), (2), (3) and (4). Destination (2) is the entrance of the visible POI

while the others are the ends of the scene. Choosing one of these as a starting point, we compute trajectories of the safest and the shortest paths and everything in between to all the other exits. By repeating this at each exit, a map of areas of most probable pedestrian positions emerges. For instance on Fig. 3, two destinations – (1) and (4) – were chosen and the different trajectories determined. Superimposing areas leads to the pedestrian trajectory probability map at right on the same figure.

Pedestrians were detected using the YOLOv2 framework [18]. Trajectories were tracked based on these detections. By accumulating these trajectories, a probability map based on observations was created (Section III-B2). Fig. 3 compares the predicted probability areas from our model and the observation.



Fig. 3: Comparison between observed probability map and predicted pedestrian trajectory probability map. The left image shows the ground truth. Right image is the predicted probability map of the scene. Darker the colour, higher the probability of pedestrians being present in that area.

In the observed probability map, it can be seen that there is a high incidence of pedestrians on the side-walk. The probability of a legal crossing is much higher compared to an illegal one, as observed. The predicted pedestrian trajectory probability map shows that there is a high probability that this scene has more chances of pedestrians crossing legally compared to an illegal crossing. It also predicts a high number of pedestrian trajectories into destination (3). Comparing to the ground truth, even though there are a few stray illegal crossings, they are overwhelmingly outnumbered by the number of legal ones. It is also observed that pedestrians continue to walk towards destination (3) compared to people crossing across into destination (2) in the scene.

TABLE I: Different behavioural cases observed in the dataset. Destinations can be seen on Fig 2.

Case	Destinations		Crossing Legality
	From	To	
I	1	3	Legal
II	1	2	Legal
III	4	2	Legal
IV	4	1	Legal
V	1	4	Illegal
VI	1	4	Legal
VII	4	1	Illegal
VIII	4	3	Legal/illegal

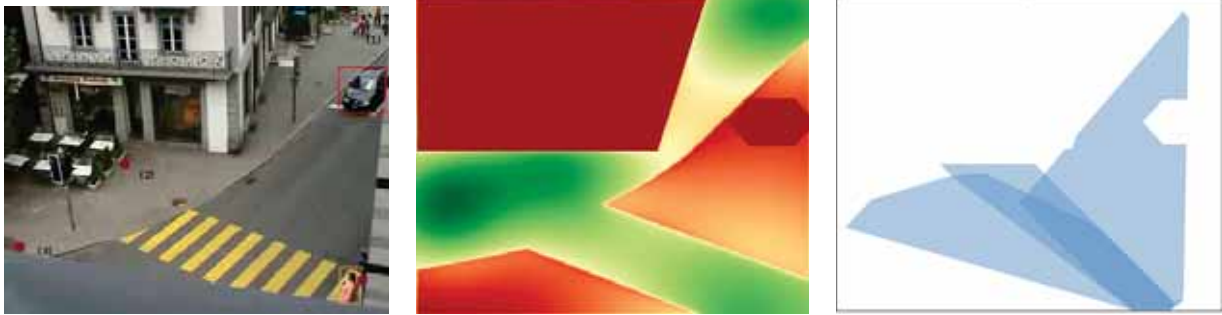


Fig. 4: Real scene juxtaposed against resultant potential field “Map” and pedestrian trajectory probability. Scene contains a dynamic obstacle and a pedestrian.

2) *Secondary Validation*: A secondary validation of the framework is to prove that it predicts behaviour based on *Natural Vision*. Pedestrians stay within certain bounds while going towards a destination. Knowing this, areas of illegal crossings can be predicted. From the dataset, 38 trajectories were chosen and classified into eight different behaviours. Each case is classified based on the entry and exit destinations of the scene and the legality of crossing between them. The behavioural classes for pedestrians walking in the scene (Fig. 2) can be found in Table I.



Fig. 5: Validation of the two most common behaviours in the dataset: case I and case VI (see table I). The dashed line corresponds to the A* predicted regions for each case. The coloured boundaries represent the extended zone.

As seen in section III-B1, the A* algorithms returns a sharp trajectory, going straight for the goal based on the heuristic provided. It does not meander when there are no explicit potential modifications. The zones predicted by just using the A* algorithm can be seen as the dashed line in Fig. 5. Human motion very rarely follows a perfect straight line. Thus, to account for these random motions, the zones were dilated by 40 cms on all sides. This leads to a much better prediction score as can be seen in Table II. This is sufficient to show

that the principle of *Natural Vision* is valid and can be used to determine pedestrian behaviour.

B. Discussion

For some behavioural cases, like cases II and III, not many pedestrian trajectories could be found in the chosen dataset. As can be seen, a large percentage of all observed trajectories for all cases are present within the predicted zones. These zones are bounded by the shortest route to the goal and the safest. Our results are a validation of this. These scores can be ameliorated by substituting better parameters to build the potential field model. For example, the *attractiveness* parameter of the POIs in the scene were assumed to be equal for all POIs in the scene. These parameters could change based on the POIs global importance, the time of day, etc., all of which could be encoded on a map for a given city. By accurately estimating the values, pedestrian behaviour in urban centers could be much better predicted. Thus, a conclusion can be reached that pedestrian behaviour is not random in nature. Their movement can be accurately predicted by utilising well established sociological ideas of *attractors* and *Natural Vision* as our work demonstrates.

1) *Towards Prediction*: As an inference, the A* algorithm can be used to determine the prior destination probabilities of every tracked pedestrian. At each time step, a tracked pedestrian’s probable regions of movement can be calculated as explained earlier. These prior probabilities can then be used to perform spatio-temporal predictions on the grid. An example for such an area is shown in Fig. 4. The first image shows the observed scene with an obstacle and the four possible destinations in the scene. The second image shows the resultant potential field due to all the component features. The third image shows the trajectory probability regions of a tracked pedestrian. Darker the region, higher the belief that the pedestrian will take that path. The left image is that of a pedestrian waiting to cross the street while a car is waiting at the traffic light. The generated potential field is used to determine the trajectory regions of the pedestrian to each destination and the probabilities are scored. This is seen in the rightmost image in Fig. 4. This result shows that there is a high belief that the pedestrian will cross via the crosswalk. Yet, given her position, there is a significant probability

TABLE II: Quantitative analysis of trajectories within predicted regions

Case	Nb Trajectories	A* Predicted zone		Extended zone (40cms)	
		Inside zone (%)	Outside zone (%)	Inside zone (%)	Outside zone (%)
1	10	84.11	15.88	96.88	3.11
2	2	30.07	69.92	88.17	11.82
3	1	29.10	70.89	51.49	48.50
4	9	68.48	31.51	77.43	22.56
5	7	72.38	27.61	83.26	16.73
6	9	39.63	60.36	50.26	49.73
7	12	76.79	23.20	83.28	16.71
8	10	64.67	35.32	69.98	30.01

that she can cross illegally. The belief can be ameliorated and made more accurate by taking into account the direction and velocity of the observed pedestrian. A drawback of this method is that the A* finds the optimal trajectory given the heuristic. This does not take into account similarly values potential fields, leading to a loss of information. A possible solution to this could be to use a greedy algorithm that can generate all possible trajectories between the pedestrian position and the destination.

V. CONCLUSION

In this work, we have established a new framework for increasing the Situational Awareness of an autonomous car on urban roads. This is done by adapting the sociological principle of Natural Vision as a function of a potential field composed of different elements of the urban environment. This allows the car to intuitively understand pedestrian behaviour in previously unobserved areas. We have also demonstrated that pedestrian behaviour in urban areas is not random but is a function of the built environment they are in. The main contributions of this paper are – a) the usage of sociological principles and the integration of POIs into understanding pedestrian behaviour, b) quick computation of probable pedestrian movement zones even when there are no pedestrian observations in the scene.

The current work utilises a dataset that captures pedestrian behaviour with a mounted, stationary camera overlooking a single view. Future work will deal with the application of this framework from the ego-perspective of an autonomous car. Algorithms like the CMC-DOT [19] can be used for estimating the occupancy grid and positions of dynamic obstacles in real time. This can then be used as an input for our framework. Another work that needs to be done is to have online extraction of POIs and track individual pedestrians based on the priors determined by our framework.

ACKNOWLEDGMENT

These researches have been conducted within the VALET project, funded by the French Ministry of Education and Research and the French National Research Agency (ANR-15-CE22-0013-02).

REFERENCES

[1] L. Gugerty, "Evidence from a partial report task for forgetting in dynamic spatial memory," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 40, no. 3, pp. 498–508, 1998.

[2] M. R. Endsley, "Toward a theory of situation awareness in dynamic systems," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 37, no. 1, pp. 32–64, 1995.

[3] B. Hillier, A. Penn, J. Hanson, T. Grajewski, and J. Xu, "Natural movement: or, configuration and attraction in urban pedestrian movement," *Environment and Planning B: planning and design*, vol. 20, no. 1, pp. 29–66, 1993.

[4] J. J. Gibson, "The ecological approach to visual perception." 1979.

[5] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[6] E. Papadimitriou, G. Yannis, and J. Golias, "A critical assessment of pedestrian behaviour models," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 12, no. 3, pp. 242–255, 2009. [Online]. Available: <http://dx.doi.org/10.1016/j.trf.2008.12.004>

[7] K. Yamaguchi, A. C. Berg, L. E. Ortiz, and T. L. Berg, "Who are you with and where are you going?" in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 1345–1352.

[8] S. Pellegrini, K. Schindler, and L. van Gool, "You'll Never Walk Alone: Modeling Social Behavior for Multi-target Tracking," *Proceedings of the IEEE 12th International Conference on Computer Vision (ICCV)*, no. Iccv, pp. 261–268, 2009.

[9] T. Bandyopadhyay, C. Z. Jie, D. Hsu, H. Marcelo, A. Jr, D. Rus, and E. Frazzoli, "Intention-Aware Pedestrian Avoidance," *The 13th International Symposium on Experimental Robotics*, pp. 963–977, 2013.

[10] B. Ziebart, N. Ratliff, G. Gallagher, and K. Peterson, "Planning-based prediction for pedestrians," *Intelligent Robots and ...*, 2009. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5354147

[11] K. Kitani, B. Ziebart, J. Bagnell, and M. Hebert, "Activity forecasting," *Computer Vision–ECCV 2012*, pp. 201–214, 2012.

[12] D. Vasquez, "Novel planning-based algorithms for human motion prediction," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 3317–3322.

[13] S. Bonnin, T. H. Weisswange, F. Kummert, and J. Schmuedderich, "Pedestrian crossing prediction using multiple context-based models," *2014 17th IEEE International Conference on Intelligent Transportation Systems, ITSC 2014*, pp. 378–385, 2014.

[14] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986. [Online]. Available: <http://dx.doi.org/10.1177/027836498600500106>

[15] M. C. Montel, T. Brenac, M.-A. Granie, M. Millot, and C. Coquelet, "Urban environments, pedestrian-friendliness and crossing decisions," in *Transportation Research Board 92nd Annual Meeting*, 2013, p. 13p.

[16] R. Volpe and P. Khosla, "A theoretical and experimental investigation of impact control for manipulators," *The International Journal of Robotics Research*, vol. 12, no. 4, pp. 351–365, 1993.

[17] J. Varadarajan and J.-M. Odobez, "Topic models for scene analysis and abnormality detection," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 1338–1345.

[18] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," *arXiv preprint arXiv:1612.08242*, 2016.

[19] L. Rummelhard, A. Nègre, and C. Laugier, "Conditional monte carlo dense occupancy tracker," in *Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on*. IEEE, 2015, pp. 2485–2490.

Constant Space Complexity Environment Representation for Vision-based Navigation

Jeffrey Kane Johnson¹

Abstract—This paper presents a preliminary conceptual investigation into an environment representation that has constant space complexity with respect to the camera image space. This type of representation allows the planning algorithms of a mobile agent to bypass what are often complex and noisy transformations between camera image space and Euclidean space. The approach is to compute per-pixel potential values directly from processed camera data, which results in a discrete potential field that has constant space complexity with respect to the image plane. This can enable planning and control algorithms, whose complexity often depends on the size of the environment representation, to be defined with constant run-time. This type of approach can be particularly useful for platforms with strict resource constraints, such as embedded and real-time systems.

I. INTRODUCTION

A significant issue in planning and control when solving real-world navigation problems is that there are often large numbers of individual agents with whom a mobile robot might interact. Consider navigating a busy roadway or crowded sidewalk or convention hall, where there may be multitudes of other agents sharing the space. Conventional approaches to planning in multi-agent systems often explicitly consider interactions between all agents and so become overwhelmed as the number of agents grows [1], [2], [3]. More scalable conventional approaches often have strict requirements on system dynamics [4] or observability of agent policies [5].

This paper presents a preliminary conceptual investigation into the use of a fixed-size environment representation for vision-based navigation. The representation is modeled after a camera image space, which is chosen because cameras are a ubiquitous sensor modality, and image space is by nature discrete and fixed size. The proposed representation additionally allows planning and control routines to reason almost directly in sensor space thereby avoiding often complex and noisy transformations to and from a more conventional Euclidean space representation. The intent of this new representation is to help vision-based mobile robots navigate complex multi-agent systems efficiently, and to take a step toward satisfying the strict resource requirements often present in real-time, safety critical, and embedded systems [6].

The next section briefly surveys related work, then the environment representation is presented along with an illustrative example of how it can be used. Finally, conclusions and future work are discussed.

¹Jeffrey Kane Johnson received his PhD from Indiana University and is principal of Maeve Automation, Mountain View, CA 94043
contact@maeveautomation.com

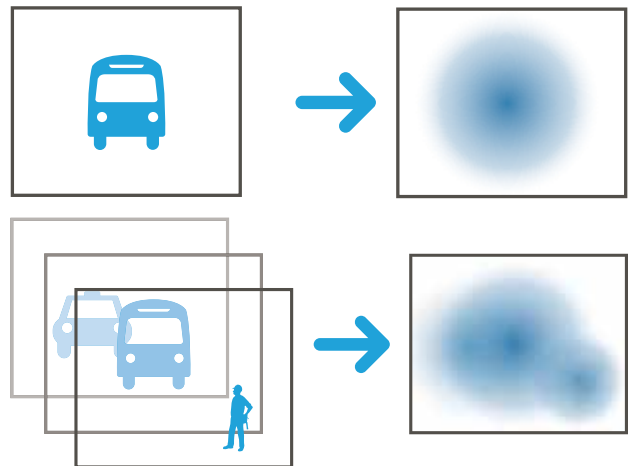


Fig. 1: Top: Illustration of an approaching object in the image plane (left), and the image space potential field (right). Bottom: Multiple object detections (left) can be composed into a single field (right). Black boxes represent ROIs.

II. RELATED WORK

The approach in this work is based on *potential fields* [7]. These fields represent attractive and repulsive forces as scalar fields over a robot’s environment that, at any given point, define a force acting upon the robot that can be interpreted as a control command. This type of approach is subject to local minima and the *narrow corridor* problem [8], particularly in complex, higher-dimensional spaces [9]. Randomized approaches can partially overcome these difficulties [10], [11], and extensions to globally defined *navigation functions* [12], [13], [14], while often difficult to use in practice, theoretically solve them. This work uses potential fields defined over a virtual image plane, which limits the possibility of narrow corridors, and is designed such that additional information can be used by the controller to interpret potential gradients, as suggested in [15]. [16] described a scheme related to that presented in this paper, but places potentials in a Euclidean space, which this approach explicitly avoids. The potential fields computed by the approach in this paper are intended to inform a *visual servoing* [17], [18], [19], [20] control scheme. This is a class of controllers that computes control commands directly from image space data.

In order to define values for the potential fields, this approach draws on a wealth of related works in optical flow and monocular collision avoidance, notably [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [24], [23]. The intuition of these approaches is that the information contained in a

sequence of monocular stills provides sufficient information to compute *time-to-contact* (Definition 2), which informs an agent about the rate of change of object proximity. The primary contribution of this work is a sensor-inspired representation space and algebra for enabling planning and control algorithms to reason effectively and efficiently with the output of this class of perception algorithms.

III. IMAGE SPACE POTENTIAL FIELDS

Before defining image space potential fields, the notion of a potential field used in this paper is defined below:

Definition 1. A *potential field* (also *artificial potential field*) is field of artificial forces that attracts toward desirable locations and repels from undesirable locations. In this work, a potential field is defined by a potential function that maps an image pixel value $I(x,y)$ to a tuple of affinely extended reals $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$, the first of which is the potential value, and the second of which is its time derivative:

$$I(x,y) \mapsto \overline{\mathbb{R}}^2 \quad (1)$$

From the definition, the image space potential (ISP) field is modeled after an image plane. As with image planes, the potential field is discretized into a grid, and regions of interest (ROIs) are defined for it. In this work it is assumed that the fields are origin and axis aligned with the camera images, and that they have the same ROIs (as in Figure 1).

The potential function, which maps image pixel values to potential values, can be defined in arbitrary ways, either with geometric relations, learned relations, or even heuristic methods. In this paper geometric properties of temporal image sequences are used. The approach is to assume an estimation of the *time-to-contact* (defined below) is available for each pixel in an image over time. The mapping of image pixel to this value is taken as the potential function that defines the image space potential field.

Definition 2. *Time-to-contact* (τ), is the predicted duration of time remaining before an object observed by a camera will come into contact with the image plane of the camera. The time derivative of τ is written $\dot{\tau}$.

As noted often in literature (e.g. [30], [25], [27], [29]), τ can be computed directly from the motion flow of a scene, which is defined as:

Definition 3. *Motion flow* is the pattern of motion in a scene due to relative motions between the scene and the camera. In other words, it is a vector field describing the motion of objects on the image plane over time.

Unfortunately, it is typically not possible to measure motion flow directly, so it is usually estimated via *optical flow*, which is defined as the *apparent* motion flow in an image plane. Historically this has been measured by performing some kind of matching of, or minimization of differences between, pixel intensity values in subsequent image frames [31], [28], [32], while more recently deep learning techniques have been successfully applied [33].

The image space potential field is now defined using τ :

Definition 4. An *image space potential field* is defined by a potential function that maps image pixels to a tuple of scalar potential values $\langle \tau, \dot{\tau} \rangle$.

A. Computing τ

Assuming some reasonably accurate estimation of optical flow vector field exists, τ can be computed directly under certain assumptions [25]. In practice, the computation of optical flow tends to be noisy and error prone, so feature- and segmentation-based approaches can be used [24], [23]. The idea of these approaches is to compute τ from the rate of change in detection *scale*. For a point in time, let s denote the scale (maximum extent) of an object in the image, and let \dot{s} be its time derivative. When the observed face of the object is roughly parallel to the image plane, and under the assumption of constant velocity translational motion and zero yaw or pitch, it is straightforward to show that [34]:

$$\tau = \frac{s}{\dot{s}} \quad (2)$$

As shown by Lemma 1, scale has a useful invariance property for these types of calculations that can make τ computations robust to certain types of detection noise:

Lemma 1. *The scale s of an object on the image plane is invariant to transformations of the object under $SE(2)$ on the XY plane.*

Proof. Let (X_1, Y_1, Z) and (X_2, Y_2, Z) be end points of a line segment on the XY plane in the world space, with XY parallel to the image plane and Z coincident with the camera view axis. Without loss of generality, assume unit focal length. The instantaneous scale s of the line segment in the image plane is given by:

$$s = \frac{1}{Z} \sqrt{\Delta X^2 + \Delta Y^2} \quad (3)$$

Thus, any transformation of the line segment on the XY plane for which $\Delta X^2 + \Delta Y^2$ is constant makes s , and thereby \dot{s} and τ , independent of the values of (X_1, Y_1) and (X_2, Y_2) . By definition, this set of transformations is $SE(2)$. \square

In addition, as shown in [35], the time derivative $\dot{\tau}$ of τ , when available, enables a convenient decision function for whether an agent's current rate of deceleration is adequate to avoid head-on collision or not. The decision function is given below, where $\epsilon > 0$ is a buffer to prevent an agent from coming to a stop directly at the point of contact with another agent:

$$f(\dot{\tau}, \epsilon) = \begin{cases} 1 & : \dot{\tau} \geq -0.5 + \epsilon \\ 0 & : \dot{\tau} < -0.5 + \epsilon \end{cases} \quad (4)$$

Equation 2 allows the computation τ for whole regions of the image plane at once given a time sequence of labeled image segmentations, and Equation 4 enables binary decisions to be made about the safeness of the agent's current

state. The following two sections describe encoding these pieces of information into image space potential fields.

B. Computing Fields for a Single Object

Computing the image space potential field for a single object is straightforward given the discussion in §III-A. Assuming an object can be accurately tracked and segmented over time in the camera image frame, its scale s and estimated expansion \dot{s} can be used to compute τ for each pixel in the image plane that belongs to the object, and a finite differences or estimation method can be used to compute $\dot{\tau}$. Pixels that do not belong to the object, and for which no other information is available, are mapped to $\langle \infty, \infty \rangle$ by the potential function. An illustration of this mapping is shown in the top row of Figure 1.

C. Computing Fields for Arbitrary Objects

Computing the image space potential field for arbitrary objects builds on the single object case by computing the field individually for each segmented and tracked object and then composing them into a single field. For this composition to be meaningful, however, the fields cannot be simply added together; this would result in the destruction of the τ information. Instead, a composite field is defined to preserve and combine τ information meaningfully. Equation 5 defines a composite field F in terms of image space potential fields F_1 and F_2 for an image I , and where \min_{τ} selects the tuple whose τ value is minimum:

$$F(x, y) = \left\{ \min_{\tau} (F_1(x, y), F_2(x, y)) \mid (x, y) \in I \right\} \quad (5)$$

Selecting the point-wise τ -minimum tuple for the composite field effectively enforces instantaneous temporal ordinality of objects, i.e., objects that are instantaneously temporally nearer are always what is seen. It is important to note that this is **not** the same as spatial ordinality. For an illustration of this, see Figure 2.

D. Constant Space Complexity

By definition the image space potential field representation has guaranteed constant space complexity assuming that the camera images for which the fields are generated are fixed size. This can be a powerful tool in simplifying planning and control algorithms whose complexity is typically dependent on the number of objects in a scene. In many cases it may, in fact, be possible to achieve constant time for planning and control given this representation.

It is important to note, however, that computing the representation itself may have arbitrary complexity: the problem of segmenting and tracking objects in order to generate these potential fields, for instance, can be efficient or arbitrarily complex, depending on the approach. The problem of investigating efficient computation methods for these fields is a point of future work discussed in §V.

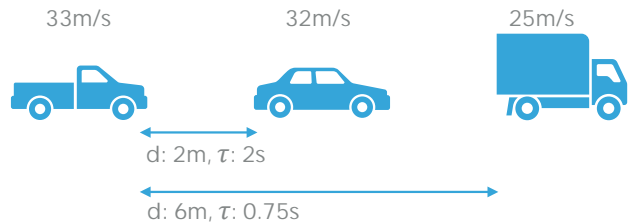


Fig. 2: Illustration comparing spatial and temporal ordinality. Consider three vehicles traveling in the same lane. For the pickup truck (left), the car (middle) has lowest spatial ordinality, i.e., is closest spatially. However, the van (right) has lowest temporal ordinality, i.e., it is nearest temporally.

IV. NAVIGATION WITH IMAGE SPACE POTENTIAL FIELDS

The intuition behind using image space potential functions for vision-based navigation is that they provide a natural space in which to compute collision avoiding controls, which then allows general navigation to be solved using a guided collision avoidance scheme, such as ORCA [4] or the Selective Determinism framework [36]. This section uses the Selective Determinism framework to define a simple control function that utilizes image space potential fields to navigate toward a visible goal in the presence of other agents. In this example only forward field of view is considered, but the extension to omnidirectional field of view is straightforward. The navigation problem considered is defined below:

Problem 1. Assume a set of acceleration-bounded agents \mathcal{A} , each operating according to known dynamics and with similar capabilities, navigating a shared space. Each agent operates according to a unique reward function that is not observable to other agents. Each agent is equipped with a camera that always faces in the direction of motion, and each agent is capable of performing image segmentation on the output. Suppose the reward function for an agent A encourages navigating toward a goal that A has in line of sight. How can A control toward the goal while avoiding collision with other agents?

Problem 1 is the type of problem that a driver may face on a busy highway when trying to navigate toward an exit or offramp. The solution in this example will take a naïve approach of decoupled steering and acceleration control while noting that more sophisticated control schemes are certainly possible. And while the example is formulated for a mobile agent traveling on a two dimensional manifold, the technique in general is equally applicable to three dimensions (such as with a UAV). The method for computing collision avoiding controls is discussed first, followed by the formulation of the navigation law.

A. Collision Avoidance

In order to address collision avoidance, the *Encroachment Detection* problem is presented.

Problem 2. Let *encroachment* refer to the reduction in minimum proximity between two or more objects in a workspace \mathcal{W} beyond desired limits as measured by a metric $\mu(\cdot, \cdot)$. Assume an agent A receives some observation input O_t of \mathcal{W} over time. Let \mathcal{A} be the set of agents that does not include A . For a sequence of observations O_1, \dots, O_t , how can A estimate the rate of change of $\min_{A_j \in \mathcal{A}} \mu(A, A_j)$?

Note that maintaining an estimate of $\langle \tau, \dot{\tau} \rangle$ directly solves the problem, as these values quantify temporal proximity and the rate of encroachment. The collision avoidance problem can now be solved by detecting encroachment and controlling such that it does not violate limits.

It was shown in [37] that collision avoidance can be guaranteed in a non-adversarial system if all agents compute and maintain collision-free stopping paths, which are contingency trajectories that bring an agent to zero relative velocity in minimal time. If agents are also self-preserving, they can each assume that all other agents will maintain such contingencies. Under these assumptions, agents should have sufficient information in the image space potential field to compute a solution to Problem 2 by maintaining non-zero time headway, which is assumed to be witness to the existence of a feasible stopping path.

For illustration, a naïve control set computation in the spirit of the braking controller in [38], [39] is sketched in Algorithm 1. This routine makes the reasonable assumption that $\dot{\tau}$ is not so large as to overwhelm τ . The idea is that steering angle and acceleration commands are computed independently and such that τ thresholds are not violated. To compute steering angles, a min filter is swept across the field of view in the potential field and a minimum potential value within the window is computed for each column in the image. Any value that meets τ thresholds is kept, and these are considered the safe steering angles (Figure 3, left). To compute the acceleration command, the minimum potential value within a centered window of a specified width is considered (Figure 3, right). If the value meets the τ threshold, the full scaled range of accelerations, $[-1, 1]$, is considered safe. If the threshold is violated, then either full deceleration $[-1, -1]$ or the range of deceleration values $[-1, 0]$ is sent depending on the value of the decision function of Equation 4. The control sets are then used by the Selective Determinism framework to compute the output control command.

B. The Selective Determinism Framework

Selective Determinism [36] is a solution framework for dynamically-constrained, non-adversarial, partially-observable multi-agent navigation problems. It belongs to a family of approaches useful for dealing with real-world problems because they remove the theoretical intractability inherent in optimal approaches [40], [41] while typically exhibiting good empirical performance. Selective Determinism, in addition, can also make certain collision avoidance guarantees even without explicitly considering interaction effects among agents.

Algorithm 1 Given an image space potential field F , compute the set of steering and acceleration commands that satisfy $\tau \geq T_s$ and $\dot{\tau} \geq -0.5 + \epsilon$, where $T_s > 0$ is some desired time headway, w_θ and w_a are kernel widths for computing steering angle and acceleration maps, and $\epsilon > 0$ is the buffer from Equation 4.

```

1: procedure SAFECONTROLS( $F, T_s, \dot{\tau}_E, w_\theta, w_a, \epsilon$ )
2:   Let  $I_c$  be the list of image column indices
3:   Let  $M_\theta$  map  $i \in I_c$  to steering angles
4:   Let  $h$  be the height (row count) of  $F$ 
5:   Let  $M_\tau$  map  $\langle \tau, \dot{\tau} \rangle$  to  $i \in I_c$  via  $w_\theta \times h$  min filter
6:   Let  $M_\theta = \{ \langle \tau, \dot{\tau} \rangle \in M_\tau : \tau \geq T_s \}$ 
7:   Let  $W$  be a centered  $w_a \times h$  window in  $F$ 
8:   Let  $\langle \tau, \dot{\tau} \rangle_{\min}$  be the min.  $\tau$  over  $W$ 
9:   Let  $L \leftarrow \emptyset$  be a container for safe accelerations
10:  if  $M_\theta = \emptyset$  then
11:     $M_\theta \leftarrow 0, L \leftarrow [-1, -1]$ 
12:  else if  $\tau_{\min} > T_s$  then
13:     $L \leftarrow [-1, 1]$ 
14:  else
15:    if  $f(\dot{\tau}, \epsilon) = 0$  then
16:       $L \leftarrow [-1, -1]$ 
17:    else
18:       $L \leftarrow [-1, 0)$ 
19:    end if
20:  end if
21:  return  $M_\theta, L$ 
22: end procedure

```

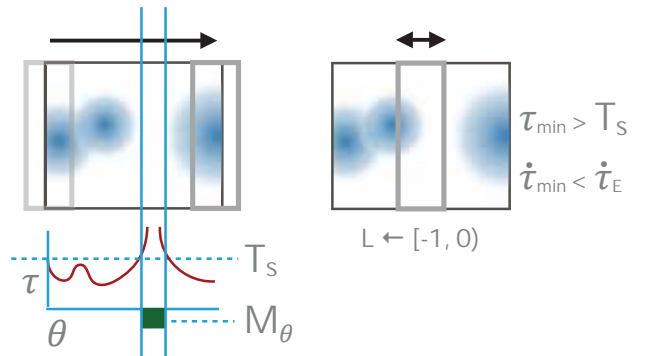


Fig. 3: Illustration of the steering angle control computation (left) and the acceleration control computation (right). On the left, a window sweeps from left to right over the image space potential field computing minimum τ for each image space column (left bottom). The set of column values that satisfy the threshold are the set of acceptable steering angles M_θ . On the right, the minimum potential value over a centered window is computed and the set L of acceptable scaled acceleration values are determined from it.

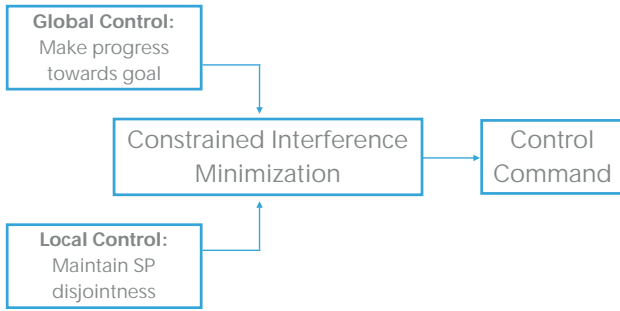


Fig. 4: Architecture of the Selective Determinism framework.

Selective Determinism works by exploiting the idea that agents in a system are capable of independently computing contingency trajectories that cover the space necessary for them to come to a stop (or to a zero relative velocity), and it assumes that agents do so, and that they will seek to maintain a non-empty set of available contingencies.

The framework casts the navigation problem in terms of a constrained interference minimization [38], [39] that utilizes a local collision avoidance controller to compute sets of controls from which an optimization chooses a control that makes maximal progress toward some goal (see Figure 4). The solution to Problem 1 is sketched in Algorithm 2.

Algorithm 2 For a desired pixel location (x_d, y_d) , and setpoint speed \dot{s}_d , compute the Selective Determinism control that safely guides the agent A toward (x_d, y_d) . See Algorithm 1 for descriptions of the other parameters.

```

1: procedure CONTROLS( $(x_d, y_d), F, T_s, \hat{\tau}_E, w_\theta, w_a, \epsilon$ )
2:   Let  $\theta_t, \dot{s}_t$  be the steering angle and speed of  $A$ 
3:   Let  $\theta_d$  be the steering angle corresponding to  $y_d$ 
4:   Let  $M_\theta, L \leftarrow \text{SafeControls}(F, T_s, \hat{\tau}_E, w_\theta, w_a, \epsilon)$ 
5:   Let  $\theta^* \leftarrow \theta_t$  contain the new steering angle
6:   for  $\theta \in M_\theta$  do
7:     if  $|\theta - \theta_d| < |\theta^* - \theta_d|$  then
8:        $\theta^* \leftarrow \theta$ 
9:     end if
10:  end for
11:  Let  $\dot{s}^* \in L$  be chosen proportionally to  $\dot{s}_d - \dot{s}_t$ 
12:  return  $\theta^*, \dot{s}^*$ 
13: end procedure
  
```

C. Complexity Analysis

In Algorithm 1 all non-trivial operations are iterations over the width of the image plane, which is assumed to be fixed for a given problem. The operations on lines 5 & 7 depend on the user defined w_θ and w_a parameters, but these are also bounded by image width. In Algorithm 2, Line 4 is a call to Algorithm 1, and so has constant complexity with respect to the image space, and Line 11 is assumed to be implemented with an $O(C)$ proportional law. Thus, the navigation algorithm as a whole has constant complexity, in space and time, with respect to the camera image space.

V. CONCLUSION & FUTURE WORK

This paper presented a conceptual investigation into an environment representation for vision-based navigation that has constant space complexity with respect to the image. This preliminary work is intended to serve as a basis for future investigations. This section outlines three primary topics of investigation.

The first topic is how to more completely combine environment information with the potential fields. As presented here, the representation is defined strictly in terms of object τ values, but a more elegant solution would build richer information about the environment into the potential field itself. An obvious extension would be to encode path information, such as lane or sidewalk boundaries, as well as goal information, into the potential field. This would enable navigation in semantically sophisticated environments and is an area of active development¹.

The second topic is a more sophisticated control law. The decoupled approach used here can lead to odd and counter-productive behavior, such as swerving out of the way of approaching objects while at the same time slowing down, or instability around control modes. A more intelligent control law would address stability issues and reason about the steering and longitudinal controls simultaneously. Additionally, allowing the potential fields to label a small, fixed-size set of objects individually could let such a control law reason about individual interactions without losing constant space complexity or information about all other objects.

Finally, the third topic, and one of great importance, is whether and how the potential fields themselves can be computed with some kind of constant complexity. A purely optical flow based approach would address this, but would require breakthroughs in the quality and efficiency of optical flow algorithms. Alternatively, a purely learning-based approach in conjunction with cheap, heuristic-based tracking approaches may provide the requisite segmentation and tracking information without runaway complexity.

REFERENCES

- [1] S. Brechtel, T. Gindele, and R. Dillmann, "Probabilistic mdp-behavior planning for cars," in *14th International IEEE Conference on Intelligent Transportation Systems, ITSC 2011, Washington, DC, USA, October 5-7, 2011*, 2011, pp. 1537–1542. [Online]. Available: <https://doi.org/10.1109/ITSC.2011.6082928>
- [2] E. Galceran, A. G. Cunningham, R. M. Eustice, and E. Olson, "Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction: Theory and experiment," *Auton. Robots*, vol. 41, no. 6, pp. 1367–1382, 2017. [Online]. Available: <https://doi.org/10.1007/s10514-017-9619-z>
- [3] F. A. Oliehoek and C. Amato, *A Concise Introduction to Decentralized POMDPs*, ser. Springer Briefs in Intelligent Systems. Springer, 2016.
- [4] J. van den Berg, S. J. Guy, M. C. Lin, and D. Manocha, "Reciprocal n -body collision avoidance," in *Robotics Research - The 14th International Symposium, ISRR, August 31 - September 3, 2009, Lucerne, Switzerland, 2009*, pp. 3–19.
- [5] K. E. Bekris, D. K. Grady, M. Moll, and L. E. Kavraki, "Safe distributed motion coordination for second-order systems with different planning cycles," *I. J. Robotic Res.*, vol. 31, no. 2, pp. 129–150, 2012. [Online]. Available: <http://dx.doi.org/10.1177/0278364911430420>

¹https://bitbucket.org/maeveautomation/maeve_automation_core/src

- [6] S. Louise, V. David, J. Delcoigne, and C. Aussaguès, "OASIS project: deterministic real-time for safety critical embedded systems," in *Proceedings of the 10th ACM SIGOPS European Workshop, Saint-Emilion, France, July 1, 2002*, 2002, pp. 223–226. [Online]. Available: <http://doi.acm.org/10.1145/1133373.1133419>
- [7] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proceedings of the 1985 IEEE International Conference on Robotics and Automation, St. Louis, Missouri, USA, March 25-28, 1985*, pp. 500–505. [Online]. Available: <https://doi.org/10.1109/ROBOT.1985.1087247>
- [8] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *IN PROC. IEEE INT. CONF. ROBOTICS AND AUTOMATION*, 1991, pp. 1398–1404.
- [9] S. M. LaValle, *Planning algorithms*. Cambridge University Press, 2006.
- [10] *A Monte-Carlo algorithm for path planning with many degrees of freedom*, 1990. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=126256
- [11] J. Barraquand and J. Latombe, "Robot motion planning: A distributed representation approach," *I. J. Robotics Res.*, vol. 10, no. 6, pp. 628–649, 1991. [Online]. Available: <https://doi.org/10.1177/027836499101000604>
- [12] D. Koditschek and E. Rimon, "Robot navigation functions on manifolds with boundary," *Advances in Applied Mathematics*, vol. 11, no. 4, pp. 412–442, 1990.
- [13] D. C. Conner, A. A. Rizzi, and H. Choset, "Composition of local potential functions for global robot control and navigation," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 4, Oct 2003, pp. 3546–3551 vol.3.
- [14] D. E. Koditschek, "Exact robot navigation by means of potential functions: Some topological considerations," in *Proceedings of the 1987 IEEE International Conference on Robotics and Automation, Raleigh, North Carolina, USA, March 31 - April 3, 1987*, 1987, pp. 1–6. [Online]. Available: <https://doi.org/10.1109/ROBOT.1987.1088038>
- [15] A. A. Masoud, "Solving the narrow corridor problem in potential field-guided autonomous robots," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, ICRA 2005, April 18-22, 2005, Barcelona, Spain, 2005*, pp. 2909–2914. [Online]. Available: <https://doi.org/10.1109/ROBOT.2005.1570555>
- [16] M. T. Wolf and J. W. Burdick, "Artificial potential functions for highway driving with collision avoidance," in *2008 IEEE International Conference on Robotics and Automation, ICRA 2008, May 19-23, 2008, Pasadena, California, USA, 2008*, pp. 3731–3736. [Online]. Available: <https://doi.org/10.1109/ROBOT.2008.4543783>
- [17] N. J. Cowan and D. E. Chang, "Toward geometric visual servoing," EECS Department, University of California, Berkeley, Tech. Rep. UCB/CSD-02-1200, Sep 2002. [Online]. Available: <http://www2.eecs.berkeley.edu/Pubs/TechRpts/2002/5822.html>
- [18] L. E. Weiss, A. C. Sanderson, and C. P. Neuman, "Dynamic visual servo control of robots: An adaptive image-based approach," in *Proceedings of the 1985 IEEE International Conference on Robotics and Automation, St. Louis, Missouri, USA, March 25-28, 1985*, pp. 662–668. [Online]. Available: <https://doi.org/10.1109/ROBOT.1985.1087296>
- [19] R. T. Fomena and F. Chaumette, "Visual servoing from spheres using a spherical projection model," in *2007 IEEE International Conference on Robotics and Automation, ICRA 2007, 10-14 April 2007, Roma, Italy, 2007*, pp. 2080–2085. [Online]. Available: <https://doi.org/10.1109/ROBOT.2007.363628>
- [20] A. X. Lee, S. Levine, and P. Abbeel, "Learning visual servoing with deep features and fitted q-iteration," *CoRR*, vol. abs/1703.11000, 2017. [Online]. Available: <http://arxiv.org/abs/1703.11000>
- [21] A. Schaub and D. Burschka, "Reactive avoidance of dynamic obstacles through optimization of their epipoles," in *2015 19th International Conference on System Theory, Control and Computing (ICSTCC)*, Oct 2015, pp. 318–324.
- [22] G. Alenyà, A. Nègre, and J. L. Crowley, "Time to contact for obstacle avoidance," in *Proceedings of the 4th European Conference on Mobile Robots, ECMR'09, September 23-25, 2009, Mlini/Dubrovnik, Croatia, 2009*, pp. 19–24.
- [23] J. Byrne and C. J. Taylor, "Expansion segmentation for visual collision detection and estimation," in *IEEE International Conference on Robotics and Automation, Kobe, Japan, May 12-17, 2009*, pp. 875–882.
- [24] T. Mori and S. Scherer, "First results in detecting and avoiding frontal obstacles from a monocular camera for micro unmanned aerial vehicles," in *2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, May 6-10, 2013*, pp. 1750–1757.
- [25] T. Camus, D. Coombs, M. Herman, and T. Hong, "Real-time single-workstation obstacle avoidance using only wide-field flow divergence," in *13th International Conference on Pattern Recognition, ICPR 1996, Vienna, Austria, 25-19 August, 1996*, 1996, pp. 323–330.
- [26] S. J. Pundlik, E. Peli, and G. Luo, "Time to collision and collision risk estimation from local scale and motion," in *Advances in Visual Computing - 7th International Symposium, ISVC 2011, Las Vegas, NV, USA, September 26-28, 2011. Proceedings, Part I, 2011*, pp. 728–737.
- [27] D. Coombs, M. Herman, T. Hong, and M. Nashman, "Real-time obstacle avoidance using central flow divergence and peripheral flow," in *ICCV*, 1995, pp. 276–283.
- [28] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Image Analysis, 13th Scandinavian Conference, SCIA 2003, Halmstad, Sweden, June 29 - July 2, Proceedings, 2003*, pp. 363–370.
- [29] R. C. Nelson and Y. Aloimonos, "Obstacle avoidance using flow field divergence," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 10, pp. 1102–1106, 1989.
- [30] Y. Barniv, "Expansion-based passive ranging," NASA Ames Research Center, Moffett Field, California, Tech. Rep. 19930023159, June 1993.
- [31] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, no. 1-3, pp. 185–203, 1981. [Online]. Available: [https://doi.org/10.1016/0004-3702\(81\)90024-2](https://doi.org/10.1016/0004-3702(81)90024-2)
- [32] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the 7th International Joint Conference on Artificial Intelligence, IJCAI '81, Vancouver, BC, Canada, August 24-28, 1981*, 1981, pp. 674–679.
- [33] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "Deepflow: Large displacement optical flow with deep matching," in *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, 2013, pp. 1385–1392. [Online]. Available: <https://doi.org/10.1109/ICCV.2013.175>
- [34] T. Camus, "Real-time optical flow," Ph.D. dissertation, Brown University, Department of Computer Science, 1994.
- [35] D. N. Lee, "A theory of visual control of braking based on information about time-to-collision," *Perception*, vol. 5, no. 4, pp. 437–459, 1976.
- [36] J. K. Johnson, "Selective determinism for autonomous navigation in multi-agent systems," Ph.D. dissertation, Indiana University, Department of Computer Science, 2017.
- [37] —, "A novel relationship between dynamics and complexity in multi-agent collision avoidance," in *Proceedings of Robotics: Science and Systems*, Ann Arbor, Michigan, June 2016.
- [38] J. Johnson, Y. Zhang, and K. Hauser, "Semiautonomous longitudinal collision avoidance using a probabilistic decision threshold," in *IROS 2011 International workshop on Perception and Navigation for Autonomous Vehicles in Human Environment*, 2011.
- [39] —, "Minimizing driver interference under a probabilistic safety constraint in emergency collision avoidance systems," in *2012 15th International IEEE Conference on Intelligent Transportation Systems*, Sept 2012, pp. 457–462.
- [40] A. Weinstein and M. L. Littman, "Open-loop planning in large-scale stochastic domains," in *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, ser. AAAI'13. AAAI Press, 2013, pp. 1436–1442.
- [41] C.-H. Yu, J. Chuang, B. Gerkey, G. J. Gordon, and A. Ng, "Open-loop plans in multi-robot pomdps," Stanford University, Tech. Rep., 2005.

Relocalization under Substantial Appearance Changes using Hashing

Olga Vysotska

Cyryll Stachniss

Abstract—Localization under appearance changes is essential for robots during long-term operation. This paper investigates the problem of place recognition in environments that undergo dramatic visual changes. Our approach builds upon previous work on graph-based image sequence matching and extends it by incorporating a hashing-based image retrieval strategy in case of localization failures or the kidnapped robot problem. We present a variant of hashing algorithm that allows for fast retrieval for high-dimensional CNN features. Our experiments suggest that our algorithm can reliably recover from localization errors by globally relocalizing the robot. At the same time, our hashing-based candidate selection is substantially faster than state-of-the-art locality sensitive hashing.

I. INTRODUCTION

The ability to localize itself is an essential capability for goal-directed robot navigation. A central ingredient of localization as well as mapping is the capability to identify that the robot is at a previously visited place, i.e., to make the data association between the current observation and a previously taken one. When operating in changing environments such as outdoor scenes, the localization system should be able to deal with substantial appearance changes. An example for such appearance changes is depicted in Fig. 1. Both images correspond to the observations taken at the same physical location but at different times during the day.

The task of localization through image matching for handling substantial changes in the appearance of a place has been tackled by several researchers [4], [5], [7], [14], [19], [21]. In line with previous work on this topic, we also rely on sequence information, i.e., exploit the fact the images are not obtained in a random order but according to the physical motion of the robot through the environment. We solve the problem by building a data association graph, where possible paths through this graph correspond to different image matching hypothesis. Efficient relocalization is achieved by a locality sensitive hashing strategy.

The main contribution of this work is a online approach for finding correspondences between the currently acquired image stream and a previously recorded image sequence even under strong appearance changes. Our work is an extension of our previous work as it uses the lazy search approach proposed in [21] but proposes several extensions. First, we provide a way for dealing with loops in the reference or database sequences and introduce new edges into the data association graph that is build up on the fly. Second, we

All authors are with the University of Bonn, Institute of Geodesy and Geoinformation, Bonn, Germany.

This work has partially been supported by the DFG under contract number FOR 1505 Mapping in Demand.



Fig. 1. Challenging image pairs for place recognition systems. Both images have been recorded at the same place but during different times resulting in strong appearance changes. The approach presented in this paper identifies such corresponding images via sequence information and can handle loops in the database sequences, recover from localization failures, as well as deal with the kidnapped robot problem.

provide an efficient way for relocalizing the robot in case it got lost. Both extensions naturally integrate with and extend [21] so that the same search approach and search heuristic can be re-used. This furthermore does not affect the online nature of the solution and the data association graph is still built incrementally.

We make the following three claims for our approach. It is able to (i) quickly relocalize the robot globally after getting lost and can handle the kidnapped robot problem, (ii) can be executed in an online fashion during navigation and requires only a small amount of image to image comparisons, and (iii) deal with loops in the reference images sequence while not relying on GPS information or similar means. These three claims are backed up through our experimental evaluation.

II. RELATED WORK

Localization is a relevant and frequently studied problem in robotics. A prominent approach to visual localization is FAB-MAP2 [5]. Dealing with substantial variations in the visual input, however, has been recognized as an obstacle for persistent autonomous navigation and one way to address it is to exploit sequence information for the image alignment [9], [13], [12], [14].

Over the past few years, different types of features have been investigated for place recognition. Some approaches use variants of HOG features such as [14] or Bag of Words models optimized to seasonal changes [16]. More recently, multiple researchers apply learned features as proposed by Sermanet *et al.* [17] and suggested for place recognition by Chen *et al.* [3]. These CNN features yield a high matching quality but are rather high-dimensional, i.e., comparisons are computationally expensive. This motivates the binarizations of such features and efficient comparisons using the Hamming distance [2]. Other alternatives are place-dependent features, which are optimized to the current location [11].

There is an increasing interest in the systems that can localize under appearance changes and different weather conditions. The experience-based navigation paradigm [4] stores multiple images or experiences for individual places and extends the place model whenever matching the current images to previous ones becomes challenging. Extension of experience-based navigation targets large-scale localization by exploiting a prioritized collection of relevant experiences so that the number of matches can be reduced [9]. SeqSLAM [13] aims at matching image sequences under strong seasonal changes and computes an image-by-image matching matrix that stores similarity scores between the images in a query and database sequence. It computes a straight-line path through the matching matrix and selects the path with the smallest sum of similarity scores across image pairs to determine the matching route. Milford *et al.* [12] present a comprehensive study about the SeqSLAM performance on low resolution images. Related to that, Naseer *et al.* [14] focus on offline sequence matching using a network flow approach and Vysotska *et al.* [21] extended this idea towards an online approach with lazy data association and build up a data association graph online on demand. Our paper extends [21] by allowing more flexible reference trajectories as well as efficient means for relocalization.

In visual place recognition, relocalization after getting lost can be achieved by comparing the query image to all images in the reference dataset as it was used by Neubert *et al.* [15]. To optimize the process of finding similar images in large datasets Gionis *et al.* [6] proposed using hashing algorithm, which was intensively used to solve text retrieval problems, to search for duplicates and even similar images in the large dataset, known as locality sensitive hashing (LSH). In LSH, slight variation in the image domain should only lead to slight variations in the hash. The disadvantage of LSH is that it relies on a quite large number hash tables (> 100 is suggested in practice) to obtain a high retrieval accuracy. To tackle this problem, Lv *et al.* [10] propose an efficient indexing strategy, which allowed to reduce the number of hash tables.

The popularity of the CNNs resulted in learned features, which are high-dimensional in comparison to those used by Lv *et al.* This slows down multi-probe LSH when matching full image sequences. An alternative approach to improve retrieval is spectral hashing [22], where a variant of spectral clustering is performed on the database before the operation in order to find better hash codes. Due to the high-dimensionality of CNN features, spectral clustering becomes computationally intractable and thus we rely on a variant of LSH proposed by Lv *et al.*

III. OUR APPROACH

A. Lazy Data Association for Image Sequence Matching

The approach proposed in this paper is an extension of our previous approach for visual place recognition in changing environments [21]. The central idea of [21] is to perform visual place recognition by matching image sequences: Given a sequence of images, also called reference sequence, find

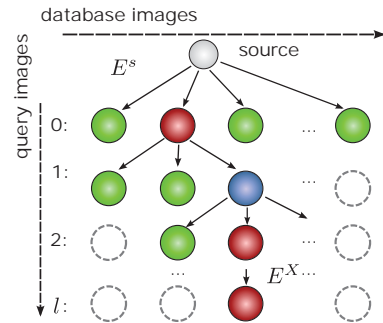


Fig. 2. Graph structure inherited from the [21]. Green circles denote expanded nodes (for which two feature vectors are compared); right circle - match; blue - non match, but support the path hypothesis.

for the stream of incoming images the visually most similar (sub-)sequence in the reference sequence. For making the required associations, the approach uses the ideas of lazy data association and solves the matching problem in an online fashion.

We formulate the problem of matching image sequences as a graph search problem, in which every node represents the potential match between two images and edges encode possible transitions between nodes. The shortest path through this graph corresponds to the most likely data association between the image sequences. Fig. 2 depicts the structure of such a data association graph: green circle denotes the nodes for which the matching cost has been computed and blue (or red) nodes are those that belong to the found path. The blue nodes are so-called hidden nodes, the ones that support the path topology, but whose matching cost is higher than a specified parameter (non-matching cost), i.e. the image appear dissimilar. The red nodes are the ones for which corresponding matching cost is lower than the non-matching cost. The search ends (for each new image in the sequence) when the latest image gets associated to a reference image, either as hidden or real node.

This approach can accurately match image sequences but has also some limitations. First, it cannot relocalize well once lost. Second, it makes assumptions that query trajectory roughly follows the reference trajectory. In this paper, we overcome both limitations.

B. Robust Image Matching Costs with CNN Features

To align image sequences, we need to match the individual images. Our approach represent each image by a single high-dimensional feature vector. In their extensive study, Chen *et al.* suggest that the 10th layer of the convolutional neural network OverFeat [17] produces robust features for changing environments. The size of the output feature vector depends on the size of the input image. We opted for the smallest acceptable size of 450×250 pixels, which results in feature vector of approx. 200,000 dimensions. Note, however, that our algorithm is not limited to this kind of features and we will plan to investigate alternative features such those from VGG-16 [18], Net-VLAD [1] and PoseNet [8] in our future work.

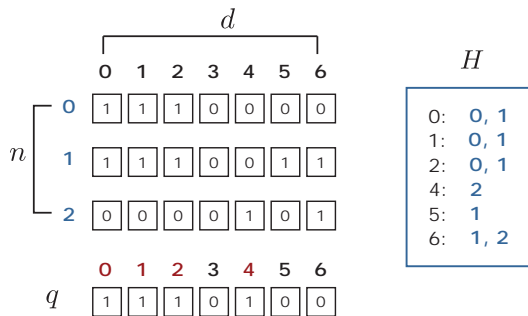


Fig. 3. Example of the proposed hashing algorithm. Here the dataset consists of 3 feature vectors of dimension 7 each. An entry of hash table $H[2]$ stores the IDs of the feature vectors 0 and 1, since for both of them, dimension 2 has the value of 1. For a query feature, the set of dimensions that take a value of 1 is $A = \{0, 1, 2, 4\}$. By collecting the values from H , the set of potential matching candidates is 0 with occurrence 3, 1 with occurrence 3 and 2 with occurrence 1. The resulting matching candidates for query q are $\{0, 1\}$.

C. Efficient Relocalization Strategy

No localization system is free of failures. Thus, it can happen that a robot gets lost, i.e., it cannot establish a correspondence between its current observations and the model anymore. In our case, this means that the (sub-)sequence of images, which the robot is currently acquiring, cannot be matched to the reference sequence anymore. A common reason for that in practice is the fact that the robot moves along a so far unseen trajectory, for example, when leaving the mapped area.

A good relocalization system should be able to detect whenever the robot reenters the previously mapped area in order to resume or restart the localization. To detect whether the robot is lost, we analyze the nodes of the best current matching hypothesis within the sliding window over time. If the percentage of the hidden nodes within this window exceeds 80%, i.e. only 20% of the images can be matched to the reference sequence, we consider the robot as lost. The size of the window depends on the framerate of the camera and potentially also on the speed of the robot¹.

A straightforward but computationally demanding way to find a reentry point is a brute force search through the whole reference database. Instead, we propose to use hashing for identifying potential reentry points. This results in comparing a query image only to the subset of the database images that are mapped to the same hash key. Hashing techniques are known to be robust and efficient to find image duplicates. In contrast to standard (cryptographic) hashing such as MD5 or SHA1, hashing for image retrieval is expected to assign similar features to the same or neighbouring buckets, i.e., to similar hash keys. This property is referred to as *locality sensitive*. Locality sensitive hashing (LSH) proposed in [6] was one of the first approaches to apply hashing for image retrieval problems.

We found that the use of an improved version of the LSH, called Multi-Probe LSH proposed by Lv *et al.* [10] is better

¹In our experiments (using car in an urban environment), the size was set to 10s.

suitable for image matching tasks. Multi-Probe LSH builds on top of the LSH but specifies an intelligent strategy to probe specified buckets in multiple hash tables to get the higher probability of finding similar images. In our work, we use the Multi-Probe LSH in the following way: The moment the robot is considered *lost*, the algorithm starts to hash every incoming image q_i and looks up for candidates $C(q_i)$, stored in the hash buckets, according to the probe strategy. More information about the probing strategy can be found in [10]. After the potential matching candidates are retrieved, we add the corresponding nodes to the graph

$$E^{\text{reentry}} = \{(x_{(i-1)j}, x_c)\}_{c \in C(q_i)} \quad (1)$$

where i is the id of the current query image q_i , the term $x_{(i-1)j}$ corresponds to a node representing current best matching hypothesis, and c refer to ids of the images in the reference dataset that were retrieved as matching candidates based on hashing.

Originally, Multi-Probe LSH was designed to match images that were taken under similar conditions and was used with relatively low dimensional features, e.g. 64 or 192 dimensions. In this work, however, we rely on high dimensional features (around 200K dimensions) as they show a better matching performance under changing conditions. This naturally leads to an increase in the querying time for computing the potential candidates from the database.

To tackle this issue, we propose an alternative hashing algorithm designed to explicitly take into account the high-dimensionality of the data and thus improve the querying time without compromising matching performance.

As in every hashing algorithm, the first step is to construct the hash table H . We start with binarizing the feature vectors that represent the images from the reference dataset. We inherit the binarization strategy for CNN features proposed by Arroyo *et al.* [2]. To receive the binary values, we first scale the feature vector so that the dimension with largest spread lies in the interval $[0, 255]$ and afterwards all dimensions higher than a middle value 128 are assigned to 1, others 0. Formally:

$$f^{\text{int}} = (f^{\text{cnn}} - \min(f^{\text{cnn}})) \frac{255}{\max(f^{\text{cnn}}) - \min(f^{\text{cnn}})} \quad (2)$$

$$f^{\text{bin}} = \begin{cases} 1, & \text{if } f^{\text{int}} \geq 128, \\ 0, & \text{if } f^{\text{int}} < 128 \end{cases} \quad (3)$$

Let us assume we need to hash a dataset that consists with N features indexed with $n \in [0, N]$ and each of the feature has D dimensions indexed with $d \in [0, D]$. Then, every entry in the hash table $H[d]$ stores the set of indices of all the features that have a value of 1 in dimension d :

$$H[d] = \{n \mid f_n^{\text{bin}}[d] = 1\} \quad (4)$$

In a query phase, the incoming image q gets also binarized using the same procedure as in Eq. (3). Afterwards, we extract a set of indices $A(q)$ of dimensions take the value of 1 for the image q :

$$A(q) = \{d \mid f_q^{\text{bin}}[d] = 1\}, \quad (5)$$

where $|A| = M < D$ and typically $M \ll D$. We collect all the feature indices from the hash table, that take a value of 1 for the dimension stored in A

$$H[A] = \hat{\cup}_{a \in A} H[a], \quad (6)$$

where $\hat{\cup}$ denotes the set union preserving duplicates. We intentionally keep the duplicates in the set to further select those feature candidates that have high number of occurrences in the set $H[A]$. This represents the fact that the query feature q and candidate features from the database share a substantial set of feature dimensions taking a value of 1. Thus, they are likely to represent the same place. For an illustration of the hashing procedure, consider the toy-example in Fig. 3.

D. Loopy Reference Sequences

While recording the reference image sequence, it can happen that the robot moves along the same route multiple times. It may even be unavoidable given the topology of the environment. In practice, this situation occurs frequently when considering a typical urban mapping run using a car. In Fig. 4 (left), we provide a sketch of a trajectory that shows the situation in which reference trajectory visits the same place in the environment twice from 'B' to 'C'. The cost matrix for a corresponding real world situation is depicted in Fig. 13 (left) in the experimental section. In these matrices, bright pixels correspond to a pair of images that appear similar given the feature vectors, whereas dark pixels suggest a low similarity. In this case after visiting the place 'C', there are two possibilities to proceed, either visiting 'C-D' or 'C-F'. As the query trajectory follows the 'C-F' route, we can see a brighter pattern on the right lower part of the cost matrix in the Fig. 13 (left), whereas if the query trajectory followed the 'C-D' direction, the pattern would appear in a left lower part of the matrix. The previous version of our approach [21], cannot handle this situation flexibly, because the query sequence is expected to roughly follow the reference one.

In this paper, we extend the approach so that the search algorithm can flexibly "jump" between similar places in reference sequence. The ability to "jump" is established by creating the edges in the graph between the nodes that correspond to the similar places in the environment. If we know that image a corresponds to image b within the reference sequence, then whenever the algorithm is requested to expand the graph from the node x_{ia} , it will also expand from the node x_{jb}

$$E^{\text{sim}} = \{(x_{ia}, x_{(i+1)k})\}_{k=b-K, \dots, b+K} \quad (7)$$

where K - is a fan-out parameter that compensates for different robot speeds or camera frame rates. The same thoughts hold if the graph gets expanded from the node x_{ib} .

To be able to establish these nodes, we need to identify which places, i.e. images, in the reference dataset represent the same place. Since the evaluations are performed only on the reference dataset, finding of the similar place can be done offline using standard place recognition algorithm such

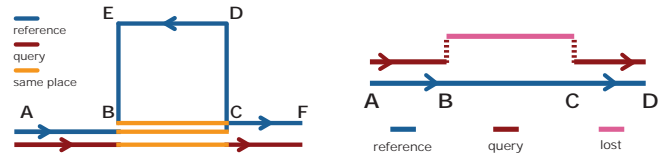


Fig. 4. Left: Sketch of similar places situation. Right: sketch of a query detour during which the robot is lost.

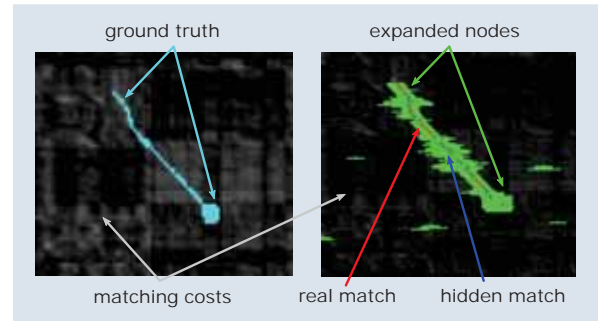


Fig. 5. Example of possible outputs in our experiments. The cost matrix stores the costs of matching individual images (not used in our algorithm). Expanded nodes - matching costs computed in our algorithm. Real matches - image pairs that represent the same place and hidden match - image pairs that support the path hypothesis, but have low matching cost. Ground truth matches that represent the same place in reality.

as FABMAP2 [5] since the images stem from the the same appearance within the reference trajectory.

IV. EXPERIMENTAL EVALUATION

Our experiments are designed to show the capabilities of our method and to support our key claims, which are: Our approach is able to (i) quickly re-localize after a period while navigating without additional pose information, (ii) handle the kidnapped robot problem, (iii) be executed in an online fashion, and (iv) deal with loops in the reference images sequence. We furthermore provide comparisons to an existing methods such as [14], [13], [21]. We perform the evaluations on own datasets as well as on publicly available ones. To support the claims made in the beginning of the paper, we have selected the datasets that explicitly represent a particular challenge for localization. Some of them stem from the *Freiburg datasets* used in [20], [21] and others from the VPRICE Challenge dataset. Additionally, we collected a more challenging dataset in terms of trajectory shapes. We collected the data in Bonn with a car and a dashboard camera. The query as well as reference trajectory contains several revisits of the same places. In this paper, we use the datasets collected in the morning with slight rain and overcast as well as in the evening and very late evening on different days. Example images can be seen in Fig. 1.

Note, that throughout all the experiments the cost matrix was only computed for visualization purposes. The matching algorithm only computes the matching costs for the image pairs visualized in green. To enhance the visualization of the cost matrix for larger datasets, we also overlaid the ground truth results, see Fig. 5 for further notations.

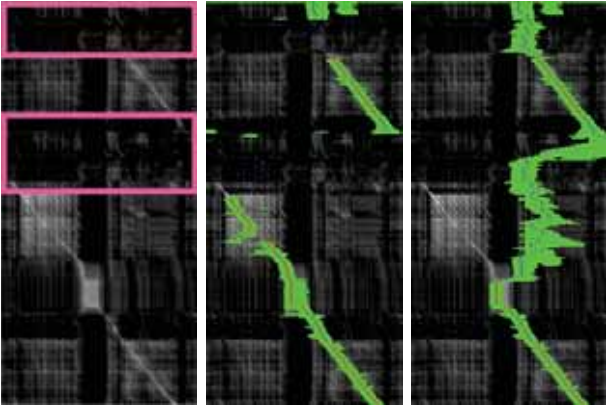


Fig. 6. Example of a cost matrix for matching two trajectories, where the query trajectory deviates from the reference one twice. Once at the beginning and then in the middle. The places are marked with pink rectangles. Left: matching matrix. Middle: result of proposed algorithm. Right: Result using previous approach. Note that the full cost matrix for matching is only shown for visualization and does not need to be computed by our approach.

A. Matching Performance and Localization Recovery

The first set of experiments is designed to show that our approach is able to quickly relocalize after the system has identified that it cannot find matching images for a certain amount of time. This typically has the reason that the robot is navigating outside the mapped area or that the robot has been “kidnapped and teleported” to a different location the map. In our setup, kidnapping is equivalent to the situation that robot has deviated from a previously taken path and now moves on a different subsequence of the reference data.

The first experiment is designed to show how our approach can deal with situations, in which the robot is navigating outside the mapped area (reference sequence). This means that there are no corresponding images with respect to the reference sequence. An example for that can be seen in Fig. 6 (left) marked by the large rectangles. Fig. 6 (middle) illustrates that our approach localizes the robot in such a situation (as can be seen from the red (matched) and blue (unmatched) pixels). In contrast, the standard lazy DA approach [21] finds the matches only partially as it searches in the wrong area of the graph, see Fig. 6 (right).

We further tested our system on the publicly available VPRICE Challenge dataset to illustrate the handling of the kidnapped robot problem. We depict here the part that contains images where a person is moving with a hand-held camera during the day in the reference sequence and repeats the same path during the day and at night within the query sequence. Since the image sequences between the day and night runs are appended to each other, this corresponds to the kidnapped robot situation, i.e. the robot was teleported from the current location (end of the sequence) to another place (beginning of the sequence). As Fig. 7 (middle) suggests also in this case we can dramatically improve the matching quality with respect to our previous approach Fig. 7 (right). Furthermore, we evaluated our approach on a more challenging dataset that has multiple revisits of the same places in query as well as reference sequence, see Fig. 8. The left

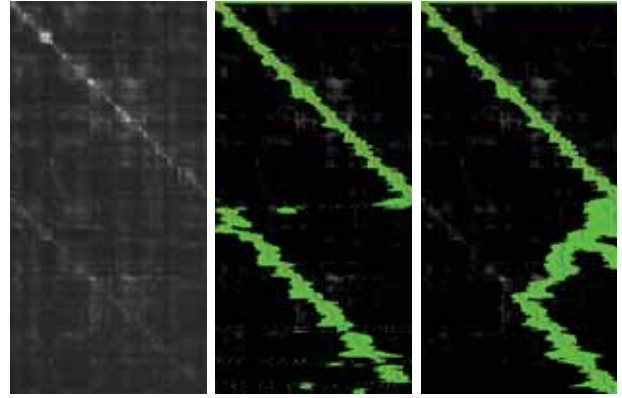


Fig. 7. Example of the trajectory matching from the VPRICE datasets. Here the query trajectory follows the reference trajectory twice, once during the day time (upper matrix part) and once during the night time (lower matrix part). Left: cost matrix. Middle: result of proposed algorithm. Right: Result using previous approach.

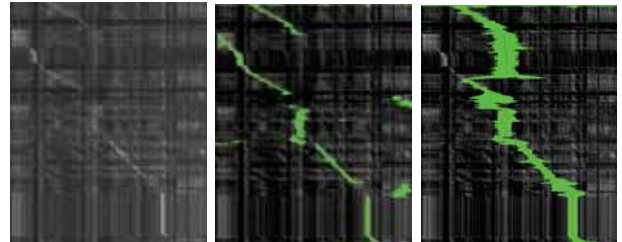


Fig. 8. Matching example from the Freiburg dataset. The trajectory contains partial revisits of the reference sequence as well as detours in the query sequence. Left: cost matrix. Middle: result of proposed algorithm. Right: Result using previous approach.

image of Fig. 9 depicts the precision-recall curve and as can be seen the quality of the result is also better than in our previous approach [21] (here labeled as “RAL’16”) exactly due to the ability to detect loops. Fig. 9 (right) visualizes the results for a dataset with a query loop and relocalization part, as in Fig. 6 and it clearly outperforms the RAL’16 approach.

The next experiment is performed using more challenging shapes of trajectories, recorded in downtown Bonn. In Fig. 10 the query sequence was collected in the morning with a slight rain, whereas the reference in the late evening around a week later. The trajectories only partially overlapped, which results in broken bright patterns in the cost matrix. As can be seen the proposed approach (middle) is able to find the underlying pattern, i.e. find the matches, whereas our previous approach can only perform reliably within the continuous pattern. Fig. 11 shows the results for another pair of trajectories. The query sequence was collected in the early evening, whereas the reference in the late evening. As can be seen the proposed approach finds the underlying pattern as well as ignores the areas, where the query trajectory deviates from the reference one, thus no matching images and minimal expansion are expected. Due to inability of our previous method to handle the loops in the trajectories, without (GPS) priors, it performs poorly on this dataset.

To evaluate the performance of our approach in a more quantitative way, we computed precision recall as well as

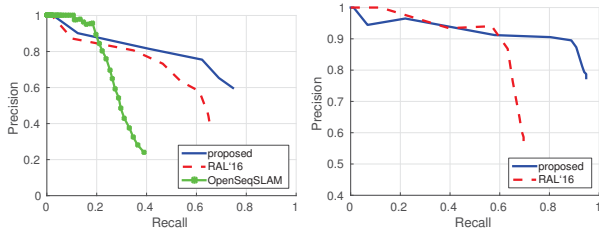


Fig. 9. Precision recall plots for the dataset with multiple loops in query in references sequences (left) and the dataset with a query connected through the similar places in the reference sequence (right).

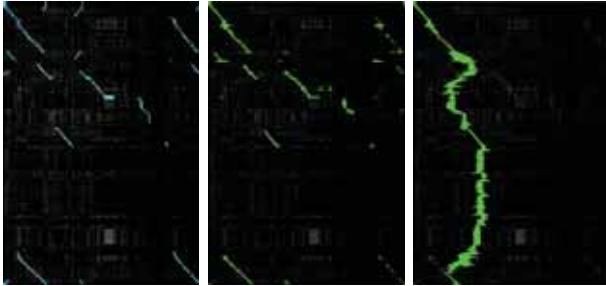


Fig. 10. Matching example of trajectories from Bonn in which both trajectories contain loops. The query also deviates for reference trajectory for a significant amount of time. Left: cost matrix with ground truth overlaid. Middle: proposed approach; Right: Result using previous approach.

F1-score statistics for the given datasets, see in the Tab. I. As it can be seen, by introducing the additional constraints and an efficient relocalization strategy, we are able to increase the number of found image matches (recall) with almost the same precision rate as in our previous paper. This naturally leads to an increase in accuracy in terms of F1-score.

B. Hashing comparison

The second experiment is designed to show that the performance of the proposed relocalization strategy and the Multi-probe locality sensitive hashing is comparable. We also confirm that the proposed hashing algorithm runs faster for the data with very high dimensional features. For the Multi-probe hashing we use the OpenCv implementation. We select the following parameters for all of the experiments: number of trees = 1, key size = 10 and probe level = 2. From our experience selecting a higher number of trees or key size does not improve the performance of the algorithm, but, dramatically increases the computation time. Fig. 12 depicts only small deviations of the precision recall

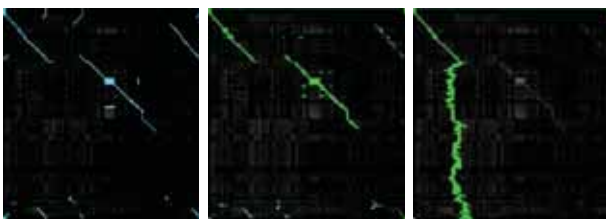


Fig. 11. Additional matching example from Bonn. Left: cost matrix with ground truth overlaid. Middle: proposed approach; Right: Result using previous approach.

TABLE I
QUALITATIVE COMPARISONS BETWEEN THE IMAGE SEQUENCE
MATCHING APPROACH FROM [21] (RAL'16) AND PROPOSED METHOD.
TRAJECTORIES OF VARIOUS SHAPES.

	RAL'16			Proposed		
	exp	pr; re	F1	exp	pr; re	F1
1	7,3%	0.98; 0.35	0.51	6%	0.95; 0.92	0.93
2	12,6%	0.89; 0.63	0.74	11%	0.89; 0.91	0.86
3	10,3%	0.55; 0.64	0.59	4,2%	0.7; 0.71	0.70
4	2,9%	0.99; 0.31	0.48	1,5%	0.95; 0.76	0.84
5	2,7%	0.72; 0.35	0.46	1,8%	0.8; 0.81	0.80

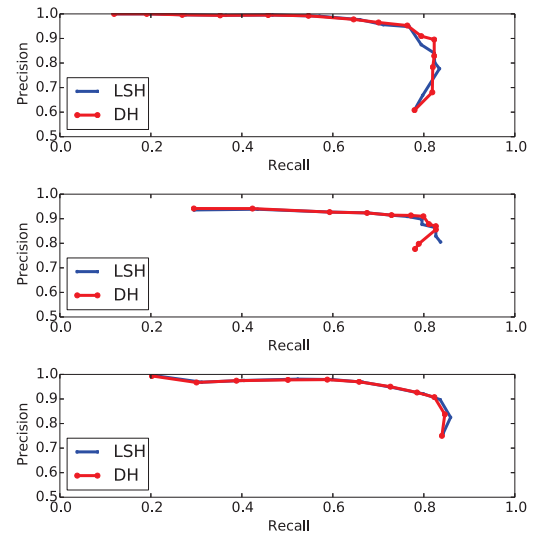


Fig. 12. Precision recall plots for different pairs of trajectories from Bonn dataset. LSH - results for locality sensitive hashing, DH - proposed hashing algorithm (dimension hashing).

curves between the locality sensitive hashing (LSH) and the proposed dimension hashing (DH), which indicates that both hashing algorithm perform equally good on multiple datasets. On the other hand, the run time of the individual hashing strategies differ dramatically. For querying a set of candidates the LSH on average takes 120 ms, whereas DH retrieves the candidates in on average in 600 μ s, which makes the candidates extraction time around 200 times faster. This speedup has a substantial impact on the overall timing.

Given the OverFeat feature vectors, we obtain the following timings. Processing a single image while being localized takes 2 – 3 ms. In contrast, processing a single image while being lost takes 30 – 80 ms using our DH hashing and 150 – 200 ms using LSH. Thus, our approach reduces the runtime for the relocalization by a factor of 2.5-5 in our experiments.

C. Loops in Reference Sequences

The last experiment is designed to show that taking into account the similarity of the places in the reference sequence leads to better localization results. As it can be seen, in Fig. 13 (middle), our proposed approach finds the underlying pattern as oppose to the approach that does not take into account the notion about the place similarity Fig. 13 (right). Again this is an expected results and was the reason for

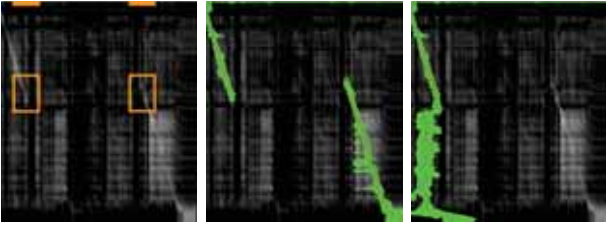


Fig. 13. Example of trajectories, where the reference sequence traverses the same place in the environment twice (marked with orange squares) and deviates in two different directions upon exiting similar area. The query trajectory then also passes the "marked" area and follows one of the directions in the reference sequence. Middle: result of proposed algorithm. Right: Result using previous approach.

implementing this approach. Experiments with other datasets show similar results, but are omitted here in sake of brevity.

V. CONCLUSION

In this paper, we presented a new approach for quickly finding correspondences between a currently observed image stream and a previously recorded image sequence under strong appearance changes. Related to [21], we build a data association graph incrementally and search for a data association sequence using an effective search heuristic. The work proposed here overcomes two key limitations of our previous method. First, we provide an efficient way for re-localizing the robot in situations, in which it got lost, after the robot has left the previously mapped areas and is reentering the known part of the environment or to solve the kidnapped robot problem. Second, our new approach can deal with loops in the reference sequences effectively without additional pose priors, like GPS. We implemented and evaluated our approach on different publicly available datasets. Our evaluations and comparisons show that we can handle the above mentioned situations, which could not be solved with the approach in [21]. We furthermore show through the experiments that our approach runs online, provides an effective image matching, and supports all claims made in this paper.

ACKNOWLEDGMENT

The authors would like to thank Nived Chebrolu for providing hardware support during the experiments and Andres Milioto for hacking the GoPro GPS interface for simplifying the evaluations.

REFERENCES

- [1] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
- [2] R. Arroyo, P.F. Alcantarilla, L.M. Bergasa, and E. Romera. Fusion and binarization of cnn features for robust topological localization across seasons. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 4656–4663, 2016.
- [3] Z. Chen, O. Lam, A. Jacobson, and M. Milford. Convolutional neural network-based place recognition. In *In Proc. of the Australasian Conf. on Robotics and Automation.*, 2014.
- [4] W. Churchill and P. Newman. Experience-based navigation for long-term localisation. *Int. Journal of Robotics Research*, 32(14):1645–1661, sep 2013.
- [5] M. Cummins and P. Newman. Highly scalable appearance-only SLAM - FAB-MAP 2.0. In *Proc. of Robotics: Science and Systems*, 2009.
- [6] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *VLDB*, volume 99, pages 518–529, 1999.
- [7] A.J. Glover, W.P. Maddern, M. Milford, and G.F. Wyeth. FAB-MAP + RatSLAM: Appearance-based slam for multiple times of day. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3507–3512, 2010.
- [8] A. Kendall, M. Grimes, and R. Cipolla. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In *Proc. of the Int. Conf. on Computer Vision (ICCV)*, 2015.
- [9] C. Linegar, W. Churchill, and P. Newman. Work smart, not hard: Recalling relevant experiences for vast-scale but time-constrained localisation. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2015.
- [10] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li. Multi-probe lsh: efficient indexing for high-dimensional similarity search. In *VLDB*, pages 950–961, 2007.
- [11] C. McManus, B. Upcroft, and P. Newman. Learning place-dependant features for long-term vision-based localisation. *Autonomous Robots*, 39(3):363–387, 2015.
- [12] M. Milford. Vision-based place recognition: how low can you go? *Int. Journal of Robotics Research*, 32(7):766–789, 2013.
- [13] M. Milford and G.F. Wyeth. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2012.
- [14] T. Naseer, L. Spinello, W. Burgard, and C. Stachniss. Robust visual robot localization across seasons using network flows. In *Proc. of the AAAI Conf. on Artificial Intelligence*, 2014.
- [15] P. Neubert, S. Schubert, and P. Protzel. Exploiting intra database similarities for selection of place recognition candidates in changing environments. In *Proc. of the CVPR Workshop on Visual Place Recognition in Changing Environments*, 2015.
- [16] P. Neubert, N. Sunderhauf, and P. Protzel. Appearance change prediction for long-term navigation across seasons. In *Proc. of the Europ. Conf. on Mobile Robots (ECMR)*, 2013.
- [17] P. Sermanet, D. Eigen, Z. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *Intl. Conf. on Learning Representations (ICLR)*, 2014.
- [18] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.
- [19] E. Stumm, C. Mei, S. Lacroix, and M. Chli. Location graphs for visual place recognition. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2015.
- [20] O. Vysotska, T. Naseer, L. Spinello, W. Burgard, and C. Stachniss. Efficient and effective matching of image sequences under substantial appearance changes exploiting gps priors. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2015.
- [21] O. Vysotska and C. Stachniss. Lazy data association for image sequences matching under substantial appearance changes. *IEEE Robotics and Automation Letters (RA-L)*, 1(1):1–8, 2016.
- [22] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Advances in neural information processing systems*, pages 1753–1760, 2009.

Experimental study of the precision of a multi-map AMCL-based localization system

Garcia G.¹, Dominguez S.¹, Blosseville J.-M.², Hamon A.¹, Koreki X.¹ and Martinet P.¹

Abstract—Autonomous navigation on the public road network, in particular in urban and semi-urban areas, requires a precise localization system, with a wide coverage and suitable for long distances. Moreover, the system must be adapted to higher speed, as compared to usual indoor mobile robots. Conventional GPS is not precise enough to satisfy the requirements. In addition, GPS suffers from signal fading and multi-path (signal reflections on nearby surfaces), very common in urban environments because of the buildings. This paper shows the methodology and statistical results of performance, over nearly 100 km, of the localization system developed at LS2N based on the classical probabilistic Monte Carlo localization, adapted for multiple maps. The environments under study are urban and suburban roads.

I. INTRODUCTION

Urban environments have different characteristics than other environments like indoor areas or highways. They usually are very dynamic in terms of traffic, require frequent and pronounced steering actions, and are usually more structured because of the surrounding buildings. On the other hand, suburban residential areas are usually structured, as well as quiet regarding traffic. In both scenarios, the average speed is limited by law to 50 km/h, less in some areas. Ring roads are another important urban traffic environment. Their distinctive features are a dynamic traffic, a globally linear structure and lots of vehicles running in the same direction, with higher speed limits than in urban centers.

For autonomous navigation, it is important to be able to self-localize with good precision in all these environments and this is one of the main motivations of this study.

Even though they are built using a 3D LiDAR, our approach uses 2D occupancy maps in order to simplify the processing while still integrating as much surrounding information as possible. 2D maps limit the amount of information to be processed, with respect to 3D maps.

In such a map-based localization system, determining the correct vehicle position is easier if the maps contain only static information. To satisfy this constraint, we take only LiDAR measurements located a minimum height above road surface (see section II). This strategy effectively discards most non stationary objects: cars, vans, motorbikes, people... and significantly contributes to stable localization. For more information, in [1] a comparison of the results using this technique with a Velodyne PLS-16 versus three laser range Sick LMS151 covering 360° is described.

Prior to the localization phase, the maps are built using Simultaneous Localization and Mapping (SLAM). Highly effective SLAM techniques exist, and state-of-the-art SLAM solvers are now available that achieve good accuracy in real-time (e.g. *GMapping* [2] and *Hector SLAM* [3]). We have implemented an extended version of the *GMapping* package of ROS (Robot Operating System). Our algorithm uses as inputs odometry and planar laser scans, as does the *AMCL*. It is a Large Scale 2D SLAM which uses multiple sub-maps [7], with the generated sub-maps being geo-positioned thanks to RTK-GPS (Real Time Kinematic GPS). Each time the system starts building a new sub-map, the previous ones are geo-positioned and saved. A map-manager is in charge of deciding when to store the current map on disk and start building a new one.

During the localization phase, the sub-maps are loaded as required along the path. Adaptive Monte Carlo localization (AMCL) [4], [5], [6]) is applied in the current sub-map to compute the position relative to the map.

It is important to precise that, for positioning the sub-maps at centimetric level, a RTK-GPS Wide Area Differential GPS has been used during the map building phase. A good positioning of the sub-maps ensures good absolute position of the car if the local map localization is precise. The RTK-GPS, in conjunction with odometry, is also used during the localization phase to calculate the instantaneous position and heading of the vehicle used as a reference.

The remainder of this paper is organized as follows. Section II presents the hardware configuration used for the experiments. Section III contains the methodology for building and geo-positioning the sub-maps, while section IV explains the localization method. Sections V and VI respectively present the experiments and the statistical localization results.

II. HARDWARE SETUP

A. Vehicle and sensors

The vehicle used for this study is an electric car Renault Fluence ZE (Figure 1), equipped with the following sensors:

- Wheel tachometers which are read at 50 Hz on the CAN bus of the car. These are used to compute the speed of the vehicle, which is one of the inputs of the odometry generator.
- “Strap-down” Inertial Measurement Unit Xsens MTI 100. It provides the angular speed of its three orthogonal axes at 200 Hz. The vertical angular speed is used to compute the increments of angular rotation of the car, which is the second input of the odometry generator.

¹ These authors are with LS2N, Laboratoire des Sciences du Numérique de Nantes, École Centrale de Nantes, 1 rue de la Noë, 44321 Nantes, France

² J.-M. Blosseville is with Sherpa Engineering
Corresponding author: Gaetan.Garcia@ec-nantes.fr

- RTK-GPS receiver ProFlex 800 (www.spectraprecision.com/eng/proflex-800.html). Provides positions with + 1 cm accuracy in RTK mode. It is used for ground truth generation and sub-map positioning.
- Puck Velodyne VLP 16 LiDAR (<http://velodynelidar.com/vlp-16.html>). Provides range measurements in 16 planes at different pitch angles between -15° and 15° , over 360° in horizontal, with a resolution of 0.25° and a range accuracy of ± 3 cm for a maximum range of 100 m. Used for map building and localization.



Fig. 1. Renault Fluence ZE used for the experiments

B. 360° laser scan generation

The VLP-16 LiDAR sensor (figure2) has been placed above the roof surface.



Fig. 2. The Velodyne VLP-16 on the roof of the Renault Fluence ZE.

Extrinsic calibration of the sensor provides the position of the VLP-16 relative to the car frame. We convert each measurement from the sensor frame into the car frame by simple reference frame transformation (Eq. 1). The raw data provided by the VLP-16 corresponds to 3D measurements. Therefore, in order to obtain a planar scan information, we project the points belonging to the height range [1.8 m - 2.8 m] on a horizontal plane (fig.3). The points chosen are higher than most cars and people, so in this step we remove from the system most of the perturbation due to moving objects, thus increasing the stability of the localization [1]. Moreover, the amount of information to be processed at each

iteration is drastically reduced when projecting from 3D to 2D laser scan, which allows the algorithm to run in real time on a conventional computer. Once the points are transformed into the car frame, the scan measurement can be addressed by an index in the vector of range measurements (Eqs. 2 and 3).

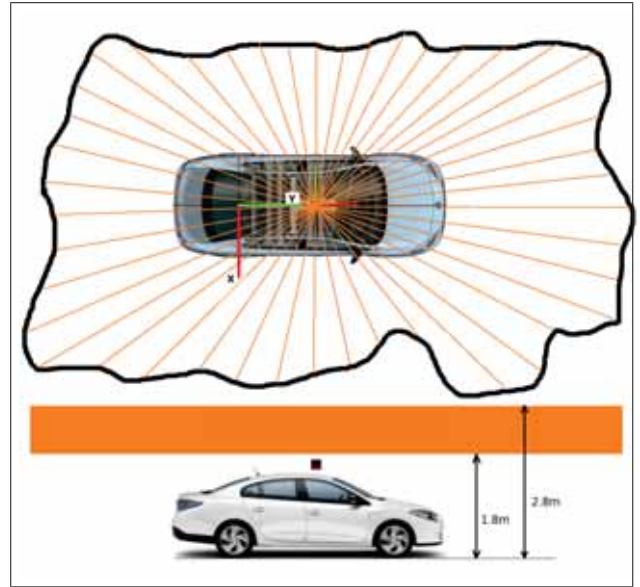


Fig. 3. Sensor configuration for Fluence. A 360° laser scan is obtained from the 3D point-cloud generated by the VLP-16. The bottom of the figure shows the point-cloud range involved in the generation of the planar scan.

The relation between the points in car and sensor frames is:

$${}^cP = {}^cT_s * {}^sP \quad (1)$$

where sP is a point expressed in the sensor frame, cT_s is the constant transformation between the sensor frame and the car frame and ${}^cP = [x, y, z]^T$ is point P expressed in the car frame. From these coordinates, we can extract the bearing angle α of P. Given the angular resolution of the scan, each value of α corresponds to a unique index in the output scan vector, given by equation 2.

$$i(\alpha) = \text{round}\left(\frac{\alpha}{\Delta\alpha}\right) \quad (2)$$

where $\Delta\alpha$ is the angular resolution, in our case 0.5° , and $\alpha \in [0, 360^\circ]$.

The range measured by the sensor for measurement index i is:

$$\text{range}_i = \sqrt{x^2 + y^2} \quad (3)$$

C. Ground truth generation

The ground truth position is computed using a Kalman filter that uses:

- The odometry pose. It is the result of integrating over time the speed of the car and its vertical angular velocity.
- The RTK-GPS position provided directly by the GPS receiver.

The GPS receiver provides localization at 1 Hz and the odometry is generated at 50 Hz, frequency of the speed measurements on the CAN bus of the car. The Kalman filter generates ground truth localization at the same frequency than the odometry (50 Hz). However, the RTK-GPS position is not always available, since it requires a minimum constellation of visible satellites and correct reception of corrections from the GPS ground station. In Section V, aerial views of the paths are provided. For each path, a color code reflects the precision of the GPS data, depending on the mode the system is in.

- GPS-Fix-RTK in green (+/- 1 cm)
- GPS-Float-RTK in blue (+/- 15cm)
- GPS-Float in red (+/- 2-5m)

III. MAP BUILDING METHODOLOGY

The localization method is based on a multi-map system that covers the length of the trip using 2D occupancy grid sub-maps. For building each of those sub-maps we apply 2D SLAM using ROS Gmapping [2] then each sub-map is geo-positioned using an optimization process that matches the local paths inside each sub-map with the global path obtained using the EKF GPS+odometry position. The reason why this method was chosen is because with respect to a local map the precision is quite acceptable for navigation, even though absolute precision might not be that good. On the other hand, the EKF GPS position can suffer from various disturbances, so its precision may locally be insufficient, while remaining globally consistent, meaning that it doesn't diverge from the real position. This method takes advantage of both good qualities: good local sub-map precision and good global GPS consistency for positioning the sub-maps.

The sub-maps are positioned forming a chain in which two consecutive sub-maps are joined at common points called *connection points*. During the building phase, for each sub-map i we simultaneously record, at a predefined interval of distance $L_{threshold}$, the path point in the map $\mathcal{P}_m^i = \{P_{j_m}^i\}_{j=0\dots N_i-1}$, and the path point produced by the EKF for the same instant. We also record the variance of the localization error of the EKF solution $\mathcal{P}_g^i = \{P_{j_g}^i, \sigma_{j_g}^2\}_{j=0\dots N_i-1}$, where N_i is the number of points of the local path i . In this way we obtain two paths, the *map path* and the *global path*, produced respectively by the SLAM and EKF processes (See figure 4). The optimization by relaxation process for geo-positioning the maps is only applied when the sub-map under construction reaches a maximum size, so it takes little computing resources and can be run in real time on a standard computer.

The points of both types of path are computed with respect to a common *global reference system* (see figure 5). The sub-maps are connected at the connection points to ensure continuity between them. We apply an iterative process of energy relaxation to fit the map and the global paths in such a way that the points that have a better EKF global precision intervene with higher weight than those with lower precision. The idea consists in considering the points of the global path like fixed points linked to the corresponding point in the map path by a virtual elastic force proportional

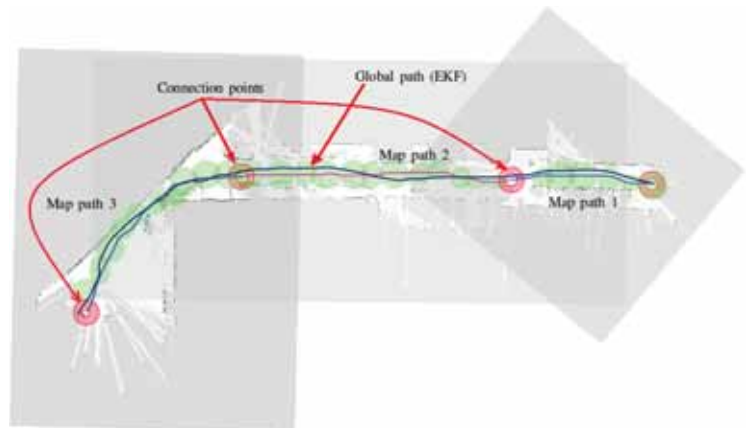


Fig. 4. *Global path and Map path*. Different sub-map paths are represented in different colors as a chain of sub-maps connected at the *connection points*. The green circles represent the localization error of the EKF solution.

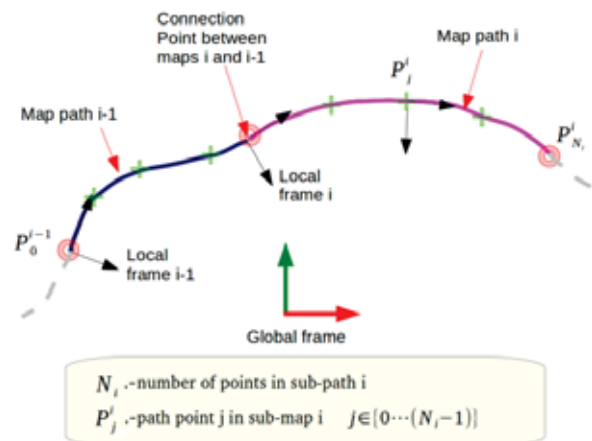


Fig. 5. Global and local reference systems used in the optimization process.

to the distance between both points. The elasticity constant is defined as the inverse of the covariance of the global position error (figure 6). The output of the relaxation process is the sequence of sub-maps poses that produce a minimal energy, so the coherence of the information recorded on the global path and the map path are optimal. In practice, areas where the quality of the GPS is lower (for instance in urban canyons) or where it is inexistent (tunnels) produce global path points in which the uncertainty of the localization is high, so the corresponding forces with the respective map path point are weak or null. In this way, the method can cross areas of poor GPS coverage, provided they are not longer than the size of a sub-map, and yet ensure a good geo-positioning of the sub-maps.

IV. LOCALIZATION METHODOLOGY

The localization uses a probabilistic process called Adaptive Monte Carlo Localization (AMCL)[5], based on a particle filter [4]. The main ideas of such an algorithm are:

- Each particle represents a likely position and orientation (posture) of the vehicle.

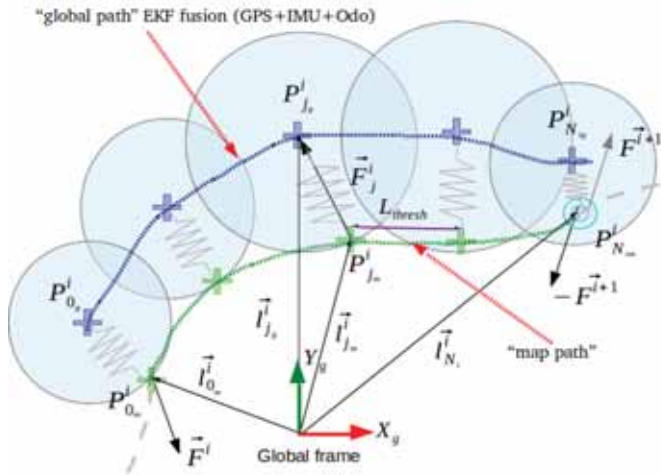


Fig. 6. Every point $P^i_{j_p}$ of the sub-map is attracted towards the corresponding point $P^i_{j_g}$ with a force $F^i_{j_g}$ proportional to the inverse of the covariance of the position error $cov^i_{j_g}$ and the relative position of the points $\vec{l}^i_{j_g}$. From Newton's action-reaction principle, at the connexion points $P^i_{N_m}$ the forces between the sub-maps are opposed.

- The odometry equations are used to predict the posture of the particles at each iteration.
- Periodically, the filter calculates how well the LiDAR measurements fit the local sub-map, for each particle, with the result corresponding to the probability of the particle to hold the true posture of the vehicle.
- Only the particles with a probability higher than a given threshold survive, others are discarded.
- The particle with the highest probability is assumed to correspond to the current posture of the vehicle.
- Whenever the number of remaining particles becomes too low, a resampling around the last best fit is performed, in order to maintain a minimum number of particles.

In our extension of the algorithm, when the vehicle leaves a sub-map, a module called "map manager" orders to load the next sub-map and initializes the local pose of the car on the new sub-map to ensure continuity of the localization process. As shown in figure 4, there is a certain amount of overlap between sub-maps, which guarantees the absence of "gap" in the process.

V. EXPERIMENTAL CONDITIONS

Three scenarios have been considered for testing the localization system:

- Ring road of Nantes: little structured environment, higher speed and continuous traffic.
- Residential area: more structure than in the previous case, lower speed, and little traffic.
- Urban zone: structured zone like in the previous case, low speed and continuous presence of traffic in any direction.

Remarks about ring road scenario:

- On the ring road, the RTK coverage varies significantly from one test to another. In figure 7 a typical example is

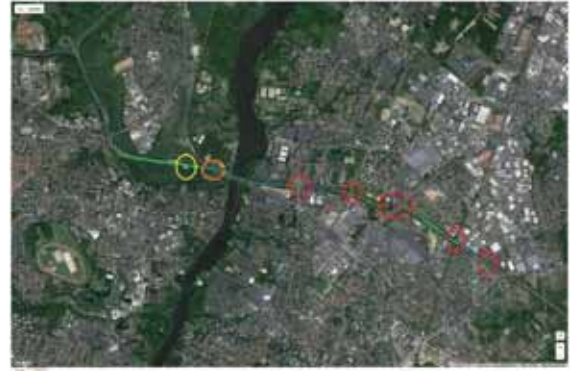


Fig. 7. RTK-GPS coverage - Ring road of Nantes (RTK zones in green)



Fig. 8. RTK-GPS coverage - Residential zone (RTK zones in green)

- shown. The total length in RTK mode is about 2.6 km.
- In the chosen trajectory, the RTK-GPS coverage east of the river Erdre is, in general, quite bad. The precision of the localization in this part has essentially been evaluated on those short portions in which the GPS was in RTK mode. This is because there are multiple bridges crossing over the road that mask the signals coming from the satellites (red circles on the image of figure 7).
 - In the part with good RTK coverage, the infrastructure is almost non-existent, except the borders of the roadway.

Remarks about residential scenario:

- The journey is located in the municipality of "La Chapelle sur Erdre", in the north of Nantes.
- The availability of RTK-GPS is quite repeatable from one test to another. Figure 8 shows a representative example.
- The RTK mode is available practically along the full trip. The total length in RTK mode is about 3.6 km.

Remarks about the urban scenario:

- The journey is located in the north of Nantes.
- The availability of the RTK mode is globally good, but in the southern part it is less stable. Figure 9 shows a representative example. The total length in RTK mode is about 3.4 km.



Fig. 9. RTK-GPS coverage - Urban zone (RTK zones in green)

Road	Lat Error	Long Error
Average[m]	0,07	-0,02
Std. Dev[m]	0,22	0,22
Min[m]	-1,15	-0,62
Max[m]	1,35	0,96
Number of measurements	1533	
Total distance [km]	26,2	
Residential	Lat Error	Long Error
Average[m]	0,03	0,01
Std. Dev[m]	0,13	0,17
Min[m]	-0,56	-0,66
Max[m]	0,56	1,25
Number of measurements	4487	
Total distance [km]	37,7	
Urban	Lat Error	Long Error
Average[m]	0,01	-0,01
Std. Dev[m]	0,17	0,18
Min[m]	-1,06	-0,88
Max[m]	0,90	0,97
Number of measurements	3825	
Total distance [km]	33,9	

Fig. 10. Statistical summary for each scenario

VI. RESULTS AND ANALYSIS

Figure 10 shows a statistical summary of the results of the three scenarios. The minimum and maximum errors, the average error and its standard deviation over nine executions of the path are given.

Figure 11 shows the histograms of the lateral and longitudinal errors for each scenario and gives a visual idea of error distribution.

Figure 12 represents the series of standard deviations of the lateral and longitudinal errors for the individual experiments of each scenario. This figures shows in a visual way the repeatability of the experiments for the same trajectory by comparing the different individual tests.

The total distance analyzed along the ring road of Nantes is about 26 km, near 38 km in residential zone and about 34 km in urban zone. The experiments were conducted at different hours of the day, on different days. The total distance covered was about 100 km.

Here are a few observations we consider important:

- The errors can be considered as centered.

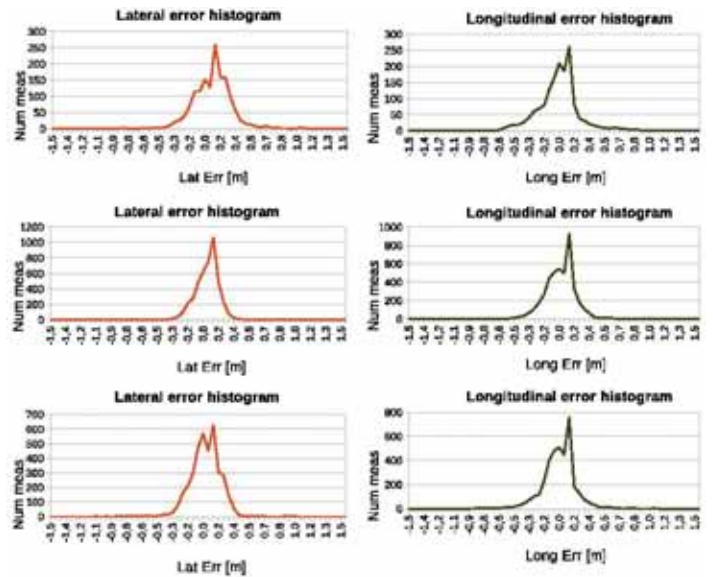


Fig. 11. Lateral and longitudinal error histograms (ring road, residential, urban)

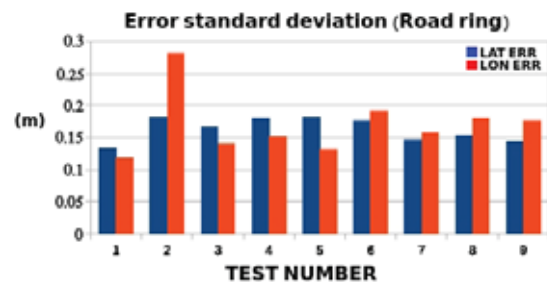
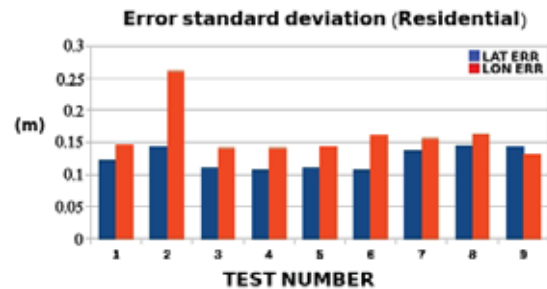
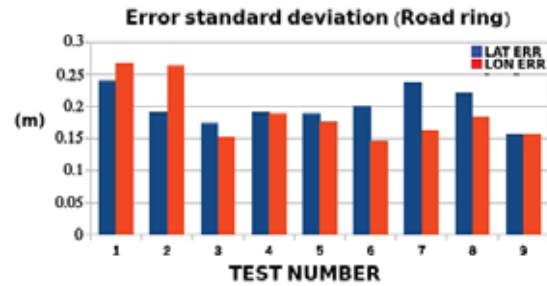


Fig. 12. Repeatability of the standard deviation error along the experiments

- The symmetry of the lateral error histograms is good, a little less for the longitudinal ones. In any case we cannot say there is any anomaly.
- The standard deviations are between 0.13 m and 0.22 m. As expected, the standard deviation of the error is higher on the ring road, with less structured environment than in the other two cases.
- Considering figures 10 and 12, we observe that the poorer quality of the localization on the ring road is duly reflected by the algorithm's estimation of the standard deviation of the error.
- Figure 12 shows that the standard deviation obtained on different executions of a given scenario are repeatable. However, experiment number two (10 March 2017) shows a standard deviation of the error higher than the others. This particular experiment was conducted in foggy conditions. At this point, it would be premature to conclude that the fog is the cause of the result based on a single occurrence of foggy conditions, but the matter certainly deserves further attention.

We have also noted that, in some of the experiments, there are rare situations where the error exceeds the $\pm 3\sigma$ interval which, for a gaussian probability distribution, corresponds to an event of probability 0.27%. The standard deviation of the errors are produced by the AMCL algorithm, based on the dispersion of the particles and is just an approximation. Nevertheless, it fits quite well the real probability distribution, so it can be used as a quality estimator.

At this point one of the main questions that emerges is, which are the sources of error in the overall process? It would be difficult to enumerate all the error sources but we can at least mention the most important ones:

- Time stamping of the RTK-GPS measurements. The GPS measurements used in the experiments were time stamped with the time at which the NMEA message from the GPS reader arrives, minus a fixed delay that was estimated at 35 ms, which is approximately the time taken by the receiver to compute the position and send the result to the computer. Obviously, that is not optimal and should be computed dynamically by using techniques that uses the GPS PPS (Pulse Per Second) signal to obtain the correct time of measurement in computer's time. The error introduced in the results by bad time stamping of the GPS measurements are related directly to the generation of the ground truth, not with the localization algorithm but, as we compare the ground truth and our localization, it affects the results.
- Time stamping of the LiDAR data. The time stamp of the VLP-16 sensor is transferred to the planar 360° scan, and this time stamp is of great importance when performing the "update" phase in the particle filter. A delay in the laser time affects the SLAM process. As in the case of the RTK-GPS, the laser data was stamped with the time of arrival of the data. In order to be precise it should be stamped using the same PPS input as for the GPS.
- Noise in laser measurements. The laser sensor has an intrinsic measurements error of a few centimeters.

Additionally, the angular discretization of the scan produces an uncertainty in the position of the measurement as well. These two effects combined cause the 3D measurement corresponding to a fixed point of the environment not to be totally static, specially if the vehicle moves.

- Projection of points from 3D to 2D. The fact that we project a range of height from the 3D cloud to a single 2D plane can also introduce some error, specially if the environment is not geometrically vertical, that is, for the same angle, all points in different heights do not project onto the same 2D position.
- Odometry error. Even though the odometry was carefully calibrated, the short term prediction of the motion between two measurements is still affected by some noise.

VII. CONCLUSIONS AND FUTURE WORK

Results of an intensive campaign of evaluation of a large scale mapping and localization experiment have been reported. The algorithms have proved to be very robust and their evaluation of the precision of the result reliable. Current accuracy may not yet be sufficient for autonomous navigation in the urban areas, but is getting close to the requirements. We feel that there is room for improvement of the performance with the same set of sensors, in particular through more precise data time stamping, which is the problem we wish to address in the near future.

ACKNOWLEDGMENTS

This work has been performed at the request of the French company Sherpa Engineering S.A. in April 2017. The LS2N is grateful to Sherpa for placing their trust in them and allowing the publication of the results of the study.

REFERENCES

- [1] S. Nobili, S. Dominguez, G. Garcia, M. Philippe, "16 channels Velodyne versus planar LiDARs based perception system for Large Scale 2D-SLAM," in *Proceeding in 7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles, PPNIV'15, IEEE/RS*, September 28, 2015, pp. 131-136, Hamburg, Germany.
- [2] Grisetti, G. and Stachniss, C. and Burgard, W., "Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters", *Robotics, IEEE Transactions on*, February, 2007, pp. 34-46, ISSN 1552-3098.
- [3] 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.
- [4] G. Grisetti and C. Stachniss and W. Burgard, "Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling", in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, April, 2005, pp. 2432-2437, ISSN 1050-4729.
- [5] D. Fox, W. Burgard, F. Dellaert, S. Thrun, "Monte Carlo Localization: Efficient Position Estimation for Mobile Robots", in *Proceedings of the National Conference on Artificial Intelligence*, July, 1999, pp. 343-349, ISBN 0262511061.
- [6] Hantén, Richard and Buck, Sebastian and Otte, Sebastian and Zell, Andreas, "Vector-AMCL: Vector based Adaptive Monte Carlo Localization for Indoor Maps", *emphIntelligent Autonomous Systems (IAS)*, The 14th International Conference on, July, 2016, Shanghai.
- [7] S. Dominguez and B. Khomutenko and G. Garcia and P. Martinet, "An Optimization Technique for Positioning Multiple Maps for Self-Driving Car's Autonomous Navigation", in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, September, 2015, pp.2694-2699, ISSN 2153-0009.

PedLearn: Realtime Pedestrian Tracking, Behavior Learning, and Navigation for Autonomous Vehicles

Aniket Bera and Dinesh Manocha¹

Abstract—We present a real-time tracking algorithm for extracting the trajectory of each pedestrian in a crowd video using a combination of non-linear motion models and learning methods. These motion models are based on new collision-avoidance and local navigation algorithms that provide improved accuracy in dense settings. The resulting tracking algorithm can handle dense crowds with tens of pedestrians at realtime rates (25-30fps). We also give an overview of techniques that combine these motion models with global movement patterns and Bayesian inference to predict the future position of each pedestrian over a long time horizon. The combination of local and global features enables us to accurately predict the trajectory of each pedestrian in a dense crowd at realtime rates. We highlight the performance of the algorithm in real-world crowd videos with medium crowd density.

I. INTRODUCTION

The sensing and computation (*detection, tracking, and prediction*) of human crowd motion has received considerable attention in the literature. It is a well-studied problem that has many applications in surveillance, behavior modeling, activity recognition, disaster prevention, and crowded scene analysis. Despite many recent advances, it is still difficult to accurately track and predict pedestrians in real-world scenarios, especially as the crowd density increases.

The problem of tracking pedestrians and objects has been studied in computer vision and image processing for three decades. However, tracking pedestrians in a crowded scene is regarded as a difficult problem due to intra-pedestrian occlusion (i.e. one pedestrian blocking others) and changes in lighting and pedestrian appearance. Similarly, predicting the trajectory of a pedestrian in a dense environment can also be challenging. In general, pedestrians have varying behaviors and can change their speed to avoid collisions with obstacles and other pedestrians in a scene. In high density or crowded scenarios, the pairwise interactions between the pedestrians tend to increase significantly, affecting their behavior and movement. As a result, the highly dynamic nature of pedestrian movement makes it difficult to estimate their current or future positions. Furthermore, many applications such as surveillance, robot navigation, and autonomous driving need realtime prediction capabilities to estimate the positions of the pedestrians.

We present an algorithm that offers better tracking and prediction using improved motion models and local navigation models and demonstrates improved navigation for autonomous robots and vehicles.

¹All the authors from the Department of Computer Science, University of North Carolina at Chapel Hill, USA



Fig. 1: Pedestrian Prediction and Multi-Robot Navigation: We developed an algorithm to predict the path of each pedestrian (i.e., close to the ground truth) and use the predicted path for collision-free multi-robot navigation. We also classify the behavior of each pedestrian from his or her trajectory and adjust some of the components of social constraints (e.g., personal space, entitativity behavior, etc.) accordingly.

II. RELATED WORK

In this section, we give a brief overview of prior work on motion models and pedestrian path prediction.

A. Motion Models

There is an extensive body of work in robotics, multi-agent systems, crowd simulation, and computer vision on modeling pedestrian motion in crowded environments. These models can be broadly classified into the following categories: potential-based models, which model virtual agents in a crowd as particles with potentials and forces [20]; boid-like approaches, which create simple rules to steer agents [35]; geometric optimization models, which compute collision-free velocities [40]; and field-based methods, which generate fields based on continuum theory [39]. Many of these models have been used for offline and online pedestrian tracking and

trajectory computation. Among these approaches, velocity-based motion models [22], [23], [41], [40], [33] have been successfully applied to the simulation and analysis of crowd behaviors and to multi-robot coordination [37]. Velocity-based models have also been shown to have efficient implementations that closely match real human paths [18].

B. Pedestrian-Tracking with Motion Models

Prior work in pedestrian tracking [10], [25] attempts to improve tracking accuracy by making simple assumptions about pedestrian movement, such as constant velocity and constant acceleration. More recently, long-term motion models and pairwise interaction rules have been combined with tracking to improve the accuracy. Bruce et al. [9] and Gong et al. [16] first estimate pedestrians' destinations and then predict their motions along the path towards the estimated goal positions. Liao et al. [28] extract a Voronoi graph from the environment and predict people's motion along the edges. Many techniques have been proposed for short-term prediction using motion models. Lubner et al. [29] apply Helbing's social force model to track people using a Kalman filter based tracker. Mehran et al. [30] also apply the social force model to detect people's abnormal behaviors from videos. Pellegrini et al. [32] use an energy function to build up a goal-directed short-term collision-avoidance motion model. Bera et al. [4], [5], [3] use reciprocal velocity obstacles and hybrid motion models to improve the accuracy of pedestrian tracking.

C. Path Prediction and Robot Navigation

There has been considerable work on predicting pedestrian trajectories in robotics and computer vision. Most of this work relies on local pedestrian interactions, crowd flows, motion models, Kalman filters, particle filters, and their variants [38], [2]. However, current methods are limited to sparse settings with only a few pedestrians. Our goal is to develop accurate methods for dense settings that consist of pedestrians as well as moving robots. There has been some recent work on using hybrid approaches, combining local and global pedestrian features. In practice, each of these methods only capture some interactions and movements; they are unable to capture the overall pedestrian behavior and hence fail in many situations.

Understanding the behavior of pedestrians within a crowd, which can be as simple as a person walking towards a destination, involves several complex human-centric decisions such as which route to take and the various ways to avoid collision with other pedestrians and obstacles. As a result, different pedestrians will accomplish the same goal in different manners.

Robots navigating in complex, noisy, and dynamic environments have prompted the development of other forms of trajectory prediction algorithms. Fulgenzi et al. [15] use a probabilistic velocity-obstacle approach combined with

the dynamic occupancy grid; this method assumes constant linear velocity motion of the obstacles. DuToit et al. [11] present a robot planning framework that takes into account pedestrians' anticipated future location information to reduce the uncertainty of the predicted belief states. Other techniques use potential-based approaches for robot path planning in dynamic environments [34]. Some methods learn the trajectories from collected data. Ziebart et al. [43] use pedestrian trajectories collected in the environment for prediction using Hidden Markov Models. Bennewitz et al. [2] apply Expectation Maximization clustering to learn typical motion patterns from pedestrian trajectories before using Hidden Markov Models to predict future pedestrian motion. Henry et al. [21] use reinforced learning from example traces, estimating pedestrian density and flow with a Gaussian process. Kretzschmar et al. [26] consider pedestrian trajectories as a mixture probability distribution of a discrete as well as a continuous distribution, and then use Hamiltonian Markov chain Monte Carlo sampling for prediction. Kuderer et al. [27] use maximum entropy based learning to learn pedestrian trajectories and use a hierarchical optimization scheme to predict future trajectories. Many of these methods involve a priori learning, and may not work in new or unknown environments.

Trautman et al. [38] have developed a probabilistic predictive model of cooperative collision avoidance and goal-oriented behavior for robot navigation in dense crowds. Guzzi et al. [19] present a distributed method for multi-robot human-like local navigation. Variations of Bayesian filters for pedestrian path prediction have been studied in [36], [31]. Some of these methods are not suitable for real-time applications or may not work well for dense crowds.

III. PEDESTRIAN TRACKING

We use the term *pedestrian* to refer to independent individuals or agents in a crowd. We use the notion of *state* to specify the trajectory characteristics of each pedestrian. We assume that the output of the tracker corresponds to discrete 2D positions. Therefore, our state vector, represented using the symbol $\mathbf{x} \in \mathbb{R}^6$, consists of components that describe the pedestrian's movements on a 2D plane:

$$\mathbf{x} = [\mathbf{p} \ \mathbf{v}^c \ \mathbf{v}^{pref}]^T, \quad (1)$$

where \mathbf{p} is the pedestrian's position, \mathbf{v}^c is its current velocity, and \mathbf{v}^{pref} is the preferred velocity on a 2D plane. The preferred velocity corresponds to the predicted velocity that a pedestrian would take to achieve its intermediate goal if there were no other pedestrians or obstacles in the scene. We use the symbol \mathbf{S} to denote the current state of the environment, which corresponds to the state of all other pedestrians and the current position of the obstacles in the scene. The state of the crowd, which consists of individual pedestrians, is a union of the set of each pedestrian's state $\mathbf{X} = \bigcup_i \mathbf{x}_i$, where subscript i denotes the i^{th} pedestrian.

The trajectories extracted from a real-world video tend to be noisy and may have incomplete tracks [13]; thus, we use Bayesian-inference technique to compensate for any errors and to compute the state of each pedestrian [24]. At each time step, the observation of a pedestrian computed by a tracking algorithm corresponds to the position of each pedestrian on the 2D plane, denoted as $\mathbf{z}^t \in \mathbb{R}^2$. The observation function $h(\cdot)$ provides \mathbf{z}^t of each pedestrian's true state $\hat{\mathbf{x}}^t$ with sensor error $\mathbf{r} \in \mathbb{R}^2$, which is assumed to follow a zero-mean Gaussian distribution with covariance Σ_r :

$$\mathbf{z}^t = h(\hat{\mathbf{x}}^t) + \mathbf{r}, \mathbf{r} \sim N(0, \Sigma_r). \quad (2)$$

$h(\cdot)$ is the tracking sensor output.

We use the notion of a state-transition model $f(\cdot)$ which is an approximation of true real-world pedestrian dynamics with prediction error $\mathbf{q} \in \mathbb{R}^6$, which is represented as a zero-mean Gaussian distribution with covariance Σ_q :

$$\mathbf{x}^{t+1} = f(\mathbf{x}^t) + \mathbf{q}, \mathbf{q} \sim N(0, \Sigma_q). \quad (3)$$

We can use any local navigation algorithm or motion model for function $f(\cdot)$, which computes the local collision-free paths for the pedestrians in the scene.

We use particle filters as the underlying tracker approach. The particle filter is a parametric method that solves non-Gaussian and non-linear state estimation problems [1]. Particle filters are frequently used in object tracking, since they can recover from lost tracks and occlusions. The particle tracker's tracking uncertainty is represented in a Markovian manner by only considering information from present and past frames. For more details we redirect the reader to [8].

IV. PEDESTRIAN PATH PREDICTION USING BAYESIAN LEARNING

We use Ensemble Kalman Filter (EnKF) and Expectation Maximization (EM) with the observation model $h(\cdot)$ and the state transition model $f(\cdot)$ to estimate the most likely state \mathbf{x} of each pedestrian. EnKF uses an ensemble of discrete samples assumed to follow a Gaussian distribution to represent the distribution of the potential states. EnKF is able to provide state estimation for a non-linear state-transition model. During the prediction step, EnKF predicts the next state based on the transition model and Σ_q . When a new observation is available, Σ_q is updated based on the difference between the observation and the prediction, which is used to compute the state of the pedestrian. In addition, we run the EM step to compute the covariance matrix Σ_q to maximize the likelihood of the state estimation.

Next, we compute clusters of these movement features and the entry point for each pedestrian. That provides information about the pedestrians' global level trajectory behaviors. At every w steps, we compute new behavior features for each agent in the scene. We group similar features and find K most common behavior patterns, which we call *movement flow clusters*. We use multivariate Gaussian mixture model

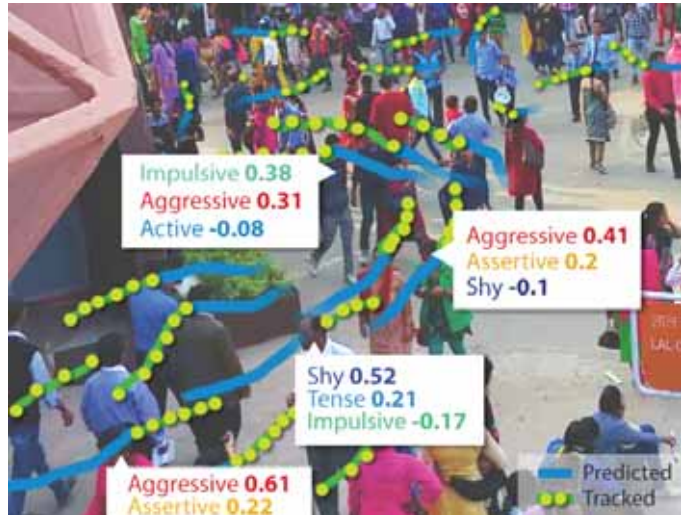


Fig. 2: Behavior Learning: Our goal is to identify the personality of each pedestrian in a robot's field of vision based on his or her trajectory and learning methods. Once we can classify the personalities of nearby pedestrians, the multi-agent planning algorithm will take the personalities of the pedestrians into account and ensure that the resulting robot behaviors satisfy the social constraints.

to learn the time-varying distribution of entry points, which will be used as the initial position \mathbf{x}^0 for a newly added pedestrian in a data-driven crowd simulation.

A. Pedestrian Behavior Classification

To track or predict the motion of the pedestrians, the robots should be able to closely estimate pedestrians' behavior and thereby adjust their movement by accounting for social constraints. While there are many factors that govern people's overall behaviors, we focus on capturing these behavior variations that occur due to pedestrians' inherent personalities and the environment.

To automatically classify every pedestrian's behavior based on his or her trajectory and motion model, we developed an approach that can classify pedestrian behaviors using Personality Trait Theory from psychology and the Eysenck 3-factor model [14]. This model identifies three major factors that are used to characterize the personality: *Psychoticism*, *Extraversion*, and *Neuroticism* (commonly referred to as **PEN**). Each individual personality is identified based on how they exhibit each of these three traits. These individual behaviors can be classified into six weighted behavior classes: aggressive, assertive, shy, active, tense, and impulsive. There is prior work on mapping the personality factors and different motion models used for multi-agent navigation [12], [17].

In our formulation, each pedestrian can be described by a weighted combination of different personality traits based on his or her movement pattern. We use learning methods and a precomputed database on pedestrian behaviors classified

using Personality Trait Theory. During online planning and navigation, we compute pedestrian state information using tracking and behavior learning methods. In this approach, the personality classification is reduced to performing a high-dimensional nearest neighbor query for behavior classification and dynamic computation of the motion parameters. Based on this classification, the pedestrians are classified into different behavior models such as shy, aggressive, active, etc., as shown in Fig. 4. Moreover, we adjust the social constraints parameters corresponding to entitativity and culture elements based on different personalities.

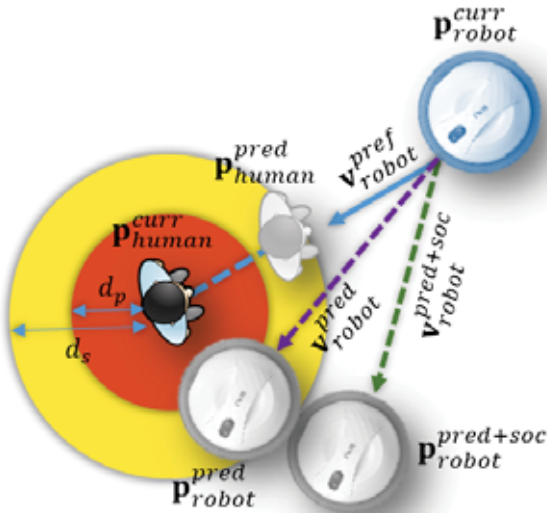


Fig. 3: Our robot navigation algorithm satisfies the proxemic distance constraints, including personal space (red) and social space (yellow). The trajectory computed by our algorithm (green trajectory) does not intrude on the personal/social space of the pedestrian, whereas a robot that fails to account for the social constraints (purple trajectory) may cause pedestrians discomfort.

In our formulation, we use the well-known Personality Trait Theory from psychology and use the Eysenck 3-factor model [14] to classify such behaviors. We adopt a data-driven approach and derive a mapping between simulation parameters and perceived agent behaviors based on the results of the perceptual study. For more details we refer the reader to [6].

As an initial proof-of-concept, we demonstrate our behavior learning pipeline on the *PBS* video stream from the 2017 Presidential Inauguration ceremony at Washington, DC, USA. We extract a representative sample of the crowd by selecting 130 pedestrians from a camera angle and learn their behaviors. As part of our crowd prediction, we changed the number of pedestrians in the scenario (e.g., 1 million pedestrians), and estimated the distribution and shape of the resulting crowd at the National Mall. The resulting crowd of

1M pedestrians has the same behavior classification as the original 130 representative pedestrians. Readers can find the video here - <https://www.youtube.com/watch?v=Hyy122qpc9I>

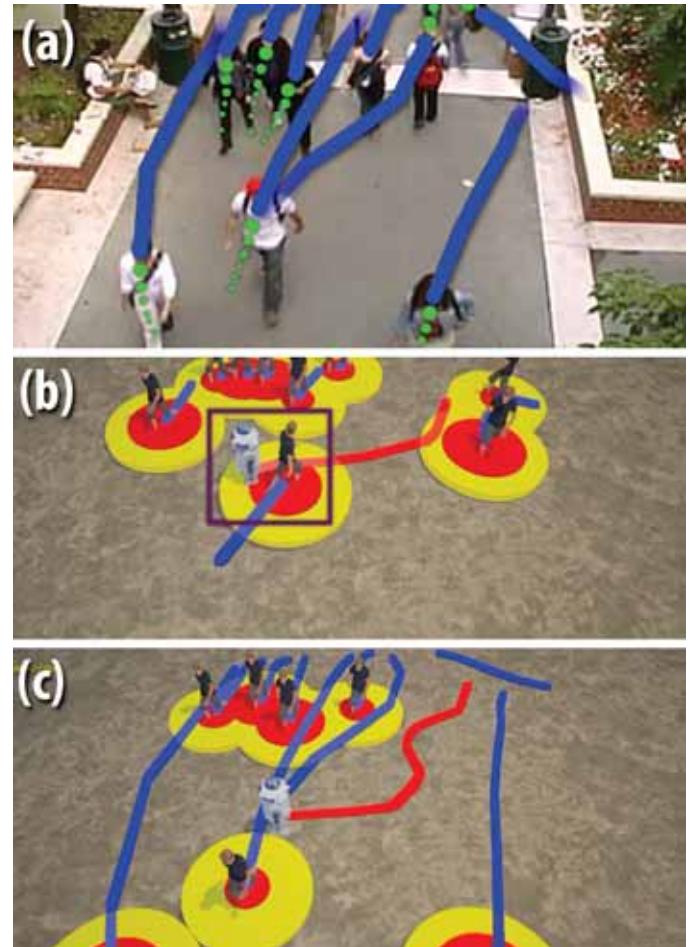


Fig. 4: Improved navigation based on social and physical constraints in a simulated environment: (a) shows a real-world crowd video and the extracted pedestrian trajectories in blue. The green markers are the predicted positions of each pedestrian that will be computed by our algorithm for collision-free navigation. The red and yellow circles around each pedestrian in (b) and (c) would highlight their personal and social spaces respectively, computed using the social constraints described in Sections 2 and 3. The use of such psychological and social constraints in (c) can result in better trajectories over the existing navigation method that takes into account only the physical constraints with no social consideration, as shown in (b).

B. Socially-Aware Multi-Robot Multi-Person Navigation

The multi-agent planning and interaction algorithms described in the earlier sub-sections will be integrated and evaluated in multi-human environments. It is necessary for multiple robots working around human pedestrians to be

able to successfully navigate to perform their tasks without colliding with pedestrians or other obstacles in the scene. One of our goals is to ensure that the resulting set of robots has fewer entitative behaviors. We also hope to account for cultural elements.

We developed a new navigation algorithm for multi-robot multi-person scenarios. Furthermore, the level of interaction may vary depending on passive, active or dynamic surveillance. Our approach combines pedestrian prediction and behavior learning methods with multi-agent navigation schemes that account for both social and physical constraints as well as social saliency.

We use the distances computed using a person's psychological constraints and social saliency to enable socially-aware collision-free robot navigation through a crowd of pedestrians. Our navigation method is based on Generalized Velocity Obstacles (GVO) [42], which uses a combination of local and global methods. The global metric is based on a road-map of the environment. The local method computes a new velocity for the robot and takes these distances into account. Moreover, we also consider the dynamic constraints of the robot in this formulation.

Figure 3 illustrates how a robot avoids an approaching pedestrian based on these distances. At a given time instance, the pedestrian is located at $\mathbf{p}_{human}^{curr}$, and has two proxemic distances: a personal distance of d_p (red) and a social distance of d_p (yellow). At the same time instant, the robot is located at $\mathbf{p}_{robot}^{curr}$ and has a preferred velocity $\mathbf{v}_{robot}^{pref}$ which is computed based on global navigation module. This is the velocity that it would have for navigating to its goal position, in the absence of any static or dynamic obstacles. The robot predicts that during the next time frame, the pedestrian will move to the position $\mathbf{p}_{human}^{pred}$ using the path prediction described in Section III(D), and it computes its new velocity to avoid a collision with the pedestrian. However, this path prediction is not sufficient for socially-aware navigation since the robot fails to take into account the pedestrian's proxemic distances. Based on these distances, the robot alters its goal position to $\mathbf{p}_{robot}^{pred+soc}$ and its velocity to $\mathbf{v}_{robot}^{pred+soc}$ to accommodate both social and psychological constraints. Notice that the velocity $\mathbf{v}_{robot}^{pred}$ causes the robot to intrude on the pedestrian's personal distance, shown by the red circle centered around the pedestrian, whereas the updated velocity $\mathbf{p}_{robot}^{pred+soc}$ successfully accounts for the pedestrian's personal distance as well as its social distance. For more experimental details and results, we refer the reader to [7].

V. CONCLUSION

We present an overview of a collection of interactive approaches for computing trajectory-level behavior features from crowd videos and demonstrate their application to surveillance and training applications. The approaches are general, can handle moderately dense crowd videos, and can compute and predict the trajectory (past, present and future)

for each agent during each time-step. A key benefit of these approaches is that they can capture dynamically changing movement behaviors of pedestrians and therefore can be used for dynamic or local behavior analysis.

Limitations: The performance and accuracy of this algorithm are governed by the tracking algorithm, which can be noisy, sparse, or may lose tracks. Furthermore, current realtime methods may not work well in very dense crowd videos, e.g., those with thousands of agents in a single frame.

VI. ACKNOWLEDGEMENTS

This work was supported by National Science Foundation award 1305286, ARO contract W911NF-16-1-0085, and a grant from the Boeing company.

REFERENCES

- [1] M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *Signal Processing, IEEE Transactions on*, 2002.
- [2] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun. Learning motion patterns of people for compliant robot motion. *The International Journal of Robotics Research*, 24(1):31–48, 2005.
- [3] Aniket Bera, Nico Galoppo, Dillon Sharlet, Adam Lake, and Dinesh Manocha. Adapt: real-time adaptive pedestrian tracking for crowded scenes. In *ICRA*, 2014.
- [4] Aniket Bera and Dinesh Manocha. Realtime multilevel crowd tracking using reciprocal velocity obstacles. In *ICPR*, 2014.
- [5] Aniket Bera and Dinesh Manocha. REACH: Realtime crowd tracking using a hybrid motion model. *ICRA*, 2015.
- [6] Aniket Bera, Tanmay Randhavane, and Dinesh Manocha. Aggressive, tense, or shy? identifying personality traits from crowd videos. In *Proceedings of the International Joint Conference on Artificial Intelligence (To appear)*, 2017.
- [7] Aniket Bera, Tanmay Randhavane, Rohan Prinja, and Dinesh Manocha. Sociosense: Robot navigation amongst pedestrians with social and psychological constraints. 2017.
- [8] Aniket Bera, David Wolinski, Julien Pettré, and Dinesh Manocha. Real-time crowd tracking using parameter optimized mixture of motion models. *arXiv preprint arXiv:1409.4481*, 2014.
- [9] A. Bruce and G. Gordon. Better motion prediction for people-tracking. In *Proc. of the International Conference on Robotics and Automation (ICRA), New Orleans, USA*, 2004.
- [10] J. Cui, H. Zha, H. Zhao, and R. Shibusaki. Tracking multiple people using laser and vision. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2116–2121. IEEE, 2005.
- [11] N.E. Du Toit and J.W. Burdick. Robotic motion planning in dynamic, cluttered, uncertain environments. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 966–973, 2010.
- [12] F. Durupinar et al. How the ocean personality model affects the perception of crowds. *Computer Graphics and Applications, IEEE*, 31(3):22–31, may-june 2011.
- [13] Markus Enzweiler and Darius M Gavrilă. Monocular pedestrian detection: Survey and experiments. *IEEE PAMI*, 2009.
- [14] H.J. Eysenck and M.W. Eysenck. *Personality and individual differences: A natural science approach*. Plenum Press New York, 1985.

- [15] C. Fulgenzi, A. Spalanzani, and C. Laugier. Dynamic obstacle avoidance in uncertain environment combining pvos and occupancy grid. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 1610–1616, 2007.
- [16] H. Gong, J. Sim, M. Likhachev, and J. Shi. Multi-hypothesis motion planning for visual object tracking. 2011.
- [17] S. Guy, S. Kim, M. Lin, and D. Manocha. Simulating heterogeneous crowd behaviors using personality trait theory. *Proc. of Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, pages 43–52, 2011.
- [18] S. J. Guy, J. Chhugani, S. Curtis, P. Dubey, M. Lin, and D. Manocha. PLEdestrans: a least-effort approach to crowd simulation. In *Symp. on Computer Animation*, 2010.
- [19] J. Guzzi, A. Giusti, L.M. Gambardella, G. Theraulaz, and G.A. Di Caro. Human-friendly robot navigation in dynamic environments. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 423–430, May 2013.
- [20] D. Helbing and P. Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282, 1995.
- [21] P. Henry, C. Vollmer, B. Ferris, and D. Fox. Learning to navigate through crowded environments. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 981–986, 2010.
- [22] I. Karamouzas, P. Heil, P. van Beek, and M. Overmars. A predictive collision avoidance model for pedestrian simulation. *Motion in Games*, pages 41–52, 2009.
- [23] I. Karamouzas and M. Overmars. A velocity-based approach for simulating human collision avoidance. In *Intelligent Virtual Agents*, pages 180–186. Springer, 2010.
- [24] Sujeong Kim, Stephen J Guy, Wenxi Liu, David Wilkie, Rynson WH Lau, Ming C Lin, and Dinesh Manocha. Brvo: Predicting pedestrian trajectories using velocity-space reasoning. *The International Journal of Robotics Research*, page 0278364914555543, 2014.
- [25] L. Kratz and K. Nishino. Tracking pedestrians using local spatio-temporal motion patterns in extremely crowded scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (99):11, 2011.
- [26] Henrik Kretschmar, Markus Kuderer, and Wolfram Burgard. Learning to predict trajectories of cooperatively navigating agents. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 4015–4020. IEEE, 2014.
- [27] Markus Kuderer, Henrik Kretschmar, Christoph Sprunk, and Wolfram Burgard. Feature-based prediction of trajectories for socially compliant navigation. In *Robotics: science and systems*, 2012.
- [28] L. Liao, D. Fox, J. Hightower, H. Kautz, and D. Schulz. Voronoi tracking: Location estimation using sparse and noisy sensor data. In *IROS*, 2003.
- [29] M. Luber, J.A. Stork, G.D. Tipaldi, and K.O. Arras. People tracking with human motion predictions from social forces. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 464–469, 2010.
- [30] R. Mehran, A. Oyama, and M. Shah. Abnormal crowd behavior detection using social force model. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 935–942, 2009.
- [31] Andreas Mogelmoose, Mohan M. Trivedi, and Thomas B. Moeslund. Trajectory analysis and prediction for improved pedestrian safety: Integrated framework and evaluations. In *Intelligent Vehicles Symposium (IV), 2015 IEEE*, pages 330–335, June 2015.
- [32] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *ICCV*, 2009.
- [33] J. Pettré, J. Ondřej, A. H. Olivier, A. Cretual, and S. Donikian. Experiment-based modeling, simulation and validation of interactions between virtual walkers. In *Symp. on Computer Animation*, page 189198, 2009.
- [34] N. Pradhan, T. Burg, and S. Birchfield. Robot crowd navigation using predictive position fields in the potential function framework. In *American Control Conference (ACC), 2011*, pages 4628–4633, 2011.
- [35] C. W. Reynolds. Steering behaviors for autonomous characters. In *Game Developers Conference*. [http://www. red3d. com/cwr/steer/gdc99](http://www.red3d.com/cwr/steer/gdc99), 1999.
- [36] Nicolas Schneider and Dariu M. Gavrilă. Pedestrian path prediction with recursive bayesian filters: A comparative study. In *Pattern Recognition - 35th German Conference, GCPR 2013, Saarbrücken, Germany, September 3-6, 2013. Proceedings*, pages 174–183, 2013.
- [37] J. Snape, J. van den Berg, S.J. Guy, and D. Manocha. The hybrid reciprocal velocity obstacle. *IEEE Transactions on Robotics (TRO)*, 27(4):696–706, 2011.
- [38] P. Trautman, J. Ma, R.M. Murray, and A. Krause. Robot navigation in dense human crowds: the case for cooperation. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 2153–2160, May 2013.
- [39] A. Treuille, S. Cooper, and Z. Popovi. Continuum crowds. In *ACM SIGGRAPH 2006*, pages 1160–1168, 2006.
- [40] J. Van Den Berg, S. Guy, M. Lin, and D. Manocha. Reciprocal n-body collision avoidance. *Robotics Research*, page 319, 2011.
- [41] J. van den Berg, S. Patil, J. Sewall, D. Manocha, and M. Lin. Interactive navigation of individual agents in crowded environments. In *Symp. on Interactive 3D Graphics and Games (I3D 2008)*, 2008.
- [42] David Wilkie, Jur van den Berg, and Dinesh Manocha. Generalized velocity obstacles. In *Proceedings of the 2009 IEEE/RSJ international conference on Intelligent robots and systems, IROS'09*, pages 5573–5578, Piscataway, NJ, USA, 2009. IEEE Press.
- [43] Brian D. Ziebart et al. Planning-based prediction for pedestrians. In *IROS*, 2009.

Safe Navigation in Dynamic, Unknown, Continuous, and Cluttered Environments

Mike D’Arcy, Pooyan Fazli, and Dan Simon

Abstract—We introduce PROBLP, a probabilistic local planner, for safe navigation of an autonomous robot in dynamic, unknown, continuous, and cluttered environments. We combine the proposed reactive planner with an existing global planner and evaluate the hybrid in challenging simulated environments. The experiments show that our method achieves a 77% reduction in collisions over the straight-line local planner we use as a benchmark.

I. INTRODUCTION

Safe navigation through Dynamic, Unknown, Continuous, and Cluttered (DUCC) environments is a crucial ability for autonomous robots in search and rescue, self-driving cars, and servicing tasks. A robot traveling through such environments must make fast decisions to react to moving obstacles, while still finding an efficient path through the partially or fully unknown area.

Path planning approaches can be broadly divided into global and local/reactive methods. Global planners attempt to find a complete path from the robot to the goal. A common way to extend these methods to handle moving obstacles is to add a time dimension to the planning space and modeling obstacles as spacetime volumes. This has the advantage of making it possible to consider the long-term effects of actions but can also result in long planning times in large or complex environments. However, in dynamic environments, standing still for a long time while replanning could result in a collision. In addition, many global approaches expect that a full map of the target area is given *a priori* [1, 6, 15, 17], which is an unrealistic assumption in many scenarios.

The limitations of global planners are often addressed by combining the global planner with a local method. Local or reactive planners only attempt to plan one or more steps in the future, without necessarily finding a complete path to the goal. This results in much faster decision-making for autonomous robots in the vicinity of dynamic obstacles but typically has weak goal-directedness, making it hard to provide completeness or optimality guarantees. Many reactive planners assume deterministic knowledge of obstacles, such as most Velocity Obstacle based approaches [4, 5, 18].

In this paper, we define the safe navigation problem in DUCC environments, with no constraints on the shape or size of the environment or the shape or velocity of the moving obstacles. The proposed local planner, PROBLP, consolidates information about the target and the obstacles detected by

the range sensor to identify a safe path in the environment. We run experiments in which we combine PROBLP with the Dynamic Rapidly-exploring Random Tree (DRRT) [3] algorithm to produce a hybrid capable of navigating through complex environments while still remaining safe around moving obstacles. We show empirically that this combined approach significantly outperforms the DRRT algorithm, which by default uses a straight-line local planner.

II. RELATED WORK

Many existing navigation approaches work only in discrete state spaces. This is the case for D* [16], AD* [11], and also for the SIPP based methods [14, 12]. While it is usually possible to discretize a continuous environment into an occupancy grid [2] or roadmap [9], doing so can require large amounts of memory and processing power, and it destroys some of the information about the environment.

Many of the methods that do work in DUCC environments are based on Rapidly-exploring Random Trees (RRT) [10]. RRT methods use random sampling to build a tree structure over the map. Given enough samples, the RRT will eventually find a path if one exists, so it is called a *probabilistically complete* algorithm. Due to the path being constructed randomly, there is no upper bound on the cost of the path produced in regular RRT, and it can be proven that the probability of the basic RRT algorithm producing an optimal path is zero [7]. This path optimality problem is addressed by RRT* [7], an extension of RRT that will converge to an optimal solution asymptotically as the number of random samples increases. While this improves the speed of path execution, the planning time to find an optimal path can still be quite large, making it unsafe for environments where obstacles may be moving during the planning process. Anytime RRT* [8] was developed to reduce this drawback by quickly finding a suboptimal solution and improving it throughout the execution of the path, but some challenges remain. For both RRT and RRT*, if the path to the goal requires going through a very narrow set of acceptable states, as is the case with a narrow tunnel, the planning time can increase significantly as the probability of sampling one of the acceptable states is low.

In dynamic environments where the initial solution may have to be replanned several times, the potentially high planning time of RRT methods is amplified. A variant of RRT that focuses on efficient replanning with a time dimension is the Multipartite RRT (MP-RRT) algorithm [19], which reuses parts of the previous tree to significantly reduce the required planning time. However, while MP-RRT is

Mike D’Arcy, Pooyan Fazli, and Dan Simon are with the Electrical Engineering and Computer Science Department, Cleveland State University, Cleveland, OH 44115, USA {m.m.darcy,p.fazli,d.j.simon}@csuohio.edu

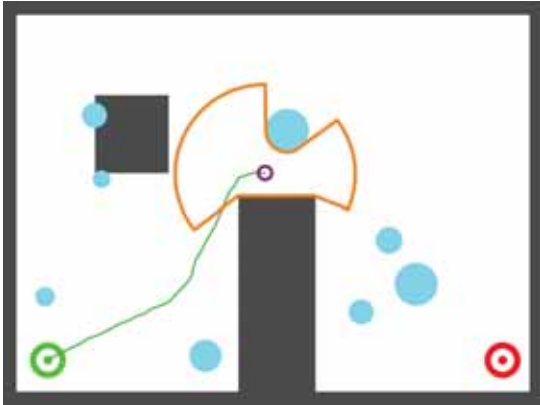


Fig. 1: A sample environment. Static obstacles are dark grey and moving obstacles are light blue. The start point is shown in green on the bottom left of the map, and the target region is shown in red on bottom right. The small purple circle represents the robot, the orange line depicts the robot’s range sensor readings, and the green line shows the robot’s path.

capable of planning in continuous, unknown environments with moving obstacles and has a low replanning time relative to other RRT methods, the planning time still increases with the size and complexity of the environment due to the nearest-neighbor search over all the nodes in the tree. This makes the approach unsafe for some real-world applications, because obstacles may collide with the robot while it is replanning.

A method that does not plan a complete path to the goal was proposed by Petti and Fraichard [13]. Called Partial Motion Planning (PMP), this approach still uses RRT as the exploration strategy but handles real-time constraints by stopping the growth of the tree when planning time runs out and provides provable safety conditions using the Inevitable Collision States (ICS) concept. PMP provides a significant improvement in real-time planning ability over the global RRT methods, but there remains the possibility that PMP will not find any collision-free trajectories, even when one exists. If there is only a narrow path to escape obstacles, PMP’s uniform sampling strategy may not be as effective as an intelligently biased sampling approach.

III. PROBLEM STATEMENT

The problem takes place in a continuous two-dimensional configuration space C with static and dynamic obstacles. Our agent is a holonomic robot that can move at a fixed speed. It has no prior information about the size and shape of the map or the velocity of the moving obstacles in the environment. The robot has a limited-range 360° sensor, with which it can determine the distance to the nearest obstacle in any direction. We additionally assume the robot can perfectly determine its own location and the location of the goal in terms of (x, y) coordinates in the map. The objective of the robot is to navigate from an initial position to the goal, without colliding with any of the static or dynamic obstacles.

Figure 1 shows a sample environment and a robot navigating toward the target point.

IV. PROBABILISTIC LOCAL PLANNER (PROBLP)

We introduce PROBLP, a probabilistic local planner, to enable an autonomous robot to navigate safely in DUCC environments. The algorithm works by sampling a set of candidate trajectories from the robot’s current position and then choosing the best by scoring each candidate on safety and on how much closer it brings the robot to the goal. We define a trajectory as a sequence of waypoints p_0, p_1, \dots, p_n , and we say the robot has *executed* a trajectory when it has visited all of the waypoints in order. Our approach does not sample trajectories entirely at random and instead biases the sampling using a probability distribution to increase the likelihood of choosing favorable candidate trajectories. We will first describe the construction of the distribution, and then how the trajectory sampling is performed.

A. Trajectory Sampling: Probability Distribution

The probability distribution f_Θ that the robot uses to sample trajectories is a distribution over direction angles $\theta \in \Theta = [0, 2\pi)$, so $f_\Theta(\theta)$ is the probability of θ being the best direction for the robot to travel next. We construct f_Θ using two other distributions: a *target distribution*, f_Θ^g , and an *obstacle distribution*, f_Θ^o . Each of these are also distributions over $\theta \in \Theta \in [0, 2\pi)$. Note that while f_Θ is a probability distribution, f_Θ^g and f_Θ^o are only pseudo probability distributions, because their integrals do not necessarily sum to 1.

1) *Target Distribution*: The target distribution f_Θ^g represents the extent to which moving at a direction angle θ would bring the robot closer to the goal. It is defined as a Gaussian distribution:

$$f_\Theta^g(\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\text{angleDiff}(\theta, \alpha)^2}{2\sigma^2}}, \quad (1)$$

where α is the direction angle between the current location of the robot and the goal, and σ is the standard deviation. The choice of σ will affect the behavior of the robot, with small values making it strongly prefer to go directly towards the goal, and large values making it more willing to take a roundabout path to stay safe around obstacles. $\text{angleDiff}(\theta_1, \theta_2)$ returns the absolute value of the smallest difference between two angles. It can be formally written as:

$$\text{angleDiff}(\theta_1, \theta_2) = \min((\theta_1 - \theta_2) \bmod 2\pi, (\theta_2 - \theta_1) \bmod 2\pi)$$

2) *Obstacle Distribution*: The obstacle distribution f_Θ^o represents the extent to which moving in a direction θ would move the robot into free space. It is defined as:

$$f_\Theta^o(\theta) = \frac{\text{range}(\theta)}{\lambda}, \quad (2)$$

where $\text{range}(\theta)$ is the distance between the robot and the nearest obstacle at direction θ , and λ is a scale factor. The

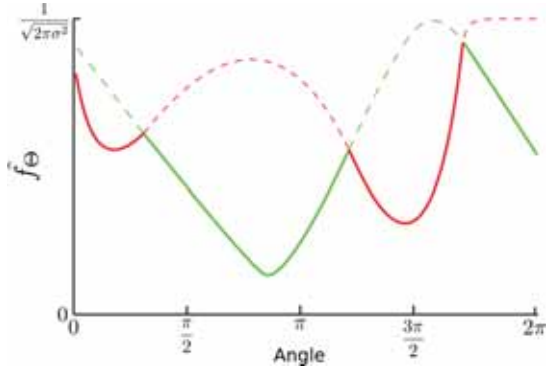


Fig. 2: The red and green lines show obstacle and target distributions respectively. The dashed part is cut away, and the solid line is the resulting final distribution.

specific λ chosen can vary by use case and affects the robot's sensitivity to obstacles, with higher values increasing the sensitivity. In our experiments, we set it to:

$$\lambda = \maxRange \times \sqrt{2\pi\sigma^2} \quad (3)$$

where \maxRange is the maximum range of the robot's range sensor.

To combine the target and obstacle distributions into the final distribution f_Θ , we take the element-wise minimum and then normalize the area under the curve to be 1:

$$\hat{f}_\Theta = \min(f_\Theta^g, f_\Theta^o) \quad (4)$$

$$f_\Theta = \frac{\hat{f}_\Theta}{\int_{\theta=0}^{2\pi} \hat{f}_\Theta} \quad (5)$$

Figure 2 shows an example of the result of combining two distributions.

B. Trajectory Sampling: Selecting the Candidate Trajectories

We use f_Θ to sample a set of candidate trajectories, from which the robot will select its next movement. Each candidate trajectory is constructed as follows: The first waypoint p_0 is set to the current location of the robot. We then sample an angle θ at random from $[0, 2\pi)$, biased by f_Θ . The location of p_1 is determined by predicting the location the robot would have if it started at p_0 and moved in a direction of θ for Δt seconds. Δt is the computation time allowed for the robot. The remaining waypoints up to p_n are computed in a similar way, with the caveat that a new f_Θ must be constructed for each waypoint.

Recall that f_Θ is composed of the target and obstacle distributions. The target distribution is constructed using the direction angle between the current location of the robot and the goal. However, when picking p_2 , we must consider that the robot will start from the previous waypoint p_1 and not from its current position p_0 , so the target distribution should be constructed using the direction angle from p_1 to the goal. For the obstacle distribution, we must consider that

not only the location of the robot will change, but also time will pass as it progresses along the trajectory. Therefore, we predict the future locations of the moving obstacles to construct an accurate distribution. To this end, we define a predictor function that returns the probability of an obstacle obs occupying position pos at time $t \geq t_0$:

$$P(obs|pos, t) = \min\left(4\frac{1 + (t - t_0)}{1 + d}, 1\right), \quad (6)$$

where t_0 is the current time, and $d = \min_{c \in C_{obs}} \|c - pos\|$. C_{obs} is the set of all locations currently occupied by obstacles, which is computed by the robot's range sensor. If no information is available with which to predict future obstacle positions, it can simply be assumed that $\forall t P(obs|pos, t) = P(obs|pos, t_0)$, which is trivial to compute directly from the range sensor. However, better predictions improve safety and reduce the need to replan. Likewise, there is no specific reason to prefer the predictor function in Equation 6 to any other obstacle prediction method, but through empirical testing of many possible predictor functions in the experiments in Section VI we found that this function works well.

Using the obstacle predictor function, we can define a $range_i(\theta)$ function giving the predicted range sensor value for angle θ at time $t_i = t_0 + i\Delta t$, with the robot predicted to be located at p_i . This function can be used in Equation 2 in place of $range(\theta)$ to compute the predicted f_Θ^o for future times. Let C_{obs}^i be the set of all points pos such that $P(obs|pos, t_i) > \gamma$, where γ is some cutoff value in the range $[0, 1]$. Then let $C_{obs}^{i,\theta} \subseteq C_{obs}^i$ be the subset of points that lie on the line segment from p_i to $p_i + (\maxRange)(\hat{u})$, where \hat{u} is a unit vector with direction θ . Then:

$$range_i(\theta) = \min_{c \in C_{obs}^{i,\theta}} \|c - p_i\| \quad (7)$$

The selection of the cutoff value γ can be adjusted depending on the use case, but in general it is most important not to pick a value that is too low. For example, if $\gamma = 0.01$ and there is some uncertainty in future obstacle locations, even points relatively far from obstacles may exceed the small cutoff value, unnecessarily restricting the robot's options. A large value of γ will increase the likelihood of sampling waypoints with low safety scores, increasing the number of samples needed to find a good trajectory, but it is less likely to eliminate desirable trajectories. We found that a cutoff of $\gamma = 0.3$ worked well in our experiments.

In general, when picking waypoint p_i , f_Θ should be computed relative to the location of p_{i-1} and relative to time $t_{i-1} = t_0 + (i-1)\Delta t$. After f_Θ is constructed for this location and time, p_i can be chosen as described previously, by picking a θ from f_Θ and projecting the location the robot would have after starting at p_{i-1} and traveling in direction θ for Δt seconds. The process is repeated up to waypoint p_n , at which point the generated trajectory is ranked by a combination of distance and safety scores.

C. Trajectory Sampling: Selecting the Final Trajectory

Our algorithm is modular with respect to the metrics used for the distance and safety scores, but we calculated them as follows:

$$\text{Distance Score} = f_{\Theta}(\beta) \times \frac{\|p_n - p_0\|}{\sum_{i=0}^{n-1} \|p_{i+1} - p_i\|}, \quad (8)$$

where β is the direction angle between p_0 and p_n , and f_{Θ} is constructed relative to the current location of the robot and to the current time. The second term in Equation 8 is used for smoothing, as the ratio of straight-line distance to total path length is larger for more straight trajectories.

$$\text{Safety Score} = \prod_{i=0}^n [1 - P(\text{obs}|p_i, t_0 + i\Delta t)]. \quad (9)$$

This is equivalent to the probability of the robot being able to follow the trajectory without having any collisions. We set a minimum safety threshold and eliminate all candidate trajectories with a safety score below this threshold. This prevents unsafe trajectories from being considered even if they score highly on distance.

The final score of each trajectory is computed by taking a weighted sum of the distance and safety scores. The safety score is weighted by w and the distance score by $(1 - w)$, where w can be adjusted on a per-use-case basis to make the robot more cautious (high w) or aggressive (low w). The robot picks the trajectory with the highest score and attempts to follow it to completion but continuously updates the estimate of the safety as new information is obtained. If the safety score goes below the minimum threshold, the robot immediately plans a new trajectory to avoid the danger. Otherwise, the robot will not replan until it reaches the terminal waypoint p_n . This makes the robot prefer to continue on its planned smooth path, instead of replanning and changing direction at each decision step.

V. COMBINING PROBLP WITH A GLOBAL PLANNER

PROBLP is weakly goal-directed, so the robot will attempt to move directly towards the goal when it is safe to do so. In an environment with complex arrangements of static obstacles, such as a maze or office building, the reactive planner alone may not be able to reach the goal due to the need to backtrack. Therefore, it is desirable to combine ProbLP with a global planner. The global planner plans a path to the goal and divides it into a series of configurations with straight lines between them. The goal for the reactive planner is then to navigate to the next configuration instead of to the target point. This allows the complex long-term path planning to be handled by the global planner, and the reactive planner can remain simple and fast for computing safe trajectories around moving obstacles.

To this end, we selected the Dynamic Rapidly-exploring Random Tree (DRRT) algorithm [3] as the global planner to combine with PROBLP. DRRT is an extension of the RRT algorithm that improves replanning speed by reusing

parts of the old tree when regrowing it. DRRT generates configurations to guide our planner through the static map and only considers static obstacles in its plan. The avoidance of moving obstacles is entirely handled by PROBLP to avoid the relatively expensive DRRT replanning. Because ProbLP may deviate significantly from the DRRT path for safety reasons, it will ask DRRT to replan if it cannot reach the next configuration within 10 seconds.

DRRT initializes a tree with the root node at the goal, and then grows the tree from the goal to the robot by repeatedly (1) sampling a random point p , (2) finding the node n_{near} in the tree that is closest to the sampled node, and (3) adding a new node n_{new} to the tree at a distance of at most η along the line from p to n_{near} , as long as the line from p to n_{new} does not intersect any obstacles. This is done until one of the nodes added to the tree is within some distance ϵ of the robot's location. The list of intermediate configurations given to the local planner is the list of ancestors of the node closest to the robot.

To work under limited-vision constraints, the DRRT builds its own internal map for collision checking, which starts empty and adds obstacles as they are observed. When it replans, the tree is first checked for collisions, and any branches that intersect an obstacle are pruned. The pruned nodes are inserted into a fixed-size waypoint cache with random replacement. Then the tree is regrown, and the local planner is re-initialized with the resulting intermediate configurations.

For our experiments, $\eta = 3$ m and $\epsilon = 0.7$ m. The size of the waypoint cache is 200 nodes and the random sampling for growing the tree is biased to pick the robot's location 10% of the time, a random point from the waypoint cache 40% of the time, and a point anywhere on the map the remaining 50% of the time. In addition, after the DRRT finds a path, we smooth it by searching for pairs of nodes along the existing path that could be connected by a straight line, skipping the nodes between them.

VI. EXPERIMENTS

We evaluate the proposed algorithm in six simulated environments. Each environment has a different static layout, as shown in Figure 3. All maps are 80 m \times 60 m. Twenty dynamic obstacles are also generated randomly for each trial, ten circular and ten square. The obstacles have random sizes ranging between 0.5 m (radius for circles, edge length for squares) and 3 m. Obstacles move randomly around the map, and collisions between obstacles are not considered (i.e., obstacles can overlap and move through each other).

The robot speed was set to a constant 1.0 m/s for all the experiments, but we tested a variety of obstacle settings. These settings can be divided into two *movement modes* and four *speed modes*. The movement modes are as follows:

- **MM-1:** Each obstacle picks a random point on the map and moves straight towards it. When it reaches the point, it picks a new point to move towards, and this continues indefinitely.

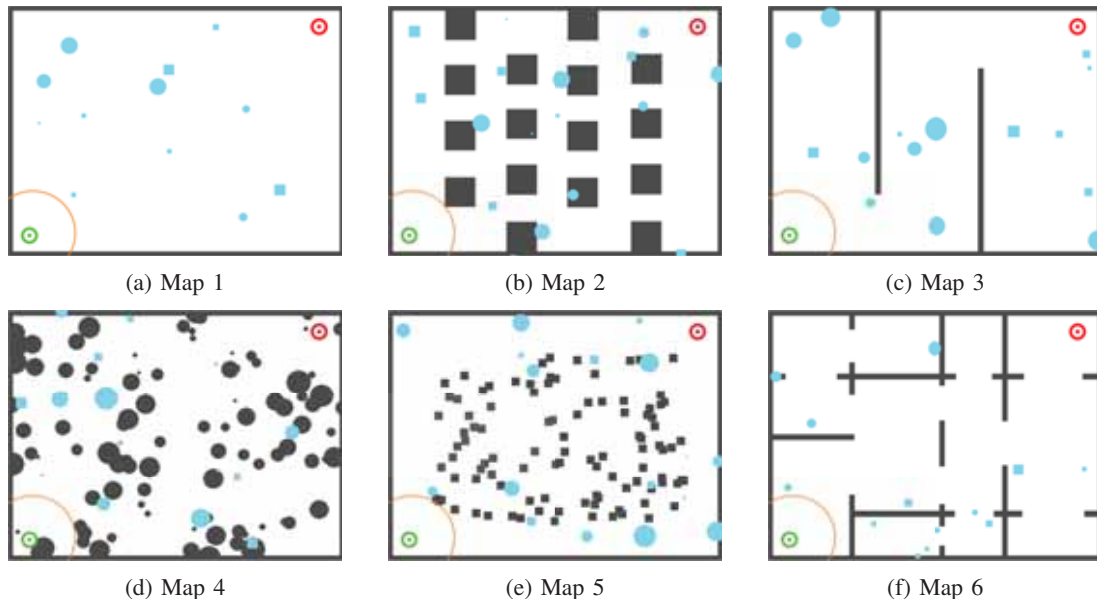


Fig. 3: Maps used for experiments. Static obstacles are dark grey and moving obstacles are light blue. The start point is shown in green on the bottom left of each map, and the target region is shown in red on top right. The orange circle represents the robot's sensor range.

- **MM-2:** Each obstacle moves back and forth between two random points picked at the start of the simulation. The speed modes are as follows:
 - **SP-1:** All obstacles move 0.5 m/s (slower than the robot)
 - **SP-2:** All obstacles move 1.0 m/s (same as the robot)
 - **SP-3:** All obstacles move 1.5 m/s (faster than the robot)
 - **SP-4:** Each obstacle moves randomly between 0.5-1.5 m/s

For each trial, all obstacles use the same movement and speed modes (i.e., for a given trial, there cannot be some obstacles using movement mode 1 and others using mode 2), and we evaluate all combinations of maps, movement modes, and speed modes for a total of $6 \times 2 \times 4 = 48$ experimental setups.

The robot is equipped with a range sensor with an effective range of 10 m, and we assume the sensor can distinguish between static and dynamic obstacles. In practice, this could be achieved with an obstacle-tracking algorithm, and this allows the obstacle predictor to be set to $P(obs|p, t) = 1$ if p is an observed static obstacle, and otherwise the predictor remains the same as defined in Equation 6. The robot speed was set to a constant 1.0 m/s for all the experiments.

For scoring candidate trajectories, we set the safety threshold to 0.1 to immediately filter out any trajectories that the obstacle predictor determined had lower than a 10% chance of being collision-free. When taking the weighted sum to combine the distance and safety scores, we set the weights to 0.5 for both distance and safety. The standard deviation, σ , of the target distribution is set to 100, and trajectories are sampled with a fixed length of two waypoints (i.e., two timesteps into the future are considered). In our testing, we found that these settings produce a good balance of path optimality and safety.

We compare our hybrid approach (DRRT-PROBLP) with a hybrid of DRRT and a straight-line local planner (DRRT-SLLP). The straight-line local planner simply travels in a straight line towards its next configuration. Unlike the DRRT in DRRT-PROBLP, which only handles static obstacles and leaves dynamic obstacle avoidance to PROBLP, the DRRT in DRRT-SLLP does handle dynamic obstacles, because the straight-line planner does not have an obstacle avoidance strategy of its own. The straight-line planner asks DRRT to replan if there are any obstacles obstructing its current path within its range of vision.

Our experiments consist of 100 trials of each of the 48 combinations of speed mode and map. Both the DRRT-PROBLP robot and the DRRT-SLLP robot run simultaneously on the same map with the exactly same obstacles, to reduce error in the comparison. We allow a maximum of 5000 time steps to get to the goal before marking the trial as a failure, and also mark the trial as a failure if the DRRT is ever unable to find a path to the goal within 5000 nodes added to the tree (which could happen if a moving obstacle blocks the only path). Failed trials are not used when calculating the averages in the results.

VII. RESULTS AND DISCUSSION

Of the 4800 trials, 4375 (91%) were successful and 425 (9%) failed. The results of the successful trials are shown in Table I. In all of the speed modes and map combinations, DRRT-PROBLP had fewer collisions than DRRT-SLLP, and the overall averages were 0.27 collisions for DRRT-PROBLP and 1.20 for DRRT-SLLP. Furthermore, even in SP-3, where obstacles are all faster than the robot, DRRT-PROBLP robot was able to receive less than one collision on average for all maps. The overall average path lengths were

TABLE I: Results of experiments. Numbers represent average of collisions

MAP	DRRT-PROBLP					DRRT-SLLP				
	SP-1	SP-2	SP-3	SP-4	Average	SP-1	SP-2	SP-3	SP-4	Average
1	0.01	0.07	0.43	0.13	0.16	0.60	0.91	1.22	0.91	0.90
2	0.09	0.19	0.62	0.19	0.27	0.50	1.19	1.67	1.28	1.15
3	0.05	0.18	0.85	0.25	0.33	1.12	1.71	2.46	1.74	1.75
4	0.03	0.10	0.65	0.24	0.25	0.54	1.06	1.73	1.13	1.10
5	0.13	0.14	0.65	0.26	0.30	0.39	0.90	1.47	1.02	0.94
6	0.07	0.17	0.78	0.31	0.33	0.69	1.48	1.89	1.26	1.33
Average	0.05	0.13	0.68	0.24		0.65	1.15	1.72	1.22	

similar for both robots, being 132.09 for DRRT-PROBLP and 128.85 for DRRT-SLLP. This indicates that DRRT-PROBLP robot did not need to deviate far from the global path to achieve the safety improvement.

We also tracked the number of individual trials in which DRRT-PROBLP received fewer collisions than DRRT-SLLP. DRRT-PROBLP had fewer collisions in 2470 trials, DRRT-SLLP had fewer in 285 trials, and in 1620 trials both had the same number. Interestingly, the DRRT-PROBLP robot reached the goal first in 2035 of the trials, with the DRRT-SLLP robot arriving first in 2240. This again indicates that the safety gains came at almost no detriment to path length.

In SP-1, with the obstacles being slower than the robot, DRRT-PROBLP averaged only 0.05 collisions per trial, and had no collisions in 94% of trials. DRRT-SLLP had a much higher average of 0.65, and only 56% of trials were collision-free. In SP-2, the obstacles moved with the same speed as the robot, and DRRT-PROBLP had 0.13 collisions on average, with 88% of trials being collision-free. In this speed mode, DRRT-SLLP had only 32% collision-free trials. Not surprisingly, SP-3 was the most challenging one, as having the obstacles moving faster than the robot meant there could be situations in which the robot simply did not have time to maneuver around the obstacle before being hit. This had a large effect on the performance of DRRT-SLLP, which averaged 1.72 collisions and had collisions in 80% of the trials. Despite the difficulty of this mode, however, DRRT-PROBLP averaged only 0.65 collisions and had zero collisions in 54% of the trials. Speed mode 4 averaged 0.24 collisions and 81% collision-free trials for DRRT-PROBLP, and 1.22 collisions and 33% collision-free trials for DRRT-SLLP.

Unsurprisingly, both robots had the lowest average number of collisions on Map 1, which has no static obstacles. The averages were 0.16 and 0.90 for DRRT-PROBLP and DRRT-SLLP respectively. The most challenging map for both robots was Map 3, having 0.33 average collisions for DRRT-PROBLP and 1.75 for DRRT-SLLP. The second most challenging map was Map 6, having 0.33 collisions for DRRT-PROBLP and 1.33 for DRRT-SLLP. Maps 2, 4, and 5 performed slightly better, having 0.27, 0.25, and 0.30 average collisions for DRRT-PROBLP, and 1.15, 1.10, and

0.94 average collisions for DRRT-SLLP.

We conjecture that the reason some maps performed better is related to the way the static obstacles are arranged. Maps 2, 4, and 5 have a scattered obstacle layout, with many small static obstacles spread across the map. In contrast, Maps 3 and 6 are more structured, consisting of long, contiguous walls with large free spaces between them. In the scattered-obstacle maps, the static obstacles can be avoided with relatively small adjustments to the trajectory of the robot, making it possible to take a relatively direct path to the goal. In the environments with walls, the robot may have to make a long detour if it finds itself trapped in a dead end, increasing the time spent in the environment and therefore the number of chances to be hit by an obstacle.

We measured the planning times for our implementation of each algorithm and found that DRRT-PROBLP took an average of 117 ms to plan each action, whereas DRRT-SLLP took 149 ms. Intuitively, DRRT-PROBLP should have a higher planning time because it has a more complicated local planner, but this result suggests the opposite. We believe the reason for the superior performance of DRRT-PROBLP is due to the difference in the DRRT for each algorithm. The DRRT in DRRT-PROBLP only needs to consider static obstacles and therefore only needs to replan when new static obstacles are observed, but the DRRT in DRRT-SLLP has to handle dynamic obstacles as well as static obstacles, causing it to undergo the expensive replanning operation much more frequently.

VIII. CONCLUSION AND FUTURE WORK

We have presented PROBLP, a probabilistic local planner, for navigating safely in dynamic, unknown, continuous, and cluttered environments. We showed that this algorithm outperforms the straight-line local planner algorithm we used for comparison.

In future work, we would like to use informed hyperparameter optimization techniques, such as evolutionary algorithms, to tune the algorithm automatically. We would also like to extend our algorithm to multi-robot safe navigation and to non-holonomic robots, and to make it more robust with respect to uncertainties in the range sensor observations and in the positions of the robot and the goal. Finally, because we have so far only developed this approach for use

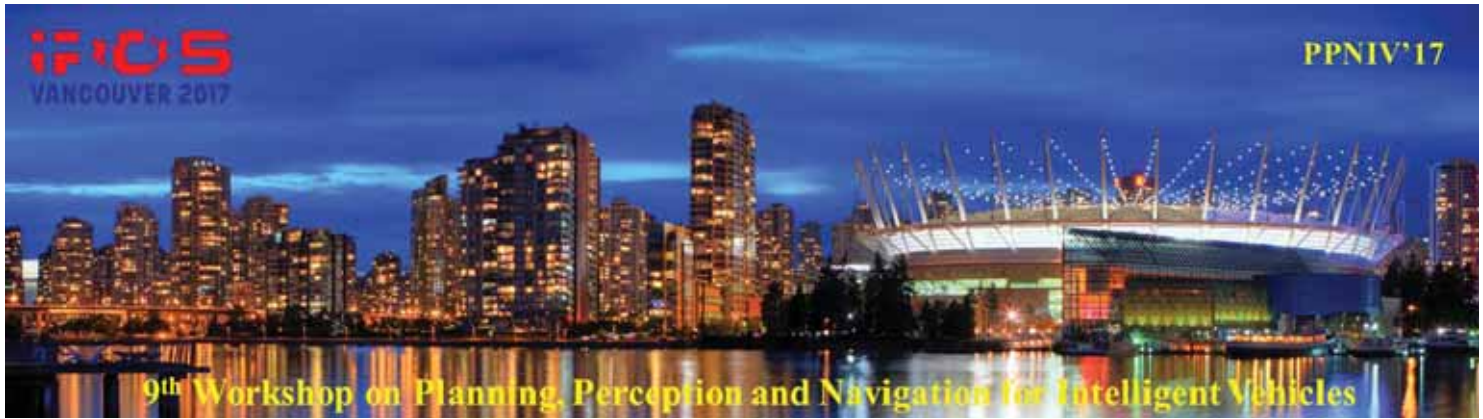
in 2D environments, we would be interested in extending the algorithm to work in higher-dimensional planning spaces.

IX. ACKNOWLEDGMENT

The authors thank Arash Roshanineshat and Utkarsh Patel for their assistance in developing the simulator.

REFERENCES

- [1] J.P. van den Berg and M.H. Overmars. “Roadmap-based motion planning in dynamic environments”. In: *IEEE Transactions on Robotics* 21.5 (2005), pp. 885–897.
- [2] A. Elfes. “Using Occupancy Grids for Mobile Robot Perception and Navigation”. In: *Computer* 22.6 (1989), pp. 46–57.
- [3] Dave Ferguson, Nidhi Kalra, and Anthony Stentz. “Replanning with RRTs”. In: *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*. 2006, pp. 1243–1248.
- [4] P. Fiorini and Z. Shiller. “Motion Planning in Dynamic Environments Using Velocity Obstacles”. In: *The International Journal of Robotics Research* 17.7 (1998), pp. 760–772.
- [5] Oren Gal, Zvi Shiller, and Elon Rimon. “Efficient and safe on-line motion planning in dynamic environments”. In: *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*. 2009, pp. 88–93.
- [6] M. Kallman and Maja Mataric. “Motion planning using dynamic roadmaps”. In: *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*. 2004, pp. 4399–4404.
- [7] Sertac Karaman and Emilio Frazzoli. “Incremental sampling-based algorithms for optimal motion planning”. In: *Proceedings of the Robotics Science and Systems, RSS*. 2010.
- [8] Sertac Karaman et al. “Anytime motion planning using the RRT”. In: *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*. 2011, pp. 1478–1483.
- [9] Lydia E. Kavraki et al. “Probabilistic roadmaps for path planning in high-dimensional configuration spaces”. In: *IEEE Transactions on Robotics and Automation* 12.4 (1996), pp. 566–580.
- [10] Steven M. Lavalle. *Rapidly-Exploring Random Trees: A New Tool for Path Planning*. TR 98-11. Iowa State University, 1998.
- [11] Maxim Likhachev et al. “Anytime Dynamic A*: An Anytime, Replanning Algorithm.” In: *Proceedings of the International Conference on Automated Planning and Scheduling, ICAPS*. 2005, pp. 262–271.
- [12] Venkatraman Narayanan, Mike Phillips, and Maxim Likhachev. “Anytime safe interval path planning for dynamic environments”. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. 2012, pp. 4708–4715.
- [13] Stéphane Petti and Thierry Fraichard. “Safe motion planning in dynamic environments”. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. 2005, pp. 2210–2215.
- [14] Mike Phillips and Maxim Likhachev. “SIPP: Safe interval path planning for dynamic environments”. In: *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*. 2011, pp. 5628–5635.
- [15] Cyrill Stachniss and Wolfram Burgard. “An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments”. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. 2002, pp. 508–513.
- [16] Anthony Stentz. “Optimal and efficient path planning for partially-known environments”. In: *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*. 1994, pp. 3310–3317.
- [17] Jur Van Den Berg, Dave Ferguson, and James Kuffner. “Anytime path planning and replanning in dynamic environments”. In: *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*. 2006, pp. 2366–2371.
- [18] David Wilkie, Jur Van Den Berg, and Dinesh Manocha. “Generalized velocity obstacles”. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. 2009, pp. 5573–5578.
- [19] Matt Zucker, James Kuffner, and Michael Branicky. “Multipartite RRTs for rapid replanning in dynamic environments”. In: *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*. 2007, pp. 1603–1609.



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session V

Perception

- **Keynote speaker: Achim Lithental (Örebro University, Sweden)**
Title: Where the Intermediate is the Big Step Intralogistics with Safe and Scalable Fleets of Autonomously Operating Vehicles in Shared Spaces
- **Title: Fast Image-Based Geometric Change Detection in a 3D Model**
Authors: Emanuel Palazzolo, Cyrill Stachniss
- **Title: Fast Graph-Based Place Recognition**
Authors: Mattia G. Gollub, Renaud Dubé, Hannes Sommer, Igor Gilitschenski and Roland Siegwart



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session V

Keynote speaker: **Achim Lilienthal**
(Örebro University, Sweden)

Where the Intermediate is the Big Step Intralogistics with Safe and Scalable Fleets of Autonomously Operating Vehicles in Shared Spaces

Abstract: Today, intralogistic services have to respond quickly to changing market needs, unforeseeable trends and shorter product life cycles. These drivers pose new demands on intralogistic systems to be highly flexible, rock-solid reliable, self-optimising, quickly deployable and safe, yet efficient in environments shared with humans. In this presentation I will report on ILIAD, an H2020 project that set out to enable the transition to automation of intralogistic services in the food distribution sector, where the challenges mentioned are particularly pressing. ILIAD develops robotic solutions that can integrate with current warehouse facilities, extending the state of the art to achieve self-deploying fleets of heterogeneous robots; life-long self-optimisation; manipulation from a mobile platform; efficient, legible and safe operation in environments shared with humans; and efficient fleet management with formal guarantees. I will present first results obtained regarding tracking and analysing humans; quantifying map quality; learning activity patterns inferred from long-term observations; action and intention recognition for improved human-robot interaction; and integration of task allocation, coordination and motion planning for heterogeneous robot fleets.

Biography: Prof. Achim J. Lilienthal is head of the Mobile Robotics and Olfaction Lab at Örebro University, Sweden. His research interests are rich 3D perception and navigation of autonomous transport robots, mobile robot olfaction, human robot interaction and mathematics education research. Achim Lilienthal obtained his Ph.D. in computer science from Tübingen University, Germany and his M.Sc. in Physics from the University of Konstanz, Germany. The Ph.D. thesis addresses gas distribution mapping and gas source localisation with mobile robots. The M.Sc. thesis is concerned with structure analysis of $(C60)_n^+$ clusters using gas phase ion chromatography.



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems



Where the Intermediate is the Big Step

Intralogistics with Safe and Scalable Fleets of
Autonomously Operating Vehicles in Shared Spaces



Achim J. Lilienthal

contact:

www.mrolab.eu/achim-lilienthal

achim.lilienthal@oru.se



Where the Intermediate is the Big Step

Intralogistics with Safe and Scalable Fleets of
Autonomously Operating Vehicles in Shared Spaces

www.iliad-project.eu

H2020 project funded by the EC

Start: Jan 2017

Duration: 48 months

Funding: 7M€

Partners: 9



Intra-
Logistics with
Integrated
Automatic
Deployment



Where the Intermediate is the Big Step

Intralogistics with Safe and Scalable Fleets of Autonomously Operating Vehicles in Shared Spaces

www.iliad-project.eu

H2020 project funded by the EC

Start: Jan 2017

Duration: 48 months

Funding: 7M€

Partners: 9



ILIAD in a nutshell

Why the intermediate is the big step



ILIAD Concept image

More quickly changing market needs, unforeseeable trends, shorter product life cycles, ...



ILIAD Flexible intralogistics needed



- Today's highly automated goods-to-man solutions
 - require dedicated warehouses,
 - are unsuitable for fresh food, bulky goods, etc.
- ILIAD's aim is to provide solutions for flexible intralogistics for the transition to automation in shared spaces.



Flexible intralogistics needed

Therefore we need intralogistic systems that are F-RE-SE-QUD-SA-EFF!
(1) highly flexible, (2) rock-solid reliable, (3) self-optimising,
(4) quickly deployable and (5) safe yet (6) efficient
in environments shared with humans.

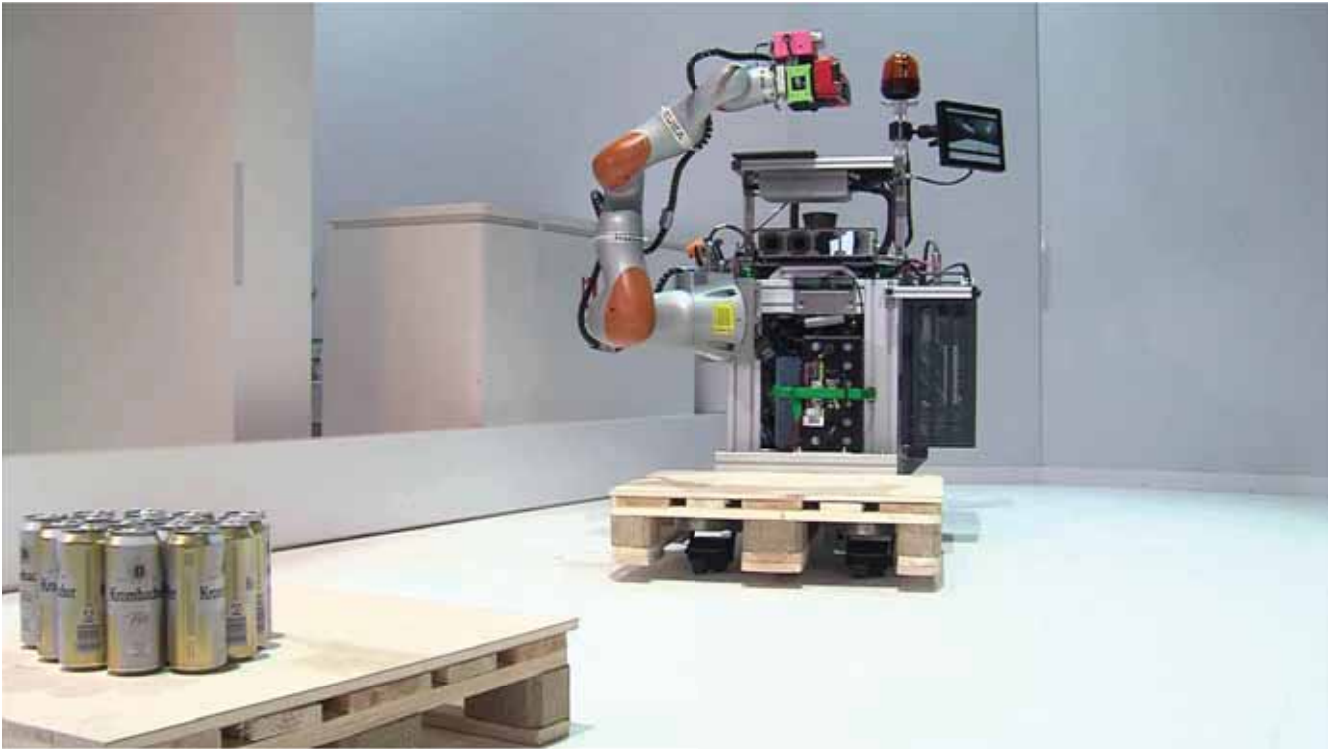


Approach



ILIAD Platform (CitiTruck)

Early prototype (APPLE platform at Hannover fair 2015)



Demonstrators

ILIAD Demonstrators

- Demos at key stakeholders from food distribution sector.
- Distribution of fresh food products particularly challenging:
 - sensitive products,
 - short shelf life,
 - rapid response to consumer needs.
- Food industry largest manufacturing sector in EU: 4.2 million jobs.

ILIAD Orkla Foods



Not always well-defined aisle environment

Heterogeneous and brittle (and sometimes large or heavy) items to pick.



ILIASD NCFM



- National Centre for Food Manufacturing

ILIASD ASDA



Photo credit: [Adrian Welsh](#)



Photo credit: [Redirack](#)



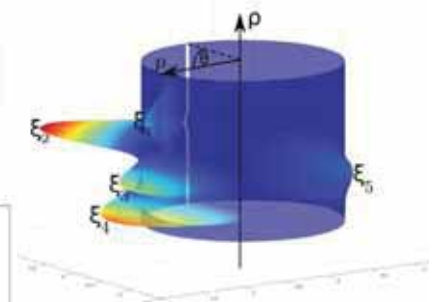
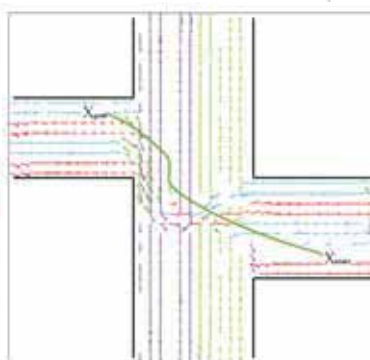
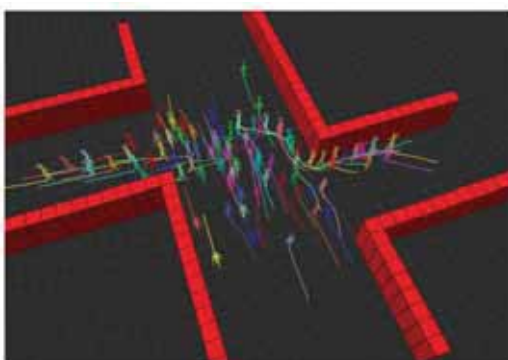
Easy deployment, long-term operation

Learning, modelling and
exploiting flow information



Learn, model and use dynamics

- Learn and map **how things usually move**.
 - Statistical multi-modal flow model.
- Use for socially compliant motion planning:
less "annoying", safer and
more efficient operation.



ILIAD Learn and model dynamics

- Learn and map **how things usually move**.
 - Extensive research on mapping geometric structure.
 - Environment typically defined by spatial movement patterns.



FP7 EU project SPENCER
<http://www.spencer.eu/>

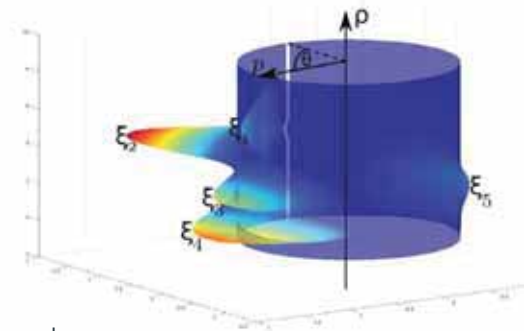
ILIAD Learn and model dynamics

- Learn and map how things usually move.
 - Extensive research on mapping geometric structure.
 - Environment typically defined by spatial movement patterns.
 - Using this information can
 - lead to safer and socially more acceptable robot trajectories;
 - allow to plan energy efficient paths for flying robots;
 - improve gas distribution mapping.



ILIAD Learn and model dynamics

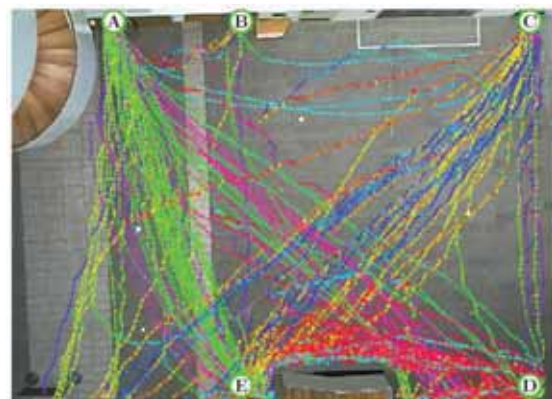
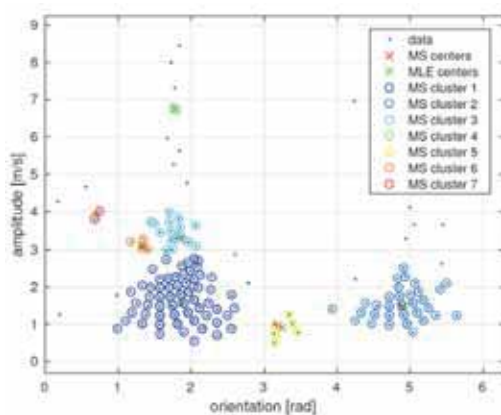
- Learn and map how things usually move.
 - Extensive research on mapping geometric structure.
 - Environment typically defined by spatial movement patterns.
 - Statistical multi-modal flow model CLiFF map (Circular–Linear Flow Field map) –
A probabilistic approach for general flow mapping



Enabling Flow Awareness for Mobile Robots in Partially Observable Environments.
T. P. Kucner, M. Magnusson, E. Schaffernicht, V. Hernandez Bennetts, and A. J. Lilienthal.
RA-L 2017 (2:2, pp. 1093-1100) / ICRA 2017

ILIAD Learn and model dynamics

- Learn and map how things usually move.
 - Extensive research on mapping geometric structure.
 - Environment typically defined by spatial movement patterns.
 - Statistical multi-modal flow model CLiFF map
 - Local elements are probability distributions of observations $V = (\theta, \rho)$
 - One circular (orientation θ) and one linear (speed ρ) random variable.



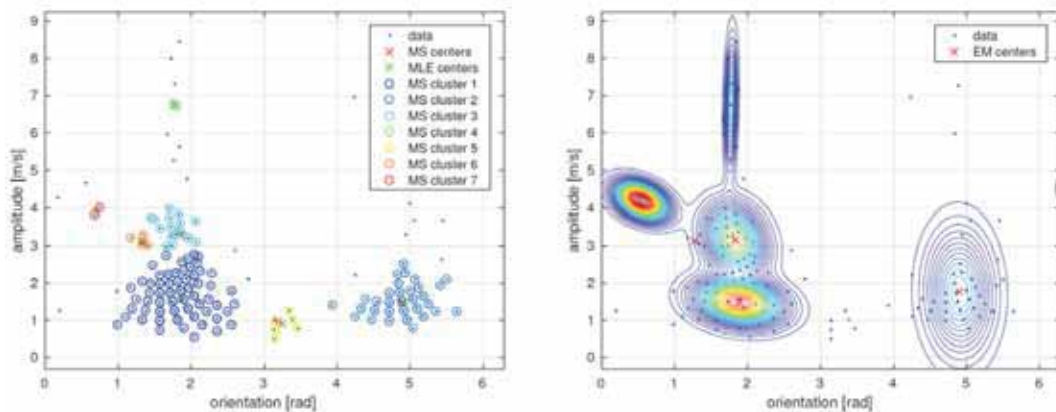
ILIASD Learn and model dynamics

- Learn and map how things usually move.
 - Extensive research on mapping geometric structure.

Learning CLiFF maps

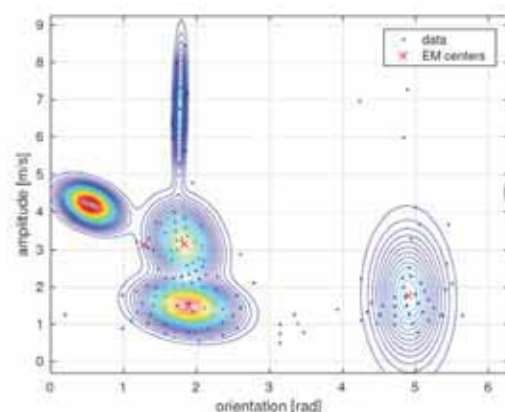
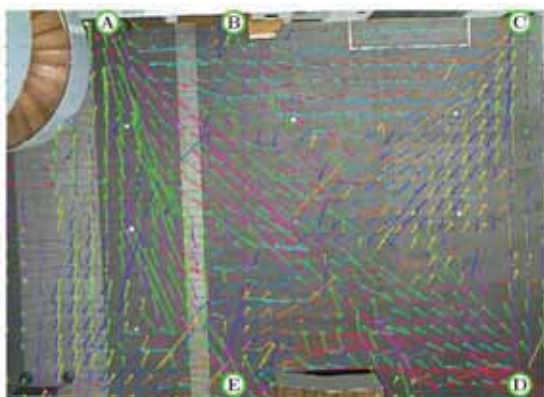
- Init: Mean Shift (MS) to determine number of clusters and their positions
- Use MS clusters to initialize Expectation Maximisation (EM)

- Semi-wrapped Gaussian mixtures



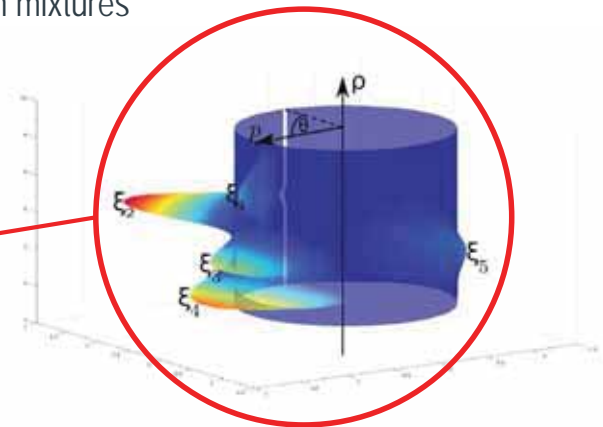
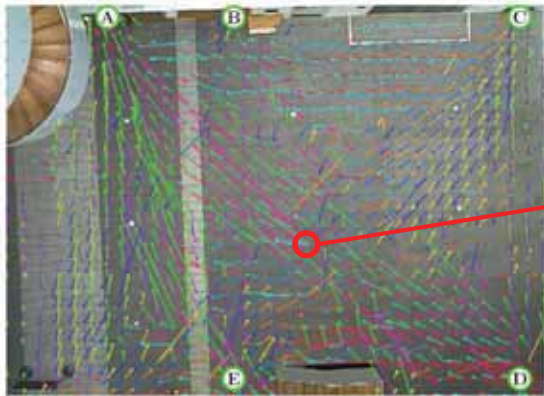
ILIASD Learn and model dynamics

- Learn and map how things usually move.
 - Extensive research on mapping geometric structure.
 - Environment typically defined by spatial movement patterns.
 - Statistical multi-modal flow model CLiFF map
 - Local elements are probability distributions of observations (θ, ρ)
 - One circular (orientation θ) and one linear (speed ρ) random variable.
 - Field of semi-wrapped Gaussian mixtures



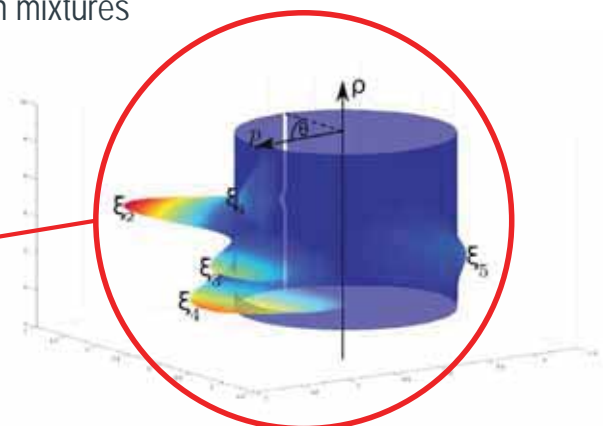
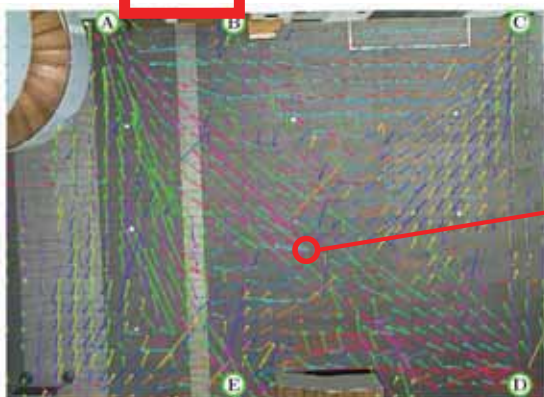
ILIAD Learn and model dynamics

- Learn and map how things usually move.
 - Extensive research on mapping geometric structure.
 - Environment typically defined by spatial movement patterns.
 - Statistical multi-modal flow model CLiFF map
 - Local elements are probability distributions of observations (θ, ρ)
 - One circular (orientation θ) and one linear (speed ρ) random variable.
 - Field of semi-wrapped Gaussian mixtures



ILIAD Learn and model dynamics

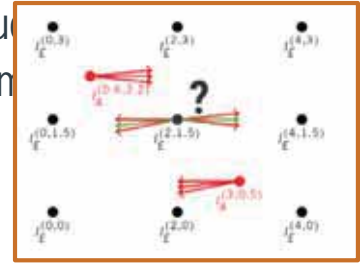
- Learn and map how things usually move.
 - Extensive research on mapping geometric structure.
 - Environment typically defined by spatial movement patterns.
 - Statistical multi-modal flow model CLiFF map
 - Local elements are probability distributions of observations (θ, ρ)
 - One circular (orientation θ) and one linear (speed ρ) random variable.
 - Field of semi-wrapped Gaussian mixtures





Learn and model dynamics

- Learn and map how things usually move.
 - Extensive research on mapping geometric structure
 - Environment typically defined by spatial movement patterns
 - Statistical multi-modal flow model CLiFF map
 - Field of semi-wrapped Gaussian mixtures
 - Observations in robotics are often sparse
 - => Data imputation to build dense maps from sparse measurements
 - Monte Carlo Imputation (MC)
 - sampling virtual observations from the surrounding
 - tends to preserve multimodal characteristics and keep sharp transitions
 - Nadaraya Watson Imputation (NW)
 - Weighted extrapolation (distance kernel)
 - smooths data and models introduces gradual changes
 - MC performed better than NW on pedestrian data (and is less sensitive to kernel size) – but results may differ in different applications

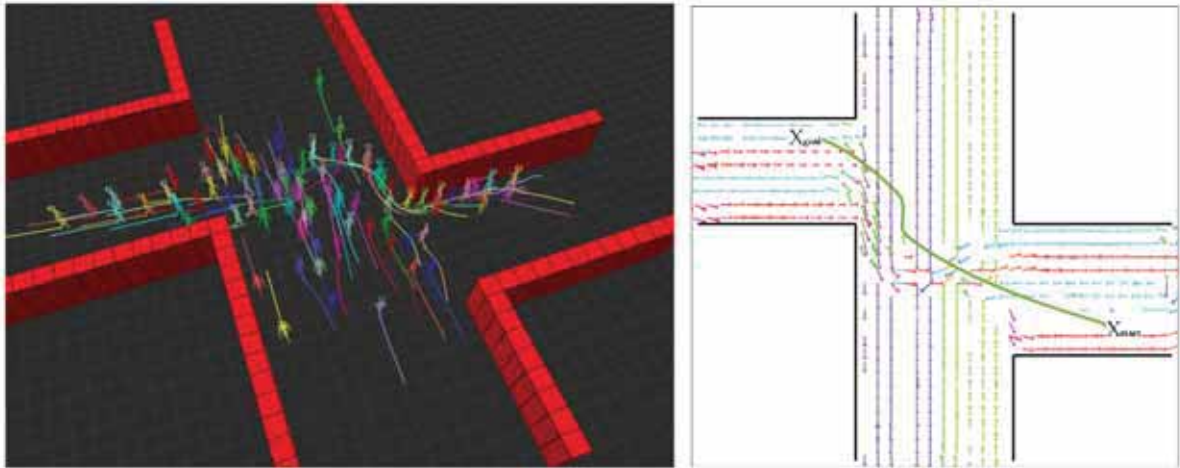


Learn and model dynamics

- Learn and map how things usually move.
 - Extensive research on mapping geometric structure.
 - Environment typically defined by spatial movement patterns.
 - Statistical multi-modal flow model CLiFF map
 - Field of semi-wrapped Gaussian mixtures
 - Observations in robotics are often sparse
 - => Data imputation to build dense maps from sparse measurements
- **Summary**
 - It is possible to accurately represent multimodal (even turbulent) flow using CLiFF map.
 - It is possible to reconstruct a dense representation based on sparsely distributed observations.

ILIAD Using dynamics models

- Use for socially compliant motion planning: less "annoying", safer and more efficient operation.



Kinodynamic Motion Planning on Gaussian Mixture Fields.
L. Palmieri, T. Kucner, M. Magnusson, A. J. Lilienthal, and K. O. Arras
ICRA 2017

ILIAD Using dynamics models

- Socially compliant motion planning using CLiFF map
 - Mobile robot motion planning approach: CLiFF-RRT*
 - CLiFF map – provides learned perception prior
 - RRT* motion planner
 - asymptotically optimal sampling-based motion planner
 - considers the robot's kinematic and its non-holonomic constraints

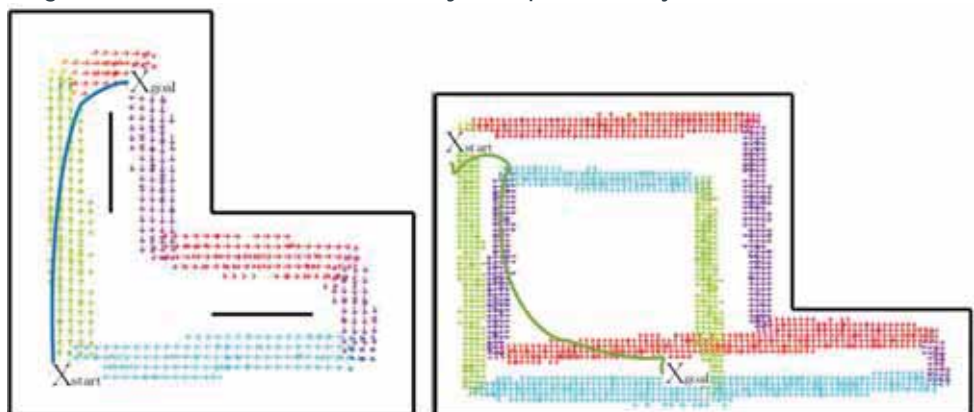
Kinodynamic Motion Planning on Gaussian Mixture Fields.
L. Palmieri, T. Kucner, M. Magnusson, A. J. Lilienthal, and K. O. Arras
ICRA 2017

ILIAD Using dynamics models

- Socially compliant motion planning using CLiFF map
 - Mobile robot motion planning approach: CLiFF-RRT*
 - CLiFF map – provides learned perception prior
 - RRT* motion planner
 - => CLiFF map guides sampling in the RRT* planner
 - low costs for paths that comply to the directions of CLiFF-map mixture components and high costs for paths in opposite directions

ILIAD Using dynamics models

- Socially compliant motion planning using CLiFF map
 - Mobile robot motion planning approach: CLiFF-RRT*
 - CLiFF map – provides learned perception prior
 - RRT* motion planner
 - => CLiFF map guides sampling in the RRT* planner
 - Results are very encouraging
 - Planner generates reasonable ("socially compliant") trajectories

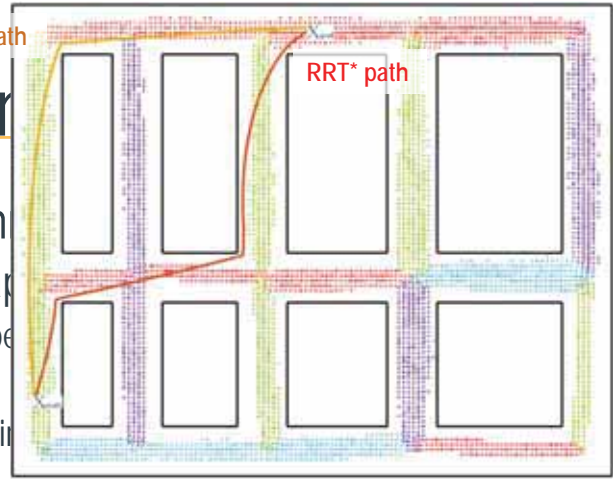




Using dynamical

CLiFF-RRT* path

RRT* path



- Socially compliant motion planning
 - Mobile robot motion planning application
 - CLiFF map – provides learned perception
 - RRT* motion planner
 - => CLiFF map guides sampling in free space
 - Results are very encouraging
 - Planner generates reasonable ("socially compliant") trajectories
 - Planner is very efficient (significantly faster than RRT and RRT*) / very fast convergence
 - Generates higher quality paths (in terms of smoothness and length)



Easy deployment, long-term operation

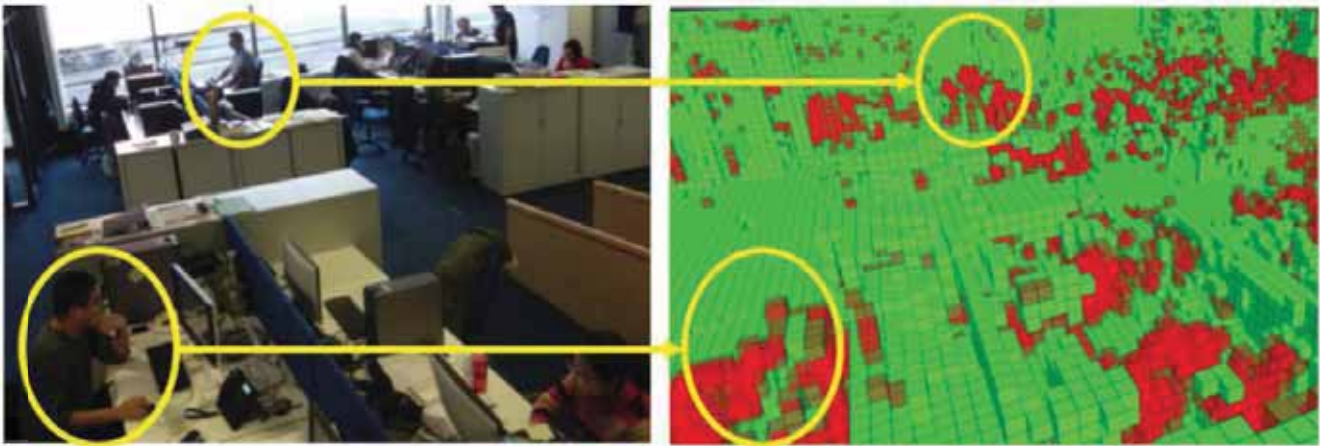
Ongoing work



ILIAD Spatiotemporal mapping

- Representations and inference for compression of past experience and prediction of future states with confidence intervals.
- Learn where and when activities happen.

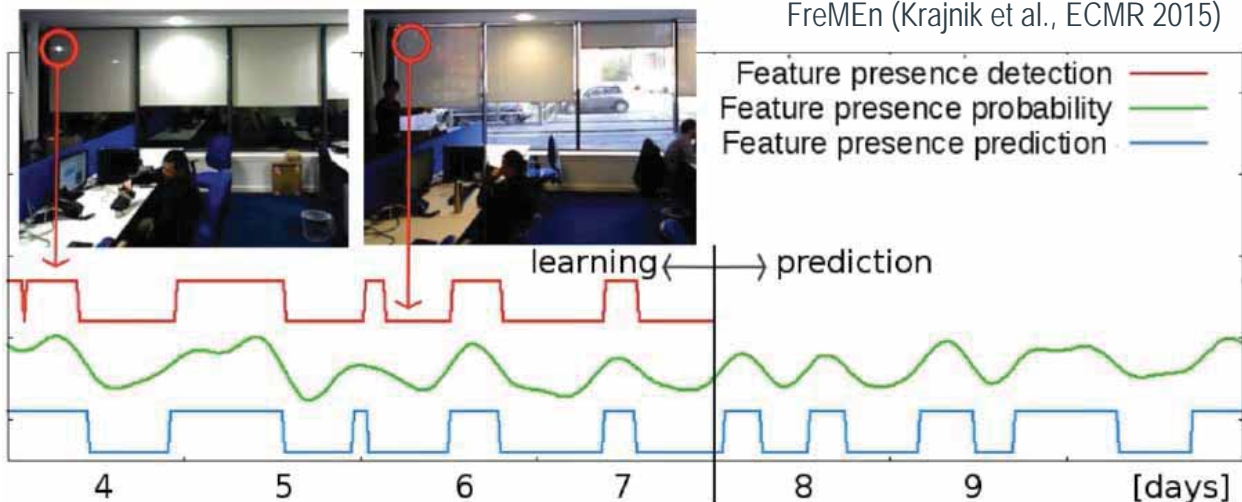
FreMEn (Krajnik et al., ECOMR 2015)



ILIAD Predicting patterns

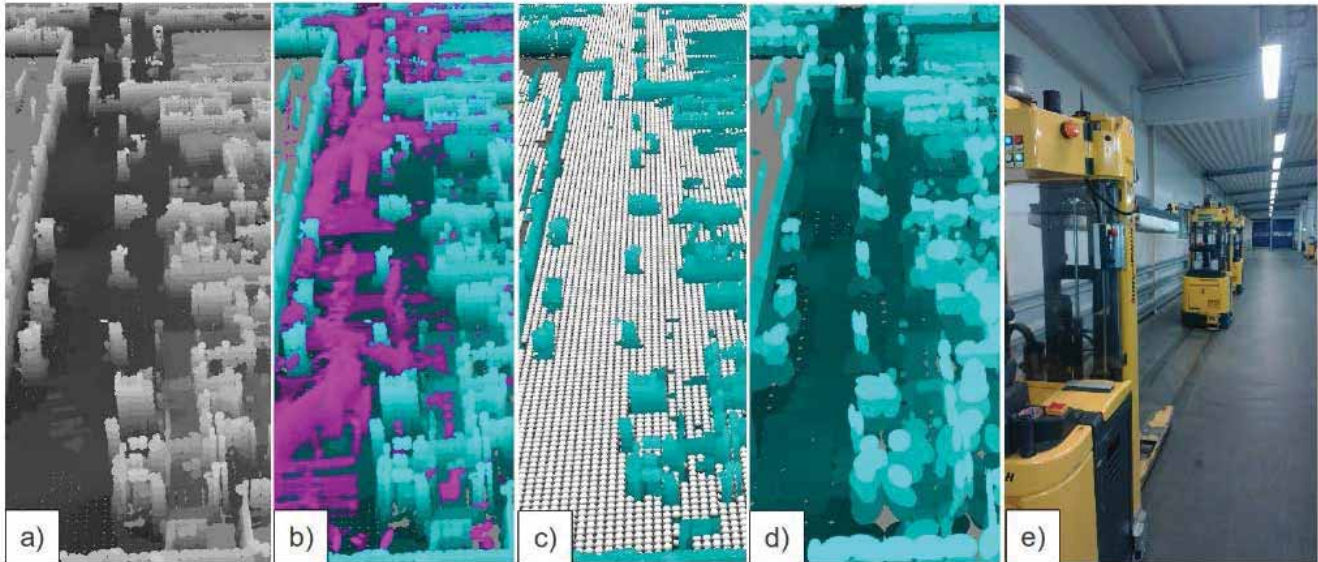
- Combine inputs from mapping and tracking.
- Actively update knowledge – plan where and when to collect data.

FreMEn (Krajnik et al., ECOMR 2015)



ILIAD Reliability-aware loc. & maps

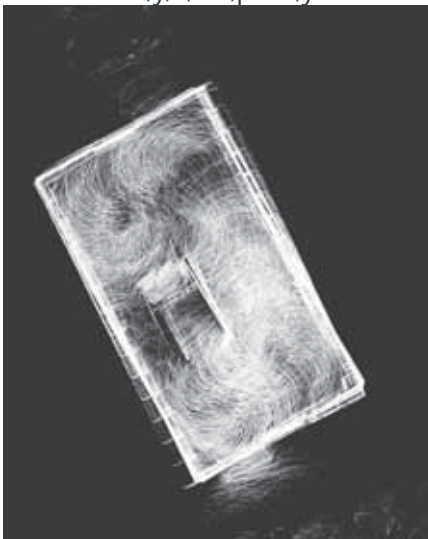
- Combining metrics to assess quality of scan registration.
- Detect mapping errors using learned structural cues.



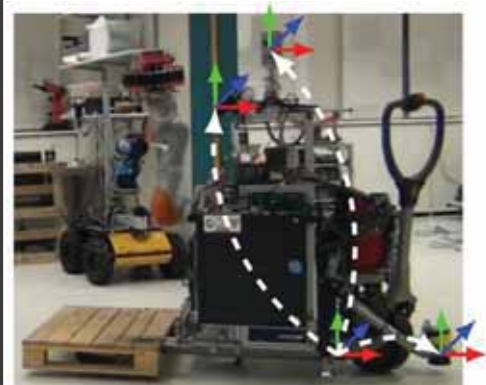
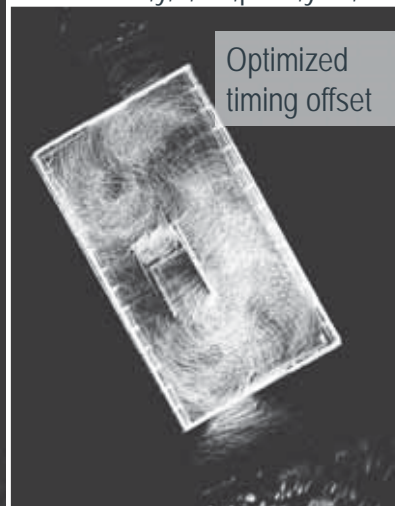
ILIAD Auto-calibration

- Unsupervised monitoring and (re-)calibration of sensors.
- Find regions with high information for calibration.

6 DOF – $x, y, z, \text{roll}, \text{pitch}, \text{yaw}$



7 DOF – $x, y, z, \text{roll}, \text{pitch}, \text{yaw}, dt$



Safe and human-aware operation

Ongoing work



ILIAD Human safety

- Study human safety in shared environments.
- **Connect injury biomechanics** to safe control and planning.
 - Associate vehicle dynamics to injury safety database.
- **Extend Safe Motion Unit** paradigm to vehicle-human and vehicle-vehicle interaction.
- Shape vehicle velocity based on
 - humans and vehicles in the environment,
 - environment observability,
 - predictive braking models.

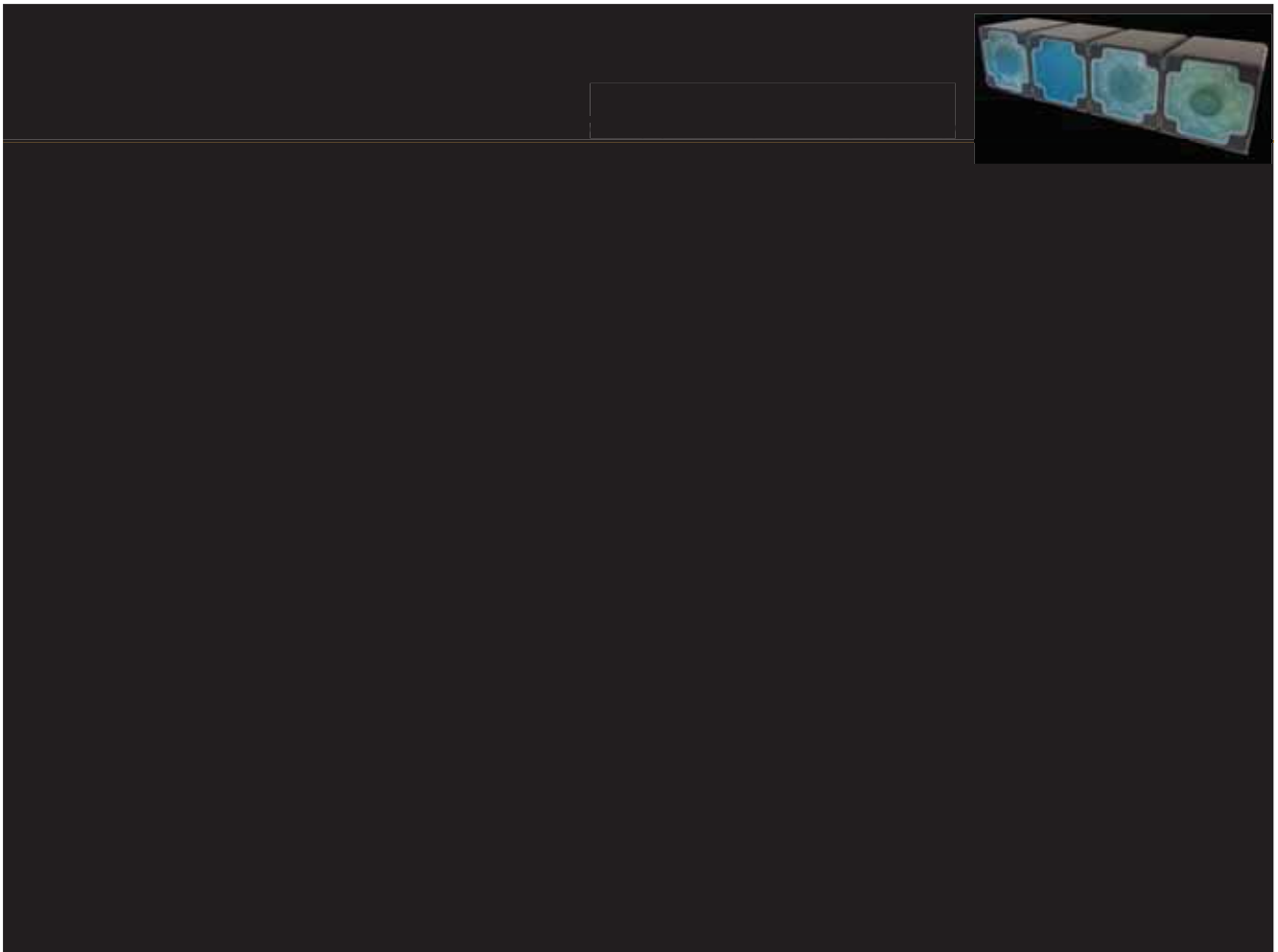




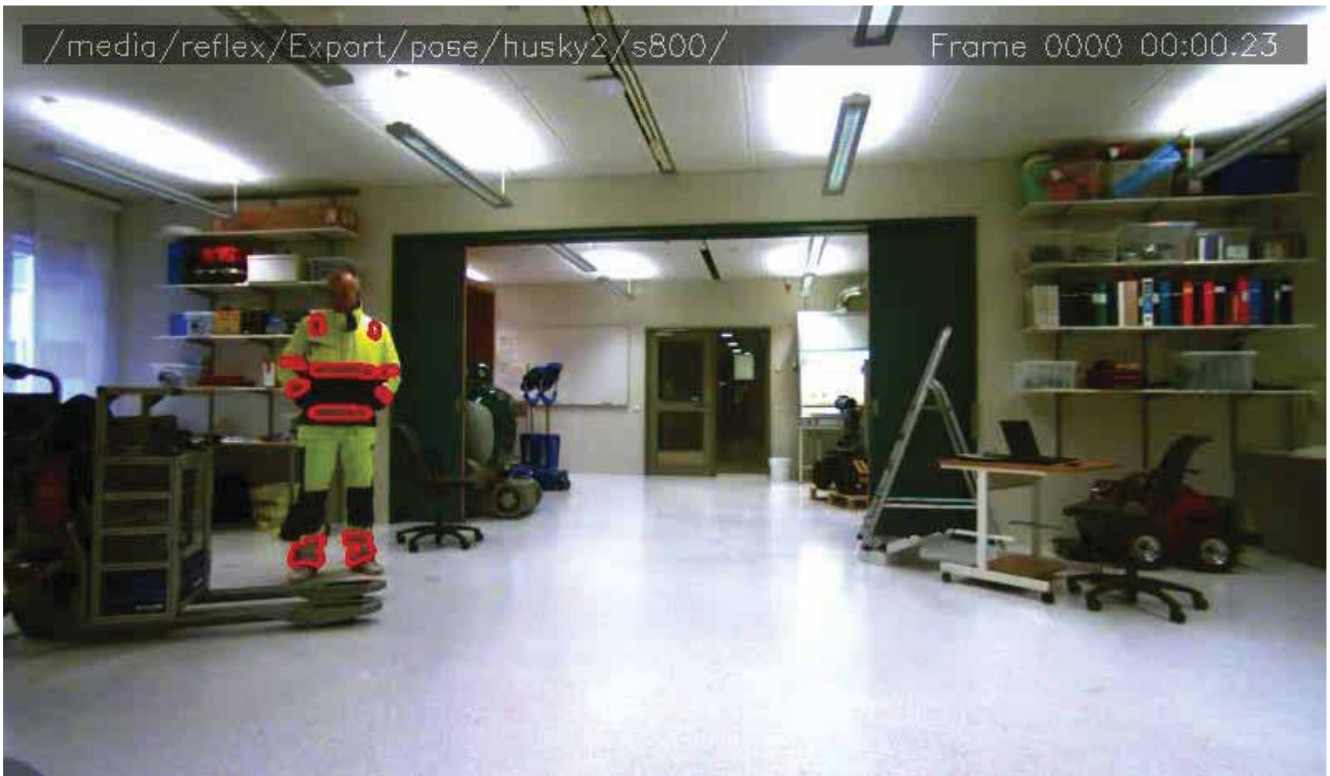
Human-aware fleets



- Increase human-robot cooperation safety & efficiency.
 - Detect, track, and analyse people.



ILIAD Human-aware fleets



ILIAD Human-aware fleets



- Increase human-robot cooperation safety & efficiency.
 - Detect, track, and analyse people.
 - => Retenua AB



ILIAD Human-aware fleets

- Increase human-robot cooperation safety & efficiency.
 - Detect, track, and analyse people.
 - Recognise **human intentions**.
 - Visually communicate **robot intentions**.
 - Socially normative **motion planning**.



Manipulation

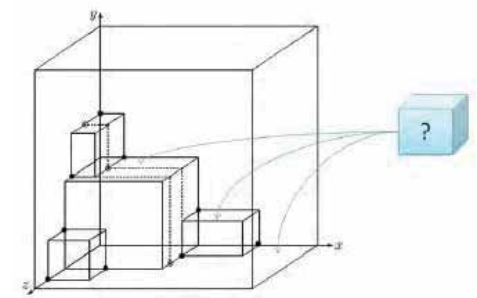


Ongoing work



ILIAD Manipulation

- Innovative end-effectors.
- Perception for dense packets, and plastic wrapping.
- Control for unwrapping, picking, palletising.
- Optimise package positions on pallets.



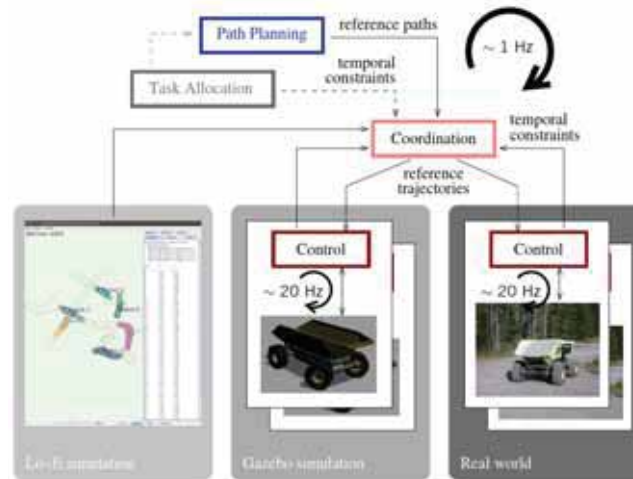
Fleet management

Ongoing work



ILIAD Fleet management

- Integrated task allocation, motion planning, and coordination.
- Guaranteed deadlock-free operation.
- Continuously revise w.r.t. changing requirements.



Where the Intermediate is the Big Step

Intralogistics with Safe and Scalable Fleets of Autonomously Operating Vehicles in Shared Spaces

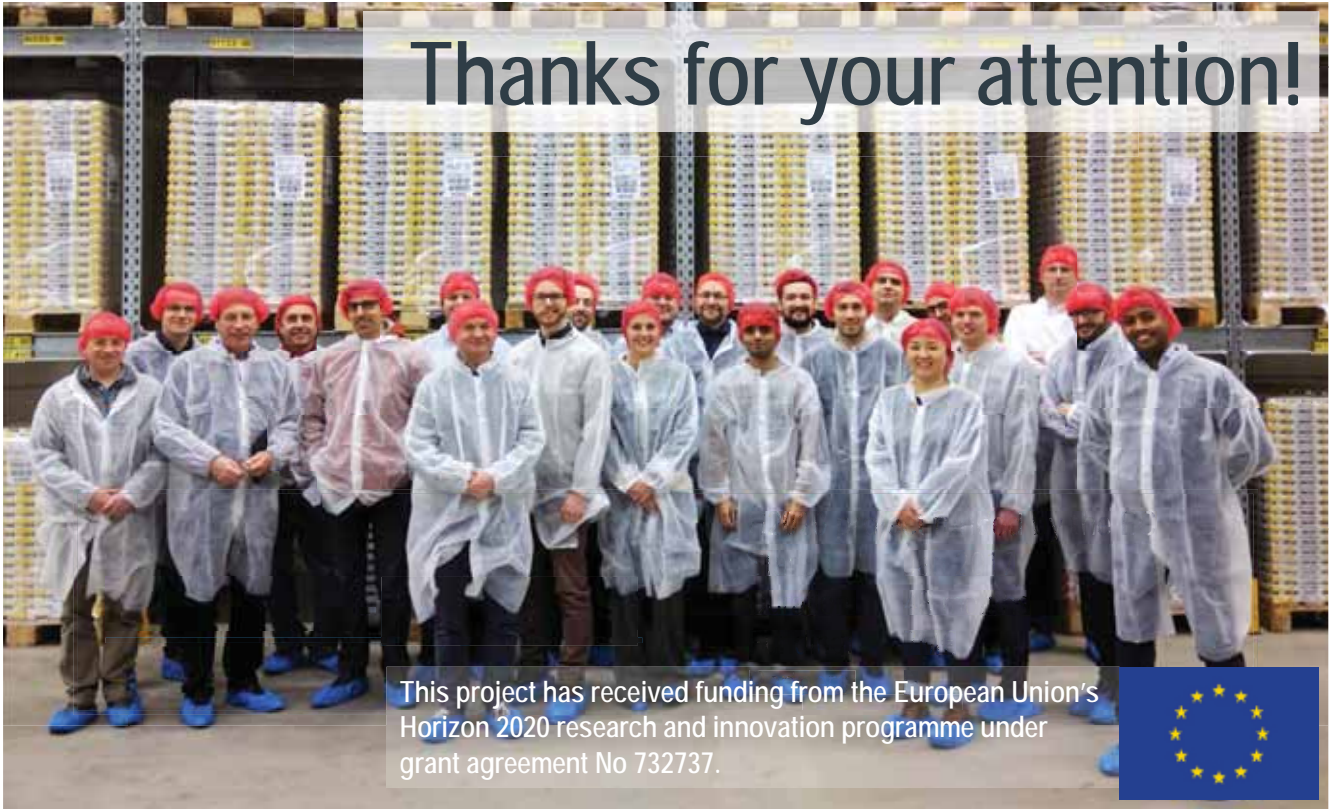
Thanks for your attention!



Intra-
Logistics with
Integrated
Automatic
Deployment



Thanks for your attention!



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732737.



Where the Intermediate is the Big Step

Intralogistics with Safe and Scalable Fleets of
Autonomously Operating Vehicles in Shared Spaces

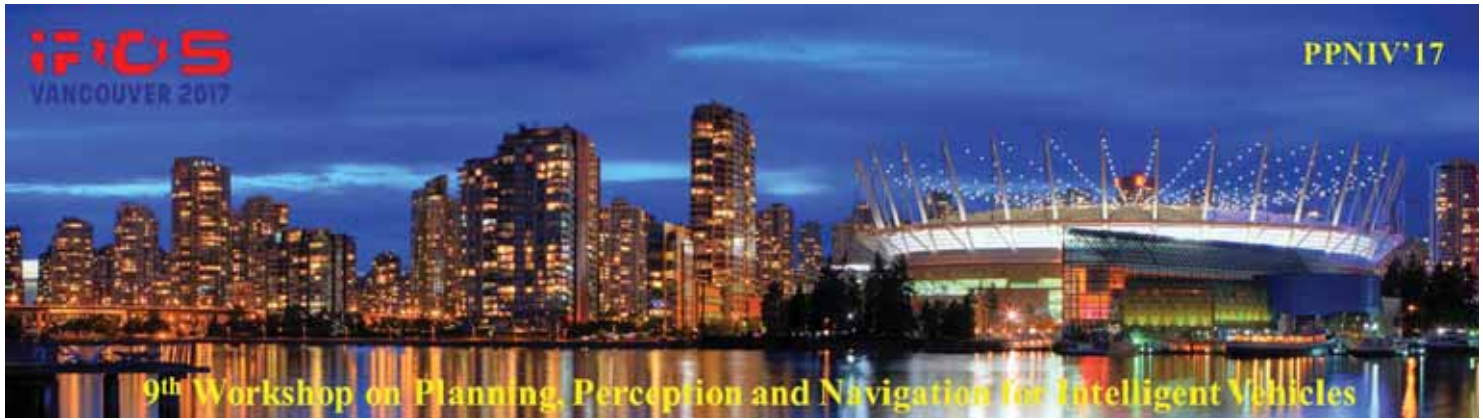


Achim J. Lilienthal

contact:

www.mrolab.eu/achim-lilienthal
achim.lilienthal@oru.se





2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Session V

Perception

- **Title: Fast Image-Based Geometric Change Detection in a 3D Model**
Authors: Emanuel Palazzolo, Cyrill Stachniss
- **Title: Fast Graph-Based Place Recognition**
Authors: Mattia G. Gollub, Renaud Dubé, Hannes Sommer, Igor Gilitschenski and Roland Siegwart



2017 IEEE/RSJ International Conference on Intelligent Robots and Systems

Change Detection in 3D Models Based on Camera Images

Emanuele Palazzolo

Cyrill Stachniss

Abstract—3D models of the environment are used in numerous robotic applications and should reflect the current state of the world. In this paper, we address the problem of quickly finding structural changes between the current state of the world and a given 3D model using a small number of images. Our approach finds inconsistencies between pairs of images by reprojecting an image onto the other by passing through the 3D model. Ambiguities about possible inconsistencies resulting from this process are resolved by combining multiple images such that the 3D location of the change can be estimated. A focus of our approach is that it can be executed fast enough to allow the operation on a mobile system. We implemented our approach in C++ and tested it on an existing dataset for change detection as well as on self recorded images sequences. Our experiments suggest that our method quickly finds changes in the geometry of a scene.

I. INTRODUCTION

Building 3D models of the environment is a frequently addressed problem in robotics as they are needed for a wide range of applications. For most applications that include autonomous behavior, such models should correspond as well as possible to the current state of the environment. In case the environment was substantially changed, existing models must be updated. For this purpose, the possibility of directing a mapping or exploring robot directly towards the possible regions that have changed instead of repeating the whole mapping process can greatly reduce the required efforts. Therefore, it is important to reliably identify locations in the environment or in a 3D model that have changed.

In this paper, we address the problem of finding changes between a previously built 3D model and its current state based on a small sequence of images (keyframes) recorded in the environment, see Fig. 1 for an illustration. Two aspects are important for us: first, we want to reliably locate changes in the model and second, the approach should have a limited computational demand so that it can be executed on a mobile platform. Our approach seeks to find changes between the current state of the world and a previously recorded, existing 3D model of the scene. For finding inconsistencies, we do not build another 3D model from the newly obtained image data. Instead, we project the currently obtained image onto the 3D model and then back to a view-point at which another image of the current sequence has been taken. Through a comparison between the back-projected images and the one observed in reality, we can identify possible regions of change. To eliminate ambiguities, this process is executed for multiple image pairs. Typically 4-5 keyframe images

All authors are with the University of Bonn, Institute of Geodesy and Geoinformation, Bonn, Germany.

This work has partly been supported by the DFG under the grant number FOR 1505: Mapping on Demand.

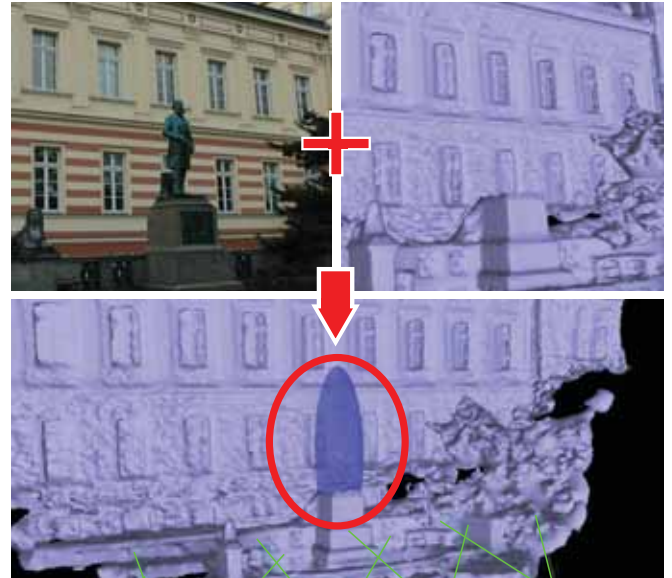


Fig. 1: Our approach aims at quickly finding changes in the environment based on an existing 3D model and a sequence of (currently recorded) images.

are sufficient to find areas of change and then estimate the 3D location where the geometry has changed. Compared to existing approaches for visual change detection such as the work by Taneja et al. [13] or Ulusoy et al. [14], our method is substantially faster towards execution on a mobile robot.

The main contribution of this paper is a new and fast approach for identifying differences between an existing 3D model and a small sequence of images recorded in the environment. Our approach identifies the approximate area of change fast enough to be executed on a navigating robot, which sets it apart from several related other techniques. We identify inconsistencies by comparing the acquired images to back-projected images that would have been obtained assuming the 3D model is correct, in combination with a forward intersection of the potentially inconsistent regions. Our experiments suggest that our method quickly finds the approximate location of the change in the scene and is fast enough to potentially guide an exploring ground robot or UAV seeking to map the changes in the environment.

We make two key claims: our approach is able to (i) identify the location of changes in the environment, in the form of 3D volumes in the world coordinate frame, using a 3D model and a sequence of images, and (ii) it is fast enough to be executed on a mobile robot, i.e. analyzing a sequence of keyframe images does not take longer than recording it (e.g., 10s for five keyframe images with a size of 1500 by 1000 pixels).

II. RELATED WORK

Building 3D models can be an expensive process as it requires a good coverage of the environment and potentially dedicated sensors or equipment. To reduce this cost, it is important to identify, on an existing model, the parts that have changed, and direct the exploration towards those locations. For this reason, 3D change detection is an increasingly popular topic, see [8].

In the past, several 2D change detection algorithms have been proposed [9]. Several of such methods are affected by lighting conditions, seasonal changes, weather conditions, and other differences that may occur between the recording of the old and the new images. Moreover, the images often do not provide information on the actual 3D location of the change. Sakurada et al. [10] try to overcome these problems by estimating the probabilistic density of the depth from the old set of images and by comparing it with the depth computed from the new set of images. Eden et al. [3] compare 3D lines in the images instead of using color or intensity information. A more recent approach by Alcantarilla et al. [1], instead use a deep convolutional neural network combined with a dense reconstruction technique.

Another approach to 3D change detection is to build a 3D model from the new images through Multi-View Stereo and then compare the new model with the old one. This is, however, often a rather time consuming activity, at least when using cameras. Golparvar-Fard et al. [4] use this approach combined with a support vector machine classifier to obtain an updated voxelized model of the environment.

A popular and effective approach is to infer the changes of the environment using a previously built 3D model and a sequence of newly acquired images. One way to achieve this is to maintain a voxelized model of the environment and detect the probability of change in it by comparing the color of a voxel and the color of the pixels in the images onto which it projects. Examples of this approach are the one by Ulusoy et al. [14] or the one by Pollard et al. [6].

Another relevant strategy that use an existing 3D model and newly acquired images is to identify changes by re-projecting images onto each other by passing through the existing model and compare the inconsistencies in the re-projection. Taneja et al. [13] use this technique on pairs of images, and apply a graph cut minimization to label the changed area in 3D in a voxelized model. In addition, Qin et al. [7] combine the pairwise detected inconsistencies by counting the rays that hit every pixel for each image, in order to get rid of the ambiguities. They stop at the image level and do not estimate the 3D location of the change.

In this paper, we use a reprojection technique similar to [13] and [7] to identify the changed regions in the images. We resolve ambiguities by fusing multiple images and introduce a fast way for estimating the rough location of change in 3D. The whole process takes only a few seconds for an image sequence. In contrast to that, state-of-the-art approaches such as [13] or [14] have execution times in the order of minutes.

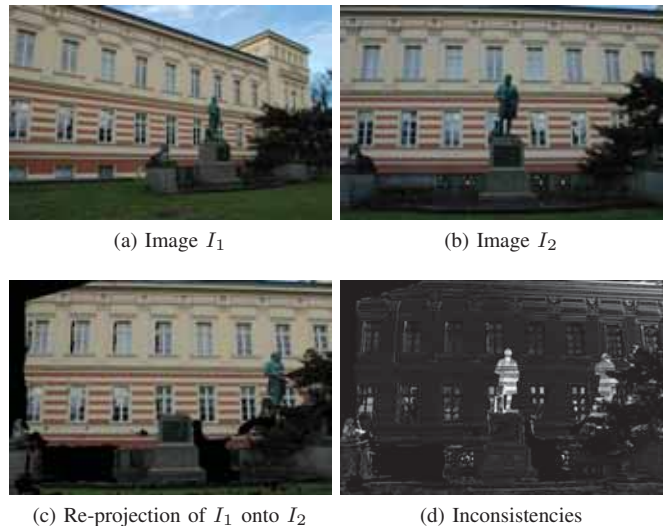


Fig. 2: A pair of images, the first image reprojected onto the second, and the inconsistencies between them.

III. FAST IMAGE-BASED CHANGE DETECTION

Our approach aims at spotting areas in an environment that have changes with respect to a previously built 3D model. It does so by exploiting a sequence of around five images through evaluating how the projections of image content from one image to the model and back to another image looks like. In terms of computational demands, this process is substantially more efficient than generating a new, dense 3D model and comparing it directly with the given one. Note that we assume a good pose estimate for the robot. We obtain the (approximate) location of the 3D model and the viewpoint of the images as described in Sec. III-A below. The first step is to detect possible inconsistencies of an image with its neighboring images assuming that the 3D model is correct. After computing pairwise inconsistency hypotheses, we fuse them to eliminate the intrinsic ambiguities and estimate the location of change by triangulation. Given that we look for inconsistencies between the 3D model and new images, our approach only finds changes from images where the rays corresponding to pixels intersect with the 3D model.

A. Camera pose estimate

Our algorithm requires an estimate of the viewpoints of the images w.r.t. the 3D model. We obtain this through direct georeferencing fusing GPS, IMU, and visual odometry, as described in [11]. The approach employs the iSAM2 algorithm, and provides uncertainty information about all sensor poses in form of a covariance matrix. In case no GPS information is available, approaches for camera to 3D model localization such as [2] can be used—although we did not directly try that here.

B. Inconsistencies Between Images Pairs

To detect inconsistencies between a pair of images consisting of the images I_1 and I_2 , we create a new image $I_{1 \rightarrow 2}$ that represents the content of I_1 as seen from view point of I_2 given the 3D model. Given the calibration matrix and

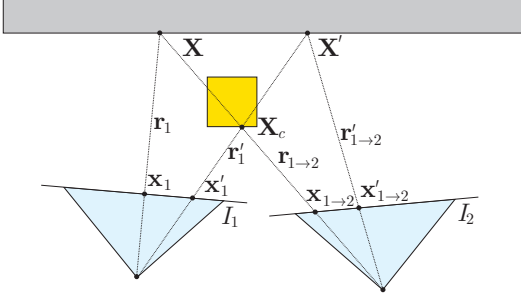


Fig. 3: Re-projection procedure. The gray rectangle represents the known 3D model, while the yellow square is a change not present in the original model. Using two images, a point \mathbf{X}_c , not present in the model, is re-projected onto two pixels $\mathbf{x}_{1 \rightarrow 2}$ and $\mathbf{x}'_{1 \rightarrow 2}$.

the pose at which the camera took I_1 , we can compute the projection of a 3D point \mathbf{X} onto the image plane resulting in a 2D point at pixel \mathbf{x}_1 :

$$\mathbf{x}_1 = \mathbf{P}_1 \mathbf{X}, \quad (1)$$

where \mathbf{x}_1 is expressed in homogeneous coordinates and $\mathbf{P}_1 = \mathbf{K}_1[\mathbf{R}_1 | -\mathbf{R}_1 \mathbf{t}_1]$ is the camera projection matrix computed from the calibration matrix \mathbf{K}_1 of the camera and the rotation \mathbf{R}_1 and translation \mathbf{t}_1 that transform the world coordinates into camera coordinates.

By inverting Eq. (1), we compute the ray from the projection center of the camera through the pixel to the 3D world. This allows us to back-project each pixel of I_1 onto the 3D model assuming the known intrinsic parameters (\mathbf{K}_1) and the rotation matrix \mathbf{R}_1 from the extrinsic parameters:

$$\mathbf{r}_1 = \mathbf{R}_1^\top \mathbf{K}_1^{-1} \mathbf{x}_1, \quad (2)$$

where \mathbf{r}_1 is the direction of the ray in world coordinates.

In the next step, we project the intersections \mathbf{X} between the rays and the 3D model onto the image plane of I_2 to obtain $I_{1 \rightarrow 2}$ (see Fig. 2c for a real example):

$$\mathbf{x}_{1 \rightarrow 2} = \mathbf{P}_2 \mathbf{X}, \quad (3)$$

where \mathbf{P}_2 is the camera projection matrix corresponding to image I_2 . In this way, we obtain a new image $I_{1 \rightarrow 2}$ that can be compared to I_2 . Since the exact poses of the cameras are unknown and the 3D model is not perfect, the point $\mathbf{x}_{1 \rightarrow 2}$ has an uncertainty represented by the covariance matrix $\Sigma := \Sigma_{\mathbf{x}_{1 \rightarrow 2} \mathbf{x}_{1 \rightarrow 2}}$. To overcome this, we compute, for every pixel of I_2 the minimum Euclidean norm of the intensity difference to each pixel of $I_{1 \rightarrow 2}$ in a neighborhood \mathcal{N} around the projected pixel. We compute the size of this neighborhood by propagating the pose uncertainty obtained while recording the images into the image points, see Sec. III-A. In detail, we search within the 3σ area given by Σ and select the pixel with the smallest difference:

$$D_{1 \rightarrow 2}(i, j) = \min_{k, l \in \mathcal{N}} \|I_2(i, j) - I_{1 \rightarrow 2}(k, l)\|_2, \quad (4)$$

where i, j, k, l are pixel coordinates and the neighborhood \mathcal{N} is defined as:

$$\mathcal{N} = \left\{ \forall (k, l) \in I_{1 \rightarrow 2} \left| \begin{bmatrix} i - k \\ j - l \end{bmatrix}^\top \Sigma^{-1} \begin{bmatrix} i - k \\ j - l \end{bmatrix} < d^2 \right. \right\}, \quad (5)$$

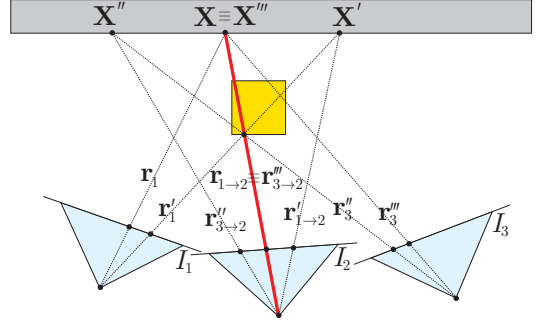


Fig. 4: Ambiguity elimination using multiple images. When re-projecting I_1 and I_3 onto I_2 , only one ray (therefore one pixel) is coincident. The thicker red line represents that coincident ray.

where $d^2 = 11.82$ is the critical value of the χ_2^2 distribution corresponding to a probability of 99.73%, i.e. a 3σ boundary on the normal distribution. Finally, we normalize $D_{1 \rightarrow 2}$ to values between $[0, 1]$. Fig. 2d shows the result of this procedure.

If there is no change in the 3D model between the acquisition time and the time when the images have been taken, all pixels in I_1 should correctly re-project onto I_2 . Therefore, I_2 and $I_{1 \rightarrow 2}$ should be identical and $D_{1 \rightarrow 2}$ should be small or equal to 0 for each pixel. If there is, however, a change in the model, pixels corresponding to the change reproject onto the wrong place in I_2 . Thus, $D_{1 \rightarrow 2}$ allows us to identify the changes (as long as not all pixels in the current images have the same RGB value, i.e. represent a large homogeneous area)

The process, however, leads to ambiguities. As Fig. 3 illustrates, a single point \mathbf{X}_c corresponding to a change in the 3D model generates two pixel locations, $\mathbf{x}_{1 \rightarrow 2}$ and $\mathbf{x}'_{1 \rightarrow 2}$, in $D_{1 \rightarrow 2}$, one corresponding to the change in I_1 reprojected onto I_2 and one corresponding to the change in I_2 reprojected onto I_1 . To eliminate this ambiguity, we use multiple pair-wise image comparisons as described in the following section.

C. Inconsistency Detection using Multiple Images

The ambiguity produced by the re-projection of an image onto another one can be eliminated by considering multiple image pairs. Fig. 4 shows how a pixel belonging to the same change in a third image I_3 re-projects onto I_2 at two different locations. It is important to note that one of the two points is mapped to the same location as a change detected by re-projecting I_1 onto I_2 . Thus, the pixels that re-project onto the same region of I_2 from the other images represent the real change.

To localize the changes, we therefore compare an image with its m neighboring keyframe images. For each image I_t , we store an inconsistency image D_t resulting from the product of all the inconsistency images obtained from the neighboring images reprojected onto I_t :

$$D_t(i, j) = \prod_{s \in \mathcal{S}(t)} D_{s \rightarrow t}(i, j), \quad (6)$$

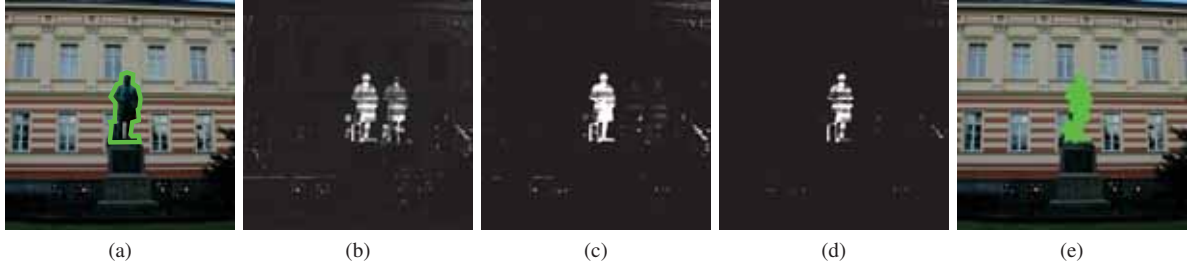


Fig. 5: (a) The statue (here manually marked in green) is not in the model. (b) Inconsistencies between 2 images ($m = 1$). (c) Inconsistencies between 3 images ($m = 2$). (d) Inconsistencies between 4 images ($m = 3$). (e) Original image masked with the segmented area obtained from the inconsistency image with $m = 3$. (best viewed in color)

where $\mathcal{S}(t)$ is the set of m neighboring keyframe images of I_t . In our implementation, we typically use the four closest images in time to I_t . Fig. 5 depicts the output of Eq. (6), for $m = 1, 2$, and 3.

D. Segmentation and Data Association

The procedure explained so far enables us to identify the pixels in each image where changes occur. For reliably computing the regions of change, we first filter out the noise with an erosion-dilation procedure, then apply a standard border following algorithm [12]. We discard all the regions with a contour shorter than a threshold (in our implementation 500 px) to filter out noise and changes that are too small. The next step is to associate the regions from the images with each other. To do that, we compute and compare hue-saturation histograms region-wise and perform standard cross-correlation together with a simple geometric consistency check using the epipolar lines.

E. Estimating the Location of Change

Once we obtain the segmented 2D regions and the association between them, we proceed to estimate the 3D location of the change.

To simplify the notation in the remainder of this section, the following equations will refer to a single change in images, i.e. dropping an index referring to individual regions. The whole procedure is repeated for every region (of detected change).

To estimate the 3D volumes in which the changes occur, we first compute, for every region identified as a change, the mean location \bar{x}_t and spread in form of the covariance Σ_t in the image. We then compute, for each change, a 3D point \bar{X} in the 3D world coordinates by triangulating the mean location in each image. Specifically, we setup a system of equations in the form

$$\mathbf{A}\bar{X} = \mathbf{0}, \quad \text{with} \quad \mathbf{A} = \begin{bmatrix} \mathcal{S}(\bar{x}_1)\mathbf{P}_1 \\ \vdots \\ \mathcal{S}(\bar{x}_n)\mathbf{P}_n \end{bmatrix}, \quad (7)$$

where \mathbf{A} is a $3n \times 4$ matrix composed by 3×4 blocks, n is the number of images, \mathbf{P}_t is the projection matrix relative to image I_t , and $\mathcal{S}(\bar{x}_t)$ is the skew symmetric matrix corresponding to the mean pixel \bar{x}_t , in homogeneous

coordinates, i.e.:

$$\bar{x}_t = \begin{bmatrix} x_t \\ y_t \\ w_t \end{bmatrix}, \quad \mathcal{S}(\bar{x}_t) = \begin{bmatrix} 0 & -w_t & y_t \\ w_t & 0 & -x_t \\ -y_t & x_t & 0 \end{bmatrix}. \quad (8)$$

We solve this system using singular value decomposition and retrieve \bar{X} by taking the right-singular vector of \mathbf{A} belonging to its smallest singular value (Fig. 6a). For each change in the image, we additionally compute the K sigma points [5] $\mathbf{v}_t^{(k)}$ ($k = 1 \dots K$) corresponding to \bar{x}_t and Σ_t and project the sigma points to the 3D space to estimate the region of change in 3D. To compute the 3D position of the sigma points, we define for each image a plane \mathcal{A}_t passing through \bar{X} with normal equal to the direction of the ray \bar{r}_t obtained through Eq. (2) for \bar{x}_t .

We can define the plane in homogeneous coordinates as a 4-dimensional vector:

$$\mathcal{A}_t = \begin{bmatrix} \bar{r}_t \\ d \end{bmatrix}, \quad (9)$$

where the last element $d = \bar{r}_t^T \bar{X}$ is the distance between the camera and \bar{X} .

The projection of $\mathbf{v}_t^{(k)}$ on \mathcal{A}_t is the intersection $\mathbf{V}_t^{(k)}$ between the plane and the ray $\mathbf{r}_t^{(k)}$ generated from $\mathbf{v}_t^{(k)}$. We compute $\mathbf{V}_t^{(k)}$ by expressing $\mathbf{r}_t^{(k)}$ in Plücker coordinates as a line $\mathbf{L}_t^{(k)}$ joining the camera projection center \mathbf{C}_t and a point $\mathbf{p} = \mathbf{C}_t + \mathbf{r}_t^{(k)}$ along the ray:

$$\mathbf{L}_t^{(k)} = \begin{bmatrix} \mathbf{L}_h \\ \mathbf{L}_0 \end{bmatrix} = \begin{bmatrix} \mathbf{C}_t - \mathbf{p} \\ \mathbf{C}_t \times \mathbf{p} \end{bmatrix} \quad (10)$$

From $\mathbf{L}_t^{(k)}$, we compute the transposed Plücker matrix

$$\mathbf{\Gamma}^T(\mathbf{L}_t^{(k)}) = \begin{bmatrix} \mathcal{S}(\mathbf{L}_0) & \mathbf{L}_h \\ -\mathbf{L}_h^T & 0 \end{bmatrix}, \quad (11)$$

where $\mathcal{S}(\mathbf{L}_0)$ is the skew symmetric matrix corresponding to \mathbf{L}_0 . Finally, we obtain $\mathbf{V}_t^{(k)}$ as

$$\mathbf{V}_t^{(k)} = \mathbf{\Gamma}^T(\mathbf{L}_t^{(k)})\mathcal{A}_t. \quad (12)$$

We repeat this procedure for the sigma points from each mean and covariance matrix of the same region in every image. In this way, we can quickly estimate the approximate 3D location of the change without computing a dense reconstruction of the scene, see Fig. 6b. The mean and the covariance of the position of these points represent the 3D area where the change occurs, see Fig. 6c.

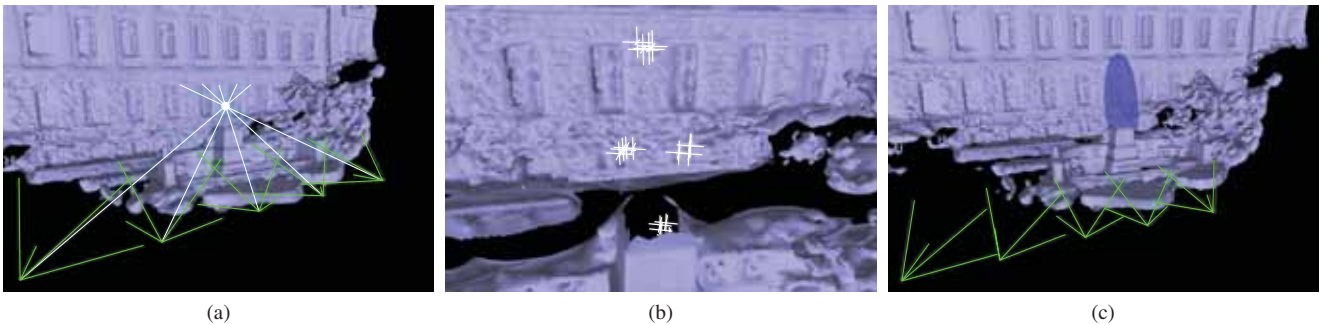


Fig. 6: (a) Example triangulation with 5 images. The white lines are the backprojected rays and the white point represent the triangulated point. (b) Sigma points projected in 3D. (c) The result of our algorithm, i.e. the 3D region where the change is. (best viewed in color)

IV. EXPERIMENTAL EVALUATION

The focus of this work is a comparably fast approach to identify changes in a previously obtained 3D model using a sequence of new images. Thus, our experiments are designed to show the performance of our approach and to support the two central claims that we made in the beginning of the paper, i.e. that our method: (i) can localize changes in the environment using a 3D model obtained in the past and a sequence of new keyframe images, and (ii) can be executed fast enough to run on an exploring robot, i.e. the average execution time should be in the order in which the sequence is recorded, here in the order of a few seconds for around 5 keyframe images.

We perform the evaluations on own datasets, which we publicly share including the 3D models at <http://www.ipb.uni-bonn.de/data/changedetection2017/>, as well as the dataset used by Taneja et al. [13], which can be obtained from: <https://cvg.ethz.ch/research/change-detection/Datasets/Structure.zip>. Throughout all experiments, we use a sequence of $n = 5$ images and for each image of the sequence, we compute the inconsistencies with $m = 4$, i.e. for these sequences all the neighboring images. We found out that using higher values of m does not improve the results substantially and that is why we used this value on all the experiments.

A. Change Identification

The first experiment is designed to illustrate the capability of our approach to localize a change in 3D given a model and a small sequence of images. Fig. 7 depicts the results of the algorithm on 4 different datasets. In all our tests, the localized 3D regions reflect the actual position of the changes. This information can allow an exploring or mapping robot to inspect the changed regions in more detail and collect more observations to update the previously built model. Note that the exploration itself is not part of this work but this works enables it.

The "Playground" dataset shown in Fig. 7c is particularly challenging. In this dataset, the house, which is not present in the model, is composed by separate wooden pieces, each one in a different color. Our algorithm is able to correctly identify the lamp and the bar as changes, but recognizes the house

TABLE I: Execution time for different datasets. The images in our datasets have resolution 1504×1000 pixels, the ones by Taneja et al. [13] have resolution 1072×712 pixels.

Dataset name	Execution time without uncertainty [s]	Execution time with uncertainty [s]
A/C Unit	4.598	10.54
Statue	7.486	12.571
Playground	8.529	13.989
Taneja et al. [13]	2.47	5.525
Average time	5.77 ± 2.758	10.656 ± 3.702

as multiple, separate changes. This does not constitute a real problem, but shows a possible limitation of our approach.

B. Execution time

The next experiment is designed to support the claim that our approach runs fast enough for processing on an exploring robot. We therefore measured the execution time of our approach on a common, lightweight laptop with an Intel Core i7 processor and an embedded Intel GPU.

Tab. I shows the average execution time needed to process sequences of 5 images from different datasets as well as the standard deviation, both with and without taking into account the uncertainty on the camera poses. The numbers support our second claim, namely that the computations can be executed fast enough for operation on an exploring robot. On our datasets, the whole process, taking into account the uncertainties, takes about 10 s, which is shorter than the time needed to record the 5 keyframe images. Even though the process is clearly not real-time in a strict sense, it is fast enough to be executed on a real robot at a low frequency to trigger exploration or additional mapping actions.

The computation time is influenced by both the number of images as well as their resolution. This is evident from our test on the dataset by Taneja et al., which took approximately half of the time for processing images with a lower resolution.

C. Comparison to an Existing Approach

Finally, we want to briefly compare our results with those obtained by Tanjea et al. [13]. The comparison is done based on the dataset that they provide and report on (Fig. 7d). Their approach uses a computationally expensive graph cut labeling on a 3D voxelization of the scene. Their method typically provides a more accurate estimate of the region

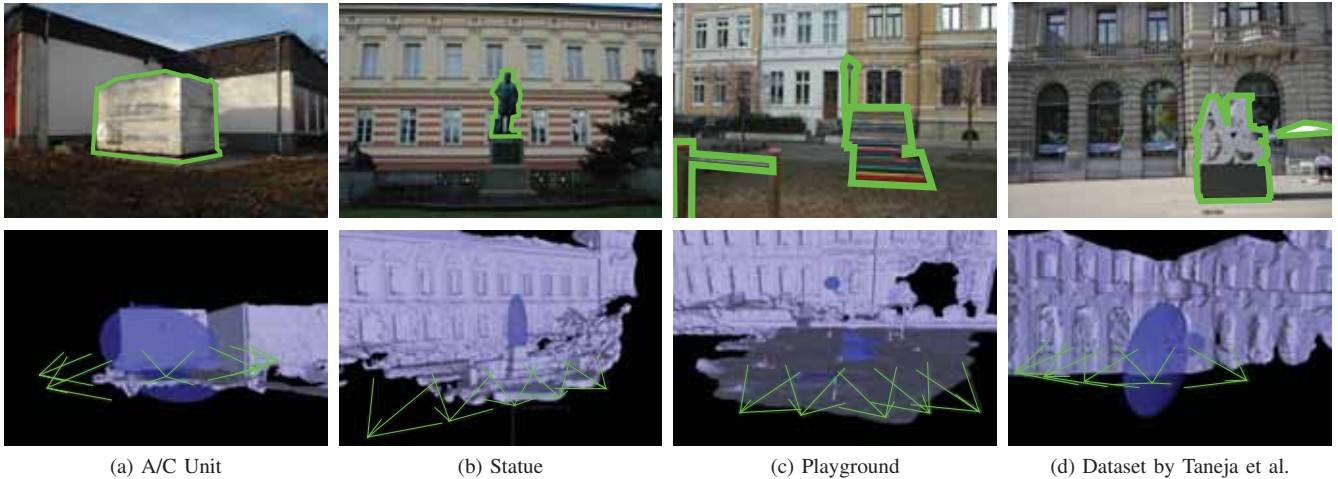


Fig. 7: Results of our experiments on 4 different datasets. For each dataset, the top image shows the changes (here manually marked in green), while the bottom image shows the 3D region, identified by our algorithm, where the changes are. (best viewed in color)

of change (in the order of $25 \times 25 \times 25 \text{ cm}^3$ voxels) than our estimate using the mean and covariance. The disadvantage of their method, however, is the computational demands as they require computation times in the order of 1 min per region, whereas we can process the same dataset in about 5 seconds. Thus, for most robotics applications, where an online feedback is expected, our approach is better suited.

To summarize, our evaluation suggests that our method can estimate the 3D localization of changes in the environment. At the same time, the algorithm is fast enough to be used by an exploring robot to focus on the areas that have changed. Thus, we supported all our claims made in the introduction with this experimental evaluation.

V. CONCLUSION

In this paper, we presented a novel approach to identify geometric changes between the current state of the environment and a previously built 3D model using a short sequence of images. Our approach operates by identifying the changes in the images by reprojecting them onto each other, passing through the 3D model. We eliminate the ambiguities about possible changes by combining the inconsistencies from multiple pairs of images. We are then able to estimate the locations of changes in 3D and identify the changed region through a mean 3D point and a covariance matrix. The computational time of the whole process using multiple images is in the order of seconds. We implemented and evaluated our approach on different datasets. The experiments suggest that our method can correctly identify the changes in the environment with only 5 images and a total computational time of around 10s, which make the algorithm suitable for running on mobile robots.

As future work, we plan to conduct more effective tests of our method on different types of datasets and extend the quantitative comparisons.

ACKNOWLEDGMENTS

We thank Johannes Schneider and Jens Behley for the fruitful discussions and valuable help during the realization

of our approach. We furthermore thank Taneja et al. for sharing their dataset from Zurich.

REFERENCES

- [1] P.F. Alcantarilla, S. Stent, G. Ros, R. Arroyo, and R. Gherardi. Streetview change detection with deconvolutional networks. In *Proc. of Robotics: Science and Systems (RSS)*, 2016.
- [2] T. Caselitz, B. Steder, M. Ruhnke, and W. Burgard. Monocular camera localization in 3d lidar maps. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 1926–1931. IEEE, 2016.
- [3] I. Eden and D.B. Cooper. Using 3d line segments for robust and efficient change detection from multiple noisy images. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, pages 172–185, 2008.
- [4] M. Golparvar-Fard, F. Pena-Mora, and S. Savarese. Monitoring changes of 3d building elements from unordered photo collections. In *Proc. of the Int. Conf. on Computer Vision (ICCV) Workshops*, pages 249–256, 2011.
- [5] S.J. Julier and J.K. Uhlmann. A New Extension of the Kalman Filter to Nonlinear Systems. *Proc. of the SPIE Conf. on Reconnaissance and Electronic Warfare System*, 3068:182–193, 1997.
- [6] T. Pollard and J.L. Mundy. Change detection in a 3-d world. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1–6, 2007.
- [7] R. Qin and A. Gruen. 3D change detection at street level using mobile laser scanning point clouds and terrestrial images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 90:23–35, 2014.
- [8] R. Qin, J. Tian, and P. Reinartz. 3D change detection – Approaches and applications. *ISPRS Journal of Photogrammetry and Remote Sensing*, 122:41–56, 2016.
- [9] R.J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam. Image change detection algorithms: a systematic survey. *IEEE Transaction on Image Processing*, 14(3):294–307, 2005.
- [10] K. Sakurada, T. Okatani, and K. Deguchi. Detecting changes in 3d structure of a scene from multi-view images captured by a vehicle-mounted camera. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 137–144, 2013.
- [11] J. Schneider, C. Eling, L. Klingbeil, H. Kuhlmann, W. Förstner, and C. Stachniss. Fast and effective online pose estimation and mapping for uavs. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 4784–4791, 2016.
- [12] S. Suzuki and K. Abe. Topological structural analysis of digitized binary images by border following. *Computer vision, graphics, and image processing*, 30(1):32–46, 1985.
- [13] A. Taneja, L. Ballan, and M. Pollefeys. Image based detection of geometric changes in urban environments. In *Proc. of the Int. Conf. on Computer Vision (ICCV)*, pages 2336–2343, 2011.
- [14] A.O. Ulusoy and J.L. Mundy. Image-based 4-d reconstruction using 3-d change detection. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, pages 31–45, 2014.

A Partitioned Approach for Efficient Graph-Based Place Recognition

Mattia G. Gollub Renaud Dubé Hannes Sommer Igor Gilitschenski Roland Siegwart*

Abstract—Place recognition is a crucial capability of autonomous vehicles that is commonly approached by identifying keypoint correspondences which are geometrically consistent. This geometric verification process can be computationally expensive when working with 3D data and with increasing number of candidates and outliers. In this work, we propose a technique for performing 3D geometric verification efficiently by taking advantage of the sparsity of the problem. Exploiting the relatively small size of the area around the vehicle, the reference map is first subdivided in partitions, and geometric verifications are only performed across relevant partitions, guaranteeing the sparseness of the resulting consistency graph. A maximum clique detection algorithm is finally applied for finding the inliers and the associated 3D transformation, taking advantage of the low degeneracy of the graph. Through experiments in urban driving scenarios, we show that our method outperforms a state of the art method both asymptotically and in practice.

I. INTRODUCTION

Efficient and reliable place recognition is one important challenge for enabling fully autonomous driving. With considerable changes in illumination occurring in driving scenarios, it is interesting to consider geometric information for performing place recognition. Autonomous vehicles are therefore often equipped with 3D time of flight sensors which permit a precise estimation of the road environment through the generation of point cloud maps.

One standard approach for recognizing places in 3D point cloud data is to compare a *local map* characterizing the vicinity of the vehicle, to a *target map* representing the full environment. This comparison can be done by extracting different basis elements such as keypoints[1], objects[2], shapes[3] or segments[4]. Place recognition is then performed by identifying correspondences between these basis elements and by verifying these correspondences for geometric consistency. This final 3D geometric verification step can be computationally expensive when working with large maps and with increasing number of correspondence candidates and outliers.

In this work, we formulate the problem of geometric verification as identifying a maximum clique in a *consistency graph* where edges connect correspondences that are geometrically consistent. A simplified example of a consistency graph is illustrated in Fig. 1c. We propose to perform the geometric verification by exploiting two important characteristics of the problem. First, we take advantage of the

*Authors are with the Autonomous Systems Lab, ETH, Zurich {gollubm, rdube, sommerh, igilitschenski, rsiegwart}@ethz.ch.

This work was supported by the European Union’s Seventh Framework Programme for research, technological development and demonstration under the TRADR project No. FP7-ICT-609763.

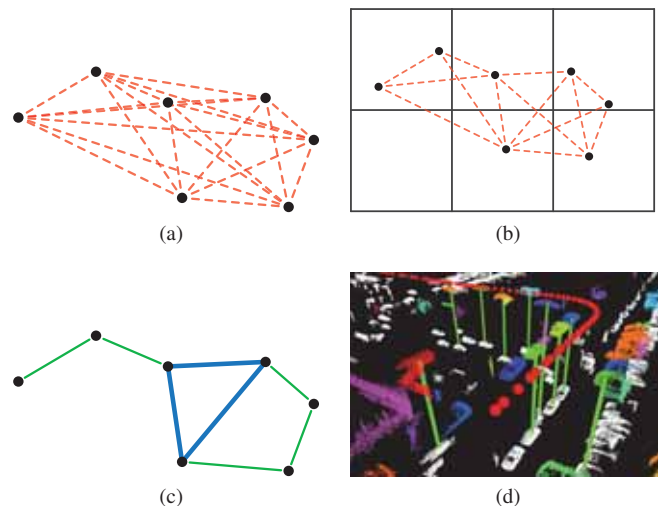


Fig. 1: Steps of the recognition process with minimum geometrically consistent set size $T = 3$: (a) Current approaches need to test all possible correspondence pairs for consistency. Nodes represent the correspondences, while the tested pairs are shown as edges. (b) Our partitioning approach allows to drastically reduce the number of consistency tests. (c) The consistency graph obtained with our method and the recognized T -clique (in blue) of geometrically consistent correspondences. (d) Example of a successful recognition in a urban driving scenario experiment. Correspondences that constitute the maximum consistent set are indicated in green.

significant difference in size between the local and target maps by subdividing the latter into partitions and by performing geometric verifications only across relevant partitions. This approach not only effectively reduces the number of consistency tests but also guarantees the sparseness of the resulting consistency graph. In a second step, we exploit this sparsity by leveraging an efficient algorithm for detecting maximum cliques in sparse graphs.

The proposed approach is compared to a state of the art baseline method in urban driving scenario experiments which demonstrate that our approach is more efficient. We show through a simple example that the baseline method can return sub-optimal solutions whereas our method always identifies a maximum clique. A derivation of the complexity of our method is presented, demonstrating that it scales linearly with the size of the target map.

To summarize, this paper presents the following contributions:

- A novel partition-based algorithm for efficiently identifying maximum geometrically consistent sets of correspondences.
- An asymptotic complexity analysis of the proposed method.

- An evaluation of the proposed method based on experiments in urban driving scenarios.

The remainder of the paper is structured as follows: Section II provides an overview of the related work in the field of geometric verification for place recognition. Section III describes our partition-based approach to place recognition. The approach is evaluated in Section IV, and Section V finally concludes with a short discussion.

II. RELATED WORK

Numerous methods have been developed for performing global registration of partially overlapping 3D objects [5–7]. Unfortunately such methods cannot always be applied for performing place recognition in real-time, as the target map can be many orders of magnitude bigger than the local map and as the matching process can produce a high fraction of false correspondences.

Chen and Bhanu [8] propose to filter these outliers by clustering correspondences into geometrically consistent sets. Two correspondences c_i and c_j are called *pairwise geometrically consistent* if the difference of the Euclidean distance between the keypoints in the local map and in the target map is below a threshold ϵ , i.e. if

$$|d_l(c_i, c_j) - d_t(c_i, c_j)| \leq \epsilon \quad (1)$$

where $d_l(c_i, c_j)$ and $d_t(c_i, c_j)$ are the keypoint distances in the local map and in the target map respectively.

This idea is employed in the *recognition* module of the Point Cloud Library (PCL) [9] where geometric consistencies are determined using a brute-force approach in which all possible correspondence pairs are checked for consistencies. This has an asymptotic complexity of $O(n^3)$, where n is the number of correspondences. This approach is well suited for scenarios with a low amount of candidates and was employed in our previous work on segment-based place recognition [4]. However, it does not scale well to cases with a large number of candidates and outliers, eg. when doing real-time place recognition in a large target map. We consider this method as our baseline in the experiments of Section IV.

Strategies for efficiently reducing the number of correspondence pairs have been proposed for stereo images and image retrieval. Ayache and Faverjon [10] describe a partitioning scheme for efficiently finding neighbor segments in stereo images. The SCRAMSAC method [11] performs RANSAC only on *spatially consistent* correspondences, i.e. correspondences that have a minimum fraction of matching neighbor features in both images. Both methods rely on assumptions about the disparity between images, thus their accuracy is influenced by the presence of high disparity and strong variation in viewing angles.

Graphical models have successfully been employed in the analogous task of recognizing places based on camera images [12, 13] In the context of geometry-based place recognition, Fernandez-Moral et al. [3] propose to leverage graphical models for executing the geometric verification. An interpretation tree is used to match the local and target graphs where vertices represent planes and edges represent

geometric relationships between these planes. Finman et al. [14] also propose a similar graph-matching strategy where vertices represent objects instead of planes. Contrastingly in our approach, vertices and edges respectively represent correspondences and geometric consistencies. Place recognition is then executed by extracting a maximum clique.

III. METHOD

This section presents our approach for performing geometric verification of correspondences in order to recognize places in 3D maps. We treat this task as a graph problem with the goal of identifying a *maximum geometrically consistent set* which is a set of maximum size consisting of correspondences that are all pairwise geometrically consistent. Pairwise geometrical consistency relationships are encoded in an undirected graph $G = (V, E)$ where $V = \{c_i\}$ is the set of correspondences c_i and $E = \{e_{ij}\}$ is the set of undirected edges e_{ij} connecting all consistent pairs of correspondences (c_i, c_j) . Once the graph is constructed, identifying a maximum geometrically consistent set is equivalent to finding a maximum clique of G . An example of a consistency graph with its maximum consistent set is given in Fig. 1c.

A. Partition-based consistency graph construction

The naive approach for identifying all pairwise consistencies is testing all possible correspondence pairs. Here we take advantage of our knowledge about the sizes of the target and local maps to reduce the number of tests.

In many recognition applications, like place recognition as in our case, the local map is significantly smaller than the target map. Taking advantage of this information we can define a criterion for reducing the amount of consistency tests a priori. Let ϵ be the tolerance for geometric consistency and b be the diameter of the bounding sphere of the local map keypoints, a pair of correspondences c_i and c_j can then only be consistent if

$$d_t(c_i, c_j) \leq b + \epsilon \quad (2)$$

Urban driving scenarios usually extend along the earth's surface only (x and y coordinates), but not vertically. Without loss of performance or correctness, we simplify the problem and consider only a bounding cylinder of diameter b . The new relaxed criterion becomes

$$d_t^{2D}(c_i, c_j) \leq b + \epsilon \quad (3)$$

where $d_t^{2D}(c_i, c_j)$ computes the distance between the target map keypoints of c_i and c_j projected on the xy -plane¹.

We create a 2D grid partitioning of the correspondences according to the position of their keypoints in the target map, where each partition $p_{u,v}$ has a square base with a side length of $b + \epsilon$ and infinite height. An example of 2D grid partitioning is illustrated in Fig. 1b. Each correspondence c_i is assigned to its partition $P(c_i) = p_{u,v}$,

$$u = \left\lfloor \frac{k_t(c_i).x - o.x}{b + \epsilon} \right\rfloor, v = \left\lfloor \frac{k_t(c_i).y - o.y}{b + \epsilon} \right\rfloor \quad (4)$$

¹The z-axis is assumed to be roughly gravity aligned.

where $k_t(c_i)$ is the keypoint of c_i in the target map and o is the origin of the grid. A good choice of o is the componentwise minimum of all $k_t(c_i)$.

With the chosen grid size $b + \epsilon$, it is guaranteed that the bounding cylinder of the model is always contained in a squared group of four adjacent partitions. Thus geometric consistency tests are necessary only on a set of candidate pairs of correspondences that is much smaller than $V \times V$:

$$\{(c_i, c_j) \in V \times V \mid i < j \wedge \exists u, v : c_i \in p_{u,v} \wedge c_j \in \mathcal{N}(u, v)\} \quad (5)$$

where $\mathcal{N}(u, v) := \bigcup_{l,m \in \{-1,0,1\}} p_{u+l, v+m}$. i.e. for a given $c_i \in V$ only correspondences in the partition of c_i and the 8-neighbor-partitions need to be tested. Since consistency is a symmetric property, each pair is tested only once, i.e. if $i < j$. Other pairs of correspondences are ignored, since they cannot be consistent. The consistency graph is constructed as an adjacency list and contains all the geometrically consistent correspondence pairs (c_i, c_j) as edges. Fig. 1a shows a set of 7 correspondences where all 21 pairs are tested for consistency. Applying this partitioning strategy reduces the number of tests to 14 (Fig. 1b).

The same approach can be used for cases where the environment extends into the third dimension. In this case, assuming that the local map is bounded by a sphere of diameter b , a 3D grid of cubes with size $b + \epsilon$ is used and consistency tests are performed over the 26-neighborhood of each partition.

B. Maximum clique detection

We consider a recognition to be successful in case the size of the detected maximum geometrically consistent set is greater than or equal to a threshold parameter T . Thus we need to identify a maximum k -clique with $k \geq T$.

A second advantage of enforcing the partitioning constraints in situations where the local map is significantly smaller than the target map is that the sparseness of the consistency graph is guaranteed. With this knowledge we can rely on a class of k -clique detection algorithms that visit the graph in *degeneracy order* to find the maximum geometrically consistent set. A graph G can be characterized by its *degeneracy* (or k -core number) d , which is the smallest number so that every subgraph of G contains a vertex with degree $\leq d$. Each graph has a *degeneracy ordering*, an order in which each vertex has at most d vertices that come later in the ordering.

We use a generalization of the maximal clique listing approach described in [15, 16] to find a maximum clique of G . An outline of the algorithm is presented in Algorithm 1. First, we use the bucket sort algorithm for sorting the vertices in increasing degree order. This is described in detail in [17] and can be performed in $O(|V|)$. Then we iterate on the vertices according to the found order. At each step, the function `CLIQUE` is called using the current vertex of minimum degree as input. If the function returns a clique that is bigger than the current maximum clique max_clique , the new maximum clique is stored. The vertex is then removed

from the graph together with its incident edges and the vertex ordering is updated in $O(d)$. The resulting *degeneracy ordering* in the visit of the vertices guarantees that at each step v has at most d neighbors, bounding the computational load on `CLIQUE`.

`CLIQUE` can be any function that returns the biggest clique $C \subseteq G$ such that $|C| \geq T$ and $v \in C$ (or the empty set, if such a clique does not exist). Instances of this function can be any clique detection approach like the branch-and-bound method presented in [18] or successive refinements like [19]. For simplicity, in this work we use a method based on heavy pruning strategies [20]. This algorithm builds a subgraph containing v and its neighbors and recursively visits its subgraphs to determine if a clique of the required size can be built. Vertex degrees are constantly checked to discard fruitless candidates as soon as possible in the recursion.

Algorithm 1 Outline of the algorithm for maximum clique detection. Iterating over the vertices in degeneracy order guarantees that v always has at most d neighbors.

```

1: procedure MAXIMUMCLIQUE( $G, T$ )
2:    $max\_clique \leftarrow \emptyset$ 
3:    $sorted\_vertices \leftarrow \text{SORTBYDEGREE}(G)$ 
4:   while  $|sorted\_vertices| \neq 0$  do
5:      $v \leftarrow sorted\_vertices[0]$ 
6:      $C = \text{CLIQUE}(v, G, T)$ 
7:     if  $|C| \neq 0$  then
8:        $max\_clique \leftarrow C$ 
9:        $T \leftarrow |C| + 1$ 
10:    end if
11:    REMOVEVERTEX( $G, v$ )
12:    UPDATEORDER( $G, sorted\_vertices$ )
13:  end while
14:  return  $max\_clique$ 
15: end procedure

```

C. Recognition

The condition on the size of the maximum geometrically consistent set can be removed in case it is safe to assume that a true recognition always exists. This is however not always possible, e.g. in loop-closure detection applications. Once an acceptable maximum clique is identified, the relative transformation between the local and target maps can be found using any rigid transformation estimation method. Here we use the least-squares approach described in [21].

D. Asymptotic complexity and scaling

In the consistency graph $G = (V, E)$, let the parameters $n = |V|$ be the number of correspondences and d be the degeneracy of the graph. Assuming that outliers are uniformly distributed in the target map and are present in a high ratio over inliers, we can say that n is proportional to the size of the target map. The average number of correspondences per partition n_p and d are considered constant as they depend on fixed parameters like the density

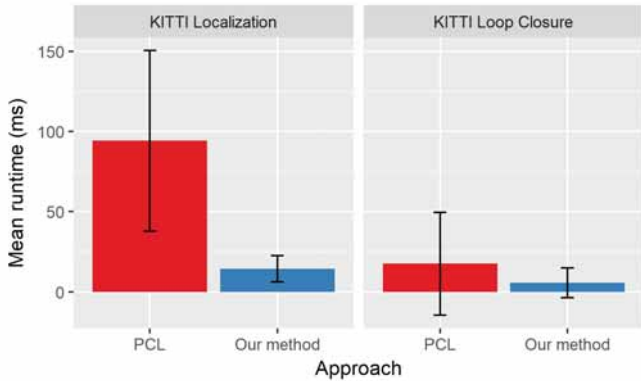


Fig. 2: Runtime comparisons between the reference PCL implementation and our new method. Error bars show the standard deviation of the measurements and are caused by the different amount of candidate correspondences found while the vehicle moves.

of objects in the environment, the size of the local map and the correspondence matching strategy.

Partitioning the correspondences is trivial and can be done in $O(n)$. All the necessary consistency tests can be performed in $O(n_p n)$ since each of the $\frac{n}{n_p}$ partitions requires $O(n_p^2)$ tests. The bucket sort is done in $O(n)$. Since every vertex visited in the loop has at most d neighbors, each of the n calls to CLIQUE has a worst case complexity of $O(d!)$, thus the complexity of the code in Algorithm 1 is $O(d!n)$. Although the $d!$ term may appear like an important bottleneck, any reasonable implementation for CLIQUE usually performs better. As shown in Table I, maximum clique detection brings a minimal contribution to the total runtime. The estimation of the 3D transformation scales linearly with the number of elements in the geometrically consistent set, thus it is bound by $O(d)$.

The total complexity of our method is $O(n(d! + n_p))$. Under the homogeneity assumption, since d and n_p are parameters that do not depend on the target map’s size, we can conclude that the performance of our algorithm scales linearly with the size of the target map.

IV. EXPERIMENTS

We tested our method in the place recognition module of our online LiDAR-based localization and mapping system described in [22]. We compare our new method with our current baseline approach implemented in the PCL (`pcl::GeometricConsistencyGrouping`). Both implementations are benchmarked in two different configurations:

- *Localization*: An autonomous vehicle drives in a known urban scenario, continuously trying to localize inside a static target map.
- *Loop-closure*: The vehicle explores an unknown urban scenario, continuously updating a dynamic target map and trying to detect loop-closures.

Each configuration uses a different dataset of the KITTI collection [23], both datasets have similar sizes. At each call we measure the runtime of the recognition. We use $\epsilon = 0.4m$

as consistency threshold and $T = 6$ as minimum consistent set size.

Figure 2 shows a runtime comparison between our method and the PCL implementation. In the localization test, our new method is 6.57 times faster than the reference method. This improvement comes from two main factors. First, only 32.6% of the consistency tests are performed. In addition, our partitioned approach enables better cache locality when accessing correspondence keypoints, increasing the performance of the consistency test functions.

In the loop-closure setting the speedup decreases to 2.05x. This is due to the fact that during the first part of the experiment the target map is relatively small and prevents partitioning from being effective. In general we observe that partitioning successfully accelerates the recognition process by reducing the number of tested correspondence pairs and our method scales better as the size of the map increases. In future work the method should be tested with bigger maps in order to prove the theorized linear scaling of the method (see Section III-D).

Step	Mean runtime (Localization)	Mean runtime (Loop-Closure)
Partitioning	$0.06 \pm 0.01ms$	$0.03 \pm 0.02ms$
Graph construction	$13.86 \pm 7.95ms$	$8.09 \pm 9.70ms$
Max clique detection	$0.13 \pm 0.08ms$	$0.17 \pm 0.23ms$
Transformation estimation	$< 0.01ms$	$< 0.01ms$
Total (Our method)	$14.33 \pm 8.16ms$	$8.52 \pm 9.98ms$
Total (PCL)	$94.23 \pm 56.28ms$	$17.49 \pm 32.10ms$
Speedup	6.57x	2.05x

TABLE I: Runtimes and speedups of the different steps of our algorithm. Clearly the most important factor is the construction of the consistency graph.

Table I shows the time needed by each step of our method. While partitioning quickly and effectively reduces the number of consistency tests required, the construction of the consistency graph is still the most expensive operation. Detecting a maximum clique is a relatively cheap task thanks to the sparseness of the graph and to the traversal strategy that bounds the complexity of the search.

We performed additional tests to determine the quality of the resulting recognition in the two methods. As metric for the quality we use the size of the identified maximum consistent set. Our method uses an exact algorithm, and is always able to find a consistent set of maximum size. On the other side, the PCL implementation uses a greedy approach and occasionally fails to detect the optimal solution. Fig. 3 shows a tested pathological case: depending on the order of the input correspondences the greedy approach can detect a maximum (3a, in blue) or a maximal (3b, in red) consistent set. Our method detects the maximum clique in both cases. Note that the PCL implementation does not explicitly build a graph, but detects the same pairwise consistency, thus we can use the same visualization.

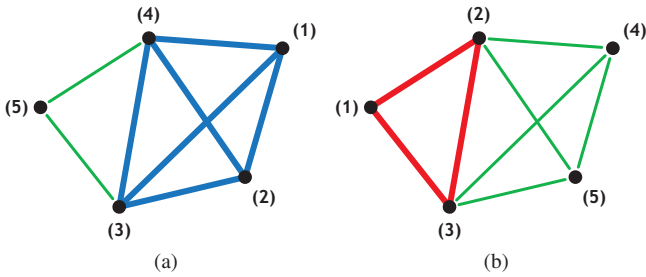


Fig. 3: Detection of consistent sets with $T = 3$ in two graphs based on the same correspondences stored in different orders, indicated by the numbers. The PCL implementation identifies the blue biggest cluster in (a), but can only find a suboptimal solution (in red) if the correspondences are stored differently (b). Our implementation uses exact k -clique detection and finds a maximum consistent set in all cases.

V. CONCLUSION

In this work, we presented a new graph-based method for place recognition in 3D point clouds that improves on our reference baseline in terms of performance. We stated the recognition task as a graph problem and used a novel partitioning approach to significantly reduce the number of consistency tests required. Taking advantage of the sparseness of the consistency graph, we use a clique detection algorithm to identify the biggest set of geometrically consistent correspondences quickly and exactly.

Experiments in urban driving scenarios show that our method performs better than the greedy approach of our selected baseline. We theoretically demonstrate that the runtime of this algorithm scales linearly with the size of the map, enabling fast localization in large environments. Benchmark results show that the present bottleneck of the method is the construction of the consistency graph. In future work we will explore other approaches to further accelerate the construction task. Moreover we would like to evaluate our method on bigger scenes to prove the theorized linear scaling behavior.

REFERENCES

- [1] M. Bosse and R. Zlot, "Place recognition using keypoint voting in large 3D lidar datasets," in *IEEE Int. Conf. on Robotics and Automation*, 2013, pp. 2677–2684.
- [2] R. F. Salas-Moreno, R. A. Newcombe *et al.*, "Slam++: Simultaneous localisation and mapping at the level of objects," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.
- [3] E. Fernandez-Moral, W. Mayol-Cuevas *et al.*, "Fast place recognition with plane-based maps," in *IEEE Int. Conf. on Robotics and Automation*, 2013, pp. 2719–2724.
- [4] R. Dubé, D. Dugas *et al.*, "Segmatch: Segment based place recognition in 3d point clouds," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5266–5272.
- [5] Q.-Y. Zhou, J. Park, and V. Koltun, "Fast global registration," in *European Conference on Computer Vision*. Springer, 2016, pp. 766–782.
- [6] A. S. Mian, M. Bennamoun, and R. A. Owens, "Automatic correspondence for 3d modeling: an extensive review," *International Journal of Shape Modeling*, vol. 11, no. 02, pp. 253–291, 2005.
- [7] Y. Guo, M. Bennamoun *et al.*, "3d object recognition in cluttered scenes with local surface features: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2270–2287, 2014.
- [8] H. Chen and B. Bhanu, "3d free-form object recognition in range images using local surface patches," *Pattern Recognition Letters*, vol. 28, no. 10, pp. 1252–1262, 2007.
- [9] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1–4.
- [10] N. Ayache and B. Faverjon, "Efficient registration of stereo images by matching graph descriptions of edge segments," *International Journal of Computer Vision*, vol. 1, no. 2, pp. 107–131, 1987.
- [11] T. Sattler, B. Leibe, and L. Kobbelt, "Scramsac: Improving ransac's efficiency with a spatial consistency filter," in *Computer vision, 2009 IEEE 12th international conference on*. IEEE, 2009, pp. 2090–2097.
- [12] E. Stumm, C. Mei *et al.*, "Location graphs for visual place recognition," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5475–5480.
- [13] S. Cascianelli, G. Costante *et al.*, "A robust semi-semantic approach for visual localization in urban environment," in *Smart Cities Conference (ISC2), 2016 IEEE International*. IEEE, 2016, pp. 1–6.
- [14] R. Finman, L. Paull, and J. J. Leonard, "Toward object-based place recognition in dense rgb-d maps," in *ICRA Workshop Visual Place Recognition in Changing Environments*, Seattle, WA, 2015.
- [15] D. Eppstein, M. Löffler, and D. Strash, "Listing all maximal cliques in sparse graphs in near-optimal time," *Algorithms and computation*, pp. 403–414, 2010.
- [16] D. Eppstein and D. Strash, "Listing all maximal cliques in large sparse real-world graphs," *Experimental Algorithms*, pp. 364–375, 2011.
- [17] V. Batagelj and M. Zaversnik, "An $o(m)$ algorithm for cores decomposition of networks," 2003.
- [18] C. Bron and J. Kerbosch, "Algorithm 457: finding all cliques of an undirected graph," *Communications of the ACM*, vol. 16, no. 9, pp. 575–577, 1973.
- [19] E. Tomita, A. Tanaka, and H. Takahashi, "The worst-case time complexity for generating all maximal cliques and computational experiments," *Theoretical Computer Science*, vol. 363, no. 1, pp. 28–42, 2006.
- [20] B. Pattabiraman, M. Patwary *et al.*, "Fast algorithms for the maximum clique problem on massive sparse graphs," *arXiv preprint arXiv:1209.5818*, 2012.
- [21] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 13, no. 4, pp. 376–380, 1991.
- [22] R. Dubé, A. Gawel *et al.*, "An online multi-robot slam system for 3d lidars," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2017.
- [23] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361.