



**2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010**

ICRA10 International workshop on Robotics and Intelligent Transportation System

**Full Day Workshop
May 7th 2010, Anchorage, Alaska**

<http://www.lasmea.univ-bpclermont.fr/Control/workshopICRA10/RITS10.html>

Organizers

**Christian Laugier (INRIA, France),
Ming Lin (University of North Carolina, USA)
Philippe Martinet (IFMA and LASMEA, France),
Urbano Nunes (ISR, Portugal)**

Contact

Professor Philippe Martinet
IFMA, Campus des Cezeaux, 63175 Aubiere, France
LASMEA-CNRS Laboratory, Blaise Pascal University
Campus des Cezeaux, 63177 Aubiere, Cedex, France
Phone: +33 473 407 653, Sec : +33 473 407 261, Fax : +33 473 407 262
Email: martinet@lasmea.univ-bpclermont.fr
Home page: <http://www.lasmea.univ-bpclermont.fr/Personnel/Philippe.Martinet>



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Foreword

Autonomous driving and navigation is a major research issue which would affect our lives in near future. 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. These new technologies can be applied efficiently for other application field like unmanned vehicles, wheelchair or assistance mobile robot. Technologies related to this area, such as autonomous outdoor vehicles, achievements, challenges and open questions would be presented, including the following topics: Object detection, tracking and classification, Collision prediction and avoidance, Environment perception, vehicle localization and autonomous navigation, Real-time perception and sensor fusion, SLAM in dynamic environments, Real-time motion planning in dynamic environments, 3D Modelling and reconstruction, Human-Robot Interaction, Behavior modeling and learning, Robust sensor-based 3D reconstruction, Modeling and Control of mobile robot, Cooperation and communications, Multi-agent based architectures, Cooperative unmanned vehicles.

Previously, four workshops were organized in the same field. The 1st edition [PPNIV'07](#) of this workshop was held in Roma during ICRA'07 (around 60 attendees), and the second [PPNIV'08](#) in Nice during IROS'08 (more than 90 registered people), the third edition [SNODE'09](#) in Kobe during ICRA'09 (around 70 attendees), and the last one [PPNIV'09](#) was organized in the next IROS'09 in Saint-Louis. A special issue in IEEE Transaction on ITS, mainly focused on Car and ITS applications, has been published last September 2009. Previous editions were focused mainly on the use of one vehicle; the 5th edition will extend the topics to control, traffic and multi-vehicle.

This workshop is composed with 4 invited talks and 12 selected papers (6 selected for oral presentation and 6 selected for interactive session. Five sessions has been organized:

- Session I: Perception & Localization
- Session II: Path Planning & Navigation Systems
- Session III: Human-Robot Interaction
- Session IV: Interactive Session
- Session V: Multi-robot Control and ITS

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, Ming Lin, Philippe Martinet and Urbano Nunes



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session I

Perception & Localization

- **Keynote speaker: Alberto Broggi (Parma University, Italy)**
Title: The VIAC Challenge: Setup of an Autonomous Vehicle for a 13,000 km Intercontinental Unmanned Drive
Co-Authors: Massimo Bertozzi, Luca Bombini, Alberto Broggi, and Paolo Grisleri
- **Title: Learning a Real-Time 3D Point Cloud Obstacle Discriminator via Bootstrapping**
Authors: Michael Samples and Michael R. James
- **Title: An Improved Flies Method for Stereo Vision: Application to Pedestrian Detection**
Authors: Hao Lee, Gwenaelle Toulminet, Fawzi Nashashibi



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session I

Keynote speaker: **Alberto Broggi (Parma University, Italy)**

The VIAC Challenge: Setup of an Autonomous Vehicle for a 13,000 km Intercontinental Unmanned Drive

Co-Authors: Massimo Bertozzi, Luca Bombini, Alberto Broggi, and Paolo Grisleri

Abstract : Autonomous vehicles have been demonstrated to be able to traverse the desert (the DARPA Grand Challenge, 2005), navigate downtown together with other traffic (the DARPA Urban Challenge, 2007), someone is even trying to emulate experienced drivers in extreme races,... In all these situations, however, the unmanned vehicles move within a semi-controlled environment. VisLab is now trying to push the unmanned vehicles technology to the limit and test their systems (both hardware and software) for a long time and in an extreme environment: on July 10, 2010, two autonomous vehicles will leave Italy and will drive for 13,000 km in Europe towards Moscow, then Russia, then Siberia, Kazakstan, then China, Mongolia, finally reaching Shanghai on October 10, 2010, after 3 months of autonomous driving. As a 'challenge into the challenge, VisLab selected electric vehicles, with the final aim of setting a new milestone in the history of robotics: goods will be transported from Italy to China on a ground trip with no driver, and without using a drop of conventional fuel. Not only these vehicles will be moving without any human intervention, but the driverless technology will be powered by solar energy thanks to a panel on the vehicle's roof. The talk will present the current state of the art and the major design challenges.

Biography: Prof. Broggi graduated in Electronic Engineering and got his Ph.D. in Information Technology at the University of Parma, Italy, in 1990 and 1994, respectively. From 1994 to 1998 he was an Assistant Professor at the Dipartimento di Ingegneria dell'Informazione of the University of Parma; from 1998 to 2001 he was Associate Professor of Artificial Intelligence at the Dipartimento di Informatica e Sistemistica (Vision Laboratory) of the University of Pavia, Italy. On November 2001 he joined again the Dipartimento di Ingegneria dell'Informazione of the University of Parma as an Associate Professor of Computer Engineering. In 2003 he got the recognition for Full Professorship by two distinct Universities and two years later became Full Professor at the University of Parma. He acted as Program Chair of the main conference in the field of intelligent vehicles (the IEEE Intelligent Vehicles Symposium 2000 in Detroit, MI) together with Jim Rillings (General Motors) and as General Chair of the same conference in 2004; from 2004 to 2008 he served the most important scientific Journal in the field of Intelligent Transportation Systems, the IEEE Trans on ITS, as Editor-in-Chief; he is now serving the IEEE Intelligent Transportation System Society as President. Prof. Broggi has been invited as keynote speaker in many different events worldwide to describe and discuss the current state of the art on intelligent vehicles and future trends in the field. He is the Founder and Director of the Artificial Vision and Intelligent Systems Lab and author of more than 150 publications on international scientific journals, book chapters, refereed conference proceedings, He is also President and CEO of VisLab srl, a spin-off company of the University of Parma, whose mission is the transfer of perception technologies to the industrial market. Prof. Broggi acted as Associate Editor for many international journals and is the Founding Editor of the regular Department on "Intelligent Transportation Systems" of IEEE Intelligent Systems Magazine, IEEE Computer Society, and the Founding Editor of the IEEE Intelligent Transportation Systems Council Newsletter. Alberto Broggi was awarded an ERC Advanced Grant in 2008 for his project OFAV (Open intelligent systems for Future Autonomous Vehicles).



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

The VIAC Challenge: Setup of an Autonomous Vehicle for a 13,000 km Intercontinental Unmanned Drive

Massimo Bertozzi, Luca Bombini, Alberto Broggi, and Paolo Grisleri
VisLab, University of Parma, ITALY – www.vislalab.it

Abstract—Autonomous vehicles have been demonstrated able to reach the end of a 220 miles off-road trail (in the DARPA Grand Challenge), and to negotiate traffic and obey traffic rules (in the DARPA Urban Challenge), but no one ever tested their capabilities on a long, intercontinental trip and stressed their systems for 3 months in a row on extreme conditions.

This invited paper presents the vehicles that will run the *VisLab Intercontinental Autonomous Challenge* and the design issues that are the base for the equipment of an autonomous vehicle that will have to drive itself without any human intervention on an intercontinental route for more than 13,000 km.

The challenge will take place from July 10, 2010 to Oct 10, 2010, therefore being currently under preparation, this paper focuses on the preparation issues and describes some important design choices.

I. INTRODUCTION

The World Expo 2010 will be held in Shanghai, China, May 1-Oct 31, 2010. It is the third most relevant worldwide event after the FIFA World Cup and the Olympic Games. The 2010 Expo theme is *better cities, better life*; therefore issues related to sustainable mobility are indeed central to the Expo, which will be a display of new ideas developed worldwide in this field.

The Expo will constitute a great opportunity to showcase new and innovative technologies in the field of intelligent mobility, especially urban mobility.

VisLab has been working for more than 15 years in the field of intelligent vehicles and participated in many worldwide events, like the DARPA Challenges. Many of VisLab's results are considered as worldwide milestones in the field of vehicular robotics, like the ARGO project (a passenger car that in 1998 drove for 2000+ km on Italian highways in automatic mode; 94% of the event was performed without human intervention), or the TerraMax vehicle.

TerraMax is an Oshkosh MTRV truck that VisLab equipped with sensing systems (primarily artificial vision) and that was able to reach the end of the DARPA Grand Challenge in 2005 (220 miles of off-road driving with no human intervention) and was qualified for the DARPA Urban Challenge in 2007 (6 hours of urban driving).

VisLab wants to set a new milestone in the domain of intelligent vehicles with a new initiative, completely conceived and sustained by VisLab: the idea is to demonstrate, through an extensive and impressive test, that the current technology

The work described in this paper has been partially funded by the Open intelligent systems for Future Autonomous Vehicles (OFAV) Project, by the European Research Council (ERC) within an Advanced Investigator Grant.



Fig. 1. The VisLab Intercontinental Autonomous Challenge route.

is mature enough for the deployment of non-polluting and no-oil based autonomous vehicles in real conditions.

II. THE CHALLENGE

The challenge, named *VisLab Intercontinental Autonomous Challenge* (VIAC), has a unique final goal: to design vehicles able to drive autonomously along a 13,000 km trip, with no human intervention. Although this goal is definitely very complex, VisLab is approaching this exciting endeavor together with additional innovative ideas. The vehicles will be electric and power to the electronic pilot will be delivered by solar panels. These additional requirements will help demonstrate that it is possible –although in a prototype version– to move goods between two continents with non-polluting vehicles powered by green energy and with virtually no human intervention. Some goods will be packed in Rome, some collected throughout the trip, and finally taken to Shanghai with virtually no impact on world's pollution.



Fig. 2. The VisLab autonomous vehicles before equipping them with sensors.

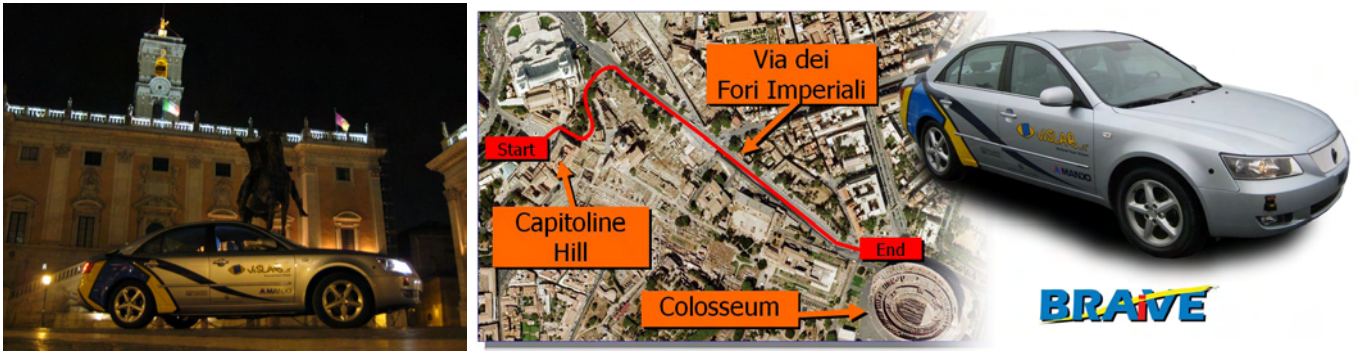


Fig. 3. BRAiVE (VisLab's latest driverless vehicle) in Rome, and the route of the demo.

A. The Route

The route will pass through different countries both in Europe and in Asia as depicted in figure 1. The main countries that will be traversed, besides few European countries, are Russia, Kazakhstan, and China, where the vehicles will spend much of the 3 months trip. Different roads and traffic conditions will be faced and the vehicles will have to deal with unexpected situations, providing ground for an excellent improvement of VisLab current ADAS systems.

B. Scientific Outcome

The Intercontinental Autonomous Challenge will be the first demonstration of autonomous driving on a route that is:

- Long: more than 13,000 km. This extensive test will allow a thorough test of the developed technologies;
- Extreme: different environments will be crossed to validate and stress the system in several different conditions.

C. The Expedition

The expedition will be composed of 4 autonomous vehicles plus support vehicles (4 Overland trucks including mechanic shop, storage, accommodation...). Other vehicles will also follow, mainly for live satellite broadcasting of the event. The complete trip will last three months.

III. AUTONOMOUS DRIVING

During the challenge two autonomous vehicles will be driving. Although the two vehicles will be exactly identical (same sensor suite and identical control system) they will have different goals:

- the first one will use the whole sensor suite (including expensive sensors) and will face a completely unknown environment;
- the second one will use a subset of sensors (only cheap ones) and will demonstrate 100% autonomy when coarse route information will be provided by the first vehicle.

A. The First Vehicle

The first vehicle will drive autonomously for most of the trip; it will conduct experimental tests on sensing, decision, and control subsystems, and will collect data throughout the whole trip. Although limited, human interventions will be needed to define the route and intervene in critical situations.

B. The Second Vehicle

The second vehicle will automatically follow the route defined by the preceding vehicle, requiring no human intervention (100% autonomous). It will be regarded as a readily exploitable vehicle, able to move on loosely predefined routes. At the end of the trip, its technology will be transferred to a set of vehicles to move in the inner part of Rome in the close future.

C. Technology Demonstration

During the trip, demonstrations will be performed in specific hot spots: autonomous vehicles will follow given routes, negotiating traffic, avoiding obstacles, and stopping when required. A first demonstration was given in Rome between the Campidoglio and the Colosseum in late October, which demonstrated autonomous driving in narrow roads, with pedestrians and traffic. The BRAiVE vehicle was used, which incorporates much of the technology installed on the electric vehicles that will on the road to China.

The Intercontinental Autonomous Challenge was officially announced by the Major of Rome in a press conference in Rome on October 29, 2009; after the presentation, the Major of Rome left the meeting on BRAiVE, VisLab's latest driverless car (www.braive.vislab.it).

IV. VEHICLE SETUP

The 4 electric vehicles are all equipped with the very same sensing and actuation technologies to optimize development time and help in case of failures.

A. The Sensing System

The vehicle sensing system is based on cameras and laser-scanners. 7 cameras are installed on the vehicle (5 forward and 2 backward looking), while 6 laserscanners with different characteristics and orientations are placed around the vehicle.

Figure 4 shows the vision sensors' placement and describe their use.

Each camera is connected at 400 Mbps to its specific processing unit in the trunk through a Firewire hub. Laserscanners are connected to the 100 Mbps ethernet switch. Each camera is capable of transmitting the Bayer/Raw image captured from the Micron MT9V022, 752×480 sensor through the IEEE 1394A bus; in this way all the color information available is transmitted at 1/3 of the bandwidth of a full RGB, 8 bit per channel image. Appropriate color reconstruction is needed on-board the processing unit. The shutter of each camera belonging to the same system is started by a 10Hz software generated common trigger signal, to ensure the images are taken at the very same time: error between the starting capture times is below a tenth of microseconds. The stitching system and the stereo system have separate triggers. A software plugin performs a very fast analysis of the incoming images from a camera and finds the best exposure parameters, like shutter and gain, to be applied to the camera to improve recognition rates. Another custom software plugin ensures the synchronization of the main exposure properties between the cameras to ensure that images feeding the same system are consistent and similar to each other. Other camera parameters like white balance and gamma are tuned at the camera boot time to be consistent.

The optical system has been tuned selecting a focal length of 4.5 mm for the Stereo Front cameras, and 3.5 mm for the Stereo Back and Stich. The forward and backward stereo vision systems locate obstacles and lane markings, while the 3-camera frontal system stitches the 3 images together to form a single panoramic view of the 180 degrees in front of the vehicle in order to locate the leader vehicle. The laserscanners are used to locate obstacles, the vehicle in front, and other traffic.

One multi-beam laserscanner unit is mounted in the center of the front bumper. This laser has four planes, a horizontal aperture of 85 degrees, a vertical aperture of 3.2 degrees, and measures distances in the range 0,3–80 m. It is set to produce data in free run mode (not triggered) at 12.5 Hz. It is positioned at 54 cm from the ground plane and the pitch orientation is a few degrees downward. Three mono-beam laserscanners create a perception plane all around the vehicle at a height of about 60 cm. This allows to detect most of the commercially available vehicles. These lasers are placed at about 60 cm from the ground plane, with the scanning plane parallel to the ground plane and with an orientation appropriate to cover the maximum area around the vehicle. Other two mono-beam laserscanners are placed at 173 cm from the ground plane, over the cabin with a pitch angle of 45 degrees downward to detect ditches and curbs in the forward looking direction. All lasers are set up to send raw data at a selected rate. Since most of them do not have a trigger input, the synchronization is obtained via software by selecting the last captured scan before a given captured set of images. The laserscanners are visible from all the computers connected to the network; in this way their data can be used for several processings at a time. All the laser sensors have

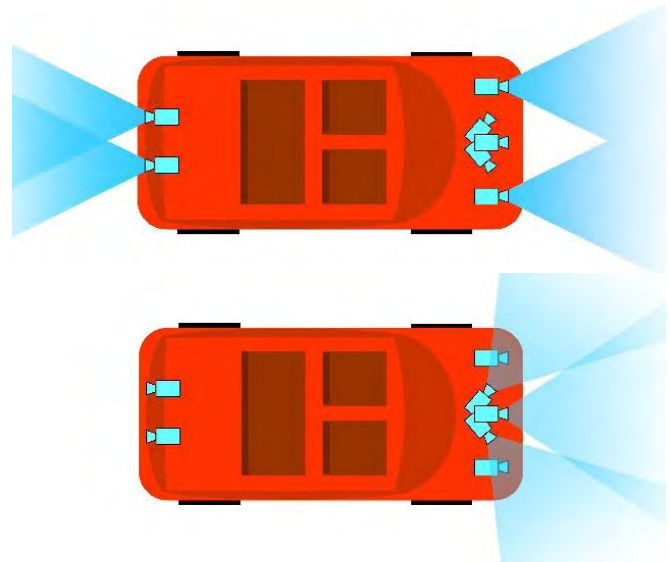


Fig. 4. The vehicle's vision sensing. Top: frontal and backward stereo vision systems able to detect lane markings and obstacles; bottom: frontal panoramic vision system, used to locate the leader vehicle.

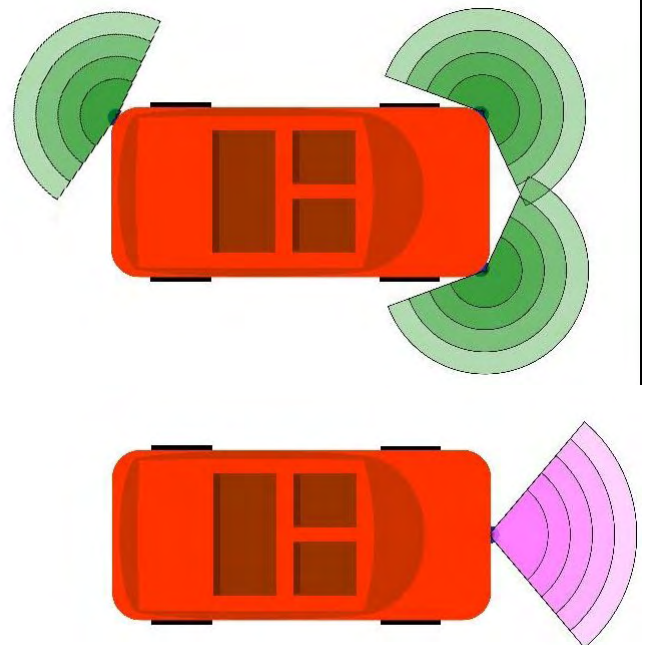


Fig. 5. The laserscanner system. Top: monobeam laserscanners covering the front and the back of the vehicle; bottom: multibeam (4) frontal laserscanner.

an IP67 waterproof grade.

Figure 5 shows the laserscanner sensing systems and their use.

The vehicle also features GPS, IMU, and intervehicle communication systems. The AGI3 GPS/IMU unit, provided by TopCon, is mounted on top of the roof and provides both GPS and inertial data through a 115200 bps RS232 serial port and through a CAN port. GPS Transmitted data are provided in NMEA format, while INS data are available on a proprietary protocol. The inertial unit provides

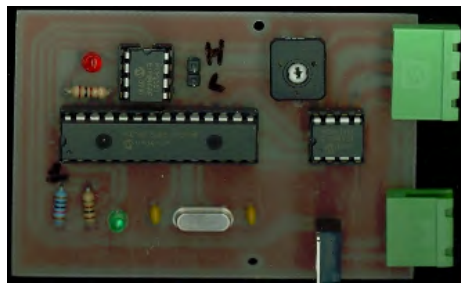


Fig. 7. The simple gas control board.

linear acceleration on three axes as well; the three angular accelerations are measured using the internal gyroscope; an additional magnetometer provides the bearing information and finally a thermometer provides the temperature of the unit.

Other data are available on the vehicle CAN bus. Among these: odometry, obtained from an incremental encoder, and the steering wheel status, which are used by the control algorithms.

Two technologies (vision and laser) are used together and their data fused in order to achieve a more robust detection in all scenarios like mountain, urban, off-road, and in all weather situations like dust, heavy rain, sunshine...

Two 240 W solar panels are mounted externally on top of the roof and provide the power to run the complete autonomous driving system. All the additional devices mounted on the vehicles for the autonomous challenge rely on an electrical system which is completely decoupled from the original vehicle power system.

B. The X-by-Wire System

The vehicles are equipped with full x-by-wire to control acceleration, braking, and steering.

In a number of vehicles, the CAN bus is used to control different on-board systems and even gas or brakes. Although directly acting on CAN messages may seem a straightforward solution, this procedure hides complex issues, since car manufacturers generally use proprietary interfaces and protocols.

In this case we selected a vehicle on which we can completely master the CAN bus and protocol. On the Porter Piaggio we developed specific control mechanisms interfaced with the CAN bus for the control of:

- gas,
- brake, and
- steering

1) *Gas*: The Porter Piaggio is an electric vehicle and its engine is controlled using pulse-width modulation (PWM). The PWM signal is generated by a mechanical potentiometer (6539S-1-502) directly connected to the gas pedal. The pot resistance varies from 0 to 5 k Ω while the current intensity ranges from 0.1 to 0.4 mA.

Therefore, a good solution for controlling the gas is the replacement of the mechanical pot using a digital one and a DIP packaged microchip PIC device has been selected. This



Fig. 8. The AES-25 installed on the Porter.

digital pot can be controlled using the standard SPI protocol. Since the gas pedal has to be controlled using a CAN bus, an additional device is used for CAN I/O and to generate the SPI signals: the PIC18F2585.

The PIC18F2585 is an 8 bits microcontroller that can work up to 40 MHz; it is equipped with a 4 MBytes flash memory, and can be programmed using the C language. An MCP2551 driver is used to interface the microcontroller to the CAN bus.

The digital potentiometer is a Microchip MCP41010 able to vary the resistance in the range 0–10 k Ω in 256 steps (in order to fit the resistance range in the required one, the pot is connected in parallel to a 10 k Ω resistor).

Figure 7 shows the resulting board.

The electric vehicle is not equipped with a complex gearbox and therefore it has only been necessary to automate the switch between forward and reverse gears: a separate circuit is again used to control a relay inside the electric engine.

2) *Steering*: Off-the-shelf components can be easily adapted to act on the steering wheel. In fact, in the agricultural environment, semi-automated or remotely driven tractors are widely exploited and the devices used to act on the tractors steering wheels can be transferred to the Porter Piaggio.

Specifically, a TopCon Accurate Electric Steering-25 has



Fig. 9. The linear actuator installed on-board of the vehicle.



Fig. 6. The first vehicle equipped with sensing technology and the on-board PCs.

been selected. This device is an electronic steering wheel that can be controlled via CAN bus through a proprietary protocol. The AES-25 can reach 30 RPM, features a 0.5 degrees resolution, and reached a 5 N m torque.

Figure 8 shows the installation of AES-25 on the Porter.

The AES-25 features the possibility of overriding the steering. If a sufficiently high external torque is applied to the steering wheel (namely, the driver tries to override the AE-25 behavior), it stops working and allows the driver to steer. This feature is mandatory for the development phase, since it is possible to safely test new control or perception strategies and to take the vehicle control in case of errors or danger.

3) *Brake*: The Porter is equipped with a mechanical brake and therefore it is not possible to intervene at electronic or idraulic level. Also in this case, it is mandatory to allow the driver to override the system behavior, namely to be able to brake even when the system is not braking.

Therefore, it has been selected to directly intervene on the brake pedal using a linear actuator. The main requirements for this actuator are a reduced size and to feature a really short latency. For these reasons, the use of an idraulic actuator was discarded and the use of an electric one was preferred.

In order to precisely define the actuator, a usage percentage ($Fu[\%]$) has to be considered:

$$Fu[\%] = \frac{t_{work}}{t_{tot}} \times 100$$

When $Fu[\%] < 30\%$ the use of a linear actuator based on trapezoidal spindle is suggested, while for $Fu[\%] > 50\%$ the use of ball screw technology is preferred.

For safety reasons, it has been decided to over-estimate the $Fu[\%]$ and therefore a ball screw based actuator has been selected: the Setec ISOMOVE 32.

This actuator is able to reach a 1500 N axial dynamic force at 200 mm/s and can be easily controlled thanks to a CAN interface.

C. The Software Architecture

The current system is depicted in figure 10.

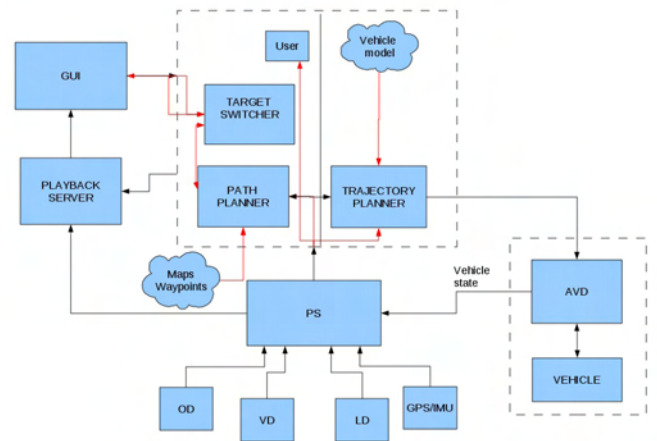


Fig. 10. The vehicles software architecture.

The solution adopted, and currently being tested, is based on 4 blocks: the Perception Server (PS), the Automated Vehicle Driver (AVD), the Graphical User Interface (GUI), and the central part which is the union of trajectory planner and path planner. In a further paper a more detailed definition of the software architecture will be given, together with an analysis of the achieved performance during the testing phase.

V. CONCLUSIONS

The 4 electric vehicles will leave Italy on July 10, 2010, and currently the VisLab team is still working on the prototypes.

Anyway, the equipment of these prototypes attracted the interest of other players and, in accordance to our ERC-funded project, VisLab is going to share the internal architecture, as well as also replicas of these vehicles, with interested research centers.



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session I

Perception & Localization

- **Title: Learning a Real-Time 3D Point Cloud Obstacle Discriminator via Bootstrapping**
Authors: Michael Samples and Michael R. James
- **Title: In Improved Flies Method for Stereo Vision: Application to Pedestrian Detection**
Authors: Hao Lee, Gwenaelle Toulminet, Fawzi Nashashibi



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Learning a Real-Time 3D Point Cloud Obstacle Discriminator via Bootstrapping

Michael Samples and Michael R. James
Toyota Research Institute, North America
{michael.samples, michael.james}@gmail.com

Abstract—Many recent mobile robotics applications have incorporated the use of 3D LIDAR sensors as an important component in scene understanding due to frequent data measurements and direct observation of geometric relationships. However, the sparseness of point cloud information and the lack of unique cues at an individual point level presents challenges in algorithm design for obstacle detection, segmentation, and tracking. Since individual measurements yield less information about the presence of obstacles, many algorithmic approaches model the joint posterior of point-labels. Such approaches can produce robust point labelings at higher computation cost. In this paper, we apply joint posterior approaches with smooth terrain priors for point cloud obstacle discrimination. The resulting labels are used to bootstrap efficient discriminators which require no human labeled data, yet are comparable in discriminative ability to the joint posterior approaches.

I. INTRODUCTION

Autonomous driving in urban environments requires the ability to quickly identify navigable terrain from potential obstacles. As mobile robots move from structured test environments to real-world scenarios, robotic perception systems must become more competent in navigating through dynamic environments. In addition to determining local navigability, perception systems should also identify the location and class of obstacles within the scene.

Entries in the recent DARPA Urban Challenge (see, e.g., [10], [13], [2]) used a combination of LIDAR, vision, and radar for obstacle detection. Each sensor has its own unique advantages and challenges, but in this paper, we focus on the use of LIDAR sensors to directly acquire 3D point clouds from objects within a scene. Mobile robotics are naturally dependent on planning paths in metric space; using point clouds greatly simplifies the problem of acquiring relative obstacle pose, but has its own unique challenges in obstacle detection and classification. LIDAR data becomes much sparser away from the sensor, and laser typically lacks high-resolution texture data that can be used for feature generation. Simple strategies—such as ground point removal by height thresholding—work in simple environments, but are not robust in real-world autonomous driving scenes. Moreover, simple systems are difficult to tune correctly: a classifier with a high false-positive rate may cause an autonomous vehicle to take sub-optimal paths, or produce uncomfortable stop-and-go behavior.

The problem of obstacle detection and classification from 3D point clouds can be approached using local feature-based classifiers [16], [10], [13], [12], solved as a joint inference

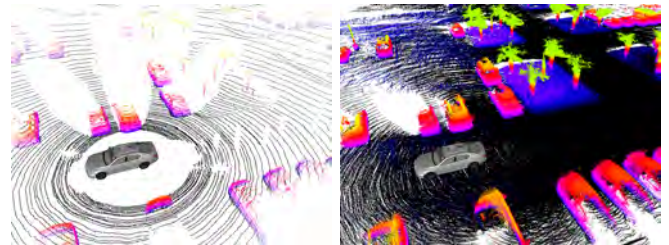


Fig. 1. Velodyne LIDAR point cloud colored by height. Data captured in a parking lot with (a) 1 rotation and (b) accumulated over several seconds.

labeling problem [1], [11], or a combination of both [14], [9]. Several approaches have also trained classifiers to match specific objects in a scene [8], [6]. In general, the techniques which use global labeling by the joint distribution of features and labels outperform the local classifier techniques. However, this improvement in classifier performance comes at greatly increased computational cost. Autonomous driving systems require both high accuracy and rapid classification to perform at modest street-driving velocities. During the DARPA Urban Challenge, vehicles with uncertain obstacle maps were able to slow down and take more sensor measurements. This option may not always be available.

In [4], Dahlkamp et al. demonstrate the bootstrapping of a vision-based classifier in cases where the features indicative of obstacles may change over time. Similarly, due to the wide variety of autonomous driving situations, it is challenging to generate many large human-labeled 3D point clouds for use as training data. This paper approaches this problem using manually-constructed weak classifiers and global terrain estimation using a random field approach. The inclusion of terrain estimation produces a more accurate point-cloud labeling, but requires greater computational resources. To meet the rapid classification requirements on the vehicle platform, we train a decision tree classifier to match the output of the combined weak classifiers and terrain model. The resulting classifier achieves high accuracy with small computational cost. This approach uses no human-labeled data, and can be run iteratively to increase the classification accuracy. In the following sections, we describe approaches to terrain estimation, including our approach using a random field model. We present results comparing the accuracy and speed of our learned classifiers, and we discuss how an efficient solution to the obstacle detection problem can be used for scene segmentation.

II. TERRAIN ESTIMATION

Several approaches for estimating the maximum likelihood labeling for a 3D point cloud involve a class of Markov Random Field problems [15]. Many such labeling problems are reduced to the Ising model, which incorporates both local evidence and neighborhood label contiguity. Work by Angelov et al. [1] and Munoz et al. [11] demonstrates how random field approaches can be used to achieve high accuracy of classification using a maximum-margin training procedure. However, margin-based approaches still require labeled training data, which may be challenging to obtain in many autonomous driving scenes.

We reduce the problem of obstacle-ground discrimination to estimating the ground surface height for some region around the vehicle, using the assumption that obstacle points can be trivially classified once the terrain is known. The problem of terrain estimation was first addressed in the GIS community for processing aerial LIDAR data (see, e.g., Sithole and Vosselman [12] for a review of large-scale terrain estimation approaches). Many of these approaches apply local geometric filters (e.g., incline thresholding) to classify aerial LIDAR points as terrain or objects.

In [9], Lu et al. labeled aerial LIDAR data using a conditional random field which explicitly maintained variables for terrain height. LIDAR points were labeled as terrain or obstacles based on an expectation maximization algorithm which incorporated local geometric features in addition to terrain smoothness. In an earlier approach, Wellington et al., [14] employed a random field which simultaneously estimated variables for terrain height and terrain type over a discrete grid. Local observations were made in a 3D voxel representation which allowed the algorithm to incorporate historical observations. The random field was solved using a Gibb's sampling approach which estimated the likelihood of each column based on a Hidden Semi-Markov model learned from labeled training data. In the approaches of both [14] and [9], points were independent of their neighbors, conditioned on their explicit variables for local terrain height and class. This conditional independence leads to a natural 2-stage algorithm in which points are classified in one stage, and the terrain is estimated in the second stage, proceeding until convergence. Similarly, in [7], Hadsell et al. estimate terrain using a kernel-based function approximation technique, assuming that ground discrimination is handled separately.

III. BOOTSTRAPPING CLASSIFIER

Similarly to the approaches of [9] and [7], we adopt a multistage approach to point classification and terrain modeling. Our overall process is shown in Figure 2. In the first step, a feature vector is computed for each point based on the local geometric arrangement to the point cloud. In the second step, weak classifiers (explained below) are manually constructed that are better than chance. These weak classifiers are applied to the point cloud, and in the third step, statistics over label type are maintained in a 3D grid centered around the vehicle. In the fourth step, a terrain height is estimated for each column in the 3D grid, based

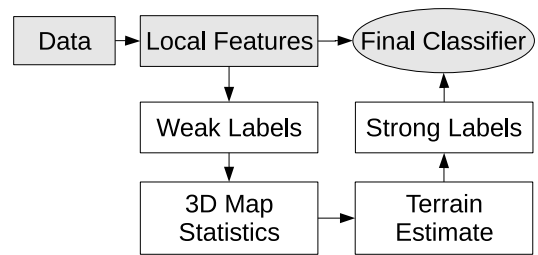


Fig. 2. Bootstrapping a classifier from weak labelings and terrain estimation. The white boxes represent an offline training process.

on the local statistics previously accumulated in that column. Finally, the original data points are relabeled based on their height above the terrain estimate, and a new classifier is trained to discriminate those points based on their original feature vectors.

Feature Generation. Feature generation for real-time point cloud classification has unique challenges from offline algorithms. First, while offline point cloud processing may use points generated from multiple timesteps, online algorithms must use features computed from instantaneously available data. Secondly, features should be principally based on local information to be robust to pose uncertainty of the sensor platform. Finally, the feature vectors must be rapidly computed. Certain features which may seem useful (e.g., collecting statistics on all neighboring points within a bounded region) may not be feasible to compute on a mobile platform with a short horizon to decision making.

Table I contains a list of features for each laser point. On our sensor platform, the Velodyne LIDAR consists of 64 lasers with unique horizontal and vertical angular offsets. We generate features using data collected during a single rotation of the unit. Due to the known internal structure of the laser, we can quickly find measurements made immediately the left, right, top, and bottom of the current beam. The last features, f_7 and f_8 are made after correcting for the roll, pitch, and yaw of the sensor platform. The last feature, f_8 , is calculated by keeping a 2D grid of the lowest measurement observed in each cell on a given rotation.

TABLE I
LOCAL FEATURES FOR POINT CLASSIFICATION.

Feature	Description
f_1	beam range
f_2	beam remission
f_3	left beam range - range
f_4	right beam range - range
f_5	top beam range - range
f_6	bottom beam range - range
f_7	distance below sensor
f_8	height above lowest measurement in grid cell

Weak Classifiers. It is challenging to construct classifiers for the features listed in Table I without large amounts of labeled training data. In order to bootstrap our classification process, we created 2 weak classifiers implemented as decision tree stumps. The first weak classifier uses the decision

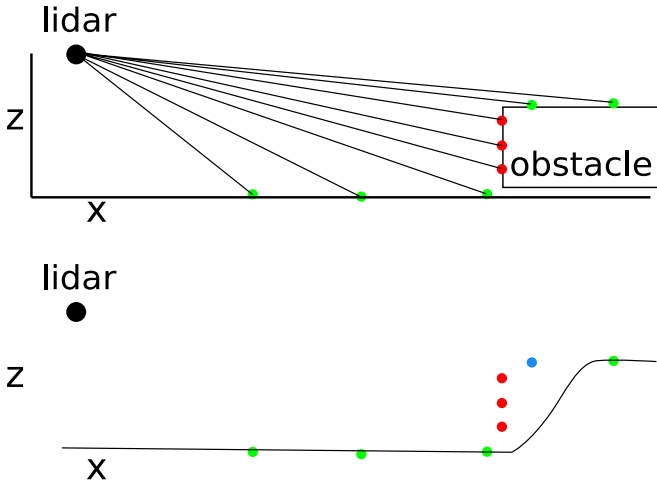


Fig. 3. Caricature of weak classification results. Features (such as comparative beam length f_6) tend to classify steep obstacles well, but misclassify flat portions of obstacles. Obstacle classification shown in red. In the lower image, a terrain surface is estimated which fits ground points and respects smoothness constraints. The smooth terrain estimate allows one point to be reclassified as an obstacle (shown in blue).

rule $f_8 > \epsilon_1$. Specifically, for a given point within a discrete 2D grid cell, it is labeled as an obstacle if the point is higher than the lowest measurement observed in the same grid cell (in our experiments $\epsilon_1 = 1.0$). This classifier is useful for labeling obstacle points on walls, trees, and the tops of vehicles. The second weak classifier labels points as obstacles if $f_6 > \epsilon_2$. In our experiments, ϵ_2 was set to -0.05 meters to reflect the confidence in LIDAR return range. This parameter relates to the local inclination around the point: steeper inclines will yield relatively larger values for f_6 . The second weak classifier is inspired by a similar application on Stanford's Urban Challenge entry Junior [10].

Summary statistics from the output of the weak classifiers is stored in a 3D grid which serves as the input to the terrain estimation procedure (as in [14]). Since terrain does not vary over time, each column in the grid maintains statistics on all of the data produced by the weak classifiers over previous timesteps. Table II shows the statistics used for each column.

TABLE II
SUMMARY STATISTICS COMPUTED FOR DISCRETE COLUMN i OF 3D GRID USING WEAK CLASSIFIER OUTPUT.

Feature	Description
μ_g^i	average height of ground hits in global coordinates
σ_g^i	variance of ground hit heights
μ_o^i	average height of obstacle hit in global coordinates
σ_o^i	variance of obstacle hit heights
μ_t^i	ground height as measured by known tire position
σ_t^i	variance of tire hit heights

Terrain CRF. We used a Conditional Random Field in Gibbsian form to find the most likely continuous labeling for the vector of terrain heights Z . Figure 4 shows the graphical model for our system, in which the height estimate z^i for each column i depends on the summary statistics

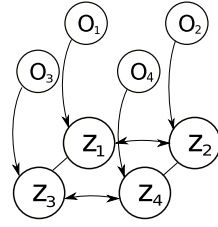


Fig. 4. CRF model for terrain estimation. Terrain height z^i depends on neighboring terrain heights and the column's summary statistics O^i .

in the column and the continuous height labels assigned to neighboring terrain cells.

The conditional likelihood of our terrain estimate Z is based on a continuous extension of the familiar Ising model:

$$\mathbb{P}(Z|O) = \frac{1}{Q} \exp(-(\omega_\psi \Psi(Z, O) + \omega_\phi \Phi(Z))) \quad (1)$$

where Ψ and Φ represent the local and neighborhood weighted potentials, and Q is the partition function [15].

The local evidence is calculated as:

$$\Psi(Z, O) = \sum_j \Psi_j(Z, O) \quad (2)$$

$$\Psi_1(Z, O) = \frac{1}{2\sigma_t^2} (z^i - \mu_t^i)^2 \quad (3)$$

$$\Psi_2(Z, O) = \frac{1}{2\sigma_g^2} (z^i - \mu_g^i)^2 \quad (4)$$

$$\Psi_3(Z, O) = -\ln \left(1 - \frac{1}{2} [1 + \operatorname{erf}(\frac{z^i - \mu_o^i}{\sigma_o^i \sqrt{2}})] \right) \quad (5)$$

The first and second terms of the local evidence function Ψ drive the terrain towards the wheel estimates and the local estimates for ground point distribution, respectively. The third component Ψ_3 creates a potential which drives the terrain below increasing densities of obstacle hits, regulated by the cumulative density function of the best-fit Gaussian.

Our neighborhood function Φ encodes the assumption that terrain height labels vary smoothly with neighboring terrain cells:

$$\Phi(Z) = \frac{1}{2} \sum_i \sum_{j \in N(i)} (z_i - z_j)^2. \quad (6)$$

Optimization. As is commonly stated, solving Equation 1 optimally is generally intractable. As in [5], we settle for a high likelihood approximation of the posterior

$$Z_m^* = \operatorname{argmax}_Z (\log \mathbb{P}(Z|O)) \quad (7)$$

which is equivalent to finding a local minima of the weighted energy $U(Z, O) = \omega_\psi \Psi(Z, O) + \omega_\phi \Phi(Z)$. This minimization can be efficiently implemented using conjugate gradient optimization:

$$\frac{\partial \Psi}{\partial z^i} = \frac{(z^i - \mu_t^i)}{2\sigma_t^2} + \frac{(z^i - \mu_g^i)}{2\sigma_g^2} + \frac{\frac{1}{\sqrt{2\pi\sigma_o^i}} \exp(-\frac{1}{2}(\frac{z^i - \mu_o^i}{\sigma_o^i})^2)}{1 - \frac{1}{2}(1 + \operatorname{erf}(\frac{z^i - \mu_o^i}{\sigma_o^i \sqrt{2}}))} \quad (8)$$

and

$$\frac{\partial \Phi}{\partial z^i} = \sum_{j \in N(i)} (z^i - z^j). \quad (9)$$

IV. EXPERIMENT AND RESULTS

We collected approximately 30 minutes of normal driving data in parking lots and urban streets. For each rotation of the Velodyne LIDAR, we computed local features and labeled each point using the hand-constructed weak classifiers. We constructed a 3D Grid at 1 meter resolution to maintain summary statistics over weak classifier labelings. We optimized the terrain CRF described above using a sliding window approach with size 100x100. Subsequently, the LIDAR points from each frame were relabeled using height above terrain model as the primary classifier. In this implementation, laser points higher than 20cm above the terrain were labeled obstacles. We used the geometric features listed in Table I and the relabeled points to train a Decision Tree classifier using a standard implementation in OpenCV [3]. We further increased the speed of our decision tree implementation by compiling the trees to machine code.

Figure 8 contains the results of our experiments. In each data set, the weak classifiers are capable of recognizing many points on an obstacle, but also misclassify a large number of points. Typically, these misclassified points would not be used for individual object classification, and the missing points might also prevent good estimates for tracking. However, incorporation of the terrain prior for smoothness allows many of these points to be correctly and efficiently relabeled as is seen in row 4 of Figure 8. The final decision tree classifier does an equivalently good job at discriminating obstacle points, yet does so without needing to explicitly estimate the local terrain.

The final decision tree implementation requires less than 30ms to classify more than 100,000 points, which includes feature vector construction. By contrast, Figure 5 shows the amount of time required to solve the terrain CRF model for an area around the vehicle. To decrease the computational cost of the conjugate gradient optimization, we seed the solution with the previous best terrain estimate. Consequently, the vehicle requires more time to solve the CRF at higher velocities, as it rapidly encounters unseen grid cells.

Evaluating the quality of the final classification is generally difficult without human-labeled data. However, the quality of the decision trees and the quality of the terrain estimation may be evaluated separately. Table III shows good correlation between the points relabeled using the results of the terrain CRF output and the rules learned by the decision tree. In some cases, the decision tree may generalize better than the CRF output. Figure 3 shows possible errors which can occur during the terrain estimation process when the weak classifiers mislabel the data. Figure 8 actually contains this type of error: points measured from a car on a side street are still considered to be ground due to numerous weak classifier mistakes. However, the final rules learned from the decision tree generalize well enough from other pieces of training data to correctly detect the obstacle.

It is difficult to empirically evaluate the quality of terrain estimation without ground truth information, but given the assumptions implicit in the CRF, one straightforward metric

is the likelihood computed in the optimal solution to Equation 7. Figure 6 shows the Gibb's cost for each of the point labeling strategies. The most costly terrain estimate arises from the weak classifiers, while the best terrain estimate is found by relabeling points with terrain data as it becomes available. The terrain estimates using labels from the decision tree validates the assumption that the decision tree approximates the maximum likelihood terrain estimate more closely than weak classifiers, and results in a terrain estimate that more closely aligns with our assumptions about terrain structure.

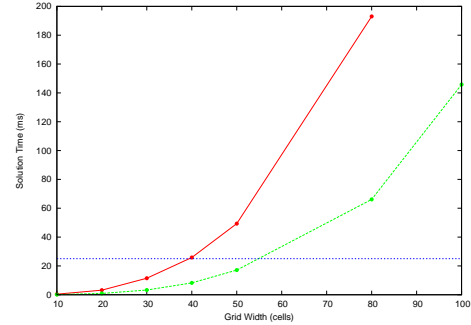


Fig. 5. Terrain estimation CPU time as a function of grid size from 10x10 through 100x100. The green dashed rate is taken from the Parking Lot data set with an average speed of 10mph. The solid red line is taken from the Surface Street data set at 40mph. Differences in solution rate are the result of greater re-use of previous solutions for the slower dataset. The blue dashed line represents the constant-time decision tree implementation.

TABLE III
DECISION TREE CLASSIFICATION AGREEMENT WITH TERRAIN
ESTIMATE DATA. '+' REPRESENTS OBSTACLE CLASS.

	Terrain +	Terrain -
Tree +	17.9	1.3
Tree -	1.5	79.2

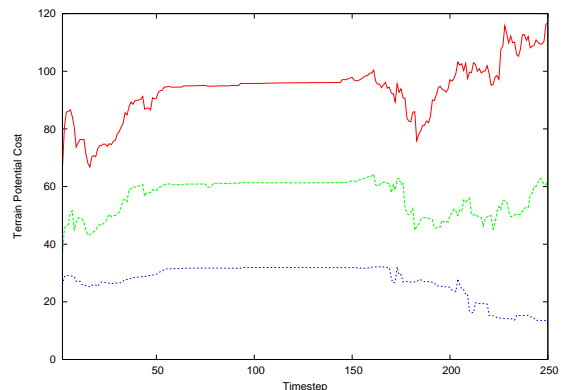


Fig. 6. Gibb's cost for best terrain CRF solution as a function of timesteps during the Parking Lot test. Terrain was optimized using Weak Labels (red), Weak Labels + Previous Terrain (Blue), and Decision Tree Labels (Green). The decision tree more closely approximates the behavior of the CRF.

V. EXTENSIONS TO TOTAL SCENE SEGMENTATION

In this paper, we've presented a machine-learning approach to train discriminative classifiers without any human labeled data. Including neighborhood priors in a point labeling procedure allows us to bootstrap weak classifiers into a competent binary classification algorithm. We have demonstrated that we are able to use this bootstrapping approach to generate a large amount of labeled training data. The success of the final classifier shows that there is sufficient information within the local geometric features to discriminate obstacles. In addition, by preventing overfitting of the strong classifier models, we are able to learn classifiers which out-perform the initial labeled data set in some cases.

In this work, our random field for estimating terrain was kept deliberately simple for ease of computation. Since the bootstrapping process can be performed offline, more computationally challenging models can be implemented as in [9] which iteratively converge on good labels for all the points within the scene.

The total computation time from LIDAR point acquisition to labeling is less than 50ms, which means this algorithm is capable of operating in real-time on our test vehicle with sensor information arriving at 10 Hz. The rapid discrimination of obstacle points allows our autonomous vehicle to spend more CPU time on object segmentation, classification, and tracking. In particular, more robust point discrimination results in higher efficacy for segmentation and classification algorithms. Figure 7 shows the result of our obstacle discrimination applied to a straight-forward distance-based segmentation algorithm. Without removal of non-obstacle points, a simple segmentation algorithm would not provide correct groupings for obstacle points. The training process described in this paper demonstrates the best approach we know for efficient and accurate obstacle discrimination.

Our future work includes extending this binary discrimination procedure to automatically train classifiers for dynamic objects within the scene, such as vehicles and pedestrians.



Fig. 7. Segmentation of obstacle points within the scene. Different colors represent different segmentation groups. Good obstacle discrimination simplifies obstacle segmentation.

REFERENCES

- [1] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng. Discriminative learning of markov random fields for segmentation of 3d scan data. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, pages 169–176, Washington, DC, USA, 2005. IEEE Computer Society.
- [2] A. Bacha, C. Bauman, R. Faruque, M. Fleming, C. Terwelp, C. F. Reinholtz, D. Hong, A. Wicks, T. Alberi, D. Anderson, S. Cacciola, P. Currier, A. Dalton, J. Farmer, J. Hurdus, S. Kimmel, P. King, A. Taylor, D. V. Covern, and M. Webster. Odin: Team victortango's entry in the darpa urban challenge. *J. Field Robotics*, 25(8):467–492, 2008.
- [3] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly, 2008.
- [4] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. R. Bradski. Self-supervised monocular road detection in desert terrain. In G. S. Sukhatme, S. Schaal, W. Burgard, and D. Fox, editors, *Robotics: Science and Systems*. The MIT Press, 2006.
- [5] J. Diebel and S. Thrun. An application of markov random fields to range sensing. In *Proceedings of Conference on Neural Information Processing Systems (NIPS)*, Cambridge, MA, 2005. MIT Press.
- [6] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *Proceedings of the European Conference on Computer Vision (ECCV)*, May 2004.
- [7] R. Hadsell, J. A. Bagnell, D. Huber, and M. Hebert. Accurate rough terrain estimation with space-carving kernels. In *Proceedings of Robotics: Science and Systems*, Seattle, USA, June 2009.
- [8] A. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(1):433 – 449, May 1999.
- [9] W.-L. Lu, K. P. Murphy, J. J. Little, A. Sheffer, and H. Fu. A hybrid conditional random field for estimating the underlying ground surface from airborne lidar data. *IEEE Transactions on Geoscience and Remote Sensing*, 47(8):2913–2922, 2009.
- [10] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Hähnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, and S. Thrun. Junior: The stanford entry in the urban challenge. *J. Field Robotics*, 25(9):569–597, 2008.
- [11] D. Munoz, J. A. D. Bagnell, N. Vandapel, and M. Hebert. Contextual classification with functional max-margin markov networks. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2009.
- [12] G. Sithole and G. Vosselman. Experimental comparison of filter algorithms for bare-earth extraction from airborne laser scanning point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 59(1-2):85 – 101, 2004. Advanced Techniques for Analysis of Geo-spatial Data.
- [13] C. Urmson, J. Anhalt, D. Bagnell, C. R. Baker, R. Bittner, M. N. Clark, J. M. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. E. Rybski, B. Salesky, Y.-W. Seo, S. Singh, J. Snider, A. Stentz, W. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson. Autonomous driving in urban environments: Boss and the urban challenge. *J. Field Robotics*, 25(8):425–466, 2008.
- [14] C. Wellington, A. Courville, and A. T. Stentz. A Generative Model of Terrain for Autonomous Navigation in Vegetation. *The International Journal of Robotics Research*, 25(12):1287–1304, 2006.
- [15] G. Winkler. *Image Analysis, Random Fields, and Markov Chain Monte Carlo Methods*. Springer, 2003.
- [16] D. Wolf, G. Sukhatme, D. Fox, and W. Burgard. Autonomous terrain mapping and classification using hidden markov models. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2005.

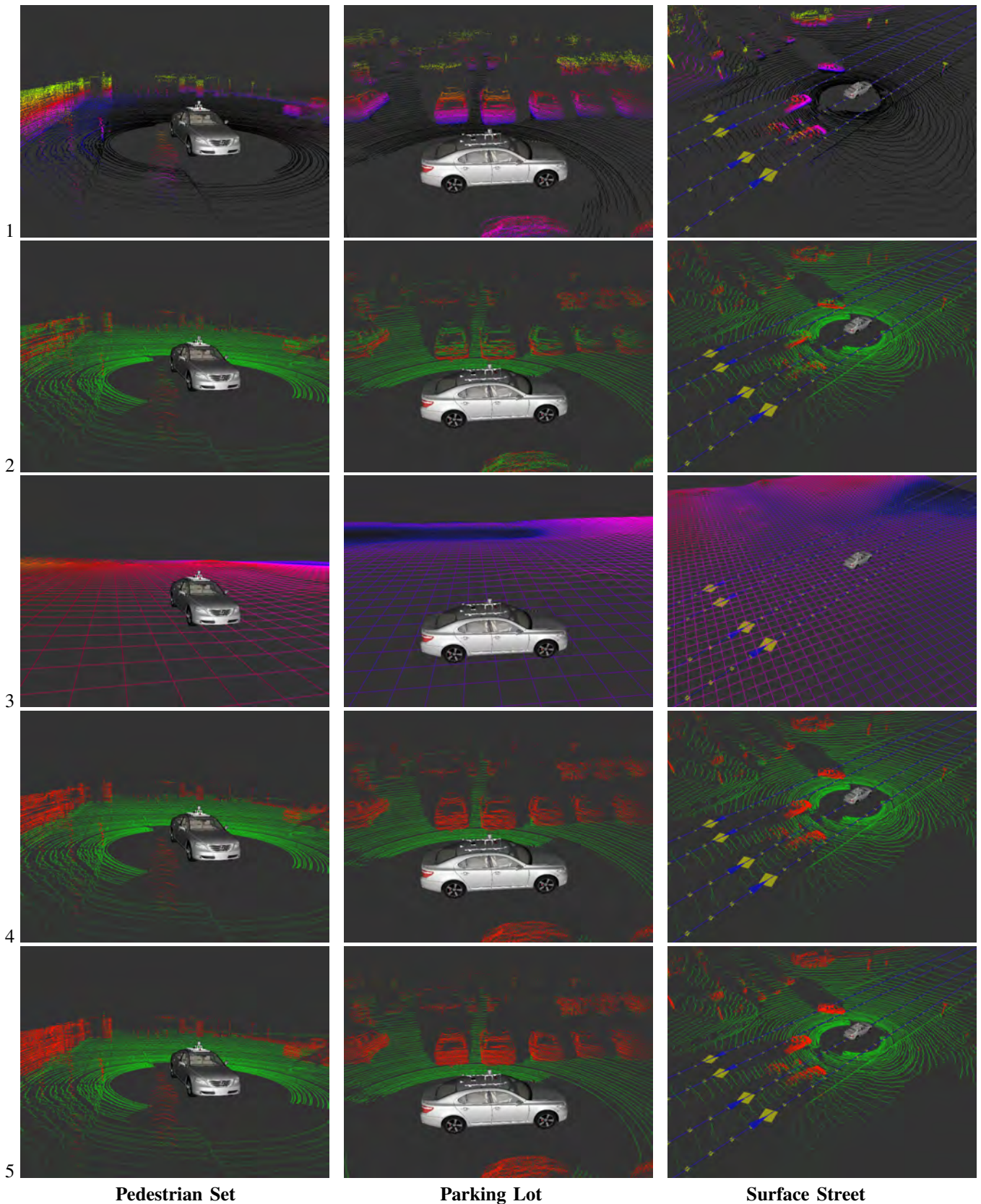


Fig. 8. Comparative classification results from 3 test sets. The first row shows the original data colored by height. The second row shows the labeling produced by the weak classifier. The third row shows the optimal terrain conditioned on the weak classification results. The fourth row shows points reprocessed with terrain information. The final row is directly labeled with a strong decision tree classifier using only local features. The parking lot data was taken at 10mph, and the surface street at 40mph.

An Improved “Flies” Method for Stereo Vision: Application to Pedestrian Detection

Hao LI, Gwenaelle TOULMINET, Fawzi NASHASHIBI

ABSTRACT — In the vast research field of intelligent transportation systems, the problem of detection (and recognition) of environment objects, for example pedestrians and vehicles, is indispensable but challenging. The research work presented in this paper is devoted to stereo-vision based method with pedestrian detection as its application (a sub-part of the French national project “LOVe”: Logiciels d’Observation des Vulnerables). With a prospect of benefiting from an innovative method i.e. the genetic evolutionary “flies” method proposed by former researchers on continuous data updating and asynchronous data reading, we have carried on the “flies” method through the task of pedestrian detection affiliated with the “LOVe” project. Compared with former work of the “flies” method, two main contributions have been incorporated into the architecture of the “flies” method: first, an improved fitness function has been proposed instead of the original one; second, a technique coined “concentrating” has been integrated into the evolution procedure. The improved “flies” method is used to offer range information of possible objects in the detection field. The integrate scheme of pedestrian detection is presented as well. Some experimental results are given for validating the performance improvements brought by the improved “flies” method and for validating the pedestrian detection method based on the improved “flies” method.

I. INTRODUCTION

Research works on intelligent transportation systems have progressed rapidly around the world and have been showing more and more promising results for enhancing urban traffic safety and efficiency. The problem of detection (and recognition) of environment objects, for example pedestrians and vehicles, is indispensable but challenging. Many research works have been devoted to the problem; take pedestrian detection as an example, mono-vision based methods have been introduced in [1][2] while laser-scanner based methods in [3]. Experiences show that methods based on single on-vehicle sensor have considerable limitations due to the limited capability of the sensor itself. Therefore, the fusion strategy among multiple on-vehicle sensors has been exploited to achieve either faster computation or more

desirable detection results. The method using fusion between laser scanner and mono-camera has been discussed in [4]. The method based on stereo-vision (i.e. the fusion between two mono-cameras) has long since been researched for indoor applications [5] and has later been extended to outdoor applications as well [6][7]. The research work presented in this paper is devoted to stereo-vision based method with pedestrian detection as its application (a sub-part of the multi-participants French national project “LOVe” (Logiciels d’Observation des Vulnerables) which aims at localization of vulnerable objects, e.g. pedestrians, in a traffic scenario [8].

Stereo-vision techniques enable the process of recovering the range information of the environment from a pair of image views. This process often involves establishing correspondence between both images, either by finding corresponding pixels i.e. the so-called disparity map (dense correspondence) [7] or by finding corresponding edges (sparse correspondence) [6][9]. The methods based on edge correspondence inevitably incur the problem of edge extraction which itself is challenging and susceptible to environment conditions; while the methods based on disparity map show robust detection result but the computational demand is forbidding for real-time applications. Besides these commonly used stereo-vision techniques, Louchet et al have put forward an innovative method which they label as the “flies” method [10][11]. Instead of establishing correspondence between 2-D images, the “flies” method directly evaluate the fitness (defined in certain way) of a group of 3-D points i.e. “flies”, and use genetic evolutionary techniques to converge these flies to places with high fitness values (which correspond to real 3-D objects). With a prospect of benefiting from the “flies” method on continuous data updating and asynchronous data reading, we have carried on the “flies” method through the task of pedestrian detection affiliated with the “LOVe” project.

Compared with former work of the “flies” method, two main contributions have been incorporated into the architecture of the “flies” method: first, an improved fitness function has been proposed instead of the original one; second, a technique coined “concentrating” has been integrated into the evolution procedure. The improved “flies” method is used to offer range information of objects in the detection field (latter referred to as “OOI”, Objects Of Interest). The paper is organized as follows: The basic philosophy of the “flies” method is briefly reviewed and the limitation of the original fitness function is analyzed in section 2; An improved fitness function as well as a

Hao LI: a PhD student at the Robotics Laboratory, Mines Paris (ParisTech) / INRIA-IMARA team (B.P.105,78153 Le Chesnay France)(e-mail: hao.li@inria.fr)

Gwenaelle TOULMINET: professor assistant at INSA of Rouen and associate researcher at JRU LaRA, Mines Paris (ParisTech)(e-mail: gwenaelle.toulminet@mines-paristech.fr)

Fawzi NASHASHIBI: with the Robotics Laboratory, Mines Paris (ParisTech) and with INRIA-IMARA team (B.P.105,78153 Le Chesnay France) (e-mail: fawzi@ensmp.fr)

“concentrating” technique is proposed in section 3; The integrate scheme of pedestrian detection is described in section 4 while experimental results are presented in section 5, followed by a conclusion in section 6.

II. THE FLIES METHOD

A. Review of the philosophy of the flies method [11]

A fly is defined as 3-D point (x, y, z) ; its projection on the left image is denoted as (u_L, v_L) and latter referred to as “left projection” while its projection on the right image is denoted as (u_R, v_R) and referred to as “right projection”; The (u_L, v_L) and (u_R, v_R) can be easily computed from (x, y, z) with calibrated camera parameter. If a fly is situated on an opaque object, the neighborhoods of its left projection and right projection are almost identical; otherwise, there is large difference between the neighborhoods of its two projections. Based on this heuristic observation, a fitness function can be defined to evaluate the degree of similarity between the neighborhoods of a fly’s left projection and right projection. In other words, the fitness function is defined in a way that a high fitness value is generally computed for a fly on an object but a low fitness value for a fly off an object. The fitness function used in former research [11] is re-written here; see Eq.(1):

$$fitness(fly) = \frac{G}{\sum_{(i,j) \in N} (L(u_L + i, v_L + j) - R(u_R + i, v_R + j))^2} \quad (1)$$

Where the denominator means the summed square of the grey-level difference between corresponding pixels in the two neighborhoods of the fly’s left and right projections; the numerator ‘G’ is “defined as the square root of an image gradient norm” [11].

The philosophy of the flies method is to detect environment objects by searching more and more flies with high fitness value through genetic evolution. Genetic evolution techniques are used to evolve a group of randomly initialized flies so that they may converge onto visible objects. The genetic evolution techniques [11] include 1) selection: retaining those best individuals; 2) sharing: reducing the fitness values of flies located in crowded areas; 3) mutation: adding random variances to the flies; 4) crossover: bearing new fly which is randomly located on the line segment of its parent flies. After a certain number of evolutionary iterations, the flies are expected to gather onto the surfaces of visible objects in the detection field.

B. The limitation of the original fitness function

Consider the fitness function Eq.(1); the inverse of the denominator gives high fitness values to flies whose left and right projections have similar neighborhoods; the numerator ‘G’, as explained in [10], is a normalizing factor that is adopted so as to reduce the fitness values of flies over insignificant regions, especially uniform regions. The fitness function Eq.(1) seems plausibly effective. However, it is

susceptible to quasi-uniform regions with small variances of high spatial frequencies.

A piece of synthetic “uniform” patch is composed for showing the limitation of Eq.(1), as shown at the left side in Fig.1 (a). There is small grey-level variance (difficult to see by naked eye) within it; the variance is indicated by a white-black alternating pattern on its immediate right side. Consider the fly ‘A’, the neighborhoods of its left and right projections are exactly the same, resulting in the denominator in Eq.(1) being zero. Although the numerator ‘G’ is small (but non-zero), the fitness value computed via Eq.(1) is high (infinite). As a consequence, the fly ‘A’ may be mistaken as

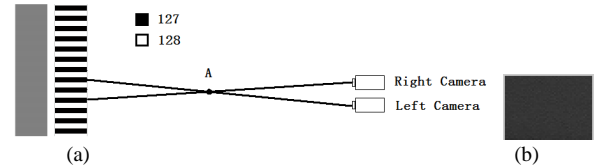


Fig.1. (a) the quasi-uniform pattern; (b) a real patch of road surface

being situated on the patch.

Above synthetic example is a bit exaggerated; in real situations, however, the problem of marking high fitness value to flies with wrong depth due to Eq.(1) still exists. This often happens where flies are in front of some quasi-uniform regions such as road surfaces; see Fig.1 (b). A simple tactic of adding a positive constant to the denominator so as to avoid it being zero does not help much.

III. AN IMPROVED FITNESS FUNCTION AND A “CONCENTRATING” TECHNIQUE

The original flies method has been briefly reviewed and the limitation of the original fitness function has been analyzed in the previous section. In this section, we are going to propose first an improved fitness function and then a concentrating technique.

A. The improved fitness function

Before any fitness function is constructed, it’s better to consider which features deserve high fitness value (called “desirable features”). As described in former work [11], desirable features are considered from two aspects: a) similarity and b) informativeness. “Similarity” means the similarity between the two neighborhoods of a fly’s left and right projections; the more similar the two neighborhoods are, the higher the fitness value is. “Informativeness” means the extent of variance within the neighborhoods. The more informative the neighborhoods are (like those with a lot of textures and contrasts), the higher the fitness value is. In the original fitness function Eq.(1), the “similarity” is measured by the summed square difference (SSD), i.e. the summed square of the grey-level difference between corresponding pixels in the two neighborhoods; while the “informativeness” is measured by the square root of an image gradient norm.

The limitation of the original fitness function has already been analyzed. Besides, using the SSD to measure the similarity (though it is simple and direct) between the two

neighborhoods has another limitation. Because of the limitation of camera calibration and image rectification and because of the influence of quantization process, even a fly is right on an object, the neighborhoods of its two projections still have some differences which are often likely to be increased in proportional to the extent of contrast within the neighborhoods. In other words, the more and larger the variations are there in an area, the more difficult it is for this area's projections on both images to be the same. As a result, the SSD tends to mark lower scores for informative regions than non-informative regions, which is not desirable.

The to-be-proposed fitness function is also constructed based on considerations from the two aspects "similarity" and "informativeness". Instead of the SSD which takes into account strictly the pixel-to-pixel difference between the two neighborhoods, a new fitness function based on covariance and the difference of grey-level mean is proposed, shown in Eq.(2):

$$\begin{aligned}
 fitness(fly) &= \sum_{j=-uSz}^{+uSz} C_j(fly) \cdot D_j(fly) \quad (2) \\
 C_j(fly) &= \sum_{i=-uSz}^{+uSz} [L(u_L + i, v_L + j) - \bar{L}_j(fly)] \cdot [R(u_R + i, v_R + j) - \bar{R}_j(fly)] \\
 D_j(fly) &= \exp(-\|\bar{L}_j(fly) - \bar{R}_j(fly)\|) \\
 \bar{L}_j(fly) &= mean\{L(u_L + i, v_L + j) \mid i = -uSz, \dots, +uSz\} \\
 \bar{R}_j(fly) &= mean\{R(u_R + i, v_R + j) \mid i = -uSz, \dots, +uSz\}
 \end{aligned}$$

Where (u_L, v_L) and (u_R, v_R) are respectively the left and right projections of the fly; their neighborhoods are respectively $L(u_L + i, v_L + j)$ (called "left neighborhood") and $R(u_R + i, v_R + j)$ (called "right neighborhood") for $i = -uSz, \dots, +uSz$; $j = -vSz, \dots, +vSz$; $L_j(fly)$ is the grey-level mean of row j in the left neighborhood while $R_j(fly)$ is the grey-level mean of row j in the right neighborhood. It is worth noting that the objects concerned (pedestrians) are normally vertical for our camera configuration; thus horizontal variances are more useful than vertical variances. This is why the new fitness function is constructed in a "row-by-row" manner.

The $C_j(fly)$ is constructed in accordance with standard definition of covariance; the $D_j(fly)$ is the exponential function in terms of the absolute difference of the grey-level means of two corresponding rows. The considerations for "similarity" and "informativeness" are integrated comprehensively in the new fitness function. If the two neighborhoods have similar grey-levels on general (then $D_j(fly)$ is large) and they are informative and similar (then $C_j(fly)$ is large), a high fitness value is computed via Eq.(2). Otherwise, either $C_j(fly)$ or $D_j(fly)$ might be small, resulting in a low fitness value. More words hovering around the "similarity" aspect, the criterion of measuring similarity by the difference of grey-level mean and the covariance is a bit "loose" compared with the SSD. It is this "looseness" that gives some tolerance to the inaccuracy of camera calibration and image rectification and the influence of quantization process. The performance improvements brought by the new

fitness function are demonstrated in latter sections.

B. The concentrating technique

After the evolution process, the result is a group of cloud-like flies which gather roughly around OOI. These fly-clouds are difficult to be used directly; a technique labeled "concentrating technique" is proposed for "concentrating" these fly-clouds (with thousands of flies) to several points which correspond to the position of OOI. The concentrating technique is incorporated into the flies method because of two reasons: first, it is used to output meaningful localization results of OOI; second, its output can be used to guide the re-generation of new fly population in the genetic evolution process. It is worth noting that a simple tactic of localize OOI by finding flies with local maximum fitness value is not practical because few flies which are far away from real objects might have high fitness value due to coincidental similarity between its backgrounds on both images.

The principle of the proposed concentrating method is to localize OOI by finding points with local maximum fitness value density. This process is similar to that of finding the modes presented in [12], where a mean-shift method is described. Here, the mean-shift method is borrowed from [12]. Besides, a step of "histogram based sampling" is used as a fast procedure to generate some sampling points which can be served as proper starting points for the mean-shift process. The proposed concentrating technique consists of two steps: 1) histogram based sampling; 2) mean-shift.

1) Histogram based sampling:

Several narrow view-angles which cover the detection field are chosen. This step is to locate a sample point for each narrow view-angle chosen; the sample point is the place with largest vote of the fitness value histogram within the view-angle. For a narrow view-angle, if there is OOI inside (Fig.3 (a)), the area around OOI is likely to be the area with highest fitness value density in it (Fig.3 (b)). Therefore, the sample point of this view-angle is normally close to the OOI. Then, starting mean-shift process from the sample point will facilitate convergence to the position of OOI. The detailed procedures of "histogram based sampling" are described below in this sub-part while the mean-shift process described in the following sub-part.

- [i] Choose several narrow view-angles which cover the detection field.
- [ii] For each view-angle, compute a histogram (Fig.3 (c)) from the flies within the view angle according to the distance, i.e. the longitudinal coordinate 'x'; the vote is weighted by the fitness value of the fly.
- [iii] For each view-angle (denote its direction angle generally as $\tan^{-1}k_i$), find the distance interval with the largest vote (suppose it to be $[x_k, x_{k+1}]$); then the sample point of this view-angle is computed via (let all sample points be situated on the ground surface, i.e. $z = 0$):

$$(x_i, y_i, z_i) = (d_i, d_i \cdot k_i, 0); d_i = (x_k + x_{k+1}) / 2;$$

2) Mean-shift:

The sample points obtained from the previous step are

served as starting points for the mean-shift process described below.

[i] Imagine all the flies are temporarily situated on ground, i.e. let their ‘z’ coordinate be temporarily 0; then project all the flies onto the left image (or right image). Latter, the mean-shift process is carried out in the left image coordinate instead of directly in the world coordinates; and the bandwidth matrix can be assigned uniformly to be the identity matrix, i.e. $\mathbf{H} = h^2\mathbf{I}$. Otherwise, different bandwidth matrixes have to be assigned to the flies according to their depth because same pixel length means different physical length at different depth. This will not only increase the complexity of the bandwidth matrixes assignment itself but also increase computational burden.

[ii] The algorithm for mean-shift is derived from the general case introduced in [12]:

$$w_i = \frac{f(p_i) \cdot \exp(-(p_{old} - p_i)^T(p_{old} - p_i)/(2h^2))}{\sum_{k=1}^n f(p_k) \cdot \exp(-(p_{old} - p_k)^T(p_{old} - p_k)/(2h^2))}$$

$$p_{new} = \sum_{i=1}^n w_i p_i$$

Where $\{p_i | i = 1, 2, \dots, n\}$ are the positions of all the flies on the left image; $f(p_i)$ is the fitness value of fly p_i ; p_{old} is the old position of the sample point while p_{new} is the new position of it. For each sample point, repeat mean-shift for few times.

[iii] Cluster the sample points using a simple distance criterion; the distance threshold is chosen to be h . Compute the geometric center of each cluster, which is referred to as ‘‘candidate point’’.

[iv] Discard those candidate points with low fitness value density; fitness value density is computed via (also derived from the general case introduced in [12]):

$$d(p) = \frac{1}{n(\sqrt{2\pi}h)^{\dim}} \sum_{k=1}^n f(p_k) \cdot \exp\left(-\frac{(p - p_k)^T(p - p_k)}{2h^2}\right)$$

For simplicity, the normalizing constant can be omitted.

[v] Re-project each candidate point from the left image onto the ground surface; the positions of these candidate points in the world coordinates are regarded as the positions of OOI and are the output of the proposed concentrating technique.

IV. THE INTEGRATE SCHEME OF PEDESTRIAN DETECTION

The improved flies method which can offer range information of OOI has been introduced in previous sections. In combination with learning-based classifier, it has been applied to pedestrian detection as a sub-part of the French national project ‘‘LOVe’’.

The range information of OOI offered by the improved flies method can be used to extract some ROI (Region Of Interest) boxes on the image. For each box (its size is 48x96 pixels after normalization), a feature vector based on HOG (Histogram of Oriented Gradient) [13] is computed and then fed to a SVM for classification; the SVM has been pre-trained

off-line with the HOG-based feature vectors computed from 10000 pedestrian examples (normalized boxes which contain pedestrians) and 18000 non-pedestrian examples (normalized boxes which do not contain pedestrians).

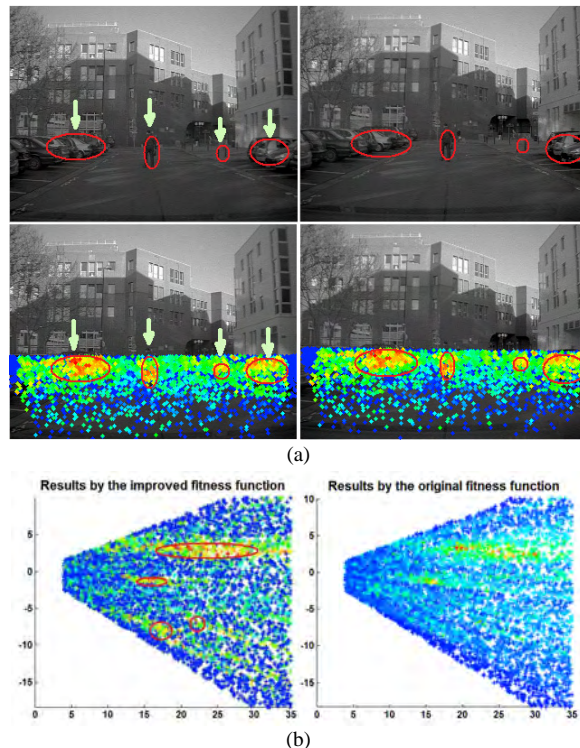


Fig.2. (a) fitness values of the flies indicated by different colors (results by the improved fitness function); (b) comparison between the performances of the improved fitness function and the original fitness function

V. RESULTS

A. The evaluation of the improved fitness function and comparison with the original one

3000 flies are initialized randomly in the detection field. The fitness values of the flies are computed via the improved fitness function Eq.(2). All the flies are displayed in the image coordinates (Fig.2 (a)) and in the world coordinates (the left part of Fig.2 (b), in bird-eye view); the fitness value of each fly is indicated by its color: the higher the fitness value is, the warmer the color is (the maximum and minimum fitness value are respectively indicated by pure red color and pure blue color). See areas marked by red circles, on the whole the color of flies around real objects (pedestrians, cars, poles) are apparently warmer than that of flies at non-object area. This shows that the improved fitness function is effective in distinguishing flies at object area from those at non-object area.

Fitness values are computed for the same flies via the original fitness function Eq.(1) (a constant is added to the denominator to avoid it being zero). Fitness values computed via Eq.(1) are also indicated by different colors like before;

see the right part of Fig.2 (b). Compare the left part (results by the improved fitness function) with the right part (results by the original fitness function) of Fig.2 (b); qualitatively speaking, the color differences between flies at object area and flies at non-object area are more prominent in the left part than in the right part of Fig.2 (b). In other words, the improved fitness function is more effective than the original fitness function in distinguishing flies at object area from those at non-object area.

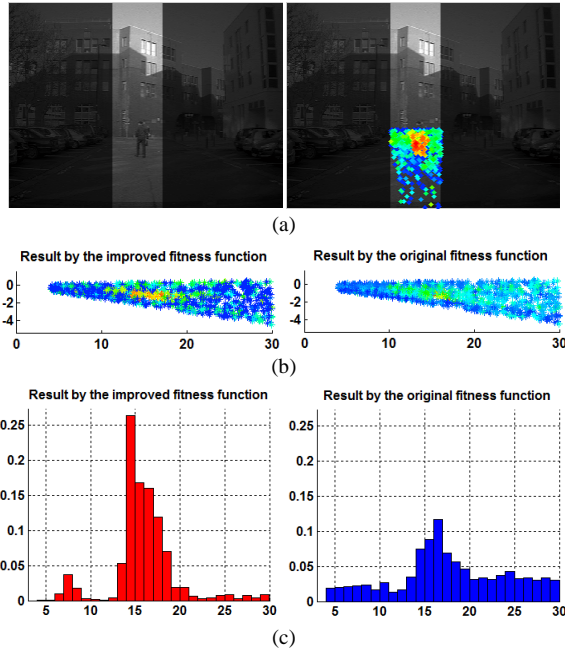


Fig.3. (a) a view-angle which contains a pedestrian; (b) fitness values of the flies indicated by different colors; (c) histograms according to distance

For quantitative comparison, a view-angle which contains a pedestrian is chosen; see Fig.3 (a). For this view-angle, compute a histogram from the flies within it according to the distance; the vote is weighted by the fitness value of the fly. First, fitness value is computed via the improved fitness function and the histogram obtained is displayed in the left part of Fig.3 (c); then fitness value is computed via the original fitness function and the histogram obtained is displayed in the right part of Fig.3 (c). The ground-truth position of the pedestrian is (15.52, -1.10, 0), i.e. 15.52 meters ahead of the vehicle. It can be seen that the votes of histogram in the left part of Fig.3 (c) are more concentrated to where the pedestrian is than in the right part of Fig.3 (c).

The root mean square distance (weighted by the fitness value of each fly) from each fly within the view-angle to the pedestrian is computed to be 2.94 meter if the improved fitness function is used while 6.86 meter if the original fitness function is used. The root mean square distance associated with the improved fitness function is noticeably smaller than that associated with the original fitness function. This quantitative result also shows that the improved fitness function is more effective than the original fitness function in

distinguishing flies at object area from those at non-object area.

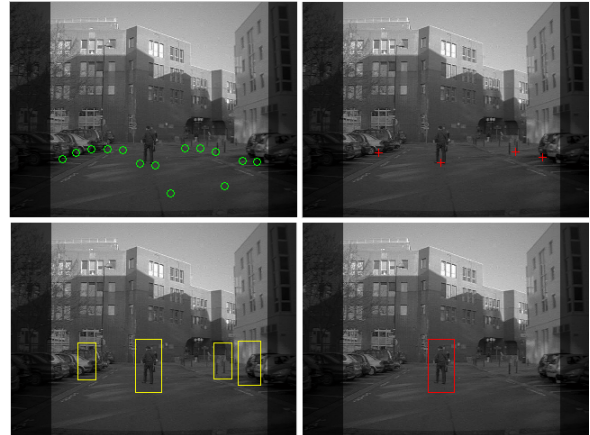


Fig.4. the process of the concentrating technique and its application to pedestrian detection

B. The performance of the concentrating technique

The concentrating technique consists of two steps: histogram based sampling and mean-shift. See the example shown before (Fig.3), the histogram based sampling step can have a rough localization result of OOI, i.e. (14.50, -0.94, 0) which is close to the ground-truth value (15.52, -1.10, 0) and can be served as good starting point for the mean-shift step. Nevertheless, this localization result still has considerable error; thus the mean-shift step is needed to advance the localization result to more accurate one. For this example, the localization result after mean-shift is (15.43, -1.13, 0) which is much closer to the ground-truth value.

For the whole image, see Fig.4 for example. As shown in the top-left part, a group of sample points generated by the histogram sampling step are marked by green circles. Some of these sample points are close to OOI, some are not. During the mean-shift step, these sample points either converge to OOI (with high fitness value density) or converge to somewhere with low fitness value density; those sample points which converge to the latter case are discarded while others are kept. The final outputs of the mean-shift step are marked by red crosses as shown in the top-right part of Fig.4; the positions of these red crosses are consistent with the positions of OOI like the pedestrian, the cars and the poles in the environment. This shows the effectiveness of the concentrating technique in localizing OOI.

C. Pedestrian detection

The outputs of the improved flies method are the 3D positions of several OOI, which can be used then to generate several ROI on the image; see the down-left part of Fig.4, the cars on both side, the poles on the right side and the pedestrian in the middle are detected by the improved flies method and several ROI boxes are generated around them. A HOG-based feature vector is computed for each box and then fed to a pre-trained SVM for classification; positive results are regarded as pedestrians, as shown in the down-right part of

Fig.4. More results are demonstrated in Fig.5.

D. Computational efficiency

The whole work is implemented in C++ in windows operating system; the CPU is 2.0GHz and the RAM is 2.0GB. The step of classification takes almost no time (<10ms) because the SVM classifier only has to deal with few ROI. The evolution part consumes most computational time which depends on the number of evolutionary iterations. It is true that more evolutionary iterations will drive flies onto more detailed 3-D structures (useful for 3-D construction) and also take more computational time. But for our application, two or three evolutionary iterations are normally enough for locating OOI, which take no more than 150ms. This computational efficiency can satisfy real-time demand.

VI. CONCLUSION

This paper presents an improved version of the genetic evolutionary “flies” method. An improved fitness function is proposed instead of the original one; a concentrating technique is incorporated into the architecture of the flies method. Experimental results are given for validating the improved performance brought by the proposed fitness function and for showing the effectiveness of the improved flies method in localizing OOI. In combination with a SVM classifier, the improved flies method is applied to pedestrian detection as a sub-part of the French national project “LOVE”.

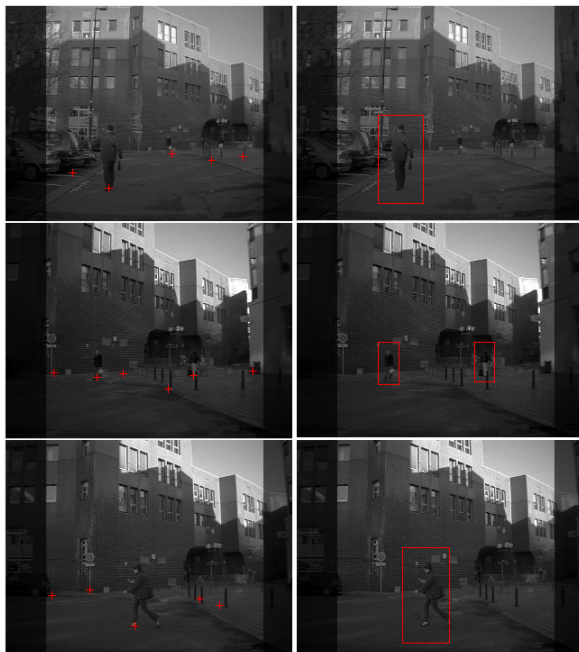


Fig.5. pedestrian detection

The strategy of using the fusion between camera and laser scanner for pedestrian detection has been showing promising results [4]. One merit of this strategy is that the laser scanner can be used to offer reliable localization results of OOI and reliable ROI on the image. The improved flies method presented in this paper, if considered from its function,

can be regarded as a laser scanner, while the device (stereo-vision) needed for it is considerably cheaper than a laser scanner. On the other hand, there are still spaces for improvements. By so far, size information of OOI located can not be offered; besides, occasionally few objects in the detection field will be missed, without being detected as OOI. To be able to offer size information of OOI and to reduce the rate of relevant objects undetected are the direction for further research.

ACKNOWLEDGEMENT

This research is supported by the French national project “LOVE” (Logiciels d’Observation des Vulnerables).

REFERENCE

- [1] Demonceaux C, Kachi-Akkouche D. Robust obstacle detection with monocular vision based on motion analysis. In: IEEE Intelligent Vehicles Symposium. Parma, Italie: 2004, pp.527-532
- [2] Giachetti A, Campani M, Torre V. The use of optical flow for road navigation. IEEE Transactions on Robotics and Automation, 1998, 14(1): 34-48
- [3] Gate G, Nashashibi F. Using targets appearance to improve pedestrian classification with a laser scanner. In: IEEE Intelligent Vehicles Symposium. Eindhoven, Netherlands: 2008, pp.571-576
- [4] Gate G, Breheret A, Nashashibi F. Centralized fusion for fast people detection in dense environment. In: IEEE International Conference on Robotics and Automation. Kobe, Japan: 2009, pp.76-81
- [5] Faugeras O. Three-dimensional computer vision: a geometric approach. Cambridge, MA :MIT Press, 1993
- [6] Li Y, Toulminet G, Benschair A. Vehicle detection based on the stereo concept of (axis, width, disparity) symmetry map. In: Proceedings of the 11th International IEEE Conference on Intelligent Transportation Systems. Beijing, China: 2008, pp.778-783
- [7] Miled W, Pesquet JC, Parent M. Robust obstacle detection based on dense disparity maps. In: Proceedings of the 11th International Conference on Computer Aided Systems Theory EUROCAST 2007, 2007, pp.1142-1150
- [8] <http://love.univ-bpclermont.fr>
- [9] Toulminet G, Mousseot S, Benschair A. Fast and accurate stereo vision-based estimation of 3D position and axial motion of road obstacles. International Journal of Image and Graphics, 2004, 4(1):1-27
- [10] Louchet J. Using an individual evolution strategy for stereovision. Genetic programming and evolvable machines, 2001, 2(2):101-109
- [11] Louchet J, Guyon M, Lesot MJ, Boumaza A. Dynamic flies: a new pattern recognition tool applied to stereo sequence processing. Pattern Recognition Letters 23, 2002, pp.335-345
- [12] Comaniciu D, Meer P. Mean shift: a robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24(5): 1-18
- [13] Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1. Washington DC, USA: IEEE Computer Society, 2005, pp.886-893



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session II

Path Planning & Navigation systems

- **Keynote speaker: Alexandre Bayen (Berkeley University, USA)**
Title: Mobile Millenium
- **Title: Optimal Vehicle Routing and Scheduling with Precedence Constraints and Location Choice**
Authors: G. Ayorkor Korsah, Anthony Stentz, and M. Bernardine Dias, and Imran Fanaswala
- **Title: Multi-Agent Planning and Simulation for Intelligent Transportation System**
Authors: Ming C. Lin, Jason Sewall, Jur Van den Berg, David Willkie, Dinesh Manocha



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session II

Keynote speaker: **Alexandre Bayen (Berkeley University, USA)**

Mobile Millennium: using smartphones to monitor traffic in privacy aware environments

Abstract : This talk describes how the mobile internet is changing the face of traffic monitoring at a rapid pace. In the last five years, cellular phone technology has bypassed several attempts to construct dedicated infrastructure systems to monitor traffic. Today, GPS equipped smartphones are progressively morphing into an ubiquitous traffic monitoring system, with the potential to provide information almost everywhere in the transportation network. Traffic information systems of this type are one of the first instantiations of participatory sensing for large scale cyberphysical infrastructure systems.

However, while mobile device technology is very promising, fundamental challenges remain to be solved to use it to its full extent, in particular in the fields of modeling and data assimilation. The talk will present a new system, called Mobile Millennium, launched recently by UC Berkeley, Nokia and Navteq, in which the driving public in Northern California can freely download software into their GPS equipped smartphones, enabling them to view traffic in real time and become probe vehicles themselves.

The smartphone data is collected in a privacy-by-design environment, using spatially aware sampling. Using data assimilation, the probe data is fused with existing sensor data, to provide real time estimates of traffic.

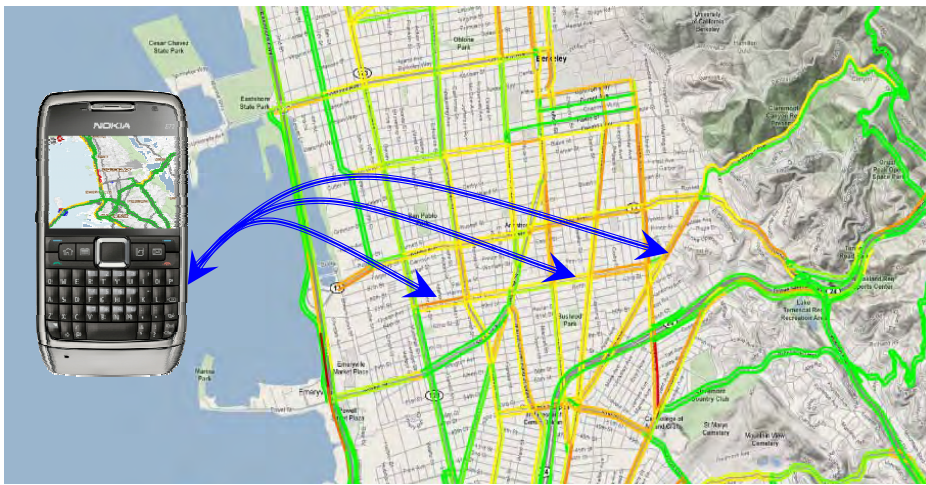
Results from experimental deployments in California and New York will be presented, as well as preliminary results from a pilot field operational test in California, with already more than 5,000 downloads.

Biography: Alexandre Bayen received the Engineering Degree in applied mathematics from the Ecole Polytechnique, France, in July 1998, the M.S. degree in aeronautics and astronautics from Stanford University in June 1999, and the Ph.D. in aeronautics and astronautics from Stanford University in December 2003. He was a Visiting Researcher at NASA Ames Research Center from 2000 to 2003. Between January 2004 and December 2004, he worked as the Research Director of the Autonomous Navigation Laboratory at the Laboratoire de Recherches Balistiques et Aerodynamiques, (Ministere de la Defense, Vernon, France), where he holds the rank of Major. He has been an Assistant Professor in the Department of Civil and Environmental Engineering at UC Berkeley since January 2005. He is the recipient of the Ballhaus Award from Stanford University, 2004. His project Mobile Century received the 2008 Best of ITS Award for 'Best Innovative Practice', at the ITS World Congress. He is the recipient of the CAREER award from the National Science Foundation, 2009. Mobile Millennium has been featured already on CBS, NBC, ABC, CNET, NPR, KGO, and the BBC.



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

MOBILE MILLENNIUM



Alexandre Bayen
Systems Engineering, CEE, UC Berkeley
<http://traffic.berkeley.edu>



The edge of the “classical” sensing paradigm

In the “classical” sensing paradigm, we

- Use the best / most accurate sensors
- Deploy a dedicated monitoring and data gathering infrastructure
- Maximize coverage of the sensor network
- Maximize autonomy of the platform (robots)

Example: autonomous water sensors patrolling water distribution infrastructure

- Floating robots
- Built for a specific application
- Goal: no human involved
- Possess suite of sensors
 - Salinity
 - Turbidity
 - Contaminants
 - Etc.
- 2010 Target: 100 autonomous in the Sacramento Delta and San Francisco Bay



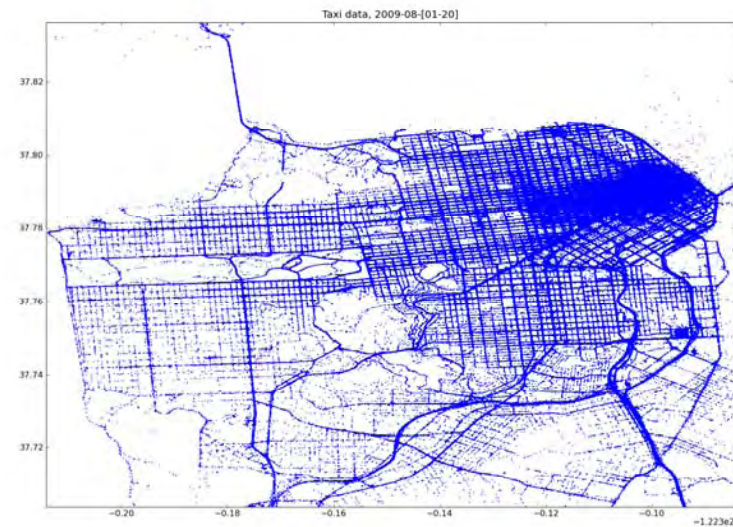
User generated content: sensor-free sensing

An example in which user generated content can be more efficient than the proper tool.

- An appropriate sensor to measure an earthquake is an accelerometer
- USGS has deployed sensor arrays in the US
- Today, most smartphones have accelerometers
 - Using them for earthquake monitoring is challenging
 - Deploying a dedicated infrastructure is expensive
- Yet Twitter can provide first alert of the type “Did you feel it” based on user generated content

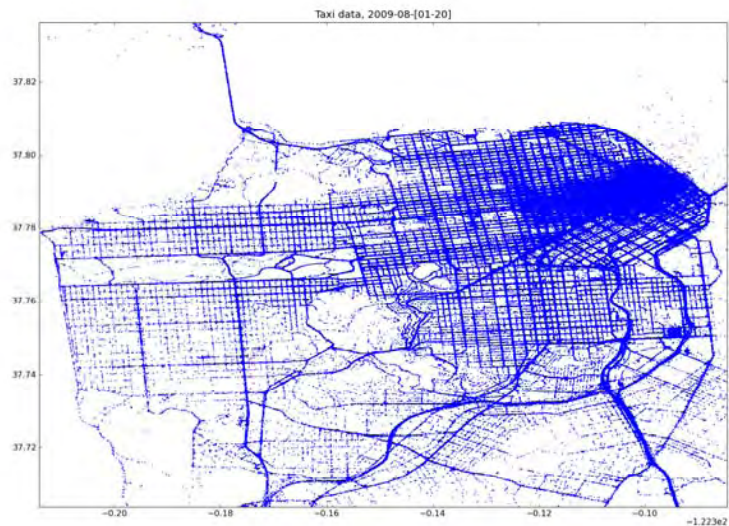


One day of Mobile Millennium data (SF taxis)

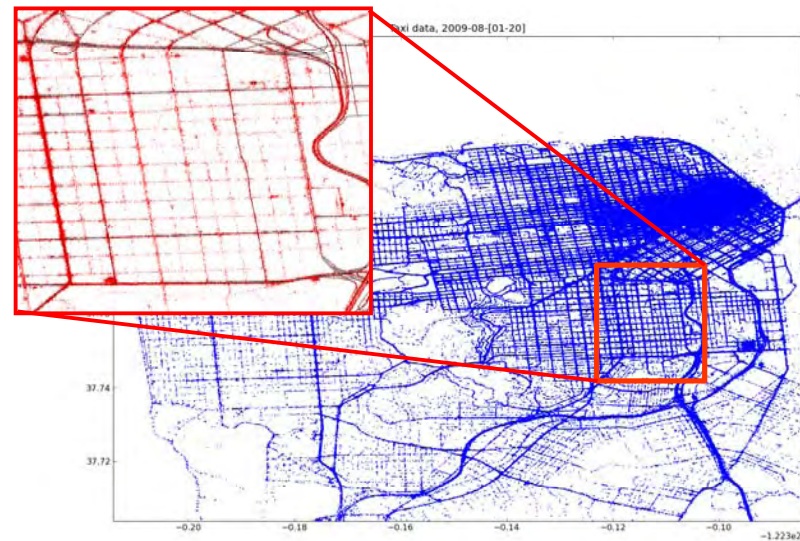




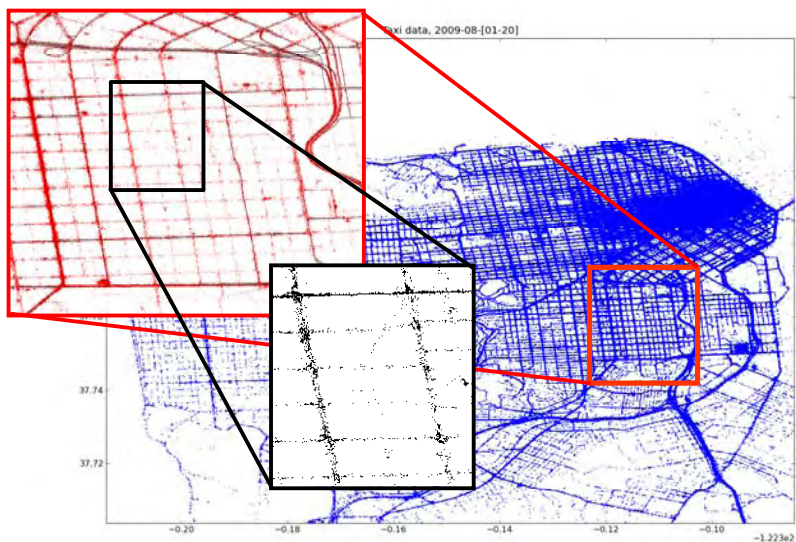
One day of Mobile Millennium data (SF taxis)



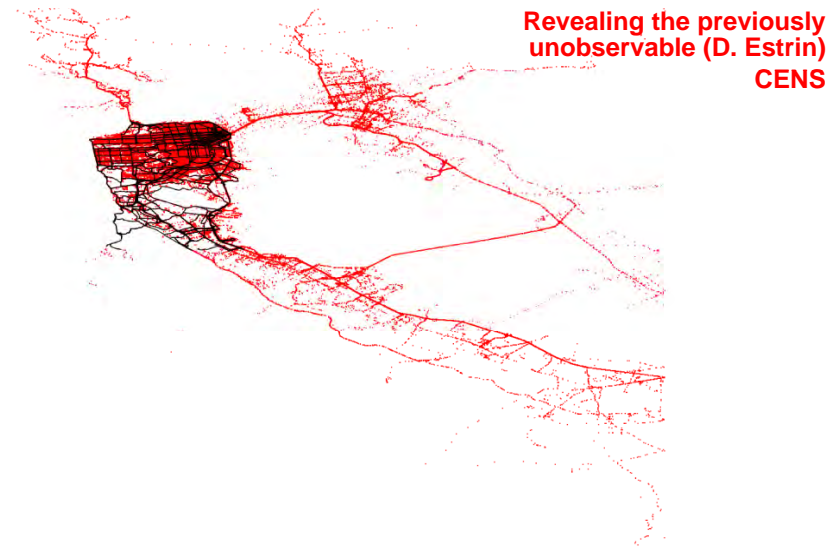
One day of Mobile Millennium data (SF taxis)



One day of Mobile Millennium data (SF taxis)



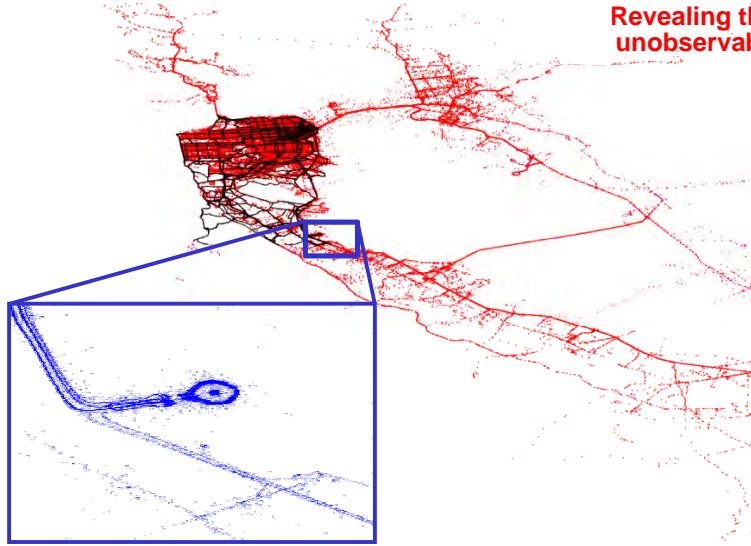
One day of Mobile Millennium data (SF taxis)



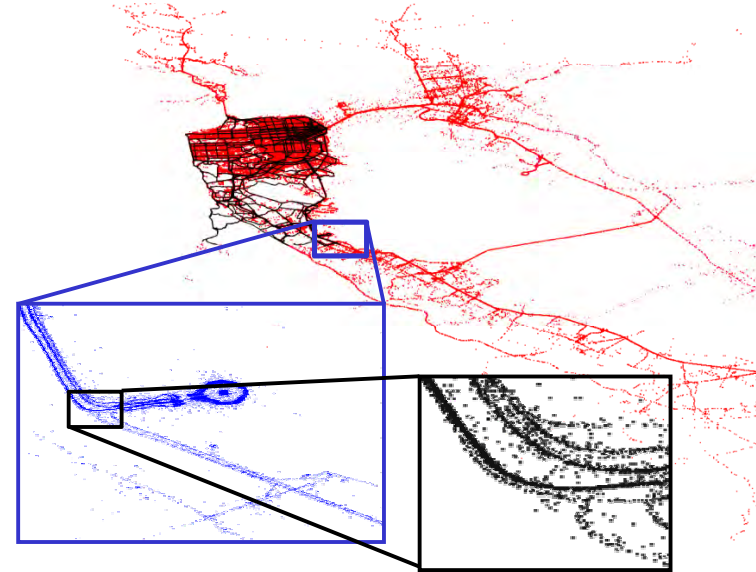


One day of Mobile Millennium data (SF taxis)

Revealing the previously unobservable (D. Estrin)
CENS



One day of Mobile Millennium data (SF taxis)



Societal need for [traffic] information systems

Rough estimates of congestion impacts

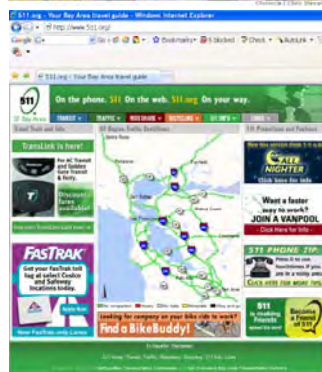
- 4.2 billion hours extra travel in the US
- Accounts for 2.9 billion gallons of fuel
- Congestion cost of 78 billion dollars

[2007 Urban Mobility Report, September 2007, Texas Transportation Institute, David Schrank & Tim Lomax]



Traffic information systems

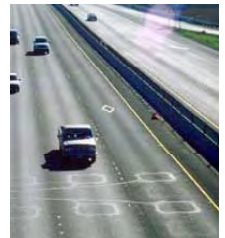
- Call in numbers (511)
- Changeable message signs (CMS)
- Online navigation devices
- Web-based commuter services:
 - www.511.org
 - www.traffic.com
 - Google Traffic
- Cellular phone web browsing
- Connected aftermarket devices



Source of today's traffic information

Dedicated traffic monitoring infrastructure:

- Self inductive loops
- Wireless pavement sensors
- FasTrak, EZ-pass transponders
- Cameras
- Radars
- License plate readers



Issues of today's dedicated infrastructure

- Installation costs
- Maintenance costs
- Reliability
- Coverage
- Privacy intrusion



Web 2.0 on wheels

Emergence of the mobile internet

- Internet accesses from mobile devices skyrocketing
- Mobile devices outnumber PCs by 5:1
- 1.5 million devices/day (Nokia)
- Redefining the mobile market: Google, Apple, Nokia, Microsoft, Intel, IBM, etc.
- Open source computing: Symbian Foundation, Android, Linux

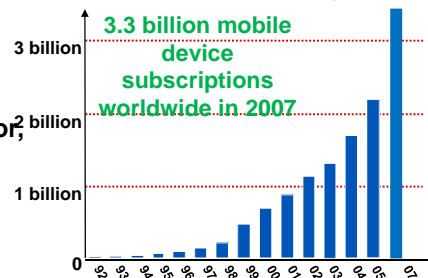


Sensing and communication suite

- GSM, GPRS, WiFi, bluetooth, infrared
- GPS, accelerometer, light sensor, camera, microphone

Smartphones and Web 2.0

- Context awareness
- Sensing based user generated content



[Courtesy J. Shen, Nokia Research Center Palo Alto]

The notion of probe

Probe historically has made extensive use of fleet vehicles

- INRIX's fleet includes approx. 1M vehicles
- NAVTEQ's fleet data ~40M data points / day
- Historically, sampling rate has been low (up to 15 mins or more)
- 2-way GPS (Dash-type)

Emergence of the Mobile Internet, two years of rapid expansion...

- Non GPS based solutions (Airsage type)
- GPS-enabled smartphone software clients: Mobile Millennium, Google Android, INRIX, etc. → participatory sensing.

... until concerns about distracted driving will reshape the landscape

- Phone can act as a data collector, interface, gateway
- Integration of the phone into the car infrastructure
 - Docking stations (BMW)
 - Bluetooth (plus eventually ODB-II)
- Fusion of phone and car will lead to car centric infrastructure

Smartphone/probe in this talk should be understood in the broad sense, not (only) phone-centric

The notion of probe

Probe historically has made extensive use of fleet vehicles

- INRIX's fleet includes approx. 1M vehicles
- NAVTEQ's fleet data ~40M data points / day
- Historically, sampling rate has been low (up to 15 mins or more)
- 2-way GPS (Dash-type)

Emergence of the Mobile Internet, two years of rapid expansion...

- Non GPS based solutions (Airsage type)
- GPS-enabled smartphone software clients: Mobile Millennium, Google Android, INRIX, etc. → participatory sensing.

... until concerns about distracted driving will reshape the landscape

- Phone can act as a data collector, ~~interface~~, gateway
- Integration of the phone into the car infrastructure
 - Docking stations (BMW)
 - Bluetooth (plus eventually ODB-II)
- Fusion of phone and car will lead to car centric infrastructure

Smartphone/probe in this talk should be understood in the broad sense, not (only) phone-centric

A cyberinfrastructure for participatory sensing

Sensing

- Millions of mobile devices as new sources for data

Communication

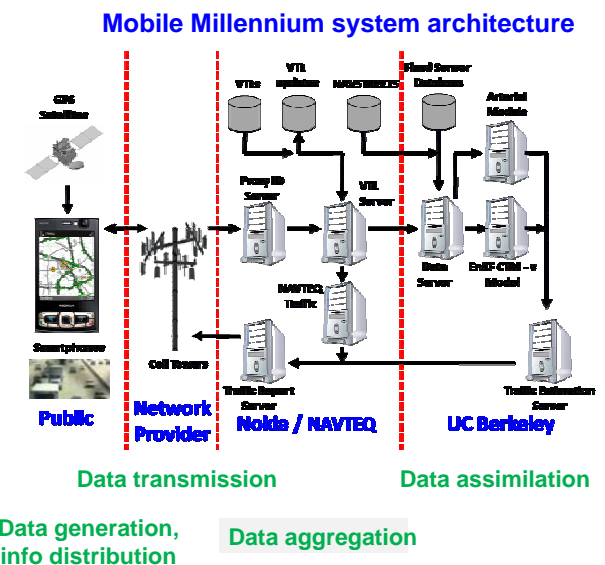
- Existing cell phone infrastructure to collect raw data and receive traffic information

Data assimilation

- Real-time, online traffic estimation

Privacy Management

- Encrypted transactions
- Client authentication
- Data anonymization



Privacy issues for location based services

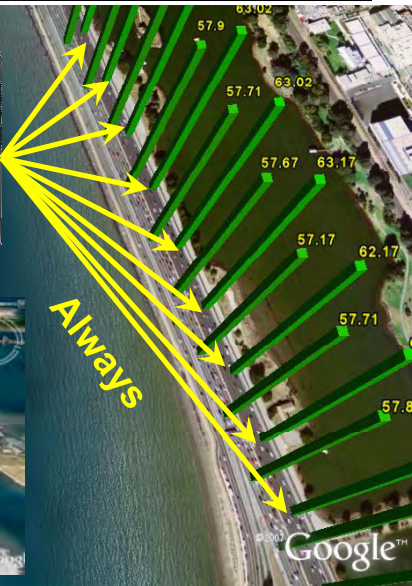


Phone can reveal

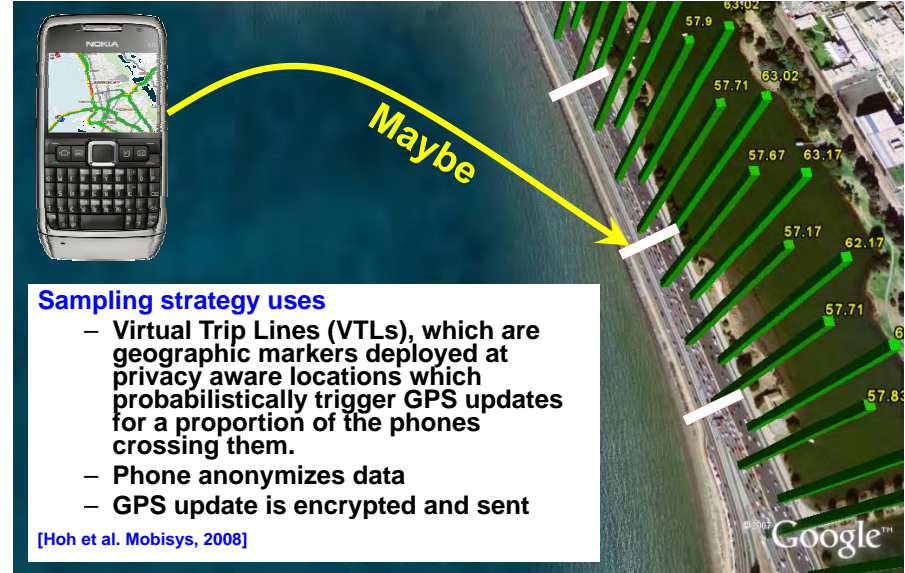
- GPS tracks
- Precise location
- Individual patterns
- User generated content
- Personal info
- Activity
- ...



Always



Data collection: spatially aware traffic monitoring



Sampling strategy uses

- Virtual Trip Lines (VTLs), which are geographic markers deployed at privacy aware locations which probabilistically trigger GPS updates for a proportion of the phones crossing them.
- Phone anonymizes data
- GPS update is encrypted and sent

[Hoh et al. Mobisys, 2008]

Nationwide deployment of a virtual infrastructure



Privacy preserving data collection infrastructure includes

- 450,000 VTLs in use for Northern California, 4.5M for the US
- 5,000 pilot users in Northern California (Nokia, Blackberry)



Illustration: trajectory based vs. VTL based



Mobile Century test (100 cars)

- Feb. 8, 2008, 100 cars, 165 students, 10 hrs, 10 miles
- Movie shows "helicopter view of 100 cars experiment)
- GPS log shown every 3 seconds for each car
- VTL subset of the data (2% of total traffic is enough for traffic reconstruction)

New York test (20 cars)

- Test was
- Only VTL data is shown
- Data is sufficient for a
- Data is gathered in a privacy aware environment





Tradeoffs (summary)

Privacy considerations

- Sampling strategy
- Randomization of the sampling
- Anonymization of the data
- Encryption of the data
- Local computations (onboard the phone)

Accuracy considerations, and data to push the estimation further

- Full trajectory collection
- Re-identification (day to day)
- Additional context collection (jam, pictures, other sensors)
- Tower information



Hardness of estimation

Privacy intrusion

Estimation results

- Patterns
- Travel time
- Aggregate data

Individual trajectories

- Inference from patterns
- Reconstruction?



Additional questions (beyond data collection)

The emphasis of the talk was only on data collection and some of the processing. This is only part of the problem. Privacy can be compromised / protected in several different other ways:

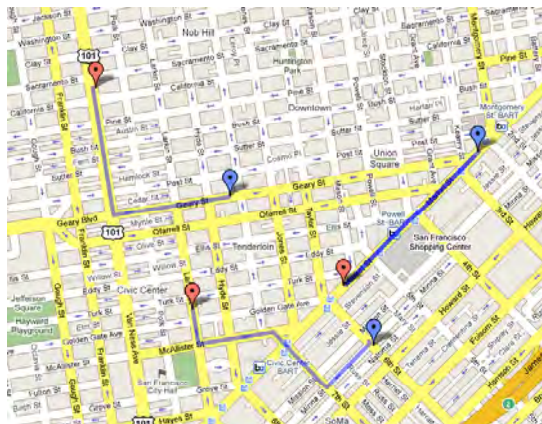
- Identification (voluntary or not, disclosure, self incrimination)
- Structure of the data sharing graph (social networks), distribution procedure
- Correlation with other sources
- Inference mechanisms
- Additional user generated content
- Chain: data creation, sharing, storage, publishing



Path inference

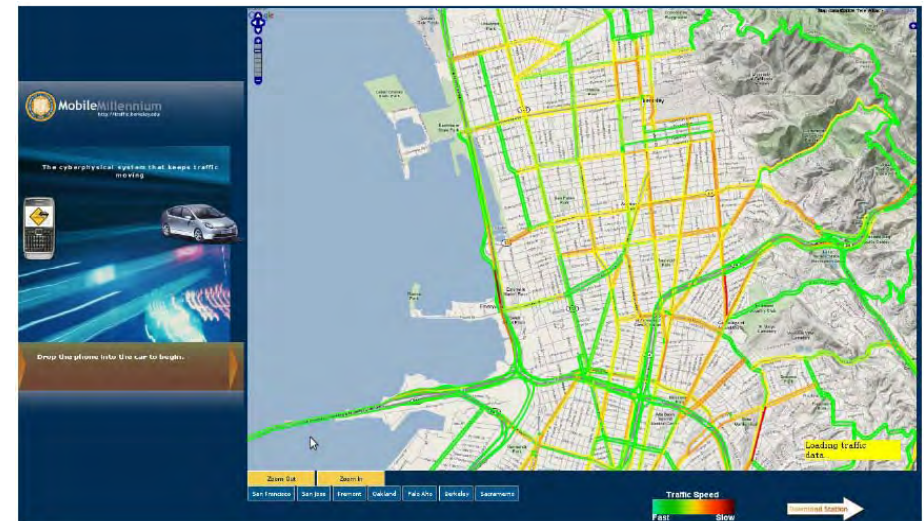
Path inference needs to be used to extract useful information from travel time between any two road locations. Time scales vary from 1 to 15 minutes, space scale from 200m to several miles

- VTL (spatial)
- Taxi (temporal)
- Bluetooth (temporal)
- Bus (temporal)



Mobile Millennium today

Traffic monitoring system freely available to the public through our website on their phone, based on participatory sensing.

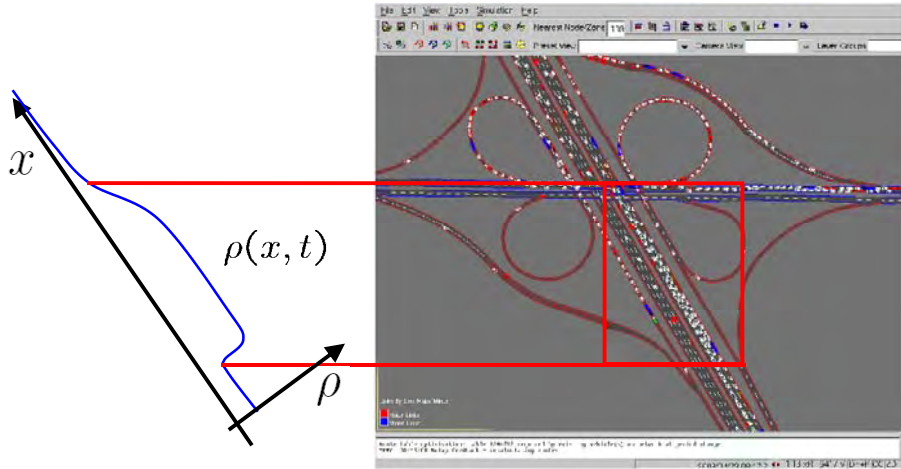


Modelling approaches to privacy aware sampling



Aggregation of data provides application-optimized privacy aware data collection infrastructure. In the case of traffic monitoring:

- Travel time
- Queue lengths, extent of traffic jams, clear up time, etc.

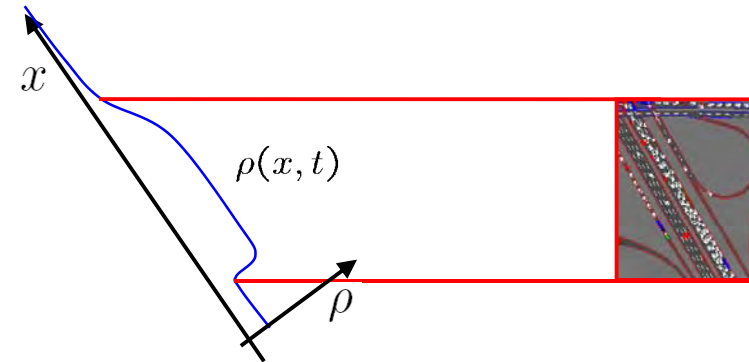


Modelling approaches to privacy aware sampling



Aggregation of data provides application-optimized privacy aware data collection infrastructure. In the case of traffic monitoring:

- Travel time
- Queue lengths, extent of traffic jams, clear up time, etc.

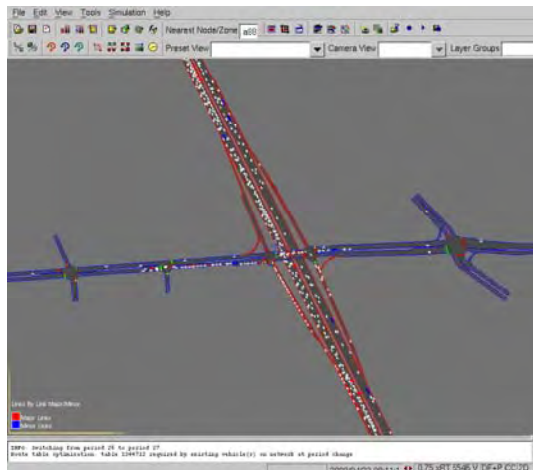
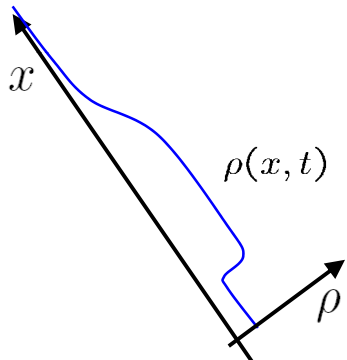


Modelling approaches to privacy aware sampling



Aggregation idea

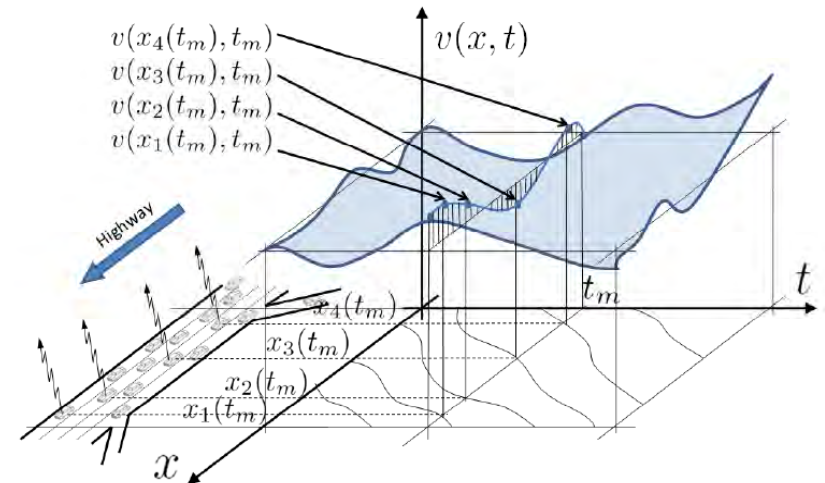
- Describe aggregate behavior (not individual behavior)
- Sample individual behavior accurately enough that it can be used to reconstruct aggregate behavior
- Additional benefit: serves computational efficiency



Data assimilation / inverse modeling



How to incorporate Lagrangian (trajectory based) and Eulerian (control volume based) measurements in a flow model.

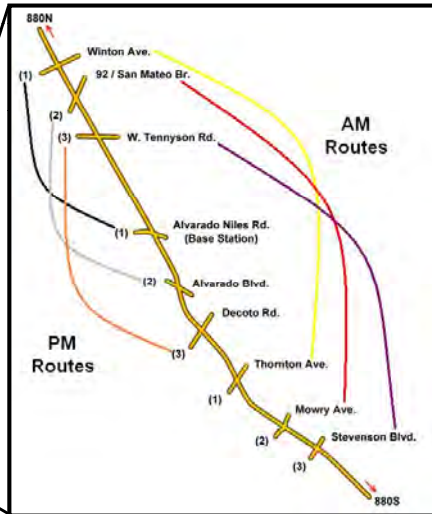
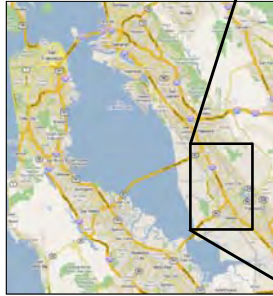




Prototype experiment: *Mobile Century*

Experimental proof of concept: the *Mobile Century* field test

- February 8th 2008
- I80, Union City, CA
- Field test, 100 cars
- 165 Berkeley students drivers
- 10 hours deployment,
- About 10 miles
- 2% - 5% penetration rate



29



Mobile Century validation video data collection

Video data:

- Vehicles counts
- Travel time validation



A glimpse of *Mobile Century* (February 8th, 2008)



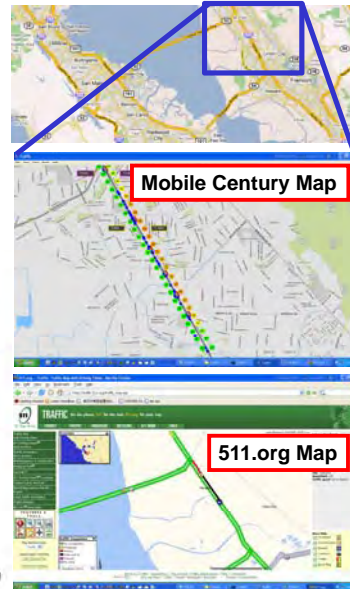
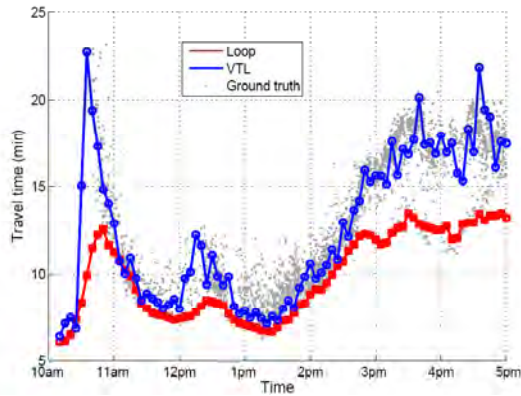
A glimpse of *Mobile Century* (February 8th, 2008)



Validation of the data (video)

Travel time predictions

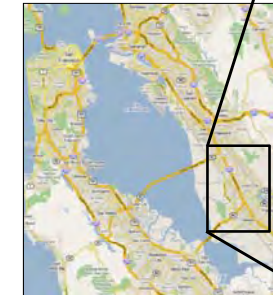
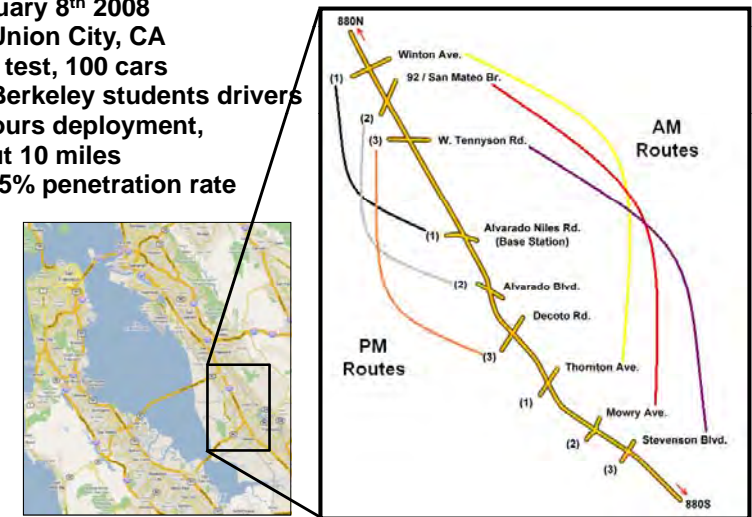
- Can be done in real time at a 2% penetration rate of traffic
- Proved accurate against data from www.511.org, with higher degree of granularity



Prototype experiment: *Mobile Century*

Experimental proof of concept: the *Mobile Century* field test

- February 8th 2008
- I80, Union City, CA
- Field test, 100 cars
- 165 Berkeley students drivers
- 10 hours deployment,
- About 10 miles
- 2% - 5% penetration rate



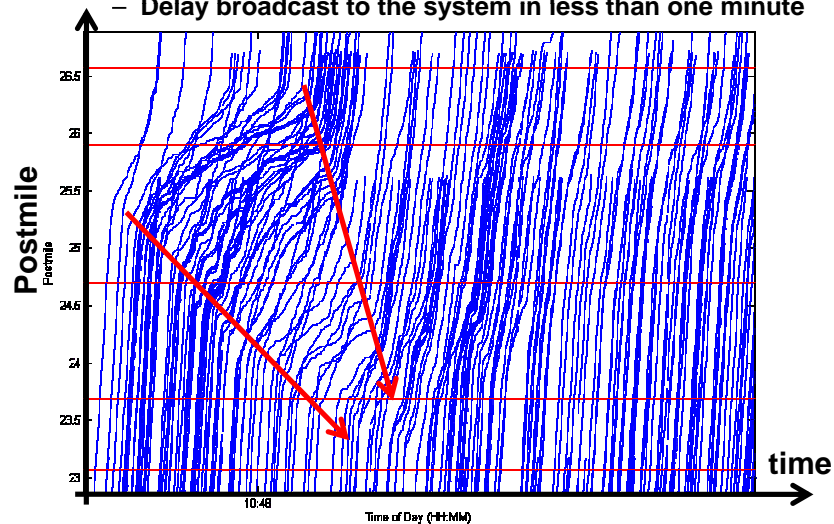
34



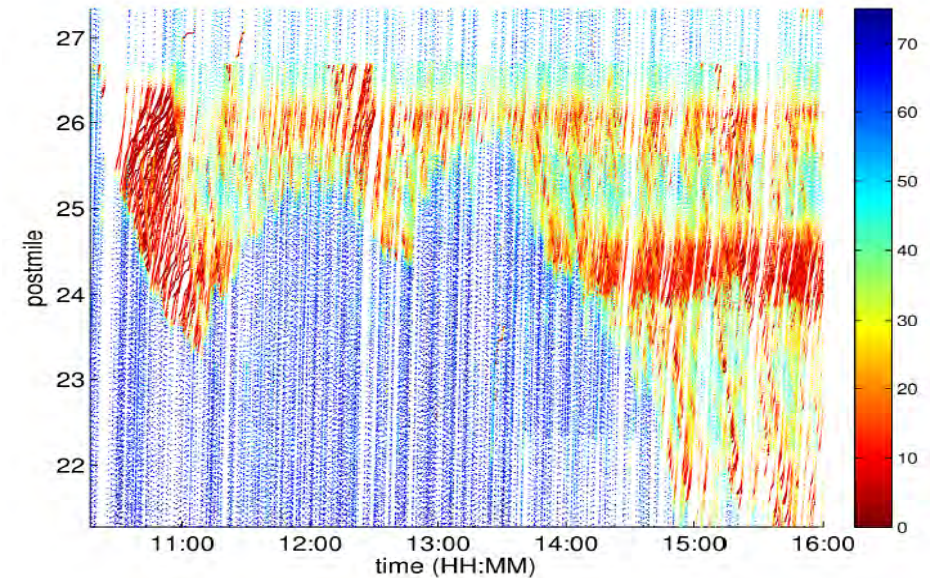
Revealing the previously unobservable

5 car pile up accident (not Mobile Century vehicles)

- Captured in real time
- Delay broadcast to the system in less than one minute

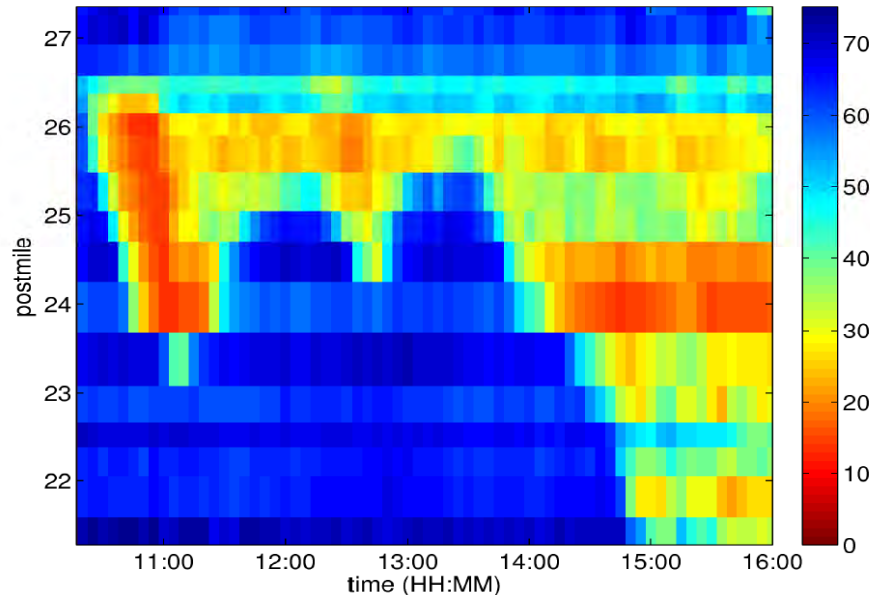


Granularity of the data (GPS data)





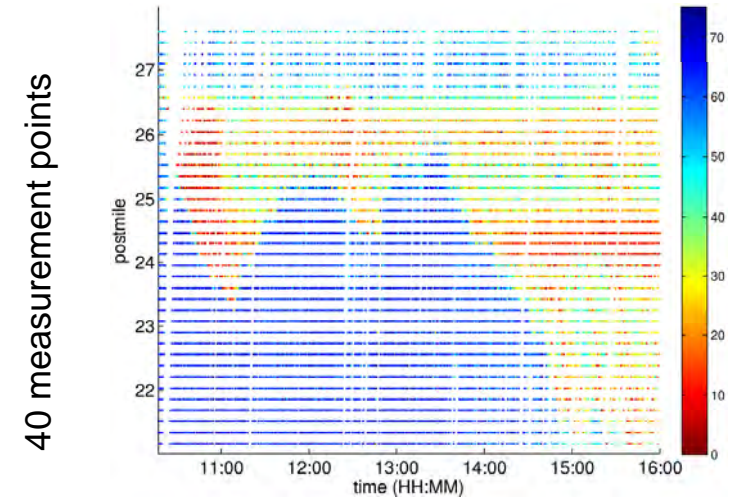
Granularity of the data (loops, for comparison)



Flow reconstruction (inverse modeling)

Physical model and data assimilation enable state estimation

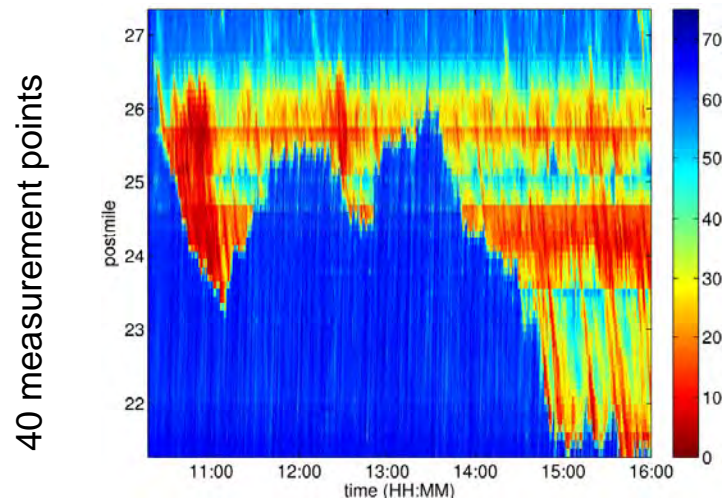
- Works even with low penetration rate
- Interpolation will just not do the job



Flow reconstruction (inverse modeling)

Physical model and data assimilation enable state estimation

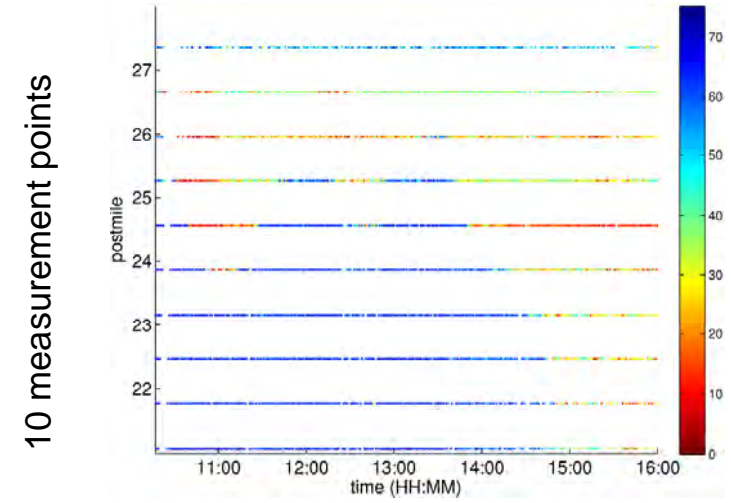
- Works even with low penetration rate
- Interpolation will just not do the job



Flow reconstruction (inverse modeling)

Physical model and data assimilation enable state estimation

- Works even with low penetration rate
- Interpolation will just not do the job

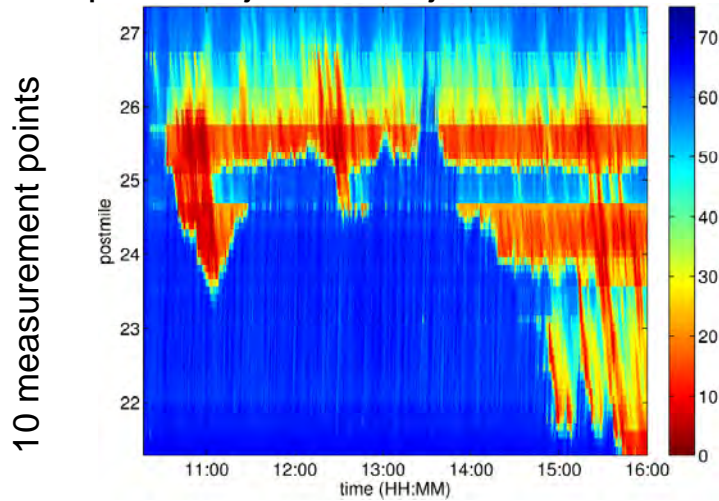




Flow reconstruction (inverse modeling)

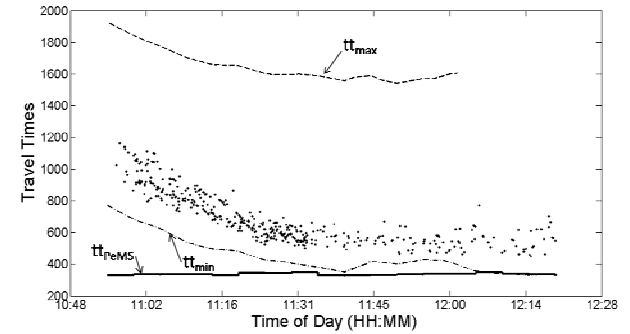
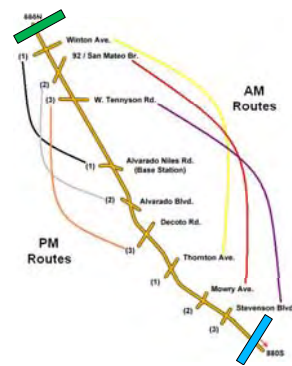
Physical model and data assimilation enable state estimation

- Works even with low penetration rate
- Interpolation will just not do the job



Bounds on travel time (PeMS)

- Outflow loop
- Inflow loop



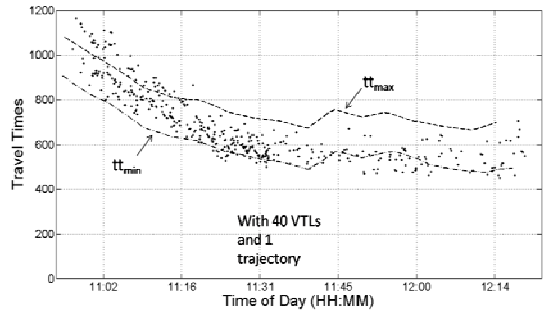
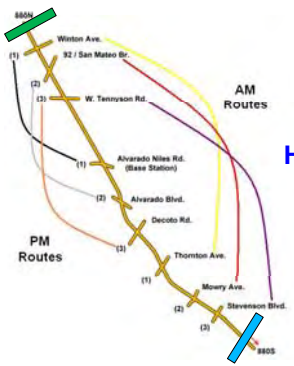
How to decrease the uncertainty on a parameter / state variable (travel time) with inverse modeling

- 2 sensors
- One accident in the middle, not captured by the sensors
- One can still estimate bounds on congestion



Bounds on travel time (PeMS and phones)

- Outflow loop
- Inflow loop



How to decrease the uncertainty on a parameter / state variable (travel time) with inverse modeling

- 2 sensors
- One accident in the middle, not captured by the sensors
- One can still estimate bounds on congestion



Mobile Millennium: a pilot project

Mission statement

The goal of *Mobile Millennium* is to establish the design of a system that collects data from GPS-enabled mobile phones, fuses it with data from existing sensors and turn it into relevant traffic information.

Mobile Millennium is a field operational test

- Deployment of thousands of cars on a network including **arterials**
- Participating users agree to **share** position and speed
- Phones receive **live information** on map application
- Project duration **12 months**
- Mobile Millennium was a **pilot**

Launch

Mobile Millennium was launched on November 10th, 2008, at 8:30am from the UC Berkeley campus, and was concluded on November 10th, 2009



The 1/10 multiplier and the 10 multiplier



Mobile Millennium Management team



Corporate and Government Relations
Tom West, Director, CCIT



Chief of Staff
JD Margulici, Senior Engineer



Project Manager
Joe Butler, Senior Software Engineer



Project Administrator
Coralie Claudel, Policy Analyst



Project Coordinator
Steve Andrews, Senior Policy Analyst

Post doctoral researchers, software eng. team



Software engineering team: Andre Lockhard, Daniel Edwards, Benson Chiu, Saneesh Apte, Xiaopang Pong, Yanli Li, Morgan Smith



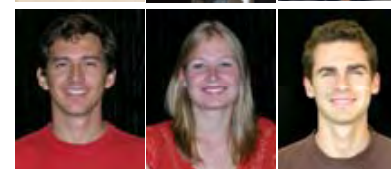
Post doctoral researchers:
Olli-Pekka Tossavainen
Jeff Ban



Graduate and undergraduate students



Graduate Students
Dan Work (CEE)
Chris Claudel (EECS)
Ryan Herring (IEOR)
Saurabh Amin (CEE)
Aude Hofleitner (EECS)
Sebastien Blandin (CEE)
Juan Carlos Herrera (CEE)
Tim Hunter (CS)

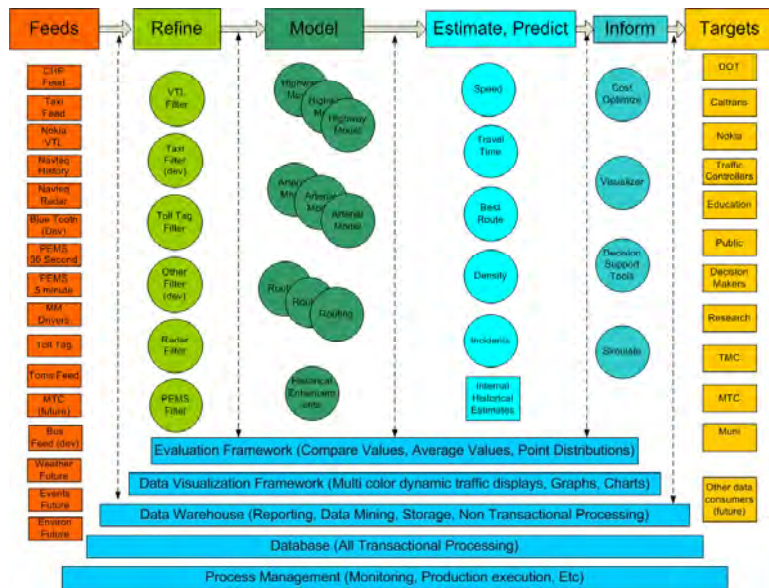


Visiting Grad. Students: Matthieu Nahoum (ENAC France), Vassili Lemaitre (ENAC France), Elena Agape (TU Aachen).

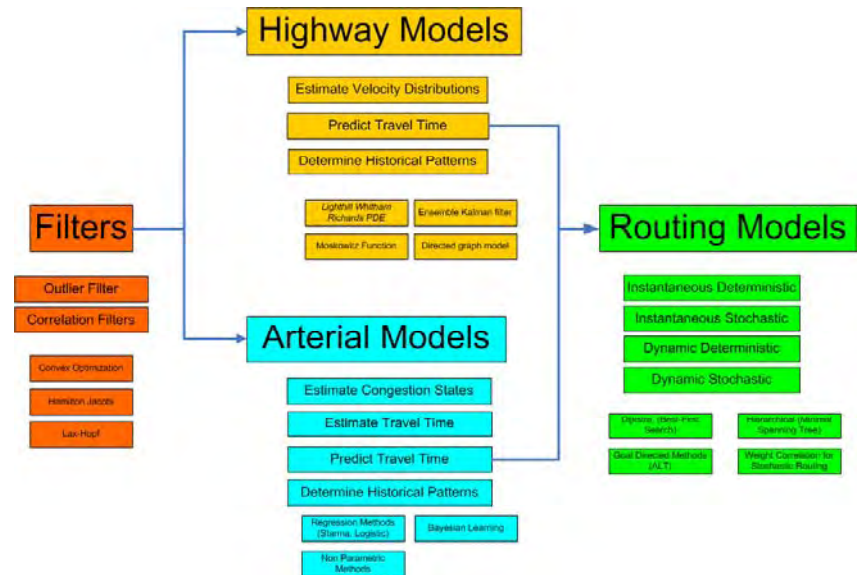


Undergraduate students: Sarah Stern (ME), Marcella Gomez (ME).

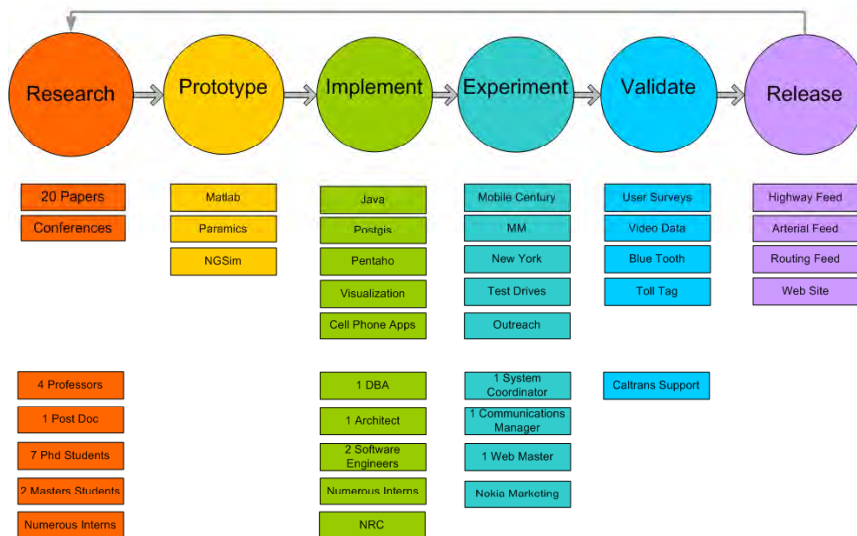
Data flow in the Mobile Millennium system



Example of process flow in the system



From academic research to product



Another example of data assimilation

Sacramento San Joaquin Delta

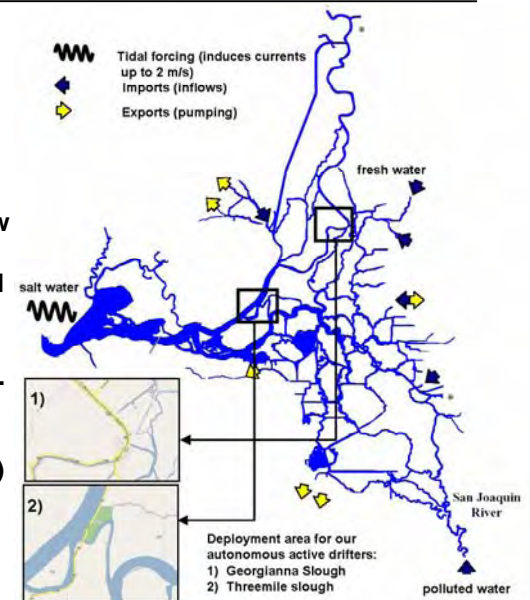
- Static sensors
- Own deployable sensors
- Floating sensors (phones)

Challenges

- Nowcast and forecast of flow and salinity in complex hydrodynamic regimes (tidal forcing and inversion)
- Providing data on factors which control fish migration.

Three deployment areas:

- Georgianna Slough (current)
- Grant line canal (current)
- 3-miles slough (future)



Traditional [Eulerian] sensing in the Delta

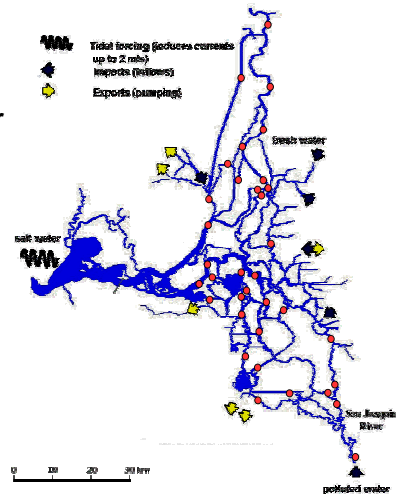


Few key locations:

50 sensors for 1000 km of channel

Inflexible (install once); expensive to install and maintain

Good for long term trends; not good for *localized* or *medium-term* phenomena

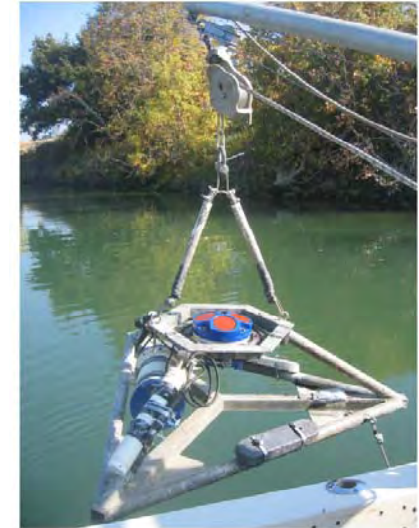


UC Berkeley [Eulerian] sensing



“Deployable” Eulerian sensors:

- Underwater sensors, autonomy 15 days.
- Measure cross sectional velocity and stage
- Need to be deployed from crane operated boat
- Need to be anchored to the ground (because of drift)
- Data upload after the experiment (no underwater comm.)



Mobile Berkeley floating sensor package

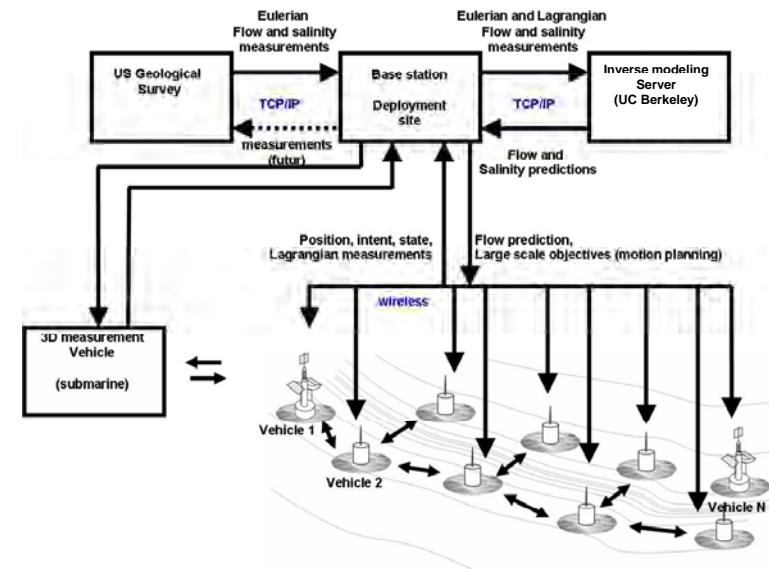


Mobile floating sensor

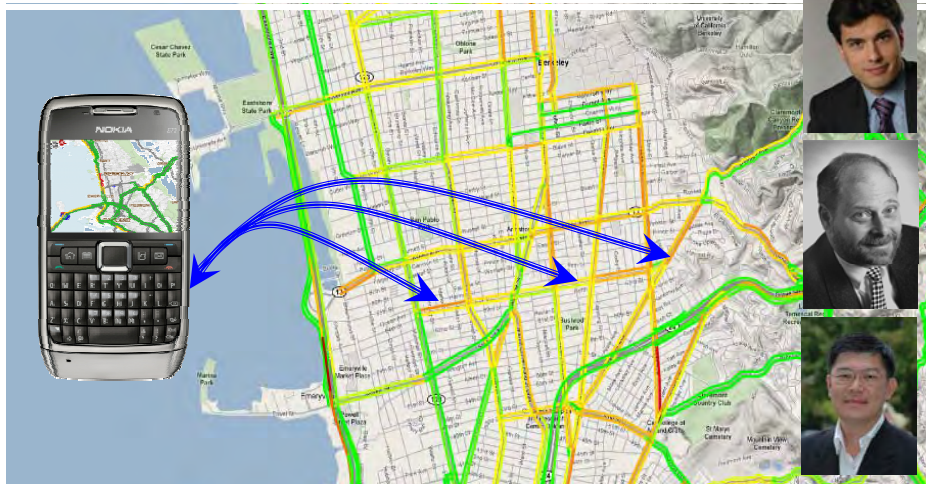
- Designed for manufacturability: standard parts, less custom machining
- Two propellers in differential drive configuration
- Internal water bag for buoyancy control
- First round of prototyping happening now; planning a fleet of 100



Water cyberinfrastructure



ClearSky

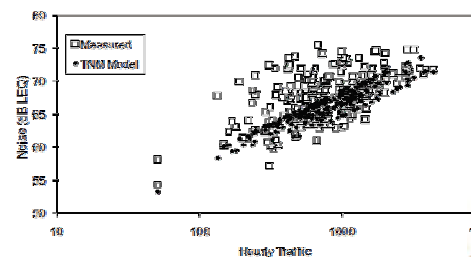


Alexandre Bayen, Steve Glaser, Edmund Seto
 Systems Engineering, CEE, School of Public Health, UC Berkeley
<http://traffic.berkeley.edu> <http://float.berkeley.edu>



Future goals (1): real time noise exposure

Inference of noise levels from traffic data (static, source: City of SF)
 – 1/6 of population in SF exposed to unhealthy noise levels



Street noise levels (Ldn dB)

- 0.0 - 55.0
- 55.1 - 60.0
- 60.1 - 65.0
- 65.1 - 70.0
- 70+

Seto, Holt, Rivard, and Bhatia, 2006



Future goals (1): real time noise exposure

Inference of noise levels from traffic data (static, source: City of SF)

- Production of noise maps for the urban network and in the vicinity of the highway network
- Real-time map based on dynamic data
- Can be used as a communication tool to rapidly inform the public



Future goals (1): real time noise exposure

Empirical data gap filling: how to understand presence of trucks?

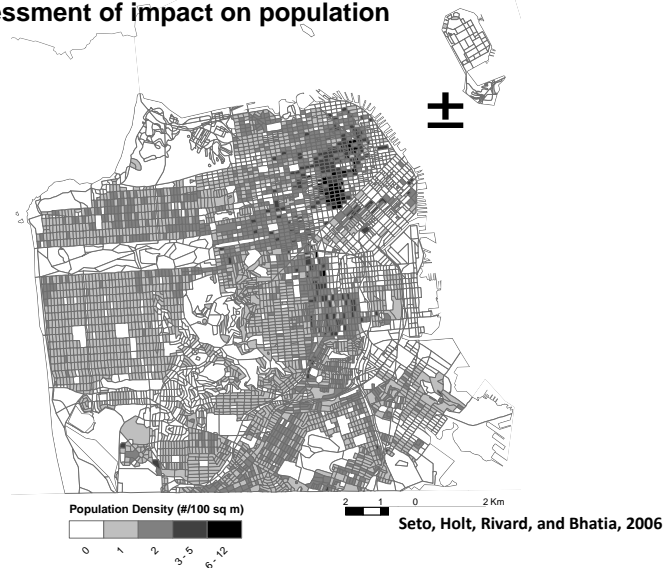
- Aerial photographs
- Spatial data



Future goals (1): real time noise exposure

Correlation of the assessment with population density:

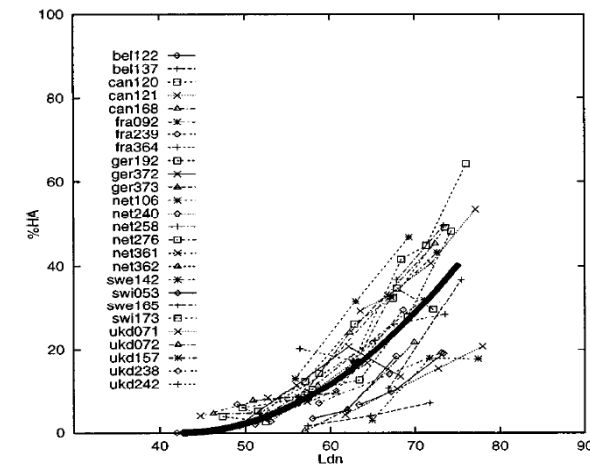
- Assessment of impact on population



Future goals (1): real time noise exposure

Impact maps on the population: building / using risk curves

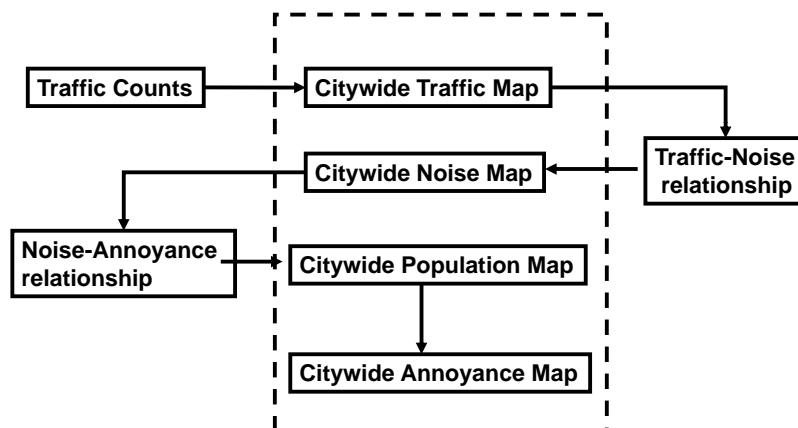
- Mortality rates can be inferred from these maps (other agents)
- In a security scenario: one hour death maps, two hours, etc.



Future goals (1): real time noise exposure

Integration process (noise, air pollution, airborne agent, etc.)

- Process will be integrated entirely in Mobile Millennium system
- Linked to population data to assess impact on population



Future goals (2): example PM2.5 aerosol

Assessment of exposure to aerosols (and other airborne agents)

- In the present case: emitted from cars, combustion, etc.
- Study can be replicated with sensor data

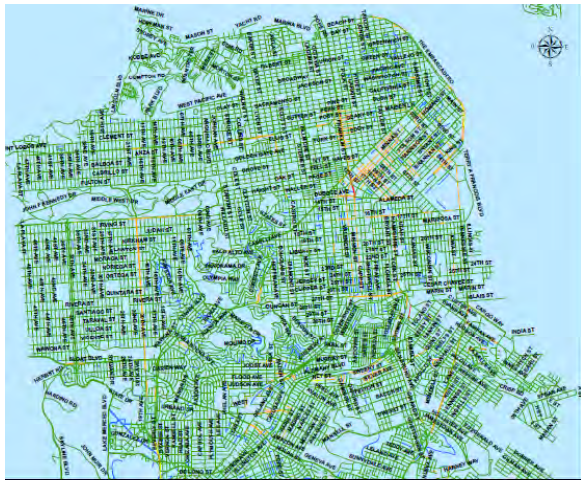




Future goals (2): example NOx aerosol

Assessment of exposure to aerosols (and other airborne agents)

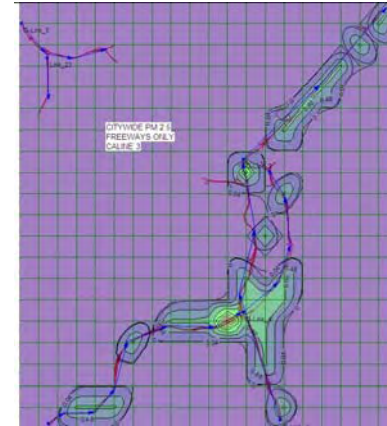
- In the present case: emitted from cars, combustion, etc.
- Study can be replicated with sensor data



Validation procedure

Scientific approach

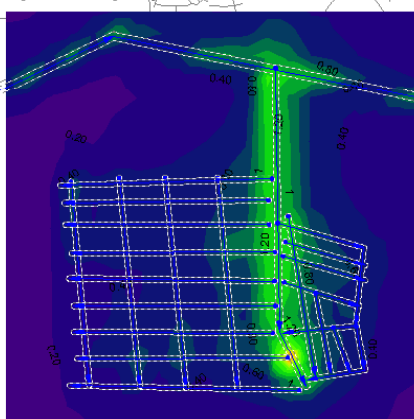
- Validation of the models (in the present case: Gaussian dispersion models)
- Model, assess concentration, measure, check that the measured data corresponds to the estimate
- Photographs: deployment of validation sensors.



Validation procedure

Scientific approach

- Validation of the models (in the present case: Gaussian dispersion models)
- Model, assess concentration, measure, check that the measured data corresponds to the estimate
- Photographs: deployment of validation sensors.



Research opportunities

Streaming data
Historical data

Data fusion
Data assimilation
Inverse modeling

Exposure
Assessment

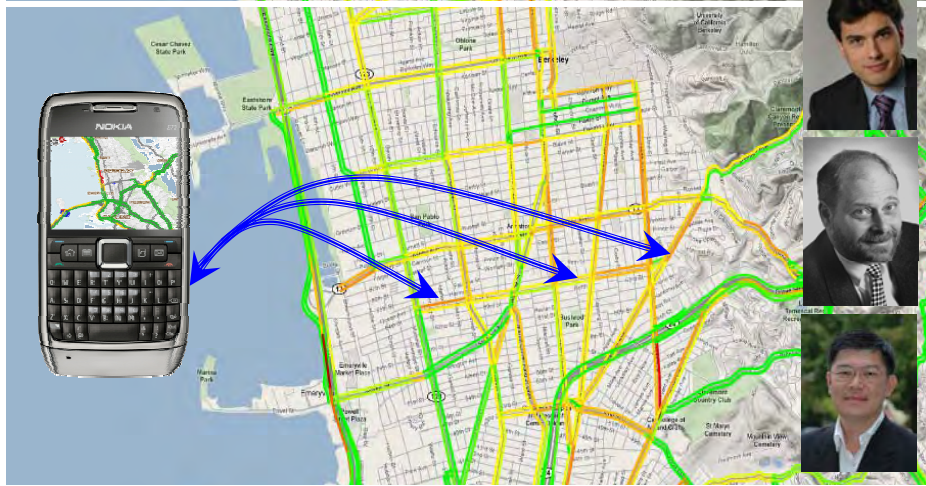
Framework for responding to potential health risks

Prediction

Preparedness

Response
Mitigation

Questions



Alexandre Bayen, Steve Glaser, Edmund Seto
Systems Engineering, CEE, School of Public Health, UC Berkeley
<http://traffic.berkeley.edu> <http://float.berkeley.edu>



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session II

Path Planning & Navigation systems

- **Title: Optimal Vehicle Routing and Scheduling with Precedence Constraints and Location Choice**
Authors: G. Ayorkor Korsah, Anthony Stentz, and M. Bernardine Dias, and Imran Fanaswala
- **Title: Multi-Agent Planning and Simulation for Intelligent Transportation System**
Authors: Ming C. Lin, Jason Sewall, Jur Van den Berg, David Willkie, Dinesh Manocha



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Optimal Vehicle Routing and Scheduling with Precedence Constraints and Location Choice

G. Ayorkor Korsah, Anthony Stentz, M. Bernardine Dias, and Imran Fanaswala

Abstract—To realize the vision of intelligent transportation systems with fully automated vehicles, there is a need for high-level planning for single vehicles as well as fleets of vehicles. This paper addresses the problem of optimally assigning and scheduling a set of spatially distributed tasks to a fleet of vehicles working together to achieve a high-level goal, in domains where tasks may be related by precedence or synchronization constraints and might have a choice of locations at which they can be performed. Such problems may arise, for example, in disaster preparedness planning, transportation of people, and delivery of supplies. We present a novel mathematical model of the problem and describe how it can be solved optimally in a branch-and-price framework.

I. INTRODUCTION

Intelligent transportation systems comprising fully automated vehicles hold promise for improved efficiency, safety, and convenience over current systems. For this potential to be realized there is a need for algorithms for high-level planning for single vehicles and fleets of vehicles, in addition to sophisticated sensing, localization and navigation.

Traditionally, vehicle routing problems (VRPs) have addressed the problem of computing efficient routes for the transportation of people and goods by a fleet of vehicles. To realize the vision of truly intelligent transportation systems, approaches to these problems must address an increasingly richer set of constraints that may arise in various problem domains. While some tasks are independent of each other, others might be related by precedence or synchronization constraints. For example, during the evacuation of a special needs population in anticipation of a disaster, medical personnel might need to visit some patients before they can be transported to shelters. Additionally, many tasks may have a pre-specified location at which they must be performed, while others may have a choice of a small set of locations where they may take place. For example, there might be choice of shelters to which individuals can be evacuated in anticipation of a disaster.

Inspired by optimal mathematical programming approaches from the Operations Research literature, this paper presents a set-partitioning model for the problem of allocating, scheduling and choosing locations for mutually-constrained tasks to be performed by a fleet of vehicles.

This work is supported by the Qatar National Research Foundation under contract NPRP 1-7-7-5.

G. A. Korsah (previously published as G. A. Mills-Tettey), A. Stentz, and M. B. Dias are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA. (Emails: ayorkor@cmu.edu, axs@ri.cmu.edu, mbdias@ri.cmu.edu). I. Fanaswala is with the Computer Science Department, Carnegie Mellon University, Doha, Qatar (Email: imranf@qatar.cmu.edu).

This model enables finding a bounded optimal solution, considering the value of tasks completed, travel costs, as well as any costs due to waiting time needed to ensure that timing constraints are satisfied. We present a branch-and-bound approach to solving small instances of this problem, as well as a branch-and-price approach for larger instances.

II. PROBLEM FEATURES AND EXAMPLE

When an area needs to be evacuated on the threat of a disaster such as a hurricane, the population of people with special needs, who are not able to make their own evacuation plans, requires particular attention. Individuals may have special transportation or sheltering needs that must be considered during planning. Considering available transportation options (e.g. vans, ambulances, helicopters), available support teams, and available shelters, an evacuation plan for this population will determine which vehicle will pick up which individuals and when. It will schedule any support teams (e.g. medical personnel) which need to be available before, at the time of, or after pickup or drop-off of an individual. It will also determine which shelter each individual will be taken to, considering the individual's particular requirements. With appropriate databases of the special needs population, an optimal evacuation plan can be created ahead of time, and this optimal plan can be adjusted as needed in the event of an actual disaster.

We consider a problem in which a set of agents, K (comprising automated vehicles and other entities in the team), is available to perform a collection of tasks, J . Each task $j \in J$ consists of one or more spatially distributed subtasks that must be performed in a given order. For example, a medical visit task comprises a single subtask, whereas transporting a customer comprises a pickup subtask and a drop-off subtask. Different tasks are suited to different types of agents in the system – medical tasks cannot be performed by automated transportation agents and vice-versa. Each subtask, $i \in I$ may have a fixed location at which it might be performed, or a choice of a very small set of locations L_i at which it may be performed. Subtasks might have time windows constraining their start time, and in the case of transporting items from one location to another, subtasks might use up a finite capacity available on the assigned agent. Pairs of subtasks in the problem might be related by precedence or synchronization constraints, thus creating constraints between different agents' schedules. These constraints need to be considered in assigning agents to tasks, and may result in delays in the agents' schedules, which may increase the cost, or conversely, reduce the total value of the solution.

III. RELATED WORK

Vehicle routing problems (VRPs) address the transportation of passengers or the distribution of goods between depots and final users. VRPs can be expressed as mixed integer programming problems (MIP), defined on a graph in which the nodes correspond to locations of tasks to be performed, and edges correspond to travel segments between these locations. These mathematical models enable the formulation of optimal solution approaches. Proposed mathematical models can be broadly categorized as 3-index models and 2-index (or set-partitioning) models. For example, Cordeau [1] defines, for the dial-a-ride (DARP) problem (a variant of the VRP), a 3-index binary variable x_{ij}^k which is equal to 1 if vehicle k travels from node i to node j in the final solution. In contrast, Savelsbergh and Sol for the DARP [2] propose a set-partitioning model in which Ω_k is the set of feasible routes for vehicle k , and the 2-index variable x_r^k is a binary decision variable that takes on the value 1 if route $r \in \Omega_k$ is performed by vehicle k and 0 otherwise. Each route in Ω_k is a path through a subset of nodes, and is feasible in that all capacity and time constraints are satisfied along the route. Note that as the problem size grows, the number of feasible routes is usually too large to enumerate exhaustively. Rather a small set of feasible routes is initially used, and subsequently, additional “profitable” feasible routes are computed by a *pricing sub-problem*, and the *master* (set-partitioning) problem then selects a minimal cost set of routes satisfying the constraint that each customer must be serviced by only one vehicle.

Recent work in the vehicle routing literature has considered precedence constraints and synchronization constraints. In particular, Bredstrom and Ronnqvist present two different approaches. In one case [3], they create a three-index formulation of a vehicle routing problem, taking into consideration timing/synchronization constraints between individual tasks, each of which occurs at a fixed location. In this model, it is possible to penalize waiting time. In another case [4], they present a set-partitioning formulation that takes into consideration precedence constraints. However, in this model, waiting time cannot be penalized because time variables do not appear in the master problem formulation and so cannot be put in the objective function. Neither model addresses location choice. This paper presents a set-partitioning model which addresses location choice and precedence (and/or synchronization) constraints, while also being able to penalize waiting time as needed.

IV. MATHEMATICAL MODEL

We present a set-partitioning model with side constraints for this problem. The set-partitioning model, while representing complete feasible routes with single variables, also exposes time variables in the master problem formulation, thus allowing waiting time to be penalized by putting wait time variables in the objective function. We adopt the terminology of the vehicle routing literature and use the term *route* to represent a single agent’s plan – that is, a sequence

of subtasks that the agent / automated vehicle will perform at given locations according to the computed schedule.

In a set-partitioning approach, feasible routes for agents are represented by columns in the mixed integer linear program. In particular, a binary variable x_r^k indicates whether an agent k performs a given route r chosen from among all possible routes R_k that can feasibly be performed by agent k . In our problem, a feasible route is an ordered set of subtasks to be performed at chosen locations, such that all subtasks corresponding to the same task occur on the same route, time constraints are not violated, and agent capacity constraints are also not violated. A typical set-partitioning formulation would consist of these variables alone, with constraints specifying that each agent must perform only one route, and each task must appear on only one route.

In our formulation, however, we include additional time variables that appear in side constraints enforcing the precedence between subtasks that may appear on different routes. The real-valued variable w_i^k represents the amount of time that agent k , having arrived at the chosen location for subtask i , has to wait before it can begin execution of subtask i . This waiting time might be due to precedence constraints involving other subtasks being performed by other agents, or it might be because subtask i has a specific time window during which it must be performed. The waiting time is 0 if agent k is not assigned to subtask i . The real-valued variable t_i represents the time that execution begins on subtask i . If subtask i is not executed in the optimal solution, t_i is 0. In addition to the domain variables x_r^k , w_i^k , and t_i , the model includes helper variables $d_{i'i}$ representing the delay that the waiting time for subtask i' causes to a subtask i occurring later on the same route. If subtasks i' and i are not on the same route in the chosen solution, $d_{i'i}$ is 0. The delay variables are needed to ensure a linear formulation.

In the model below, v_j represents the value of completing a task j , which may of course comprise more than one subtask. c_{1r}^k represents the total travel cost of the route $r \in R_k$, and c_{2r}^k represents the waiting cost per unit time for agent k . The indicator π_{jr}^k is 1 if task j occurs on route $r \in R_k$ and 0 otherwise. Similarly, γ_{ilr}^k is 1 if subtask i occurs at location l on route $r \in R_k$ and 0 otherwise, and $\delta_{i'ir}^k$ is 1 if subtask i' occurs before subtask i on route $r \in R_k$ and 0 otherwise. The value τ_{ilr}^k represents the time that subtask i would be started on route $r \in R_k$ assuming no wait time was necessary, and τ_∞ represents the largest possible time in the model, that is, the end of the planning horizon. The value W_i represents the maximum allowed waiting time for subtask i ; α_{il} and β_{il} represent the earliest and latest times respectively that service can begin on subtask i when it is performed at location l . λ_{il}^k represents the service time for subtask i when it is performed at location l by vehicle k . In the model below, we use λ_i and y_i respectively as placeholders for the following expressions:

$$\lambda_i = \sum_{k \in K} \sum_{r \in R_k} \sum_{l \in L_i} \lambda_{il}^k \gamma_{ilr}^k x_r^k$$

$$y_i = \sum_{k \in K} \sum_{r \in R_k} \sum_{l \in L_i} \gamma_{ilr}^k x_r^k \equiv \sum_{k \in K} \sum_{r \in R_k} \pi_{jr}^k x_r^k$$

(where j is the task to which subtask i belongs)

TABLE I

SUMMARY OF VARIABLES AND DEFINED QUANTITIES

Variable	Definition
x_r^k	Whether or not agent k performs route r
w_i^k	Waiting time of agent k for subtask i
t_i	Execution start time for subtask i
$d_{i'i}$	Delay for subtask i caused by i'
Quantity	Definition
v_j	Value of completing task j
R_k	Set of feasible routes for agent k
c_{1r}^k	Travel cost for route $r \in R_k$
c_{2r}^k	Wait cost per unit time for agent k
π_{jr}^k	Whether or not task j occurs on route $r \in R_k$
γ_{ilr}^k	Whether or not subtask i occurs at location l on route $r \in R_k$
$\delta_{i_1 i_2 r}^k$	Whether or not subtask i_1 occurs before subtask i_2 on route $r \in R_k$
τ_{ilr}^k	No-wait start time of subtask i at location l on route $r \in R_k$
τ_∞	End of planning horizon
W_i	maximum allowed waiting time for subtask i
$[\alpha_{il}, \beta_{il}]$	Valid time window within which to begin execution of subtask i at location l
λ_{il}^k	Service time for subtask i performed by agent k at location l
λ_i	Service time for subtask i in chosen solution
y_i	Whether or not subtask i is performed in chosen solution
P	Set of precedence constraints

That is, λ_i is the service time of subtask i in the chosen solution (0 if i is not performed), and y_i indicates whether or not subtask i is performed in the selected solution. Finally, P represents the set of precedence constraints in the problem. Each precedence constraint $p = (i', i) \in P$ indicates that execution of subtask i' must end at least $\epsilon_{i'i}^P$ time units before service begins on subtask i .

The variables and defined quantities appearing in the mathematical model are summarized in Table I.

Maximize:

$$\sum_{j \in J} \sum_{k \in K} \sum_{r \in R_k} v_j \pi_{jr}^k x_r^k - \sum_{k \in K} \sum_{r \in R_k} c_{1r}^k x_r^k - \sum_{i \in I} \sum_{k \in K} c_2^k w_i^k \quad (1)$$

Subject to:

$$1 = \sum_{r \in R_k} x_r^k \quad \forall k \in K \quad (C1)$$

$$1 \geq \sum_{k \in K} \sum_{r \in R_k} \pi_{jr}^k x_r^k \quad \forall j \in J \quad (C2)$$

$$w_i^k \leq W_i \sum_{l \in L_i} \sum_{r \in R_k} \gamma_{ilr}^k x_r^k \quad \forall i \in I, k \in K \quad (C3)$$

$$t_i = \sum_{l \in L_i} \sum_{k \in K} \sum_{r \in R_k} \tau_{ilr}^k \gamma_{ilr}^k x_r^k + \sum_{i' \in I} d_{i'i} + \sum_{k \in K} w_i^k \quad \forall i \in I \quad (C4a)$$

$$t_i \geq \sum_{l \in L_i} \alpha_{il} \sum_{k \in K} \sum_{r \in R_k} \gamma_{ilr}^k x_r^k \quad \forall i \in I \quad (C4b)$$

$$t_i \leq \sum_{l \in L_i} \beta_{il} \sum_{k \in K} \sum_{r \in R_k} \gamma_{ilr}^k x_r^k \quad \forall i \in I \quad (C4c)$$

$$d_{i'i} \geq \sum_{k \in K} w_{i'}^k - W_{i'} \sum_{k \in K} \sum_{r \in R_k} (1 - \delta_{i'i r}^k) x_r^k \quad \forall i', i \in I \quad (C5a)$$

$$d_{i'i} \leq W_{i_1} \sum_{k \in K} \sum_{r \in R_k} \delta_{i'i r}^k x_r^k \quad \forall i', i \in I \quad (C5b)$$

$$d_{i'i} \leq \sum_{k \in K} w_{i'}^k \quad \forall i', i \in I \quad (C5c)$$

$$y_{i'} \geq y_i \quad \forall (i', i) \in P \quad (C7a)$$

$$t_{i'} \leq t_i - \lambda_{i'} - \tau_\infty (y_i - y_{i'}) - \epsilon_{i'i}^P (y_i + y_{i'} - 1) \quad \forall (i', i) \in P \quad (C7b)$$

The objective function (1) strives to maximize the difference between overall reward and overall travel and waiting cost. (C1) specifies that each agent is assigned to exactly one route (which may be an empty route, allowing the solution to choose not to use a given vehicle). (C2) specifies that each task is assigned to at most one agent, and can in fact be rejected by assigning it to no agent. These two are the standard set-partitioning constraints. (C3) represents the bound on the waiting times. (C4a) computes the start time for a subtask, while (C4b) and (C4c) represent the time window bounds for the subtask. (C5a-C5c) represent constraints on the delay variables, which enable the start time of each

subtask to be computed correctly, taking into consideration the wait time of all prior subtasks on the selected route. Finally, (C7a) and (C7b) capture the precedence constraints of the problem: (C7a) indicates that the second task i in the precedence constraint $(i', i) \in P$ is performed only if the first task i' is performed; (C7b) ensures that the start times of the task satisfy the precedence constraints. Similar constraints to (C7a) and (C7b) can represent synchronization constraints, by changing the inequalities to equalities, and removing the $\lambda_{i'}$ and $\tau_\infty (y_i - y_{i'})$ terms from C7b.

In this model, the capacity and some time constraints are dealt with when generating feasible routes. The process of generating feasible routes also performs location choice by fixing the location of each subtask on the route. The solution of the set-partitioning problem then selects between all feasible routes for an agent, thus finalizing the location choice for each subtask, and also fixes the time for each task by inserting waiting times as needed to ensure that between-route timing constraints are satisfied while still respecting the within-route timing constraints.

V. BRANCH-AND-BOUND ALGORITHM

In problems that are small enough to exhaustively enumerate all feasible routes, the above mixed integer programming problem can be solved in a standard branch-and-bound framework. To start with, an upper bound on the solution is computed by relaxing the integrality constraints on the x_r^k variables, and solving the resulting linear program. Subsequently, branching decisions are made on fractional x_r^k variables (e.g., forcing them to either 0 or 1), and the solution process is repeated at each node of the branch-and-bound

tree, until a solution that satisfies the integer constraints is found. In reality, it is more efficient with this problem to make higher-level branching decisions rather than simply setting fractional x_r^k to 0 or 1. We adopt the following branching decisions, variations of which are used in several VRP solution approaches: When there are fractional routing variables, we branch by forcing two tasks to be on the same route (“together”) in one branch or on different routes (“not together”) in the other branch. When the fractional routing variables represent two different routes with the same subtasks performed in different orders, we branch by constraining two subtasks to occur in a specific order.

VI. BRANCH-AND-PRICE ALGORITHM

In most situations, it will not be possible to enumerate all possible routes up front, and this is where a *column generation* process is useful. The algorithm starts out by considering only a subset of columns, and new columns are added as needed. The columns to be added are determined by solving, a problem called the *pricing subproblem*. A branch-and-price algorithm is a branch-and-bound algorithm in which column generation occurs at each node of the branch-and-bound tree.

Barnhart et al [5] provide a useful introduction to branch-and-price approaches and a detailed theoretical discussion, which is outside the scope of this paper. However, the essential idea is that the pricing subproblem finds “profitable” routes that can potentially improve the solution of the master problem. This is done by optimizing a pricing function computed from the dual variables of the master problem. For a minimization problem, a profitable column has a “price” of less than 0 (and as such, can decrease the objective function of the master problem if included), while for a maximization problem, a profitable column has a “price” of greater than 0 (and as such, can increase the objective function of the master problem if included). At each iteration of the column generation process, new columns are added to the master problem until the pricing problem, when solved to optimality, returns that there are no more profitable routes. At this point, branching can be performed on any fractional values, and the process repeated at subsequent branch-and-bound nodes.

If we designate the dual variables corresponding to constraints (C1) to (C7b) in our mathematical model as u^1 to u^{7b} respectively, then the pricing subproblem for our set-partitioning model can be derived as finding the feasible route r for agent k that maximizes the quantity:

$$\begin{aligned}
p_r^k = & -(c_{1r}^k + u_k^1) + \sum_{j \in J} (v_j - u_j^2) \pi_{jr}^k \\
& + \sum_{i \in I} \sum_{l \in L_i} (u_{ik}^3 W_i + u_i^{4a} \tau_{ilr}^k - u_i^{4b} \alpha_{il} + u_i^{4c} \beta_{il}) \gamma_{ilr}^k \\
& + \sum_{i' \in I} \sum_{i \in I} (-u_{i'i}^{5a} + u_{i'i}^{5b}) W_i \delta_{i'i}^k \\
& - \sum_{(i', i) \in P} \sum_{l \in L_{i'}} (u_{(i', i)}^{7a} + u_{(i', i)}^{7b} (\lambda_{il}^k + \epsilon_{i'i}^P - \tau_\infty)) \gamma_{i'l}^k \\
& + \sum_{(i', i) \in P} \sum_{l \in L_i} (u_{(i', i)}^{7a} - u_{(i', i)}^{7b} (\epsilon_{i'i}^P + \tau_\infty)) \gamma_{ilr}^k
\end{aligned} \tag{2}$$

To gain a better understanding of the pricing subproblem, recall that π_{jr}^k indicates whether a given task j is fully served on route r , γ_{ilr}^k indicates whether subtask i is performed at location l on route $r \in R_k$, and $\delta_{i'i}^k$ indicates whether subtask i' occurs before subtask i on route $r \in R_k$. Also, note that for a given instance of the subproblem solution process, the dual variables u are constants. Thus, for a given agent, solving the pricing problem is equivalent to finding a feasible route that maximizes the above quantity. This can be done by searching through a graph where nodes represent feasible {location, subtask} pairs and edges indicate that it is possible for agent k to travel from one location to another. No edges connect pairs of nodes that correspond to the same subtask but different locations. The transition costs in the graph are determined by equation (2). The first term in the equation represents the cost of the route modified by a constant that depends on the agent. This route cost can be broken down into a transition cost for each edge of the graph that is traversed along the route. The second term in the equation represents a value or reward for each task completed on the route. Since a feasible route must comprise all subtasks of any task that is performed on the route, and since a feasible route consists of only one location for each subtask that is performed on the route, this term can be broken down to a value for each node of the graph visited along the route. The third term in the equation represents a cost for each node that is visited along the route: 3 of the sub-terms (involving the dual variables u_{ik}^3 , u_i^{4b} , and u_i^{4c}) represent a constant cost for the node, and a fourth sub-term (involving the dual variable u_i^{4a}) represents a cost that is linear in the time that it takes to reach that node along the route, i.e., the no-wait arrival time, τ_{ilr}^k . The fourth term in the equation represents a cost for each ordered pair of tasks on the route, (i', i) , such that i' precedes i . The fifth term represents a cost for each node along the route for which the corresponding subtask appears as the first subtask in a precedence constraint. The sixth and final term represents a cost for each node along the route for which the corresponding subtask appears as the second subtask in a precedent constraint.

VII. GENERATING NEW ROUTES:

THE CONSTRAINED ROUTING PLANNER (CRP)

We have formulated an optimal dynamic programming route-planning algorithm to solve the pricing problem described above. The algorithm finds the route that minimizes $\bar{p}_r^k = -p_r^k$, and hence maximizes p_r^k . Our route-planning algorithm is based on the DD* Lite algorithm [6] for incremental search with state dominance. DD* Lite performs a best-first search, focused by a heuristic, through a multi-dimensional state space, to find a path from a start node to a goal node, exploiting domain-specific dominance relationships to prune the state space where possible in order to make the search more efficient. While DD* Lite is a general search algorithm, it needs to be customized to the given domain, and we modify and customize it in the following ways:

1) *State Space*: Each node in the state space being searched is identified by the graph node n representing a given {subtask, location} pair, the no-wait arrival time t_a of the agent at the node along the route, the unordered set S_p of subtasks that have been previously completed along the route to that state, and a boolean variable b indicating whether the route satisfies the branching constraints of the current node in the branch-and-bound tree at which column generation is being performed. $state := \{n, t_a, S_p, b\}$

The node n is a node in the graph that is being searched, that is, the collection of {subtask, location} pairs in addition to two special nodes corresponding to the agent start and end locations. We designate the set of graph nodes as N where $|N| = \sum_{i \in I} |L_i| + 2$. While n is an *path-independent parameter* of the state, whose value does not depend on the path taken to reach the state, t_a , S_p and b are *path-dependent parameters* (as described in [7]) whose values depend on the path taken to reach the state and are computed dynamically during the search process. As such, states in this large multi-dimensional search space are not instantiated up front but are generated as they are encountered in the search.

2) *Search Direction*: The original version of DD* Lite searches backwards from the goal to the start in order to facilitate efficient replanning in domains where changes in the graph are likely to occur close to the start state. In this work, however, we flip the direction of search and instead search forwards from the start state to the goal node. This is because, as described in the previous section, the transition cost to a node in the graph depends on the arrival time at the node, and this can only be accurately computed if searching in a forward direction. We do not currently use the replanning functionality of DD* Lite.

3) *Start and Goal States*: The start state is defined as the special graph node corresponding to the agent start location for n , the earliest available time for the agent for t_a , an empty set for S_p , and a value for b that depends on the particular branch constraints in question. The goal state is defined as the special graph node corresponding to the agent end location for n , and a value of `true` for b . The values of t_a and S_p for the goal state are not known ahead of time, but are computed during the search.

4) *State Transitions*: From a state s_1 in the graph, we can transition to a state s_2 corresponding to any other node $n \in N$ in the graph, with the following exceptions:

- No transitions are allowed into the start node
- No transitions are allowed out of the end node
- Transitions are not allowed from s_1 to a node corresponding to a subtask that has already been completed along this route, that is, a subtask in $s_1.S_p$.
- Transitions are not allowed from s_1 to a node corresponding to a subtask whose previous subtasks have not yet been performed (for tasks with more than one subtask).

The arrival time of s_2 is computed using the arrival time of s_1 , the service time at $s_1.n$, and the travel time from $s_1.n$ to $s_2.n$. Note that we are, in effect, computing the no-wait arrival time along the route. Waiting time is not

included in the computation of t_a , since waiting time might be affected by the schedules of other agents and as such, is determined in the branch-and-price master problem. $s_2.S_p$ is computed as the union of $s_1.S_p$ with the subtask performed at $s_2.n$. Finally, $s_2.b$ is computed by examining whether the branching constraints are satisfied for this state. To facilitate this, we also keep track of a boolean variable for each branching constraint that needs to be satisfied, in order to be able to determine when they have all been satisfied.

The state transition function checks that agent capacity as well as maximum route length constraints are not violated. It also checks some time constraints, rejecting partial routes that arrive at a node after the relevant time window. Note that although the computation of t_a does not take waiting time into consideration, the algorithm does keep track for each state of the minimum waiting time that would be required to meet the time window constraints at the nodes along the route to that state. This enables tighter checks of time constraints and reduces the number of times the algorithm will find a route that is ultimately rejected in the branch-and-price master problem because of time constraints. The state transition function also checks that no branching constraints corresponding to the current branch of the branch-and-price tree are violated. For a “not together” constraint involving tasks j_i and j_j , it ensures that if j_i is already on the current route, transitions to nodes corresponding to subtasks of j_j are not allowed, and vice-versa. For an “order” constraint involving subtasks i_1 and i_2 , it ensures that transitions to nodes corresponding to the later subtask are allowed only if the earlier subtask has been performed.

The transition cost from s_1 to s_2 is computed based on equation 2. Equation 2 represents the value for an entire route, but this needs to be incrementally computed as we transition from one state to another during the search. Note that negative transition costs are allowed in the graph, which is not problematic since cycling is disallowed.

5) *Search Heuristic*: Having computed the cost from the start state to state s , we need to compute a lower bound on the transition cost for the remainder of the route from s to the goal node. Whereas in the original version of DD* Lite a heuristic can be omitted (i.e. the heuristic value of 0 can be used), this is not possible here because transition costs can be negative. Without a heuristic that is a valid lower bound on the cost of the remainder of the route, the algorithm may terminate with a suboptimal solution. We compute a valid heuristic by computing bounds on the various dual variables, and using maximum route length constraints to bound the length of the remaining path.

6) *Dominance Relationship*: Assume that two states, s_1 and s_2 have the same node n , the same set of previous subtasks, S_p , but different arrival times, t_{a1} and t_{a2} . These states represent two different partial routes r_1 and r_2 , reaching node n , having completed the subtasks in S_p in different orders or at different locations. Assume, without loss of generality, that $t_{a1} < t_{a2}$. Then the partial route r'_1 leaving from s_1 can visit at least every node that the partial route r'_2 leaving from s_2 can visit and will arrive at these nodes no later than

if it left from s_2 . Furthermore, if the dual variable u_i^{4a} is non-positive for all subtasks, then, from the pricing equation (2), the cost of the partial route r'_1 from s_1 to the goal, will be no worse than the cost of the partial route r'_2 from s_2 to the goal. Thus, if the cost of the route r_1 from the start to s_1 is also better than the cost of the route r_2 from the start to s_2 , s_1 dominates s_2 , because there is no benefit in taking the route through s_2 .

VIII. RESULTS

We have implemented the branch-and-price algorithm as described. Below, we present the solution of an illustrative problem, highlighting the issues of precedence constraints, waiting costs, waiting time, and location choice. We then discuss the issue of how to handle dynamism, which is very relevant to intelligent transportation systems.

A. Example Problem

The example in figure 1 shows an evacuation problem where there are two transportation agents (squares), one medical agent (triangle), five individuals to be transported (crosses) and two shelters to which the victims can be transported (circles). Each of the five individuals requires a visit by the medical agent before being transported to a shelter. Thus, there are five precedence constraints in the problem. Figure 1 shows the routes computed for 2 different values of waiting cost, and Figure 2 show the corresponding schedules. In Figure 2, the horizontal axis represents time, and the three lines represent the schedules of the agents 0, 1 and 2. Circles indicate arrival at a location to perform a task, crosses indicate beginning execution of a task, dotted lines indicate waiting time, and solid lines indicate execution time. With a waiting cost of 0, the algorithm minimizes the travel time, regardless of how long the agent needs to wait/be idle before it can complete its task. Because there is only one medical agent, waiting times for some tasks are quite significant since the assigned transportation agent needs to wait for the medical agent to complete its task before it can transport the individual in question. When the waiting cost is increased to 0.5 (that is, half of the transportation cost per unit time), the transportation agents are less willing to wait and would prefer to travel in order to transport an individual that is ready. The overall distance traveled is increased, and the waiting time is reduced, though not completely.

For this example with 5 transportation tasks and 5 medical tasks to perform, the table below shows the effect of location choice on the problem size, and the potential advantage of column generation. As the number of subtasks in the problem that have a choice of locations at which they can be performed increases, the number of feasible routes increases drastically, but the number of routes generated in the column generation process stays roughly constant for this problem.

Location choices	0	1	2	3	4	5
Feasible routes	2261	3391	5073	7487	10813	15231
Generated routes	50	51	52	52	47	48

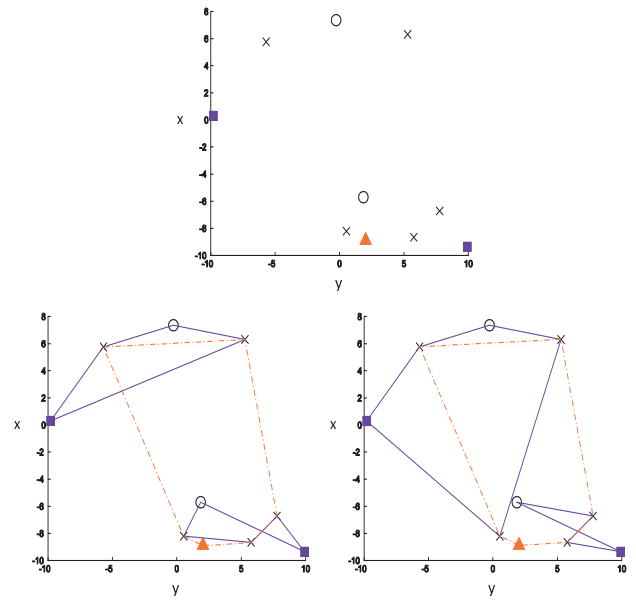


Fig. 1. Example evacuation problem and routes computed with waiting costs of 0 and 0.5

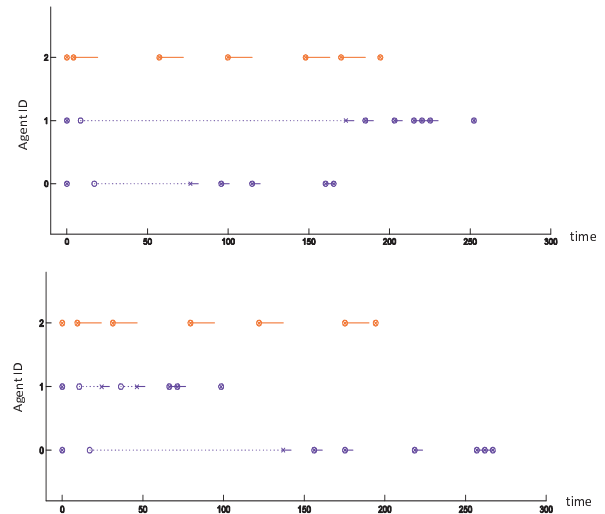


Fig. 2. Agents' schedules for waiting cost of 0 and 0.5

B. Handling dynamism

The presented approach has the advantage of computing an optimal solution to the task allocation and routing problem, but it requires that the system knows about all tasks ahead of time. By itself, it is not suited to dynamic domains in which new tasks arrive or conditions change over time. To handle dynamism, we propose a hybrid planning approach in which the optimal planner computes an initial optimal plan for the set of static tasks, that is, the tasks available at the beginning of the planning horizon. As new tasks arrive, they are incorporated into the schedule by using a decentralized market-based task allocation method [8]. The same method can be used to handle any modifications in the plan that are required as a result of unforeseen events (such as a road being closed, or a given vehicle becoming disabled).

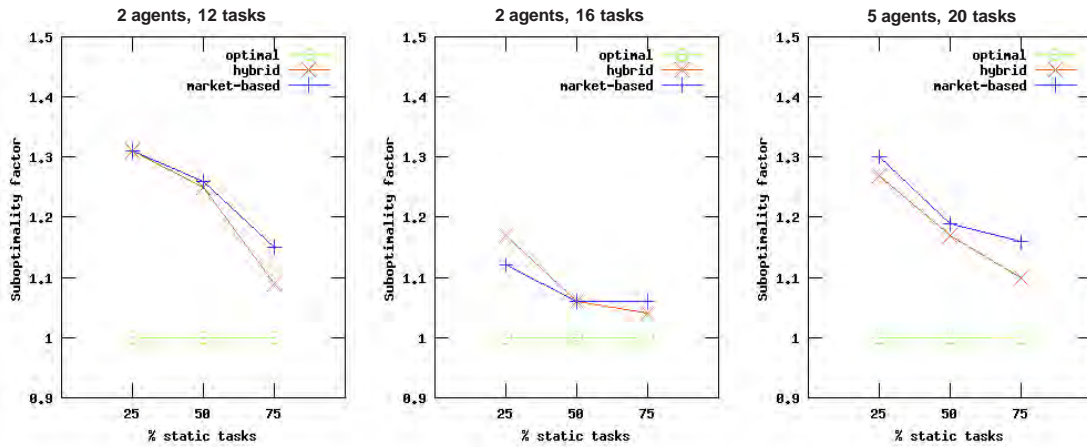


Fig. 3. Relative route costs for the optimal planner, the hybrid planner, and the market-based planner, as a function of the proportion of static tasks

Figure 3 compares the average solution cost (in this case, total team distance) for this hybrid approach to that for a solely market-based approach for three different problems configurations of a simulated scenario in which a team of vehicles needs to perform a collection of simple single-step tasks. For this scenario, there are no precedence constraints or location choices. The vehicles operate in a 40×40 area, traveling at a speed of 1 unit of distance per unit time. A set of “static” tasks is present from the begin of the planning horizon ($t=0$) while the remaining “dynamic” tasks come in at various times from $t=1$ to $t=100$. The optimal planner computes an initial plan for the static tasks, which the vehicles begin to execute. As they come in, new tasks are allocated and incorporated in the vehicle schedules with the market-based planner. The costs of the solutions computed by the optimal, hybrid and market-based allocation approaches (averaged over 5 random instances) are expressed as a ratio of the best solution computed by the optimal approach, in “hindsight”, that is, after all dynamic tasks have arrived in the system. Note that this “hindsight” plan optimizes the team distance by inserting waiting time into the plan as appropriate, since it knows the exact times at which all tasks arrive in the system. As such, it is a plan that is unattainable without knowing the future. Note also that for this problem, a timeout is set on the solution process for the constrained route planning subproblem and so the best solution produced by the optimal planner may be slightly suboptimal in some cases. Figure 3 illustrates that having an optimized seed plan is usually, although not always, advantageous over using only a market-based method. As expected, the advantage of having a seed plan increases with the proportion of tasks that are static.

IX. CONCLUSION AND FUTURE WORK

In this paper, we have presented a novel mathematical model for a complex task allocation and routing problem involving precedence constraints and location choice. We have presented a branch-and-price solution process and a

novel dynamic programming algorithm for finding profitable routes. We also presented an approach by which this method can be combined with a market-based method to address dynamic scenarios. Our ongoing work includes a comprehensive analysis of the performance of the approach for different problem configurations. It also includes developing heuristic algorithms to solve the constrained route-planning subproblem, to widen the pool of problems for which this is a tractable planning approach. The planning framework presented contributes to enabling vehicles in an intelligent transportation system to achieve autonomy, not only on the level of navigation but also on the level of coordinating with other vehicles in a fleet to achieve high-level goals.

REFERENCES

- [1] J.-F. Cordeau, “A Branch-and-Cut Algorithm for the Dial-a-Ride Problem,” *Operations Research*, vol. 54, no. 3, pp. 573–586, 2006.
- [2] M. Savelsbergh and M. Sol, “Drive: Dynamic routing of independent vehicles,” *Operations Research*, vol. 46, no. 4, pp. 474–490, 1998.
- [3] D. Bredström and M. Rönnqvist, “Combined vehicle routing and scheduling with temporal precedence and synchronization constraints,” *European Journal of Operations Research*, vol. 191, pp. 19–31, 2008.
- [4] —, “A branch and price algorithm for the combined vehicle routing and scheduling problem with synchronization constraints,” Norwegian School of Economics and Business Administration, Department of Finance and Management Science, Discussion Paper Number FOR7 2007, 2007.
- [5] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance, “Branch-and-price: Column generation for solving huge integer programs,” *Operations Research*, vol. 46, pp. 316–329, 1998.
- [6] G. A. Mills-Tettey, A. T. Stentz, and M. B. Dias, “DD* lite: Efficient incremental search with state dominance,” in *Twenty-First National Conference on Artificial Intelligence (AAAI-06)*, July 2006, pp. 1032–1038.
- [7] —, “Continuous-field path planning with constrained path-dependent state variables,” in *ICRA 2008 Workshop on Path Planning on Costmaps*, May 2008.
- [8] M. B. Dias, “Traderbots: A new paradigm for robust and efficient multirobot coordination in dynamic environments,” Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, January 2004.



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Multi-Agent Planning and Simulation for Intelligent Transportation Systems

Ming C. Lin¹ Jason Sewall² Jur Van den Berg³ David Willkie⁴ Dinesh Manocha⁵

¹lin@cs.unc.edu, ²sewall@cs.unc.edu, ³berg@cs.unc.edu, ⁴wilkie@cs.unc.edu, ⁵dm@cs.unc.edu

Department of Computer Science, University of North Carolina at Chapel Hill

Videos available at http://gamma.cs.unc.edu/TRAFFIC_RECON/ and <http://gamma.cs.unc.edu/CTS>

Abstract—In this paper, we provide a brief survey of our recent work on multi-agent planning and simulation for intelligent transportation system. In particular, we first present a novel algorithm to reconstruct and visualize continuous traffic flows from discrete spatio-temporal data provided by traffic sensors. Given the positions of each car at two recorded locations on a highway and the corresponding time instances, our approach can reconstruct the traffic flows (i.e. the dynamic motions of multiple cars over time) in between the two locations along the highway using a priority-based multi-agent planning algorithm. Our algorithm is applicable to high-density traffic on highways with an arbitrary number of lanes and takes into account the geometric, kinematic, and dynamic constraints on the cars. In addition, we describe an efficient method for simulating realistic traffic flows on large-scale road networks. Our technique is based on a continuum PDE model of traffic flow that we extend to correctly handle lane changes and merges, as well as traffic behaviors due to changes in speed limit. We show that our method can simulate plausible traffic flows on publicly-available, real-world road data and demonstrate the scalability of this technique on many-core systems.

I. INTRODUCTION

Traffic congestion management is a global challenge that transportation engineers and planners, policy makers, and the public at large will likely have to cope with for decades. Besides the obvious energy and environmental impacts, traffic congestion imposes tangible costs on society. The latest biennial report on urban mobility in the US [7] indicates that over 4 billion hours of delays are cumulatively experienced by the system users at a staggering annual cost of over \$78 billion. and this figure has doubled over a 10-year period between 1997-2007. It is unlikely that traditional physically-centered mitigation strategies by themselves will be successful. These approaches are simply not sustainable in the current economical and environmental climate.

A. Intelligent Transportation Systems

In order to address the problems caused by traffic congestion, numerous strategies have been proposed to construct Intelligent Transportation Systems (ITS), a term originally coined to represent the incorporation of sensing, information and communication technologies in transportation infrastructure and vehicles. At a broad level, research in ITS tends to minimize many undesirable impacts, such as excessive travel delay, air pollution, infrastructure utilization, fuel consumption, and also improves overall highway safety. ITS technologies encompass basic management strategies, such as

traffic signal control, variable message signs, speed cameras, and license plate readers, to more advanced technologies, such as variable speed-limit for flow control, electronic toll collection and congestion pricing, and automatic law enforcement (e.g. radars). More vehicles are increasingly equipped with cruise-control, collision avoidance systems, and GPS-guided navigation. Recent developments related to semi-autonomous or autonomous vehicles have also shown some success, e.g. DARPA Grand Challenge.

Sensing technologies have continued to make leaps with the wide availability of Radio-frequency identification (RFID), inexpensive intelligent beacon sensing, inductive loop detectors, video/camera detection systems, etc. As wireless communication networks have become ubiquitous, cell-phones and Wi-Fi networks have also been used as networks of mobile sensors to monitor or visualize traffic conditions.

B. Main Results

In this paper, we propose to develop a new algorithmic framework with efficient computational methods applied to solving challenging transportation engineering problems. Our vision is that of a seamless, integrated transportation system where traffic at the micro, meso or macro scales is sensed and re-constructed; and where estimation of future traffic states takes place in a real-time manner; where high-performance simulators is invoked and test multiple traffic control strategies; and where algorithms select the optimal control strategies to apply to the context at hand; and where quick handoffs between simulation and physical controllers implement the proposed strategies in a timely manner while the context is still valid.

Our proposed framework relies on available infrastructure-based sensors on road networks, but can also incorporate mobile sensor data. In addition to traffic monitoring and basic traffic mitigation, we propose to develop an interactive computational framework that captures current traffic states continuously by streaming and processing sensor data, performing on-the-fly traffic reconstruction and simulation for continuous analysis, prediction, intervention, and traffic control in real time by utilizing the computational capabilities of many-core processors and graphics processor units (GPUs) available on current desktop systems. Eventually, these algorithms can also be embedded in intelligent vehicles to provide more coordinated, individualized rerouting schemes

for each car in coordination with vehicle flows of the entire transportation system to avoid traffic congestion.

In this paper, we present an overview of our early results, including (1) a novel traffic reconstruction algorithm [3] based on multi-agent planning and coordination, taking into account the geometric, kinematic, and dynamic constraints on each car in Sec. II; (2) *real-time, large-scale* traffic simulation [6] based on a continuum representation with discrete, individual-vehicle behaviors in Sec. III and a fast parallel realization of this method to handle urban traffic consisting of tens of thousands of vehicles on GPUs and many-core processors for desktop and portable systems. We conclude by suggesting several research challenges in realizing this vision.

II. TRAFFIC RECONSTRUCTION USING MULTI-AGENT PLANNING AND COORDINATION



Fig. 1. Images of highway traffic synthesized by our method. Our method computes trajectories one by one for a continuous stream of cars (of possibly high-density). The trajectories fit the boundary conditions at the sensor points, and obey the geometric, kinematic and dynamic constraints on the cars. The number of lane changes and the total amount of (de-)acceleration are minimized and the distance to other cars is maximized to obtain smooth and plausible motions.

In this section, we present a multi-agent planning method for reconstructing traffic flows based on the spatial-temporal data captured from in-road sensors. Given two locations along a highway, say A and B , we assume that the velocity and the lane of each car is known at two corresponding time instances. The challenge is to reconstruct the continuous motion of multiple cars on the stretch of the highway in between the two given locations. We formulate it as a “multi-robot planning problem”, subject to spatial and temporal constraints. There are several key differences, however, between the traditional multi-robot planning problem and our formulation. First of all, we need to take into account the geometric, kinematic and the dynamic constraints of each car (though a subset of specialized algorithms have also considered these issues [2]). Second, in our formulation, not only the start time, but the arrival time of the cars is also specified. In contrast, the objective of previous literature has been for the robots to arrive at the goal location as soon as possible. Third, the domain that is dealt with here is an *open system*, i.e. the number of cars is not fixed. Instead, new cars can continuously enter the stretch of the highway. This aspect requires incremental update to the current solution as new cars arrive at the given location.

A. Overview

We extend a prioritized approach that assigns priorities to each car based on the relative positions of the cars on the road: cars in front have a higher priority. Then, in order of decreasing priority, we compute trajectories for the cars that avoid cars of higher priority for which a trajectory has already been determined. To make the search space for each car tractable, we constrain the motions of the car to a pre-computed roadmap, which is a reasonable assumption as each car typically has a pre-determined location to travel to. The roadmap provides links for changing lanes and encodes the car’s *kinematic* constraints. Given such a roadmap, and a start and final state-time on the roadmap, we compute a trajectory that is compliant with the car’s *dynamic* constraints and avoids collisions with cars of higher priority. At each time step, the car either accelerates maximally, maintains its current velocity, or decelerates maximally. This approach discretizes the set of possible velocities and the set of possible positions as well, enabling us to compute in three-dimensional state-time grids along the links of the roadmap. Our algorithm searches for a trajectory that minimizes the number of lane changes and the amount of (de-)acceleration, and maximizes the distance to other cars to obtain smooth and realistic motions.

B. Reconstructing Traffic

1) *Constructing the Roadmap*: The cars are constrained to move over a preprocessed roadmap to make the configuration space of a car tractable. We construct this roadmap as follows. First, we subdivide the highway into M segments of equal length. For each lane of the highway, we place a roadmap vertex at the end of each segment. This gives a $M \times N$ grid of roadmap vertices, where N is the number of lanes.

Each vertex (i, j) is connected by an edge to the next vertex $(i + 1, j)$ in the same lane. These edges allow cars to stay in their lane and move forward. To allow for lane changes, we also connect vertices of neighboring lanes. Each vertex (i, j) is connected to vertices $(i + a, j + 1), \dots, (i + b, j + 1)$ and $(i + a, j - 1), \dots, (i + b, j - 1)$. Here a and b denote the minimum and maximum length (in number of segments) of a lane change, respectively. The short lane changes are useful at lower velocities, the longer ones at higher velocities.

When adding the edges for lane changes, we have to make sure that they are “realistic”. That is, they should obey the kinematic constraints of a car and should be traversable without abrupt steering wheel motions. Let us look more closely at the constraint on the speed with which the steering wheel is turned. It translates into the following bound on the *curvature derivative*:

$$|\phi'(t)| \leq \omega_{\max} \Leftrightarrow |\kappa'(t)| \leq \frac{\omega_{\max}}{\lambda} \Leftrightarrow |\kappa'(s)| \leq \frac{\omega_{\max}}{v\lambda} \Leftrightarrow v \leq \frac{\omega_{\max}}{|\kappa'(s)|\lambda} \quad (1)$$

In other words: the smaller the curvature derivative (with respect to path length s), the higher the velocity with which this path can be traversed.

The roadmap resulting from the above method is valid for cars with any value of λ , so we need to construct a roadmap only once, and can use it for all cars.

2) *Trajectory for a Single Car*: Given a roadmap as constructed above and the state-time graph as defined in the previous section, we describe how we can compute a trajectory for a single car, assuming that the other cars are moving obstacles of which we know their trajectories. How we reconstruct the traffic flows for multiple cars is discussed in below.

A straightforward approach for searching a trajectory in the state-time graph is the A*-algorithm. It builds a *minimum cost tree* rooted at the start state-time and biases its growth towards the goal. To this end, A* maintains the leaves of the tree in a priority queue Q , and sorts them according to their f -value. The function $f(\langle q, t \rangle)$ gives an estimate of the cost of the minimum cost trajectory from the start to the goal via $\langle q, t \rangle$. It is computed as $g(\langle q, t \rangle) + h(\langle q, t \rangle)$ where $g(\langle q, t \rangle)$ is the cost it takes to go from the start to $\langle q, t \rangle$, and $h(\langle q, t \rangle)$ a lower-bound estimate of the cost it takes to reach the goal from $\langle q, t \rangle$. A* is initialized with the start state-time in its priority queue, and in each iteration it takes the state-time with the lowest f -value from the queue and *expands* it. That is, each of the state-time’s successors in the state-time graph is inserted into the queue if they have not already been reached by a lower-cost trajectory during the search. This process repeats until the goal state-time is reached, or the priority queue is empty. In the latter case, no valid trajectory exists.

In [8] the A*-algorithm was used to find a *minimal-time trajectory*. That is, only a goal state is specified, and the task is to arrive there as soon as possible. This makes it easy to focus the search towards the goal; the cost of a trajectory is simply defined as its length (in terms of time). However, in

our case the arrival time is specified as well, so we know in advance how long our trajectory will be. Therefore, we cannot use time as a measure in our cost function. Instead, we let the cost of a trajectory T depend on the following criteria, in order to obtain smooth and realistic trajectories:

- The number of lane changes $X(T)$ in the trajectory.
- The total amount $A(T)$ of acceleration and deceleration in the trajectory.
- The accumulated cost $D(T)$ of driving in closer proximity than a preferred minimum $d_{\text{limit}} > 0$ to other cars.

More precisely, the total cost of the trajectory T is defined as follows:

$$\text{cost}(T) = c_X X(T) + c_A A(T) + c_D D(T) \quad (2)$$

where c_X, c_A and c_D are weights specifying the relative importance of each of the criteria. $A(T)$ and $D(T)$ are defined as follows:

$$A(T) = \int_T |v'(t)| dt \quad (3)$$

$$D(T) = \int_T \max\left(\frac{d_{\text{limit}}}{d(t)} - 1, 0\right) dt \quad (4)$$

where $v(t)$ is the velocity along the trajectory as a function of time, and $d(t)$ is the distance (measured in terms of time) to the nearest other car on the highway as a function of time.

The distance $d(t)$ to other cars on the highway given a position s in the roadmap and a time t is computed as follows. Let t' be the time closest to t at which a car configured at s would be in collision with another car, given the trajectories of the other cars. Then, $d(t) = |t - t'|$. We obtain this distance efficiently by — prior to determining a trajectory for the car — computing for all positions in the roadmap during what time intervals it is in collision with any of the other cars. Now, $d(t)$ is simply the distance between t and the nearest collision interval at s . If t falls within an interval, the car is in collision and the distance is zero. As a result, the above cost function would evaluate to infinity.

In the A*-algorithm, we evaluate the cost function per edge of the state-time graph that is encountered during the search. The edge is considered to contain a lane change if a lane-change edge of the roadmap is entered. The total cost $g(\langle q, t \rangle)$ of a trajectory from the start state-time to $\langle q, t \rangle$ is maintained by accumulating the costs of the edges the trajectory consists of. The lower bound estimate $h(\langle q, t \rangle)$ of the cost from $\langle q, t \rangle$ to the goal state-time $\langle q_{\text{goal}}, t_{\text{goal}} \rangle$ is computed as follows:

$$v_{\text{avg}} = \frac{x(q) - x(q_{\text{goal}})}{t_{\text{goal}} - t} \quad (5)$$

$$h(\langle q, t \rangle) = c_X |\text{lane}(q) - \text{lane}(q_{\text{goal}})| + c_A (|v(q) - v_{\text{avg}}| + |v(q_{\text{goal}}) - v_{\text{avg}}|) \quad (6)$$

where v_{avg} is the average velocity of the trajectory from $\langle q, t \rangle$ to $\langle q_{\text{goal}}, t_{\text{goal}} \rangle$, and $x(q)$, $\text{lane}(q)$ and $v(q)$ are respectively

the the position along the highway, the lane and the velocity at state q . If $v_{\text{avg}} > v_{\text{max}}$, we define $h(\langle q, t \rangle) = \infty$.

An advantage of the goal time being specified is that we can apply a bidirectional A*, in which a tree is grown from both the start state-time and the goal state-time in the reverse direction until a state-time has been reached by both searches. This greatly reduces the number of states explored and hence the running time.

Streaming: Let us assume that we acquire data from each of the sensors A and B whenever a car passes by. Obviously, for each car, we first acquire data from A and then from B . We order the cars in a *planning queue* sorted by the time at which the cars pass sensor A . The queue grows when new sensor data arrives from sensor A . Now when data from sensor B has arrived, we compute a trajectory for the car at the front of the queue. To this end, we use the algorithm of the previous section, such that the car avoids other cars for which a trajectory has previously been computed (which is initially none). The start state-time and the goal state-time are directly derived from the data acquired at sensor A and B respectively. They are rounded to the nearest point in the discretized state-time space. This procedure repeats indefinitely.

Streaming Property: The reconstructed trajectories can be regarded as a “movie” of the past, or as a function $R(t)$ of time. As new trajectories are continually computed, the function $R(t)$ changes continuously. However, the above scheme guarantees that $R(t)$ is *final* for time t if $(\forall i : t_i^A < t : t_i^B < t_{\text{cur}})$, where t_{cur} is the current “real world” time. “Final” means that $R(t)$ will not change anymore for time t when trajectories are determined for new cars. In other words, we are able to “play back” the reconstruction until time t as soon as all cars that passed sensor A before time t have passed sensor B . We call this the *streaming* property; it allows us to stream the reconstructed traffic at a small delay.

Real Time Requirements: In order for our system to run in *real time*, that is, so that the computation does not lag behind new data arriving (and the planning queue grows bigger and bigger), we need to make sure that reconstruction takes on average no more time than the time in between arriving cars. For instance, if a new car arrives every second, we need to be able to compute trajectories within a second (on average) in order to have real-time performance.

Prioritization: The above scheme implies a static prioritization on the cars within a given pair of sensor locations. Cars are assigned priorities based on the time they passed sensor A , and in order of decreasing priority trajectories are calculated that avoid cars of higher priority (for which trajectories have previously been determined). This is justified as follows: in real traffic, drivers mainly react to other cars in *front* of them, hardly to cars behind. This is initially the case: a newly arrived car i has to give priority to all cars in front of it. On the other hand, car i may overtake another car j , after which it still has to give priority to j . However, it is not likely that once car i has overtaken car j that both cars will ‘interact’ again, and that car j influences the remainder of the trajectory of car i . This can be seen as follows. If

we assume that cars travel on a trajectory with a constant velocity (this is what we try to achieve by the optimization criteria of Equation (3)), each pair of cars only interact (i.e. one car overtakes the other) at most *once*.

In fact, in a real-world application it is to be expected that multiple consecutive stretches of a highway are being reconstructed, each bounded by a pair of a series of sensors A, B, C, \dots placed along the highway. If car i overtakes car j in stretch AB , then for reconstructing the stretch BC , car i has gained priority over car j . So, when regarded from the perspective of multiple consecutive stretches being reconstructed, there is an implicit *dynamic* prioritization at the resolution of the length of the stretches.

Results: We show that this approach can successfully reconstruct traffic flows for a large number of cars efficiently, and examine the performance of our method on a set of real-world traffic flow data in [3]. Fig. 1 shows some of the challenging scenarios reconstructed and visualized by our method.

III. CONTINUUM TRAFFIC SIMULATION

Traffic forecast is usually performed by *forward simulation*, i.e. by propagating the flow model forward assuming some inflow and outflow conditions. A key component of our system framework is developing a fast and accurate traffic simulator. The main challenge is to model traffic flow: given a road network, some traffic measurements, and the initial vehicle states, how does the traffic in the system evolve? Such methods are typically designed to explore specific phenomena, such as congestion, shockwaves, unstable, stop-and-go patterns of traffic, or to evaluate network configurations to aid in real-world traffic engineering.

We develop a novel method for efficient modeling and simulations of large-scale, real-world networks of traffic using a continuum representation. Specifically, using traffic sensor measurement, we can transform discrete, moving vehicles on *multiple lanes* into continuous traffic flow, and vice versa. Our goal is to allow the individual vehicles to be introduced into this continuum representation of traffic flow at any given time. This approach of using continuum simulation with discrete vehicle behaviors could allow the simulation to capture both the macroscopic flow behavior, as well as the individual movement of each vehicle in the traffic. The sensor data collected from traffic loop detectors and camera/video sensors will be used to provide the initial states to the traffic simulator.

A. Mathematical Formulation

Next we describe the essential elements modeled in our continuum traffic simulation.

Road Networks We propose to simulate the traffic flow in a network of roads. Each road has one or more lanes and is connected to other roads through intersections and interchanges. We plan to design our data structure and software systems so that we can also model other information in the future, including road quality and conditions, information about driveways, parking spaces along the arterial roads,

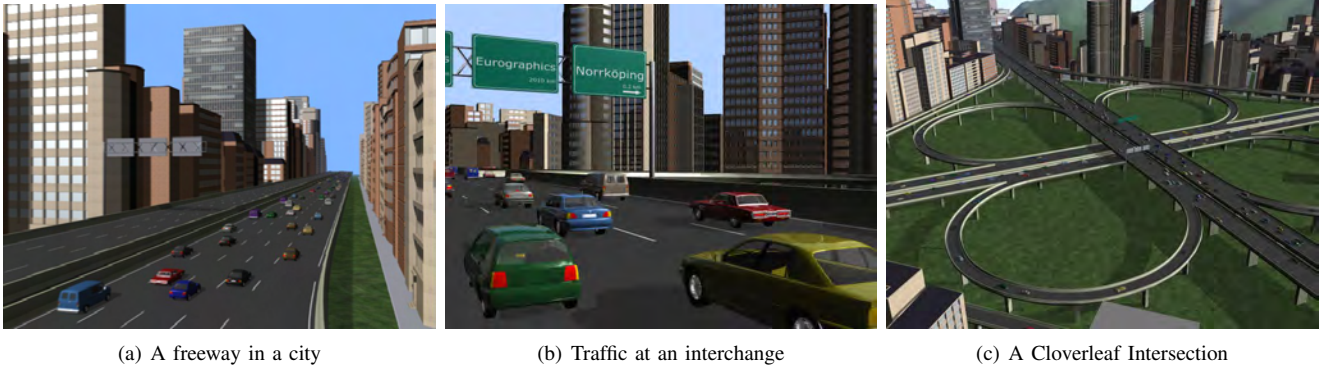


Fig. 2. Images from our simulator

etc. Each multi-lane road segment can also have variable speed limits. Moreover, driveways, parking lots, and curbside parking are treated as *sources* and *sinks* along lanes.

Traffic Flow We model the flow of traffic using a system of nonlinear hyperbolic conservation laws that represent traffic as a continuum along lanes. We use the ARZ model proposed by Aw and Rascle [1] and Zhang [9], which addressed non-physical phenomena of earlier partial differential equations for modeling traffic flows. The resulting equations can be written as [5]:

$$\mathbf{q}_t + \mathbf{f}(\mathbf{q})_x = 0, \text{ where } \mathbf{q} = \begin{bmatrix} \rho \\ y \end{bmatrix} \text{ and } \mathbf{f}(\mathbf{q}) = \begin{bmatrix} \rho u \\ yu \end{bmatrix}, \quad (7)$$

where the subscripts denote differentiation, \mathbf{q} is a vector-valued quantity of the traffic density ρ , i.e. “cars per length”, and the “relative flow” of traffic, y , and u as the velocity of traffic. $\mathbf{f}(\mathbf{q})$ is a vector-valued function, known as the *flux function*, uniquely characterizing the dynamics of the system.

Discretization: One way to discretize solutions to Eqn. (7) is based on the Finite Volume Method (FVM). If we take the integral of Eqn. (7) in space over some arbitrary interval $x \in [a, b]$, we have

$$\int_a^b \mathbf{q}_t dx + \int_a^b \mathbf{f}(\mathbf{q})_x dx = \frac{d}{dt} \int_a^b \mathbf{q} dx + \mathbf{f}(\mathbf{q}(b)) - \mathbf{f}(\mathbf{q}(a)) = 0. \quad (8)$$

We can divide Eqn. (8) by $(b - a) = \Delta x$ and discretize \mathbf{q} into quantities \mathbf{Q}_i representing the *average* of \mathbf{q} over $[a, b]$, followed by

$$\frac{\mathbf{Q}_i^{n+1} - \mathbf{Q}_i^n}{\Delta t} + \frac{1}{\Delta x} [\mathbf{f}(\mathbf{q}(b)) - \mathbf{f}(\mathbf{q}(a))] = 0, \\ \mathbf{Q}_i^{n+1} = \mathbf{Q}_i^n - \frac{\Delta t}{\Delta x} [\mathbf{f}(\mathbf{q}(b)) - \mathbf{f}(\mathbf{q}(a))] \quad (9)$$

Numerical update procedure Eqn. (9) is a straightforward update scheme; what remains to be computed are the quantities $\mathbf{f}(\mathbf{q}(b))$ and $\mathbf{f}(\mathbf{q}(a))$ — that is, the flux that occurs at the boundaries between cells. This is in fact not straightforward for nonlinear $\mathbf{f}(\cdot)$ (as in the ARZ system of equations) and accounts for the bulk of the computation in the underlying

numerical scheme. The problem of determining these fluxes is known as the *Riemann problem*.

Riemann Problem and Field Classification In order to compute fluxes, we must be able to determine the value of between the piecewise-constant states in adjacent cells. Depending on the relative values of initial and final constant states at the cell boundary, we expect the solution to consist of two or more distinct “regions” of self-similar solutions traveling with varying speeds.

The eigenstructure of the Jacobian of the flux function is the key to determining these speeds and the solution various regions of the solution. Based on the eigenvalues and eigenvectors of the Jacobian, we can perform field classification to determine when to expect nonlinear phenomena, such as shocks and rarefaction waves to appear or when the flow will propagate as constant discontinuity. Based on the field classification, we can classify the solutions for different cases of vehicle densities, velocities, etc.

Generalized Solutions for Variable Speed Limit The above discussion on the solution to the Riemann problem for the ARZ system of equations has assumed that the maximum velocity remains constant in space — i.e. that the speed limit on either side of the interface is the same. Clearly speed limits vary from road to road and changes even along a single lane, and the effects of these variations in speed limits have discernable effects on traffic flow and in fact can be used as a mechanism to regulate the flow. At a decrease in speed limit, we expect traffic to slow and increase in density, while an increase in speed limit might cause traffic to accelerate and rarefy. Whereas the solution to the Riemann problem developed above is a *homogeneous* Riemann problem, when speed limits on either side of a cell interface differ, we also need to solve these *inhomogeneous* cases. We investigate the inhomogeneous Riemann problem for the ARZ equations using the concepts of *supply* and *demand* of flow [4]. For more detail, please refer to [6].

Additional Issues: Other challenging issues include (1) handling of lane-end boundary conditions at the intersection, stopped outflow, ‘starvation’ inflow, taper in/out; (2) source terms due to driveways, parking lots, and curbside parking, (3) single-in and multiple-out Riemann solvers for handling

lane merging and splitting, etc. We are not aware of any prior work on real-time traffic simulation with this level of accuracy.

Discrete Representation of Cars: Another challenge is introducing discrete vehicle representation of vehicles in our proposed approach. In order to have the underlying continuum simulation reflect the position of these vehicles, we need to “seed” the discrete cells along each lane with the appropriate density and velocities of each car. We interpret the quantity ρ stored at each grid cells as “cars per length”; thus, for each cell i a particle j with velocity u_j overlaps, we compute the updated density (ρ'_i) and velocity (u'_i) at i from their original values $[\rho_i, u_i]^T$ and the contribution of j :

$$\Delta\rho_i = \frac{o_{i,j}}{\Delta x_{\text{lane}}}; \quad \rho'_i = \rho_i + \Delta\rho_i; \quad u'_i = \frac{\rho_i u_i + \Delta\rho_i u_j}{\rho'_i}. \quad (10)$$

Here $o_{i,j}$ is the length (in real, not parametric, space) that the particle j overlaps cell i . Our approach can also handle the movement of vehicles from one lane to another (interchangeably for a lane change or a merge) using a combination of information from particles and the density/flow data from the continuum model, similarly for the vehicle entering and existing the highway system.

Continuum-Discrete Coupling: One of the future key challenges is a truly hybrid traffic simulation that correctly provide coupling between the continuum and discrete simulation, thus enabling level-of-detail simulations for modeling traffic flows.

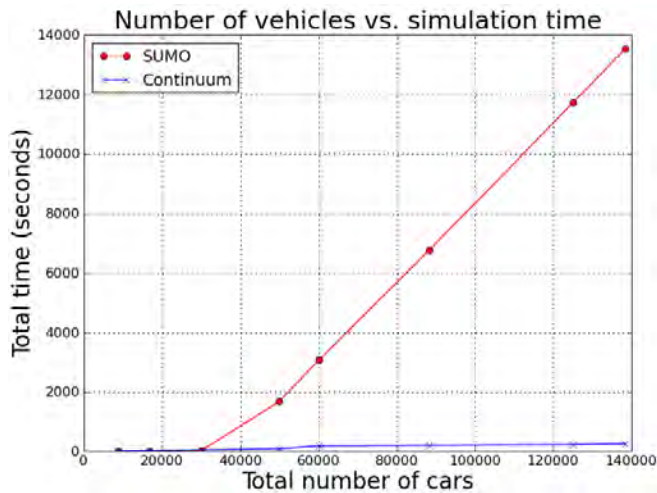


Fig. 3. Comparison on performance scaling of agent-based SUMO in red curve (top) vs. our simulator in blue line (bottom) as the number of cars increases.

B. Parallelization on GPU & Many-core Processors

The vast majority of the computation time in our simulation framework is spent in two kernels: (1) the computation of solutions to the many Riemann problems across the grid and (2) the application of these Riemann solutions to the cells of the grid to advance to the next timestep. The computation

of Riemann solutions is essentially independent across all cell interfaces. Given the two cells adjacent to a given interface, we compute the fluxes and speeds that comprise the Riemann solution at that interface. The update procedure is similarly data-parallel across each grid cell; to update a cell, we need only the global timestep being used for each pass and the Riemann solutions corresponding to the two interfaces shared with the cell’s neighbors along the current dimension.

We therefore can achieve significant performance scaling from a parallelization of these kernels across the grid. Our early multiple-CPU implementation parallelize the dynamics computations of traffic flow and show scalable performance. Some images from our traffic simulator can be seen in Fig. 2. And, a comparison against a well known agent-based simulation is shown in Fig. 3

IV. CONCLUSION AND FUTURE WORK

In this paper, we briefly summarize a novel traffic reconstruction algorithm based on a novel priority-based, multi-agent planning method [3] and an efficient continuum traffic simulation [6] for intelligent transportation systems and traffic congestion management. Next, we plan to investigate fast computations of optimal control strategies that can quickly evaluate various freeway and arterial control strategies using the traffic reconstruction and simulation techniques described in this paper and adaptively select the best rerouting options for individual vehicles to alleviate traffic congestion.

REFERENCES

- [1] A. Aw and M. Rascle, “Resurrection of “second order” models of traffic flow,” *SIAM Journal of Applied Math*, vol. 60, no. 3, pp. 916–938, 2000.
- [2] C. M. Clark, T. Bretl, and S. Rock, “Applying kinodynamic randomized motion planning with a dynamic priority system to multi-robot space systems,” *IEEE Aerospace Conference Proceedings*, vol. 7, pp. 3621–3631, 2002.
- [3] J. V. den Berg, J. Seawall, M. C. Lin, and D. Manocha, “Virtualized traffic: Reconstructing traffic flows from discrete spatio-temporal data,” *Proc. of IEEE Virtual Reality Conference*, pp. 183–190, 2009.
- [4] J.-P. Lebacque, H. Haj-Salem, and S. Mammar, “Second order traffic flow modeling: supply-demand analysis of the inhomogeneous riemann problem and of boundary conditions,” in *10th EURO Working Group Transportation Meeting*, Poznan, Poland, September 2005.
- [5] J.-P. Lebacque, S. Mammar, and H. Haj-Salem, “The aw-rascle and zhang’s model: Vacuum problems, existence and regularity of the solutions of the riemann problem,” *Transportation Research Part B*, no. 41, pp. 710–721, 2007.
- [6] J. Sewall, D. Wilkie, P. Merrell, and M. C. Lin, “Continuum traffic simulation,” *Computer Graphics Forum (Proc. of Eurographics)*, 2010.
- [7] D. Shrank and T. Lomax, “The 2007 urban mobility report,” Texas Transportation Institute, Texas A & M University, College State, Texas, Tech. Rep., 2007.
- [8] J. van den Berg and M. Overmars, “Kinodynamic motion planning on roadmaps in dynamic environments,” in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2007, pp. 4253–4258.
- [9] H. M. Zhang, “A non-equilibrium traffic model devoid of gas-like behavior,” *Transportation Research Part B: Methodological*, vol. 36, no. 3, pp. 275–290, March 2002.



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session III

Human-robot interaction

- **Keynote speaker: Oussama Khatib (Stanford University, USA)**
Title: Robot for the Human



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session III

Keynote speaker: **Oussama Khatib (Stanford University, USA)**

Robot for the Human

Abstract: Robotics is rapidly expanding into the human environment and vigorously engaged in its new emerging challenges. From a largely dominant industrial focus, robotics has undergone by the turn of the new millennium a major transformation in scope and dimensions. This expansion has been brought about by the maturity of the field and the advances in its related technologies. The new generation of robots is expected to safely and dependably co-habitat with humans in homes, workplaces, and communities, providing support in services, entertainment, education, health care, manufacturing, and assistance. Interacting, exploring, and working with humans, the new generation of robots will increasingly touch people and their lives. New design and fabrication concepts, novel sensing modalities, effective planning and control strategies, modeling and understanding of human motion and skills are among the key requirements discussed for the development of this new generation of human-friendly robots.

Biography: Oussama Khatib received his Doctorate degree in Electrical Engineering from Sup'Aero, Toulouse, France, in 1980. He is Professor of Computer Science at Stanford University. He is Co-Editor of the Springer Tracts in Advanced Robotics series, and has served on the Editorial Boards of several journals as well as Chair or Co-Chair for numerous international conferences. He co-edited the Springer Handbook of Robotics, which received the PROSE Award for Excellence in Physical Sciences & Mathematics and was also the winner in the category Engineering & Technology. He is a Fellow of IEEE and has served RAS as a Distinguished Lecturer, as a member of the Administrative Committee, and as the Program Chair of ICRA 2000. He is the President of the International Foundation of Robotics Research (IFRR) and a recipient of the Japan Robot Association (JARA) Award in Research and Development.



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010



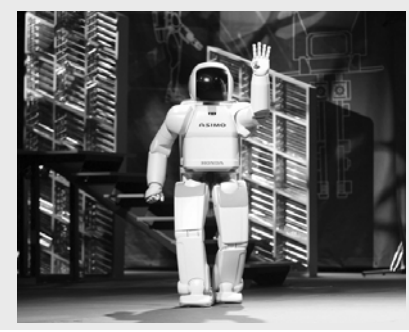
Robots for the Human

Oussama Khatib
 Artificial Intelligence Laboratory
 Department of Computer Science
 Stanford University

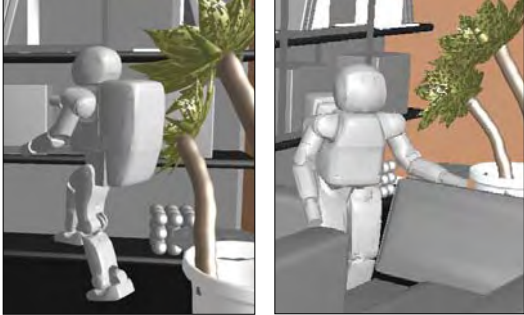
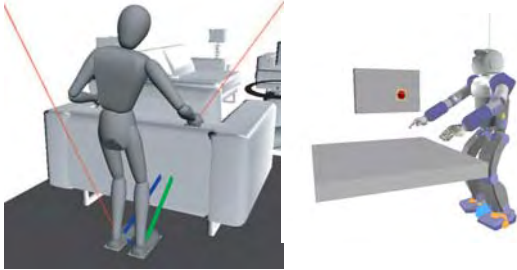
.. in the human environment



whole body control

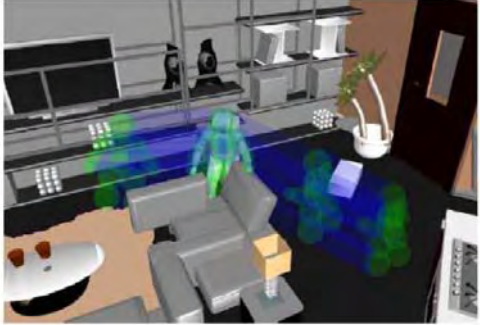


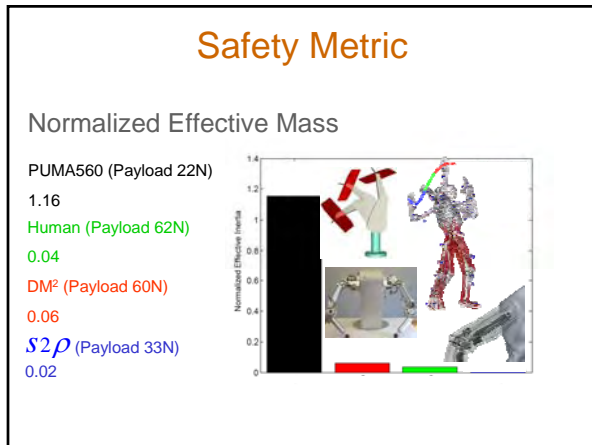
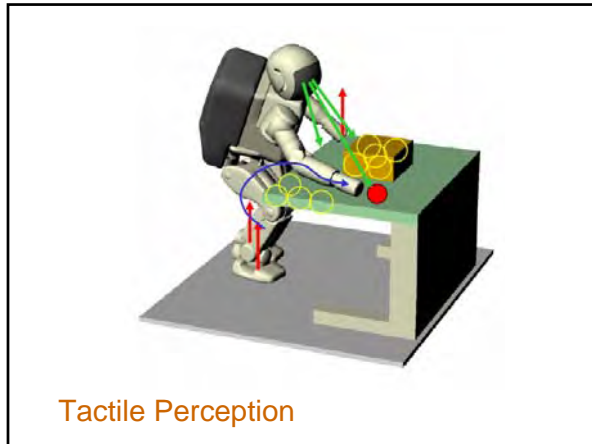
.. not only walk, but also interact with the world

.. motion in contact

perception





The Challenge

Sensing and Perception
real-time, unstructured world

Planning, Control, and Skills
*many degrees of freedom
robot control architecture
human-like skills, learning*

Human-Robot Interaction
cognitive and physical

Mechanisms and Actuation
safety & performance

Interactivity & Human-Friendly

Human-Friendly Robot Design

Requirements

- Safety
- Performance

Competing?

Technologies

Heavy structure

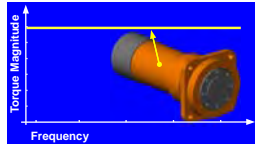
Conventional Geared Drive:

- Lighter structure
- Large reflected actuator inertia

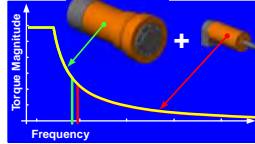
Effective Inertia
 $(J_{link} + N^2 J_{motor})$

Actuation Requirements

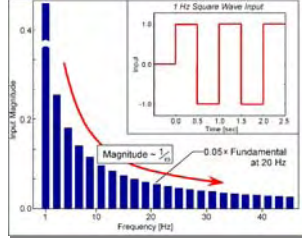
Assumed Torque Requirements



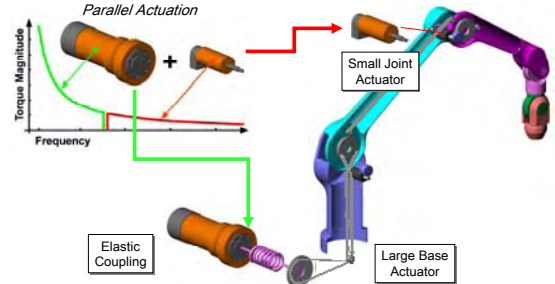
Actual Torque Requirements



Torque Vs Frequency: Square Wave



Distributed Macro Mini (DM²) Approach



DM² Performance

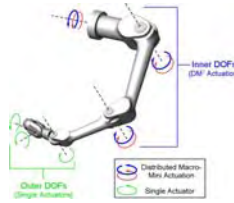
Distributed Macro-Mini Actuation → DM²

- 10x reduction in effective inertia
- 3x increase in position control bandwidth
- 10x decrease in trajectory tracking error

Safety AND Performance



DM² - Human-Friendly Robot

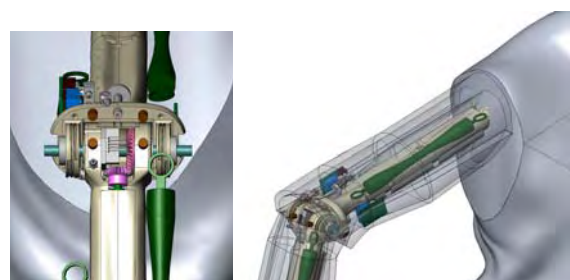


"the high capacity of a large robot with the fast dynamics and safety of a small one"

S2ρ : Stanford Human-Safe Robot



S2ρ : Stanford Human-Safe Robot

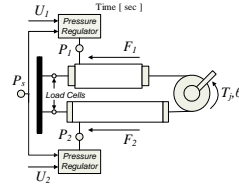
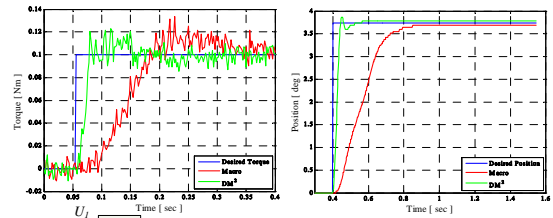


artificial muscles with electrical motors and compact pressure regulators

S2ρ : Stanford Human-Safe Robot



S2ρ : Stanford Human-Safe Robot

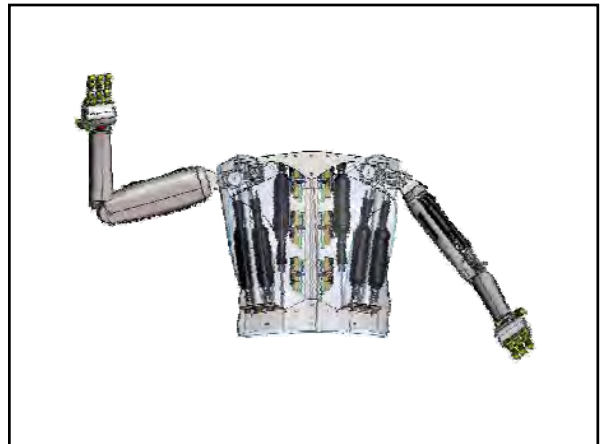
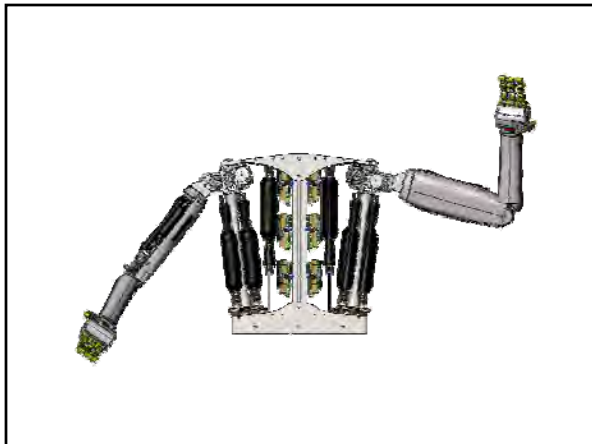
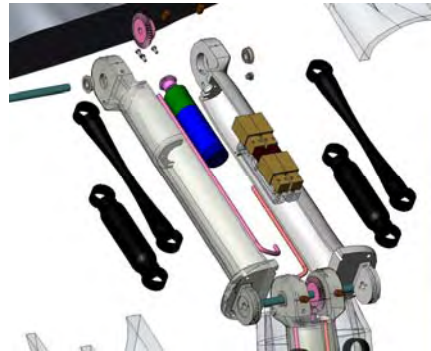


Macro.....0.5Hz
 Macro/Tension 7.0Hz
 Macro/Mini..... 35Hz

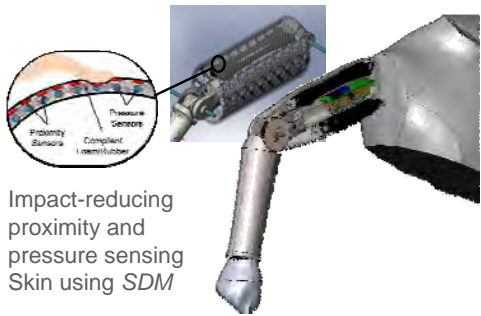
S2ρ_{1.5}: New Design



S2ρ : Stanford Human-Safe Robot



S2ρ : Stanford Human-Safe Robot



Impact-reducing proximity and pressure sensing Skin using *SDM*

The Challenge

Mechanisms and Actuation
 Safety & Performance
 Sensing and Perception
real-time, unstructured world
 Planning, Control, and Skills
many degrees of freedom
robot architecture
human-like skills, learning
 Human-Robot Interaction
cognitive and physical
 Interactivity & Human-Friendly



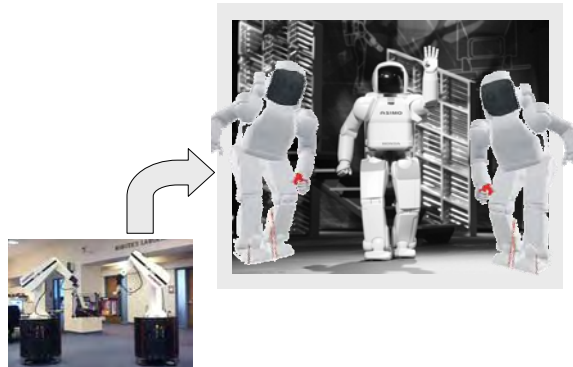
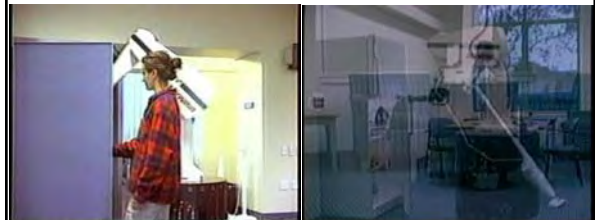
Mobile Manipulation

Human Guided Motion & Human-Robot Interaction
Romeo & Juliet (1993)



Mobile Manipulation

Human Guided Motion & Human-Robot Interaction
Romeo & Juliet (1993)



multiple postures
consistent with
 tasks
 multiple-point
 motion
 contact
consistent with
 internal constraints
 self collision
 local obstacles
 and balance
 }}

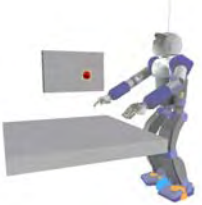


interact with the world,
 cooperate, and manipulate


Humanoid Robot Control

branching and under-actuated

- whole-body control
- constraints
- multiple contacts
- internal forces & balance
- manipulation skills



Whole-body Control



Task & Posture Decomposition

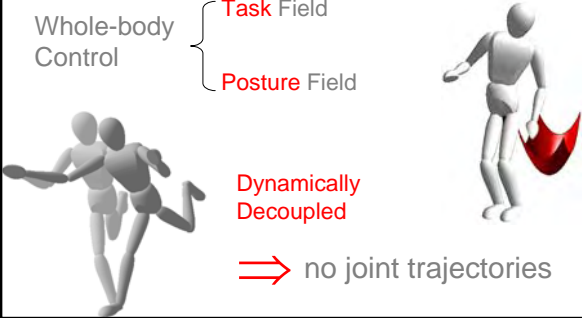

Task and Posture Control

Whole-body Control

- Task Field
- Posture Field

Dynamically Decoupled

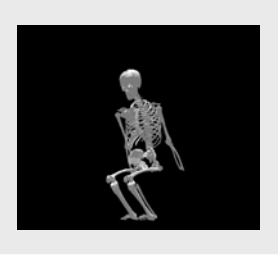
⇒ no joint trajectories

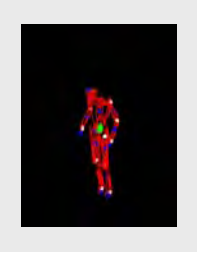
learning from the human

Simulation 79 DOF and 136 Muscles

Biometric Data & Bone Geometry



Dynamic simulation



Motion capture

Learning from the Human

Humans discover the *physio-mechanical advantage*

In learned tasks, humans minimize muscular effort, under physical and “social” constraints



Learning from the Human

In learned tasks, humans minimize muscular effort, under physical and "social" constraints



➔ Physiology-based Task/Posture Control

Physiology-based Posture Field

Human posture is adjusted to reduce muscular effort

Human-muscular Energy minimized:

$$E = cm^2$$

Function of physiology, mechanical advantage, and task

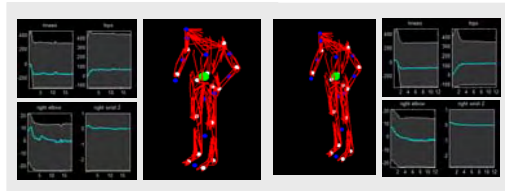
$$E = F^T \Phi F \quad \Phi(q) = J(L^T N_c^2 L)^{-1} J^T$$



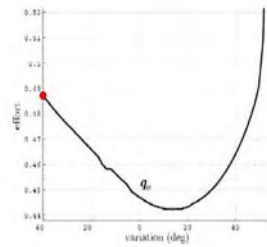
Data from Subjects



Data from Subjects

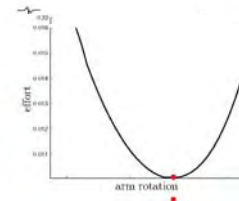


Validation - Arm Effort



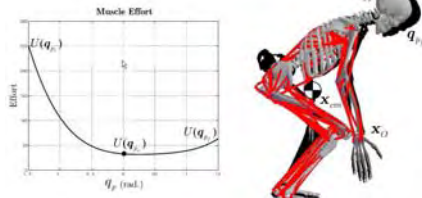
$$E = cm^2$$

Validation - Arm Effort

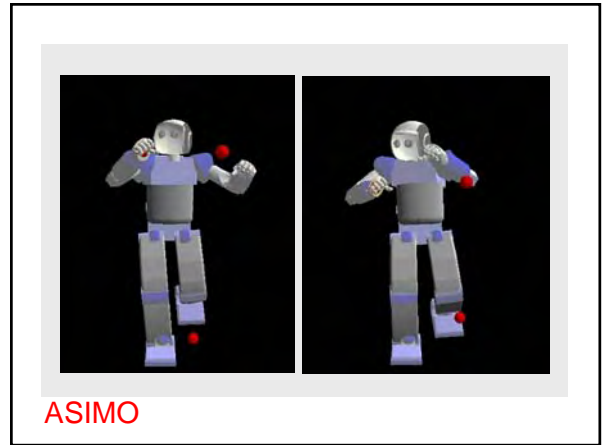
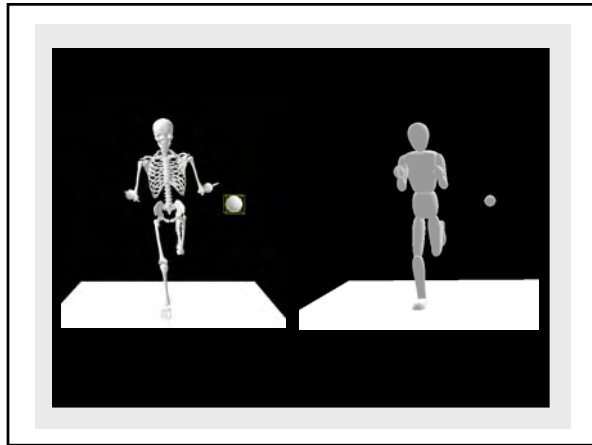
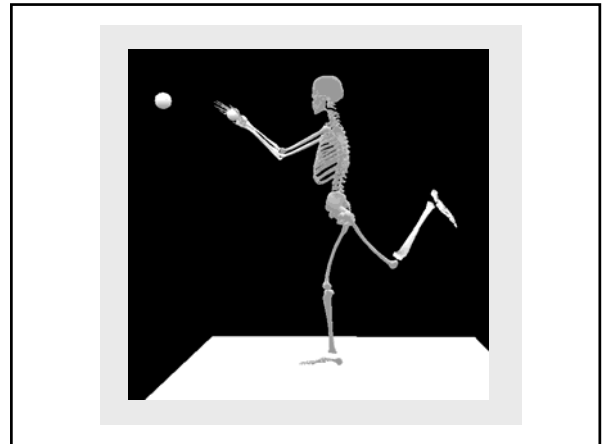


$$E = cm^2$$

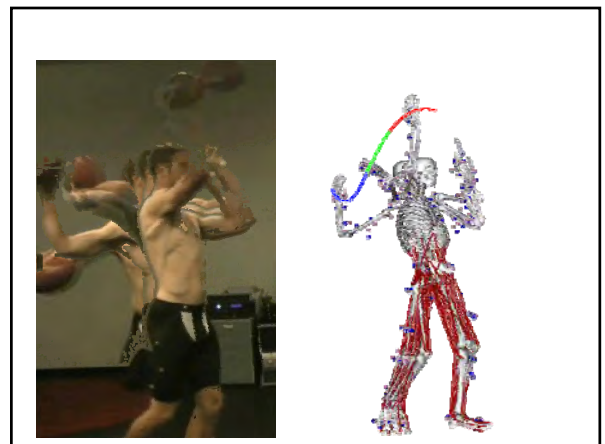
Validation – whole-body effort

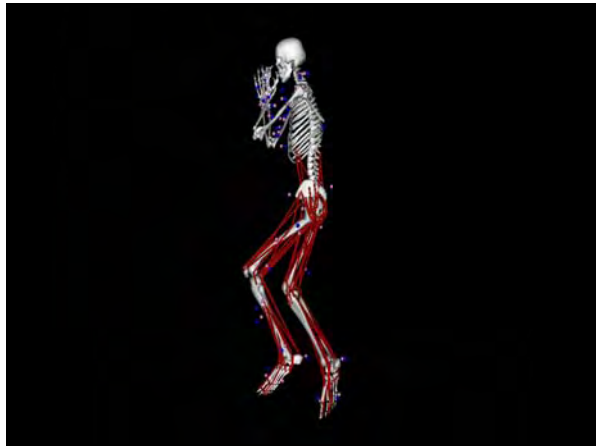


$$E = cm^2$$



SAI Neuromuscular Library





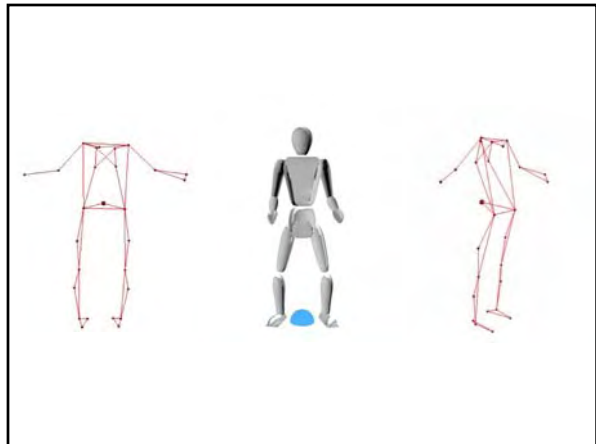
Minimum Muscle Effort Lines

$$E = cm^2$$

$$E = F^T \Phi F$$

Largest Acceleration Lines

Skill Learning – Tai Chi



Human Motion Reconstruction

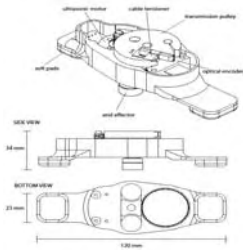
Injury prevention, Pathology Evaluation, and Athletics

```

    graph LR
      SM[Subject Motion] --> S[Sensing]
      subgraph S [Sensing]
        MC[Motion Capture]
        EMG[EMG]
      end
      S --> RP[Reconstruction Processing]
      subgraph RP [Reconstruction Processing]
        TC[Trajectory Controller]
        MC2[Muscle Controller]
        MM[Musculoskeletal Modeling]
      end
      RP --> ID[Interactive Display to Subject]
      subgraph ID [Interactive Display to Subject]
        VDT[Visual Display Tactile Display]
      end
      ID -- Multi-modal Dynamic Feedback --> SM
  
```

Human Motion Reconstruction

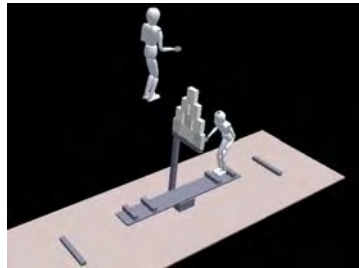
Injury prevention, Pathology Evaluation, and Athletics



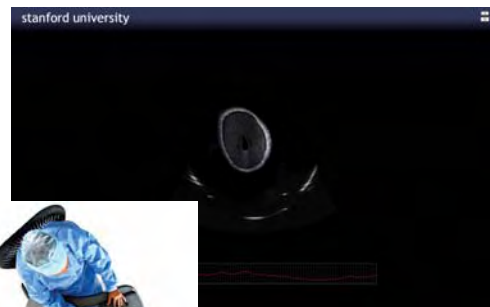
Proprioceptive Skin Stretch



Collision Detection



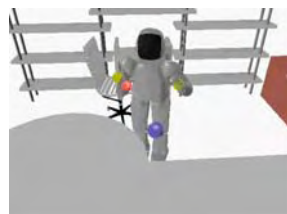
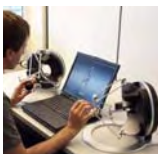
Collision Resolution



Ultrasound Imaging

Skill Learning

bi-manual haptic interface



haptics



.. simulating the sense of touch



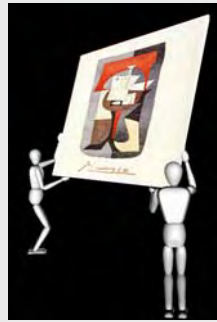
Haptics in Space



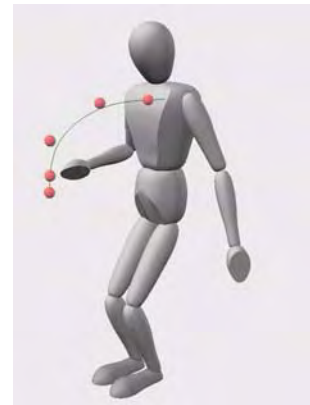
Ultrasound Imaging



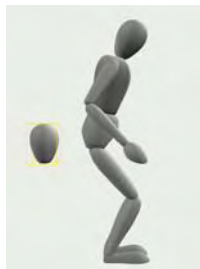
Constraints



Self Collision



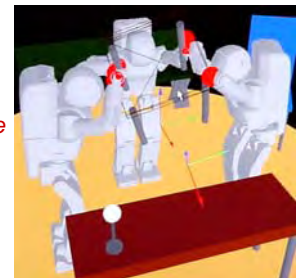
Obstacles



Elastic Planning

Real-time collision-free path modification

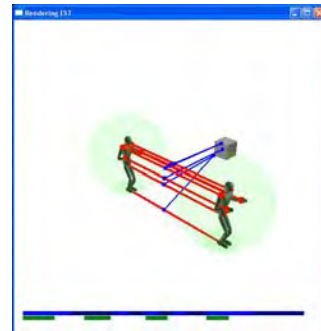
Connecting
Reactive Local Avoidance
with
Global Motion Planning



Elastic Planning



Efficient Elastic Planning

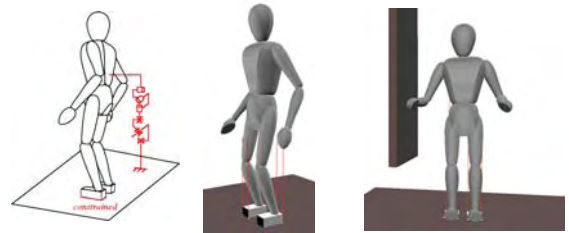


Integration of Locomotion



Multi-Contact Whole-body Control

Integration of Whole-Body Control & Locomotion



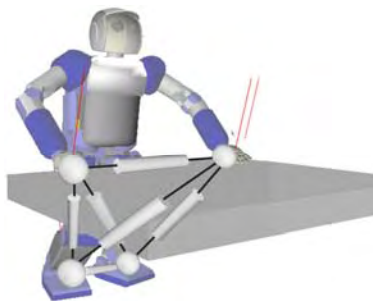
Under-actuated

Balance

Reaction forces

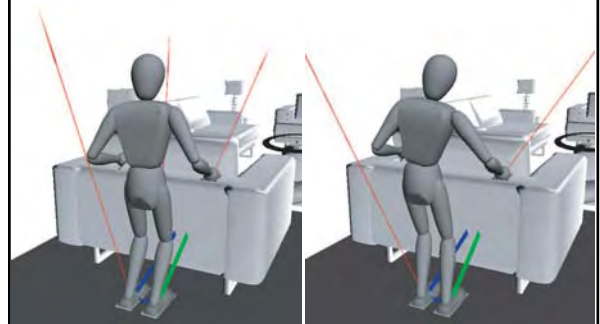
Balanced Supporting Contacts

Internal Force Control – Virtual Linkage




Unified Framework

Task, Posture, Constraints, Multiple Contacts, and Balance



Unified Framework

$$v_{\otimes} = \begin{pmatrix} v_{c|s} \\ v_{f|c|s} \\ v_{m|f|c|s} \\ v_{p|m|f|c|s} \end{pmatrix}$$

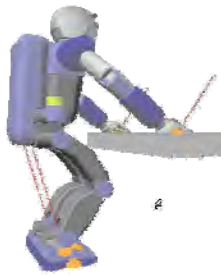


interact with the world,
cooperate, and manipulate

Unified Whole Body Control Framework

Task, Posture, Constraints, Multiple Contacts, and Balance

$$v_{\otimes} = \begin{pmatrix} v_{c|s} \\ v_{f|c|s} \\ v_{m|f|c|s} \\ v_{p|m|f|c|s} \end{pmatrix} \quad J_{\otimes} = \begin{pmatrix} J_{c|s} \\ J_{f|c|s} \\ J_{m|f|c|s} \\ J_{p|m|f|c|s} \end{pmatrix} \quad F_{\otimes} = \begin{pmatrix} F_{c|s} \\ F_{f|c|s} \\ F_{m|f|c|s} \\ F_{p|m|f|c|s} \end{pmatrix}$$



Unified Whole Body Control Framework


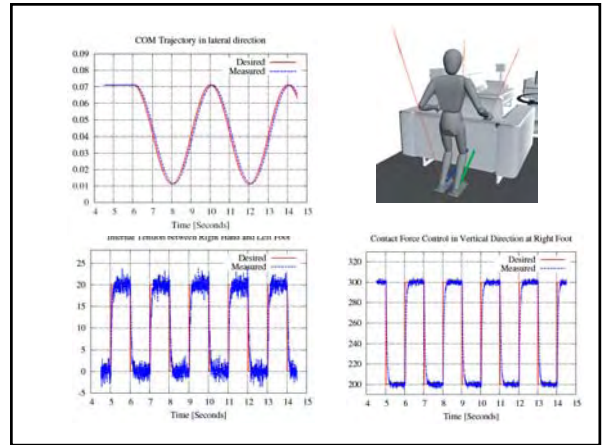
Task, Posture, Constraints, Multiple Contacts, and Balance

Dynamics

$$\Lambda_{\otimes} \dot{g}_{\otimes} + \mu_{\otimes} + p_{\otimes} + F_f = F_{\otimes}$$


Control

$$F_{\otimes} = \hat{\Lambda}_{\otimes} F_{\otimes}^* + \hat{\mu}_{\otimes} + \hat{p}_{\otimes}$$

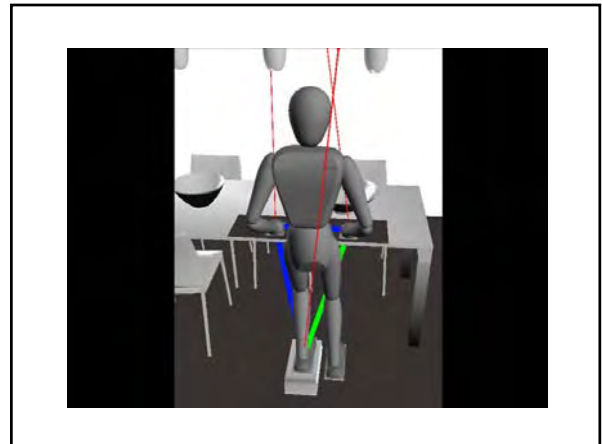
$$F_{\otimes}^* = \begin{pmatrix} F_{c|s}^* \\ F_{f|c|s}^* \\ F_{m|f|c|s}^* \\ F_{p|m|f|c|s}^* \end{pmatrix} \quad \Gamma_a = (\overline{UN}_s)^T J_{\otimes}^T F_{\otimes}$$



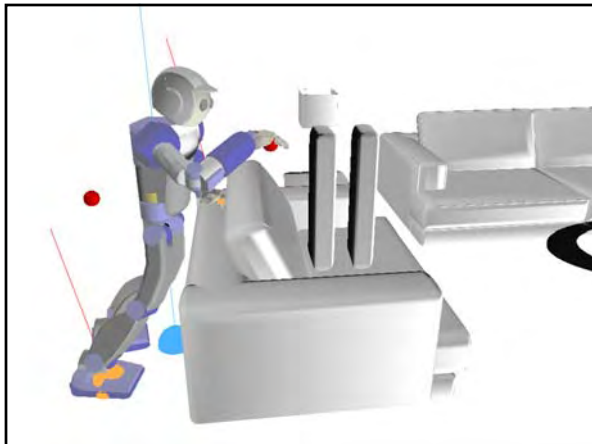
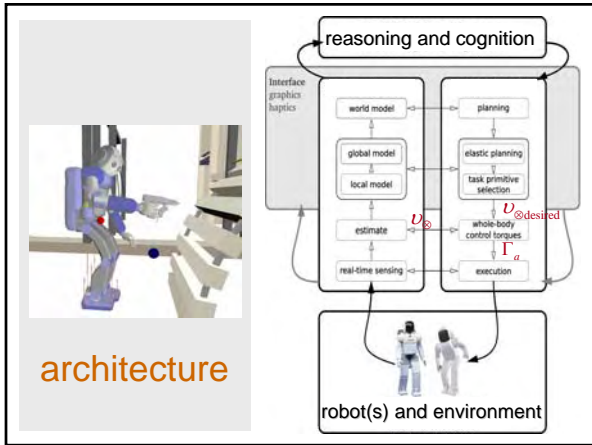
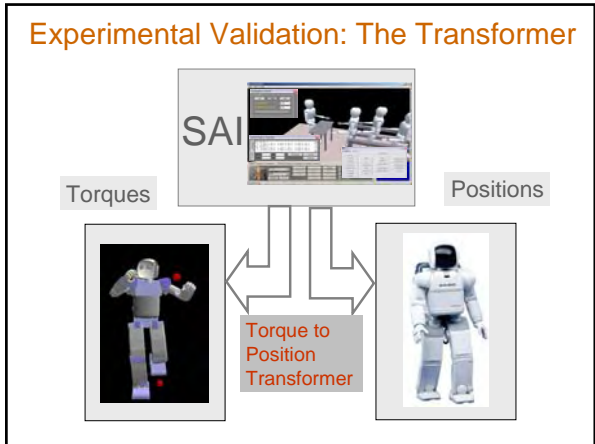
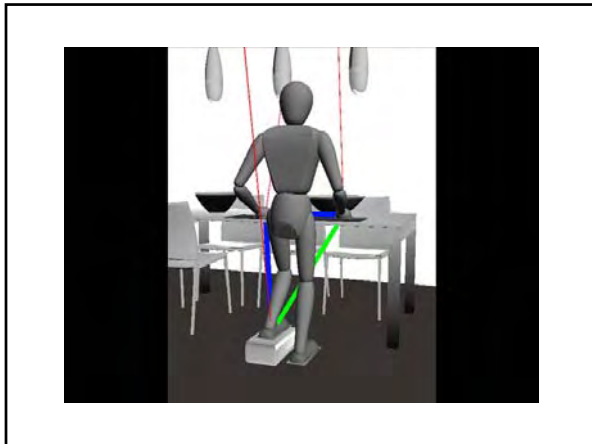
Unified Framework

- posture
- consistent with {
- task
- consistent with {
- contact
- consistent with {
- internal constraints
- self collision
- local obstacles
- consistent with {
- balance



interact with the world,
cooperate, and manipulate







2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session IV

Interactive session

- **Title: Benchmark Tools for Evaluating AGVs at Industrial Environments**
Authors: Hector Yuste, Leopoldo Armesto and Josep Tornero
- **Title: Automatic Routing System for Intelligent Warehouses**
Authors: K. T. Vivaldini; J. P. M. Galdames, T. B. Pasqual, R. M. Sobral; R. C. Araújo, M. Becker, and G. A. P. Caurin
- **Title: Coordinating the motion of multiple AGVs in automatic warehouses**
Authors: Roberto Olmi, Cristian Secchi and Cesare Fantuzzi
- **Title: ArosDyn: Robust Analysis of Dynamic Scenes by means of Bayesian Fusion of Sensor Data - Application to the Safety of Car Driving**
Authors: Christian Laugier, Igor E. Paromtchik, Mathias Perrollaz, Mao Yong, Amaury Nègre, John-David Yoder, Christopher Tay
- **Title: Real-Time Detection of Moving Obstacles from Mobile Platforms**
Authors: Chunrong Yuan and Hanspeter A. Mallot
- **Title: Studying of WiFi range-only sensor and its application to localization and mapping systems**
Authors: F. Herranz, M. Ocaña, L. M. Bergasa, M. A. Sotelo, D. F. Llorca, N. Hernandez, A. Llamazares and C. Fernandez
- **Title: Terrain Classification for Improving Traversability of Autonomous Vehicles**
Authors: Jayoung Kim, Jonghwa Lee, Jihong Lee



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Benchmark Tools for Evaluating AGVs at Industrial Environments

Héctor Yuste, Leopoldo Armesto and Josep Tornero

Abstract—The paper addresses the problem of evaluating AGVs with different degrees of autonomy by defining a methodology and benchmark tools to grade the performance of each solution. The proposed benchmark requires running different experiments, from manual driving to autonomous navigation, at different velocities and different scenarios. The goal is to evaluate the performance of AGVs, in terms of robustness to reach the goal, collisions reduction, traveling time, average speed, etc. The underlying objective is to evaluate the potential advantages of manual-assisted driving as well as autonomous navigation against standard manual driving. To obtain valid and significant results, 180 experiments have been completed on each case with drivers of different ages, sex and skills.

I. INTRODUCTION

Advanced Auto-Guided Vehicles (AGVs) involved in industrial applications constitute interesting Intelligent Transportation Systems (ITS) for researchers and engineers. Commercial AGVs like those found in [1], [2] among others, are typical based on magnets, wires or laser guidance, which require a specific infrastructure. Another type of guidance for AGVs is the inertial navigation, using accelerometers and gyroscopes aided with exteroceptive transponders embedded in the floor. The advantages are to provide a fully autonomous solution with capacity to react, to avoid obstacles and to compute the path by using the most advance techniques. The drawbacks of these solutions are, in our opinion, the high implantation cost and the general lack of flexibility make unfordable the use of this kind of technology, especially for SME (Small-Medium Enterprises). On the other hand, traditional manually maneuvering with forklifts has advantage of providing human intelligence to adapt to unstructured and cluttered environments. However, human behavior is occasionally risky and dangerous because they may underestimate a particular situation. It seems appropriate to investigate on new ITS that combine both, human intelligence and computational capabilities of an artificial system, to improve security and reduce the risk of having accidents. This kind of ITS have become increasingly common in cars, see section I-A for details, but rarely used in industrial environments.

The reduction of accidents of transportation systems, is nowadays, one of the main goals in many companies. Accidents with industrial vehicles have negative consequences for companies such as: delays in scheduled tasks affecting to

the manufacturing process, damages on the vehicles which require expensive reparations and maintenance, worker casualties and economical losses. Accidents can be caused by human driving errors, but also by unexpected obstacles, machine failures or incorrect signaling of restricted areas, among other causes.

This paper discusses advantages and disadvantages of different AGV control modes in the context of industrial environments using industrial forklifts covering several driving modes from traditional manual driving mode to autonomous navigation, including a hybrid solution where an intelligent system assists manual driving so both the human operator and the intelligent system play a cooperative role. In order to provide valid and useful ITS solutions to improve security in industrial environments, benchmarking tools are needed to evaluate techniques and methods under the same conditions.

In that sense, the main contribution of the paper is the definition of a methodology for evaluating ITS approaches, especially in the context of robustness and security performance with industrial forklifts, but also in the context of timing aspects as well as trajectory shape. The methodology establishes to run experiments under different scenarios and different velocities for each ITS approach being evaluated. In particular, we grade the performance of manual driving, that is driving with no kind of assistance, manual-assisted driving and an advance obstacle avoidance technique appropriate for troublesome scenarios for autonomous navigation.

A. Intelligent Transportation System Technologies

The Antilock Braking System (ABS) first brought to market by Bosch in 1978 prevents wheel lock during full braking. This ensures that the vehicle can still be steered and moved out of the way of unexpected obstacles. Adaptive Cruise Control (ACC) technology improves the function of standard cruise control by automatically adjusting the vehicle speed and distance to the vehicle ahead. Adaptive headlights can direct the beams by moving each headlamp left, right, up or down in reaction to steering wheel angle, speed and movement of the vehicle. The Lane Change Assistant [3] or the Blind Spot Detection systems [4] continuously monitor the rear blind spots on both sides of the vehicle. Driver Drowsiness Monitoring and Warning systems [5] can detect the driver's drowsiness in several ways: by tracking the driver's facial features, movements of hands and feet, by analyzing eye-closures and head pose or even changes in heartbeat. The Electronic Brake assist System (EBS) is a very efficient aid in emergency braking situations when the driver wants the vehicle to stop as quickly as possible which can be found in Mercedes-Benz (S-Class, SL-Class).

This work was supported by PISALA Project funded by Vicerectorado de Investigación Desarrollo e Innovación, Universidad Politecnica de Valencia and PROMETEO Program funded by Conselleria Educació, Generalitat Valenciana.

Héctor Yuste, Leopoldo Armesto and Josep Tornero are with Institute of Design and Manufacture in Automation (IDF-Automation), Universidad Politécnica de Valencia, Camino de Vera s/n, Spain. Corresponding author: Leopoldo Armesto {leoaran@isa.upv.es}

The Electronic Stability Control [6] detects the deviation between the vehicle's trajectory and the intended direction. Without any action on the part of the driver, small amounts of braking are applied separately to each wheel and this can bring the vehicle back to the intended course. Lane Departure Warning Systems (LDWS) are electronic systems, found initially in Nissan, Toyota and Citroën that monitor the position of the vehicle within its lane and warn the driver if the vehicle deviates or is about to deviate from the lane. Obstacle / Collision Warning Systems help the driver to prevent or mitigate accidents by detecting vehicles or other obstacles on the road ahead and by warning the driver if a collision becomes imminent. Current solutions with limited performance are an additional function of Adaptive Cruise Control, using information obtained from radar sensors to give visual and acoustic warnings [7] and [8]. Intelligent Speed Adaptation (ISA), also known as Intelligent Speed Assistance, is any system that constantly monitors vehicle speed and the local speed limit on a road and implements an action when the vehicle is detected to be exceeding the speed limit, see [9] for a complete review. Other assistance systems are gear shift indicator, night vision, adaptive light control, automatic parking, traffic sign recognition and hill descent control, among others.

B. Motion planning and obstacle avoidance techniques

Literature on mobile robot motion planning and reactive obstacle avoidance of mobile robots considers the problem of how to reach a goal pose without colliding with the environment. Potential Field (PF) methods [10] and [11] addressed the first sensor-based motions, where a large set of different potential functions have been proposed [12], [13], [14] among others. The Vector Field Histogram (VFH) [15], [16], [17] considered sensor uncertain sensor to avoid obstacles with application to sonar sensors using occupancy grids. Generalized Perception Vector (GPV) [18] is comparable to the VFH but linked to the sensor instead of occupancy grids. In [19] an extension of the GPV was developed by considering an orientable eccentric ellipsoid as focus of attention based on the movement direction for non-holonomic mobile robots. The Elastic Bands [20] was the first technique combining planning and reaction schema in a unified framework. The Dynamic Window Approach (DWA) was the first technique to address kinematics and dynamics to carry out motion at high speeds [21] and similarly [22]. More recently, the Nearness Diagram (ND) navigation [23] was the first technique to address motion in troublesome scenarios, where several variations of the algorithm can be found [24], [25], [26], [27].

II. ITS BASED ON INDUSTRIAL FORKLIFTS

In this section, we summarize our research on ITS for industrial forklifts. In the context of a several past Research Projects named Auto Trans, GATA and LITRONA, we have developed several approaches for automating industrial vehicles for different applications such as teleoperation [28] and vision-based line tracking [29], [30]. More recently, we have proposed an unified and general approach for automating

vehicles in a range from manual driving to high-level automation modes in autonomous navigation, including several medium automation levels solutions such as "guided" driving [31] and manual-assisted driving in addition to teleoperation, vision-based line tracking, etc.

A. Vehicle Automation

An auto-guided vehicle based on the Nichiyu FBT15 industrial forklift with three wheels in tricycle configuration (an orientable wheel at the rear and two fixed wheels at the front). A mechanical link joints the rear motor to the steering-wheel with power-assistance. In additional, two DC motors attached to the two fixed wheels, are coordinates with the rear wheel through an electronic differential system for avoiding slippery.

The automated industrial vehicle is based on a PLC for low-level vehicle control and an industrial PC at high level, implementing the intelligence of the AGV. The purpose of the PLC is to provide an industrial solution to the most critical aspects regarding with emergency stops and signal interface. The devices have been installed on the vehicle are:

- Two lasers rangers form SICK providing a 180° range scan with 0.5° angular precision at a rate of 75Hz (see Fig. 1) at the rear and front of the Vehicle. They provide warning and protection zones that activate a digital output in case of intrusion to implement emergency stops.
- Sonar rangers and infrared sensors to cover lateral areas of the vehicle for moving in narrow areas.
- Two incremental encoders measure speed of fixed wheels an an absolute encoder to measure the angle of the rear steering wheel.
- PLC controller analog inputs: one sensing the throttle pedal and another one sensing the torque applied to the steering wheel.
- PLC controller analog outputs: one for controlling the vehicle drive velocity that replaces the original throttle pedal signal and another signal for controlling the rear steering wheel as if a power-assisted steering wheel.
- PLC controller digital inputs and outputs: for sensing and generating signals related with laser warning and protection intrusion, level gear (forward, neutral and reverse) and other non-relevant signals.

B. Teleoperation

Likewise manual-assisted driving, teleoperation is human driven but with the main difference that vehicle control has to be performed remotely. Therefore, wireless communications should be established to transmit remote control and sensing and that a "replica" of the vehicle, namely a teleoperation cabin, to reproduce vehicle commands. In this application force feedback control becomes more important to sense the environment through the steering wheel.

C. Vision-based Line Tracking

In vision-based line tracking applications an "intelligent" system processes images from a camera pointing to the floor where a line circuit defines the route to follow. As a result,

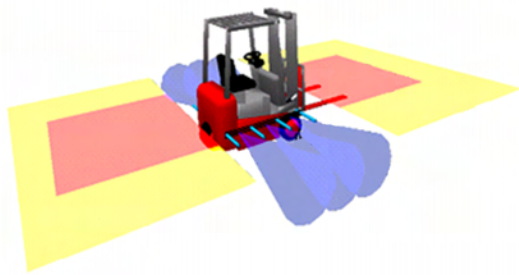


Fig. 1. Security zones covered by range sensors.

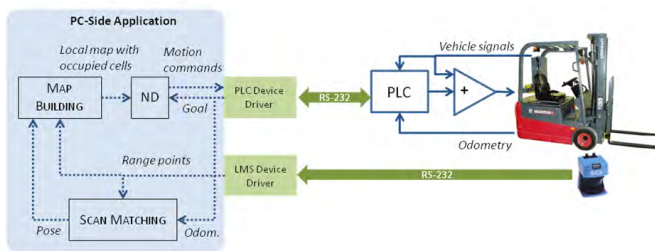


Fig. 2. Component-based software architecture. On the PC-side application there are three main components: scan matching, map building and nearness diagram (ND), apart from others such as display and data-logging modules (not included here). The PC communicates through RS-232 with the PLC and the LMS Devices.

the detected line on the image is used as reference to guide the vehicle. In this application, the “intelligent” head that processes images should be robust enough to detect lines under several lighting conditions and floor backgrounds. The proposed solution is based on the CMUCAM3 embedded vision system to detect lines [31].

D. Autonomous Navigation

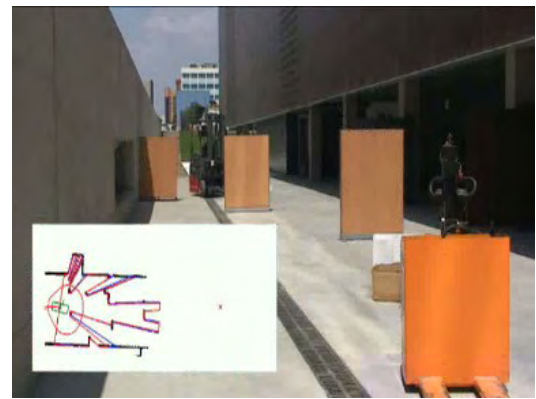
The autonomous navigation application has been designed using component-based software architecture, where the main modules are shown in Fig 2. In particular, we are using scan matching techniques to locally estimate the vehicle pose [32] and occupancy grid maps to fuse data from multiple scans [33]. In addition, we have selected ND [25] as the obstacle avoidance technique, which has shown appropriate behavior to avoid obstacles at moderate velocities.

In Fig. 3(a) we show an example of the autonomous navigation application, where the vehicle moved on a cluttered unstructured environment as shown in Fig. 3(b).

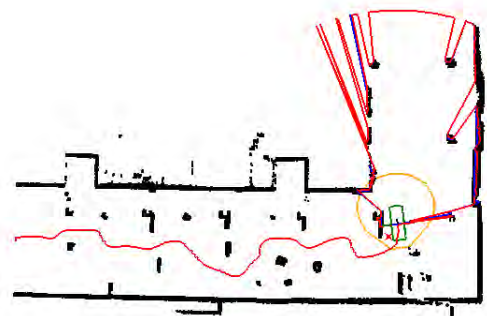
In the previous example the vehicle tries to reach the target from a start point along a very cluttered environment. There was no constrain to solve the problem.

E. Manual-Assisted Driving

In manual assisted driving, the driver normally operates the vehicle, but it can receive feedback from different kind



(a) Experiment scenario.



(b) Complete map and robot trace.

Fig. 3. Example of autonomous navigation of the industrial forklift in a very cluttered environment.

of visual, audio and haptic devices aimed to reduce the risk of collisions. The main idea is to investigate on feasible ready-to-market solutions that improves driving security aspects with industrial forklifts. Our particular approach focuses on haptic feedback devices, with which drivers can feel the danger of selecting an inappropriate steering wheel direction by applying a torque on the steering wheel opposite to that direction. In addition to this, the vehicle speed is automatically regulated as in Adaptive Cruise Control (ACC) systems. Our implementation is based on the GPV technique [19], where the main ideas can be summarized as follows: obstacles inside an ellipsoid surrounding the vehicle are taken into account to compute a coefficient indicating the risk of collision. This coefficient is used to cancel the throttle pedal commands introduced by the driver and even to generate a negative acceleration if required. This coefficient is also used to generate a proportional reactive torque on the rear motor that is feedback on the steering wheel so the driver feels higher stiffness while trying to move towards an inappropriate direction.

The main advantage of manual-assisted driving with respect to autonomous driving are easiness implantation, since it requires fewer changes to companies, with greater flexibility and adaptability to route re-planning. In addition to this, it does not required a specific infrastructure which im-

plies lower implantation costs. Moreover, the manual-assisted driving can improve pure manual driving performance in reducing collisions and correcting common human mistakes which are also costly for companies.

III. BENCHMARKING METHODOLOGY

In order to grade the performance of an AGV, we propose a complete benchmark:

- Define different types of scenarios where the vehicles move. Possible scenarios may include wide and narrow corridors, small and large isles, rooms, slalom-like parts requiring zig-zag maneuvering, moving objects (other vehicles or people). Combinations of these elements could be considered based on the application requirements.
- For each scenario a collection of pairs start-goal positions are defined (in our case 10). The criteria for choosing these group of points is as follows: the start and target points, must be located in the free space of the augmented scenario with a margin distance, the euclidean distance between start position and target position must be greater than a given threshold. Moreover, the location of each pair of points must be in different quadrants of the scenario to cover the overall parts of the scenario. The final selection of points has been filtered manually so that the problem to solve results interesting in terms of narrow corridors, slaloms, u-shaped obstacles to avoid.
- Define a set of maximum velocities that the vehicle can reach. These are interpreted as the maximum speed that the vehicle reacts when the driver fully accelerates. The speed time constant is fixed so the variation of the maximum speed from one experiment to another is sensed by the driver as a change in the sensitivity of the throttle pedal. In our case the we have chosen 5 different maximum speeds: $1m/s$, $2m/s$, $3m/s$, $4m/s$, $5m/s$.
- Each single test is performed as follows: A start-goal positions couple and a maximum speed are chosen randomly from the two sets defined before. The user is asked to go from the start position to the target without any limitation, but a maximum time. The driver has previous knowledge about the type of driving mode and is informed about the time left. Nevertheless, he does not know the maximum speed that the vehicle can reach nor how the algorithm (if exists) is working.
- A fixed number of experiments is performed for each scenario and driving mode (10 in our particular case). Some data is logged to posterior analysis of performance: start and goal positions, distance to obstacles, robot positions, time instants, collisions, etc... to obtain metrics of the performance. Related to robustness, an experiment is classified as positive (P) if it has reached the goal within the time interval and classified as negative (N) if not. Moreover, true positive (TP) experiments are those that reached the goal without colliding, while false negative (FN) experiments have at least generate one collision to reach the goal.

The simulation platform used to perform the tests was Player/Stage. To make the experiments as real as possible

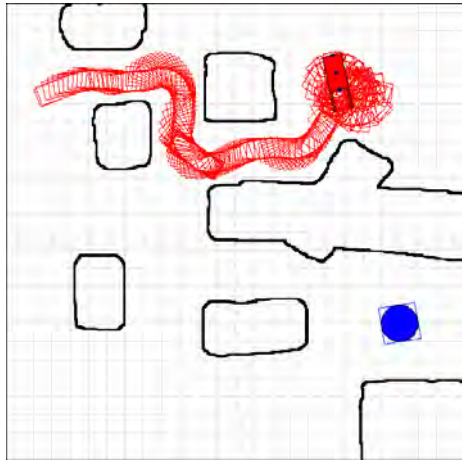
TABLE I
ROBUSTNESS OF DRIVING MODES AT DIFFERENT SPEEDS.

Speed [m/s]	Case [%]	Mode		
		Manual	Assisted	Autonomous
1	TP	88.23	95	76.92
	FP	8.82	5	2.56
	N	2.94	0	20.51
2	TP	60.52	100	56.75
	FP	34.21	0	2.7
	N	5.26	0	40.54
3	TP	70.27	94.87	85.71
	FP	29.72	5.12	0
	N	0	0	14.28
4	TP	79.41	94.73	80.95
	FP	14.7	5.26	2.38
	N	5.88	0	16.66
5	TP	56.75	96.66	92.59
	FP	37.83	3.33	0
	N	5.4	0	7.40

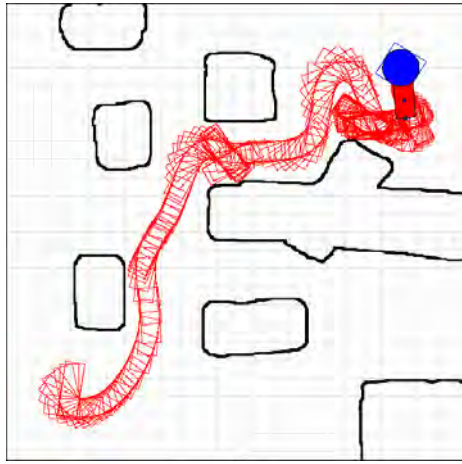
the user interact with the real truck but statically, that is, without a real movement, and the simulation is carried out on the on-board PC. All vehicle signals (throttle pedal position, steering wheel angle, break and selected gear) are sensed from the real truck in real time while laser data is supplied by the simulation platform. Vehicle signals are converted into appropriate position Player/Stage commands to simulate the robot motion. The Stage simulation is displayed on the screen as main GUI.

Table I contains the results of one of the simulated environments based on Player/Stage [34]. In particular, we are using the “cave” scenario and evaluating the performance of three driving modes. This scenario contains wide and narrow corridors and small and large isles. The driving modes that have been considered are manual driving, manual-assisted driving with a haptic device as proposed in II-E and autonomous navigation as described in section II-D. The experiments have been carried out with people external to the project in order to minimize the manipulation of results, including people of different sex and with ages between 25 to 50. A total amount of 180 experiments have been performed on each driving mode. From the total amount of people that did the experiments a 40% were experts in driving industrial trucks whilst the rest had experience only with cars.

It can be appreciated from the results of table I that manual-assisted driving is clearly more robust than manual and autonomous modes. In particular, negative cases of autonomous mode are basically due to trap situations that the ND could not solve as shown in Fig. 4(a), while negative cases of manual and assisted modes are basically due to lack of time to solve the scenario. In favor to ND, the number of false positive cases is lower than manual and manual-assisted modes, which implies lower collisions. However, in order to properly interpret the results of table I, the human factor must be taken into account, specially for non-skilled drivers who prefer to collide several times moving forward and backward until they can solve a trapping situation as shown in Fig. 4(b). The autonomous mode tries to move forward and backward but without colliding, but as a result it can not escape from the trapping situation. It is also interesting to remark that



(a) N case in autonomous mode.



(b) FP case in manual mode.

Fig. 4. Example of a negative (N) and false positive (FP) cases in the cave scenario. The blue point is the goal to reach.

manual-assisted mode has been designed to be usable as main design priority so force feedback applied to the steering wheel and speed reduction are moderate. As a consequence, drivers can collide with objects on purpose and therefore this is the trade off between usability and security we need to pay.

In addition to robustness, the performance of an AGV can be grade with several metrics as those proposed in [35] for the positive cases (including true positives and false positives). In particular, we have considered, for each velocity and driving mode, the mean time employed to reach the goal, the average speed, two security metrics as defined in [35], the bending energy and smoothness of the described trajectory as shown in table II. Regarding with the security metrics (SM), we have considered $SM2$ and $SM3$, where $SM2$ is the mean of minimum distances to the obstacles taken at each sampling time. This gives an idea of the risk taken through the entire mission, in terms of the proximity to an obstacle. On the other hand, $SM3$ is the minimum distance between the vehicle and any obstacle through the entire trajectory. This index measures the maximum risk taken throughout the entire mission. On the other hand, the

TABLE II
METRICS OF DRIVING MODES AT DIFFERENT SPEEDS.

Speed [m/s]	Metrics	Mode		
		Manual	Assisted	Autonomous
1	Mean Time [s]	53.1	46.06	95.79
	Av. Speed [m/s]	0.7	0.68	0.31
	Path Length [m]	27.15	24.18	29.02
	$SM2$ [m]	1.17	1.28	1.17
	$SM3$ [m]	0.29	0.42	0.54
	TBe	5.01	6.71	1.11
	Smoothness	0.7	0.13	0.05
2	Mean Time [s]	45.27	35.96	42.71
	Av. Speed [m/s]	1.02	0.92	0.6
	Path Length [m]	27.44	31.98	26.47
	$SM2$ [m]	1.1	1.28	1.1
	$SM3$ [m]	0.22	0.4	0.5
	TBe	20	17.31	4.72
	Smoothness	3.63	0.67	0.13
3	Mean Time [s]	45.78	34.4	33.1
	Av. Speed [m/s]	1.1	1.01	0.93
	Path Length [m]	27.49	29.93	33.44
	$SM2$ [m]	1.1	1.25	1.13
	$SM3$ [m]	0.27	0.38	0.51
	TBe	33.15	28.37	13.71
	Smoothness	9.36	1.38	1.7
4	Mean Time [s]	40.75	33.15	25.82
	Av. Speed [m/s]	1.11	1.03	1.23
	Path Length [m]	29.8	26.42	38.26
	$SM2$ [m]	1.16	1.24	1.12
	$SM3$ [m]	0.23	0.37	0.49
	TBe	33.4	34.81	28.41
	Smoothness	14.49	4.83	6.42
5	Mean Time [s]	49.5	34.83	20.04
	Av. Speed [m/s]	1.08	1.04	1.56
	Path Length [m]	28.72	30.99	27.47
	$SM2$ [m]	0.99	1.31	1.13
	$SM3$ [m]	0.15	0.34	0.48
	TBe	44.38	42.47	50.11
	Smoothness	19.2	2.75	17.48

bending energy is related with curvature of the trajectory while the *smoothness* is a measure of the variation of such a curvature.

From the results of table II we conclude that the average velocity and mean time with manual and manual-assisted modes are basically constant, though the performance of manual-assisted mode is clearly higher. Both cases seem to be invariant to the maximum speed, due to the fact that drivers adapt vehicle velocity accordingly to their skill, whatever the maximum velocity is used. On the other hand, autonomous navigation seems to be more conservative at low speeds than the other two modes and more aggressive at high speeds, showing a linear relation with the maximum velocity. Regarding with security metrics, the manual-assisted driving has the higher $SM2$, which implies that is more distant to obstacles during the whole experiment. The autonomous mode is the one that shows better $SM3$ due to the fact that has lower collisions. Finally, with respect to trajectory metrics, the autonomous and manual-assisted modes show a complementary behavior. Autonomous mode describes smoother trajectories at low speeds while it performs more abruptly at high speeds. Manual-assisted mode shows, in general, smooth trajectories though for low velocities they are slightly more abrupt than the autonomous mode. In any case, the manual mode, without any kind of assistance, shows the worst performance.

IV. CONCLUSIONS

The paper addresses the problem of AGV under the perspective of Intelligent Transportation Systems (ITS) covering different automated solutions such as line-tracking, manual-assisted driving, remote teleoperation and autonomous navigation.

The main contribution of the paper is to introduce a methodology to evaluate the performance of AGVs with different degrees of autonomy. This methodology establishes a benchmark consisting of running multiple experiments, like Monte Carlo simulations, under the different velocity conditions and measuring the performance based on several metrics, including mean time, mean speed, security metrics and trajectory bending energy and smoothness. In particular, the paper evaluates the benefits of manual-assisted driving compared to traditional manual driving. Moreover, the comparison is extended by including also an autonomous navigation mode based on ND algorithm, obtaining a solution with fewer collisions but less robust since it is based on a local planner that fails to solve specific trapping situations.

It is interesting to remark that in the manual-assisted mode a force feedback is applied to the steering wheel together with speed reduction when an obstacle is being detected. In this mode, drivers may collide with objects if they intended it on purpose, depending on the degree of freedom imposed. On the contrary, in autonomous navigation mode the goal is to reach the destination without any possible collision, therefore the priority is on security performance. The natural limit of our implemented autonomous navigation solution is on the lack of reasoning, global perception and learning which is seriously affecting to the robustness as mentioned therein after.

V. ACKNOWLEDGEMENTS

Authors want to thank to Adolfo Muñoz for his useful feedback and Daniel Tormo for the vehicle automation tasks.

REFERENCES

- [1] [Online]. Available: <http://www.egemin.com>
- [2] [Online]. Available: <http://www.robocoaster.com>
- [3] J. Kaller and D. Hoetzer, "Lane-change assistant for motor vehicles," United States Patent Application 20050155808, 2009.
- [4] B. A. Miller and D. Pitton, "Vehicle blind spot detector," United States Patent 4694295, 1987.
- [5] O. A. Basir, J. P. Bhavnani, F. Karray, and K. Desrochers, "Drowsiness detection system," United States Patent 6822573, 2004.
- [6] M. Sawada and T. Matsumoto, "Vehicle stability control system," United States Patent 7577504, 2009.
- [7] R. Lamberto, E. Stewart, R. Quimby, J. Borelli, A. Geissberger, and D. Palmieri, "Low cost 77 ghz monolithic transmitter for automotive collision avoidance systems," in *IEEE Microwave Millimeter Wave Monolithic Circ Symposium*, 1993, pp. 63–66.
- [8] T. O. Grosch, "Radar sensors for automotive collision warning and avoidance," in *Proceedings of SPIE - The International Society for Optical Engineering*, 1995, pp. 239–247.
- [9] K. Young and M. A. Regan, "Intelligent speed adaptation: A review," in *Proceedings of RS Conference*, 2002, pp. 445–450.
- [10] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.
- [11] B. H. Krogh and C. E. Thorpe, "Integrated path planning and dynamic steering control for autonomous vehicles," in *IEEE Int. Conf. on Robotics and Automation*, 1986, p. 1664–1669.
- [12] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 5, no. 19, pp. 1179–1187, 1989.
- [13] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *IEEE Int. Conf. on Robotics and Automation*, vol. 2, 1991, p. 1398–1404.
- [14] R. B. Tilove, "Local obstacle avoidance for mobile robots based on the method of artificial potentials," in *IEEE Int. Conf. on Robotics and Automation*, 1990, p. 566–571.
- [15] J. Borenstein and Y. Koren, "The vector field histogram - fast obstacle avoidance for mobile robots," *IEEE Trans. Robot. Automation*, vol. 7, no. 3, pp. 278–287, 1991.
- [16] I. Ulrich and J. Borenstein, "Vfh+: Reliable obstacle avoidance for fast mobile robots," *IEEE Int. Conf. Robotics and Automation*, pp. 1572–1577, 1998.
- [17] —, "Vfh*: Local obstacle avoidance with look-ahead verification," in *IEEE Int. Conf. Robotics and Automation*, 2001, pp. 2505–2511.
- [18] P. S. R. Brauningl and J. Ezquerra, "Fuzzy logic wall following of a mobile robot based on the concept of general perception," in *Int. Conf. On Advanced Robotics*, 1995, pp. 367–376.
- [19] A. O. F. Cuesta, *Intelligent Mobile Robot Navigation*. Springer-Verlag, 2005.
- [20] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *IEEE Int. Conf. on Robotics and Automation*, 1993, p. 802–807.
- [21] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," Tech. Rep. IAI-TR-95-13, 1 1995.
- [22] R. Simmons, "The curvature-velocity method for local obstacle avoidance," in *IEEE Int. Conf. on Robotics and Automation*, p. 3375–3382.
- [23] J. Minguez and L. Montano, "Nearness diagram (nd) navigation: Collision avoidance in troublesome scenarios," *IEEE Transactions on Robotics and Automation*, vol. 1, no. 20, p. 45–59, 2004.
- [24] J. Minguez, L. Montano, T. Simeon, and R. Alami, "Global nearness diagram navigation," in *IEEE Int. Conf. Robotics and Automation*, 2001, pp. 33–39.
- [25] J. Minguez, "The obstacle restriction method (orm): Obstacle avoidance in difficult scenarios," in *In IEEE Int. Conf. on Intelligent Robot and Systems*, 2005.
- [26] J. Durham and F. Bullo, "Smooth nearness-diagram navigation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.
- [27] C.-C. Yu, W.-C. Chen, C.-C. Wang, and J.-S. Hu, "Self-tuning nearness diagram," in *Proceedings of the International Conference on Service and Interactive Robotics*.
- [28] M. Mora, V. Suesta, and J. T. L. Armesto, "Factory management and transport automation," in *IEEE Conference on Emerging Technologies and Factory Automation*, vol. 2, 2003, pp. 508–515.
- [29] L. Armesto and J. Tornero, "Autotrans: Management and transport automation in warehouses," in *Industrial Simulation Conference*, 2005, pp. 236–241.
- [30] L. Armesto, M. Mora, and J. Tornero, "Supervisión, teleoperación y navegación de vehículos industriales y su integración en el sistema de gestión," *Revista Iberoamericana de Automática e Informática Industrial*, vol. 2, pp. 55–63, 2005.
- [31] L. Armesto and J. Tornero, "Automation of industrial vehicles: A vision-based line tracking application," in *International Conference on Emerging Technologies and Factory Automation*, 2009, pp. 1–6.
- [32] J. Minguez and F. Montesano, L. Lamiroux, "Metric-based iterative closest point scan matching for sensor displacement estimation," *IEEE Transactions on Robotics and Automation*, 2006.
- [33] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.
- [34] [Online]. Available: <http://playerstage.sourceforge.net/>
- [35] N. Muñoz, J. Valencia, and N. Londoño, "Evaluation of navigation of an autonomous mobile robot," in *Proceedings of Performance Metrics for Intelligent Systems Workshop*, 2007, p. 15–21.

Automatic Routing System for Intelligent Warehouses

K. T. Vivaldini; J. P. M. Galdames *Member, IEEE*; T. B. Pasqual, R. M. Sobral; R. C. Araújo, M. Becker *Member, IEEE*; and G. A. P. Caurin, *Member, IEEE*

Abstract— Automation of logistic processes is essential to improve productivity and reduce costs. In this context, intelligent warehouses are becoming a key to logistic systems thanks to their ability of optimizing transportation tasks and, consequently, reducing costs. This paper initially presents briefly routing systems applied on intelligent warehouses. Then, we present the approach used to develop our router system. This router system is able to solve traffic jams and collisions, generate conflict-free and optimized paths before sending the final paths to the robotic forklifts. It also verifies the progress of all tasks. When a problem occurs, the router system can change the tasks priorities, routes, etc. in order to avoid new conflicts. In the routing simulations each vehicle executes its tasks starting from a predefined initial pose, moving to the desired position. Our algorithm is based on Dijkstra's shortest-path and the time window approaches and it was implemented in C language. Computer simulation tests were used to validate the algorithm efficiency under different working conditions. Several simulations were carried out using the Player/Stage Simulator to test the algorithms. Thanks to the simulations, we could solve many faults and refine the algorithms before embedding them in real robots.

I. INTRODUCTION

THE routing task may be understood as the process of simultaneously selecting appropriate paths for the AGVs (Automated Vehicle Guided) among different solutions. One of the goals of routing for AGVs is the minimization of cargo cost. In recent years, several algorithms, distinguished in two categories: static [1]-[7] and dynamic [8]-[17], have been proposed to solve routing problems. In the first case, static routing the route from node i to node j is determined in advance and is always used if a load has to be transported from i to j . Thus, a simple assumption is to choose the route with the shortest distance from i to j . However, these static algorithms can not adapt to changes in the system and traffic conditions. In dynamic routing, the information necessary to determine efficient routes are dynamically revealed to the decision-maker and, as a result, various routes between i and j can be chosen [18]. As one can notice, the several papers found in the literature, have not take into account important features in the routing, such as path maneuvers, time curves, load and unload operations, etc.

Manuscript received February 28, 2010. This work was supported in part by FAPESP (Processes 2008/10477-0, 2008/09755-6, and 2009/01542-6), CAPES (Process 33002045011P8) and CNPq (Process 134214/2009-9).

K. T. Vivaldini; J. P. M. Galdames; T. B. Pasqual, R. M. Sobral; R.C. Araújo, M. Becker; and G. A. P. Caurin are with the Mechatronics Laboratory - Mechanical Engineering Department, at EESC-USP, SP 13566-900 Brazil (e-mails: {kteixeira}{galdames}{becker}{gcaurin}@sc.usp.br; and {thales.bueno}{rafael.sobral}{roberto.carlos.araujo}@usp.br).

In many researches, the route is calculated considering the minimum path. Broadbent et al. [1] presented the first concept of conflict-free and shortest-time AGV routing. The routing procedure described employs Dijkstra's shortest path algorithm to generate a matrix, which describes the path occupation time of vehicles.

Kim and Tanchoco [9] proposed an algorithm based on Dijkstra's shortest-path method and the concept of time windows graph for dynamic routing of AGVs in a bidirectional path network. They presented formulas to consider the time curves, but this was not considered in their results.

Maza and Castagna [12-13] added a layer of real time control to the method proposed by Kim and Tanchoco. In [12], they proposed a robust predictive method of routing without conflicts, and in [13] they developed two algorithms to control the AGV system using a predictive method when the system is subject to risks and contingences avoiding conflicts in real time manner. In both, the curve time is not mentioned.

Möhring et al. [14] extended the approaches of Huang, Palekar and Kapoor [18] and Kim and Tanchoco [9], and presented an algorithm for the problem of routing AGVs without conflicts at the time of route computation. In the preprocessing step the real-time computation for each request consists in the determination of the shortest path with time-windows and a following readjustment of these time-windows, both is done in polynomial time. By goal-oriented search, they computed times appropriate for real-time routing. Extending this concept, Klimm et al. [14] presented an efficient algorithm to cope with the problem of congestion and detours that also avoids potential deadlock situations. The authors in [14-15] did not consider that the time window for routes with curves could imply the estimated time. Only in [16], Gawrilow et al. presented an algorithm to avoid collisions, deadlocks and livelocks already at the route computation time, considering the physical dimensions of the AGVs to cope with complex maneuvers and certain safety aspects that imply particular applications.

Ravizza [17] focused on the online control of AGVs, presenting a heuristic approach for task assignment and a dynamic polynomial-time sequential routing to guarantee deadlock and conflict-free routing of the AGVs in undisturbed operations. The articles cited above used Dijkstra's Algorithm to calculate the route.

However, the shortest route is not always the most efficient method, ie. the number of maneuvers is larger than

the number necessary for the path to be executed.

In this context, as a solution to this problem, this paper presents the development of a routing system that computes optimal routes, reducing the amount of unnecessary maneuvers, allowing path planning and coordinating the task execution of the transport and handling in structured environments. It also avoids known and dynamic obstacles located in the environment. This approach investigates an efficient solution to routing AGVs, as an alternative to the various methods developed previously in the literature. We also propose a software architecture that considers the local and global tasks of the robotic forklifts (e.g.: local navigation, obstacle avoidance, and auto-localization). Figure 1 presents this architecture.

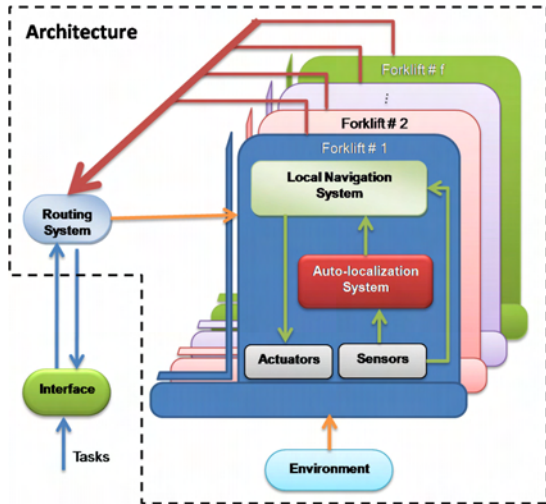


Fig. 1. Overall System Architecture proposed.[25]

In summary, it can be explained as follows: the routing system receives information about the required transportation tasks from a user interface. Based on these data, it selects the minimum quantity of robotic forklifts necessary to execute the tasks. Then, based on a topologic map of the environment, it calculates the routes for the selected forklifts, checking possible collisions, traffic jams, etc. After that, it sends the routes to each robotic forklift and regularly verifies the progress of all tasks. Taking into account the global route, each forklift calculates its own local path necessary to reach its goal and monitors its surroundings looking for mobile or unexpected obstacles during the execution of the planned path. At a certain frequency, based on sensor data and an environment map, each robotic forklift auto-localization subsystem updates its estimated position and informs these data to the local navigation subsystems. The local navigation subsystem compares the current position with the desired one. If the robot deviates from the route, the local navigation subsystem sends commands to correct its pose returning the planned route. If the local navigation subsystems verify that the route has exceeded the limit runtime determined by the routing, it communicates it to the routing system, which recalculates the route avoiding collisions or deadlocks among other robotic forklifts. The algorithm is based on Dijkstra's

shortest-path method [20] to calculate the routes of the robotic forklift adding heuristic functions to optimize the quantity of maneuvers, and using the method of routing with time-windows [21] to ensure conflict-free routes.

II. ROUTING ALGORITHM

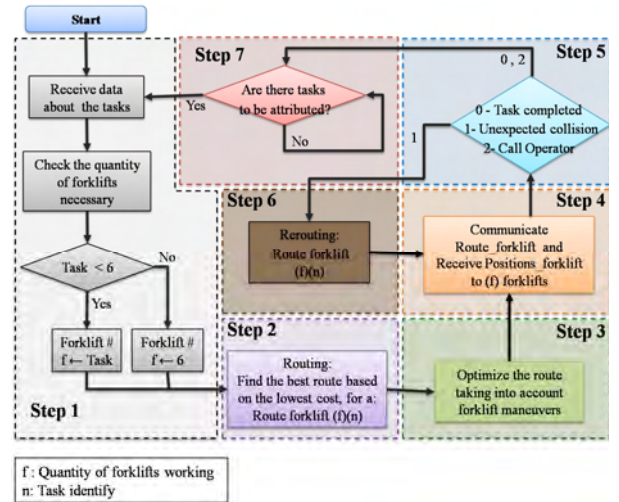


Fig. 2. Proposed Algorithm.

The routing algorithm (Fig. 2) was based on the dynamic programming approach, which consists in dividing the original problem into smaller and simpler problems. This approach was very useful to our problem, as it presents a sequence of decisions to be taken along a time sequence [22]. Therefore, our routing system could be divided into seven steps.

A. Step 1

The routing system receives a list of the requests as input (Table 1 presents a summary of the requested data considered in the storage activities). Each request is a task defined by a sequence of pairs: loading stations (origin node) and unloading stations (destination node). Then, according to the requests, the routing system checks the quantity of robotic forklifts necessary to execute the tasks, and assigns each task to a forklift.

TABLE I
REQUEST DATA, INPUTS AND OUTPUTS OF OUR ALGORITHM

Data	Input	Output
<i>Request (Orders)</i>	Quantities, loading data / unloading data of each product	Necessary number of forklifts and allocation of each request in a route for the forklifts
<i>Loading</i>	Location point for loading the pallets	Routes that the forklifts should execute to carry the pallets
<i>Unloading</i>	Location point for delivering the pallets	Routes that the forklifts should execute to unload the pallets

Several tasks can be attributed to various robotic forklifts, but one task cannot be attributed to several robotic forklifts. When all tasks have been designated and the quantity of

robotic forklifts necessary has been determined, these data are sent to Step 2 in order to calculate the routes.

B. Step 2

In this Step the routing system applies a graph-based approach. The graph is obtained by using a topological map of the warehouse environment. The route necessary to execute each task is composed of two sub-routes. Which their one has its own origin and destination nodes. Then Dijkstra's Algorithm is applied to calculate the lowest path (relation between distance and total cost) for each robotic forklift. The path is a continuous sequence of positions and orientations of the robotic forklift (e.g.: the intermediate positions and the pre-established positions and orientations present in Fig. 3). Aiming to guarantee that collisions between robotic forklifts will not occur, the forklift cannot use the same arc or node at the same time. Therefore, we know when every arc is (or not) occupied by a robotic forklift for each calculated sub-route. These data (time intervals $[a_i, b_i]$) and the time window are saved on a list (log file), allowing verifying possible conflicts between paths of the robotic forklifts. When the sub-route is a task, a value representing the time needed to load or unload the pallets is added to the destination node time interval.

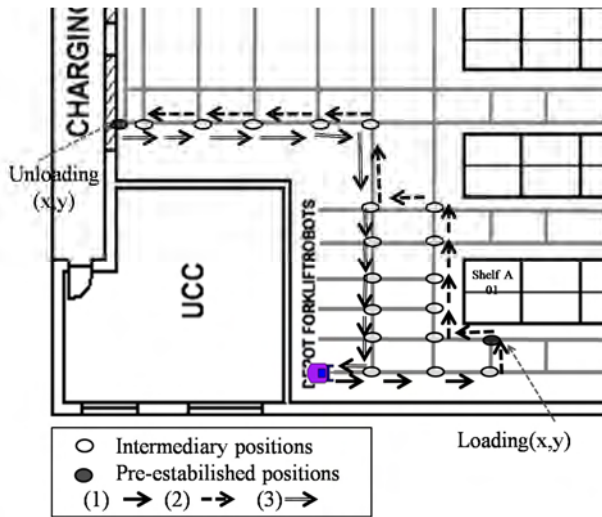


Fig. 3. Illustration of route definition. A robotic forklift will execute a task (go to Shelf A-01, load a pallet and unload it at Charging Platform F). After this task, it will return to its depot. The first (1) and the second (2) sub-routes represent the task execution. The third one (3) represents the robotic forklift returning to its depot.

Figure 3 illustrates how algorithm generally defines a route. The origin of the first sub-route is the current robotic forklift position. Its destination is defined as the task initial position (loading node). The second sub-route origin is the loading node and its destination is the unloading node. If the same robotic forklift has another task, the following sub-routes are defined in a similar way. If it is closing-time, or the robotic forklift needs to recharge its batteries, after the last task its final sub-route drives it directly to its depot. Thus, we always have the origin and destination nodes and also know if a task is (or not) in execution.

C. Step 3

In this step the quantity of maneuvers (e.g.: curves) during the path route is analyzed by heuristic functions, which verify the possibility of optimizing them. If the maneuvers are unnecessary, the route is optimized using again the Dijkstra's Algorithm with time-window, taking into account the costs previously established. If the resulting total execution time of the task is longer than the previously calculated (Step 2), the final route is not changed (it means that the route cannot be optimized). At the end of this step, if the route has been optimized, it will be conflict-free.

D. Step 4

In this step, the routes are sent to each robotic forklift and the routing system interacts with the other systems embedded in the robots (navigation and auto-localization systems), as shown in Fig. 4.

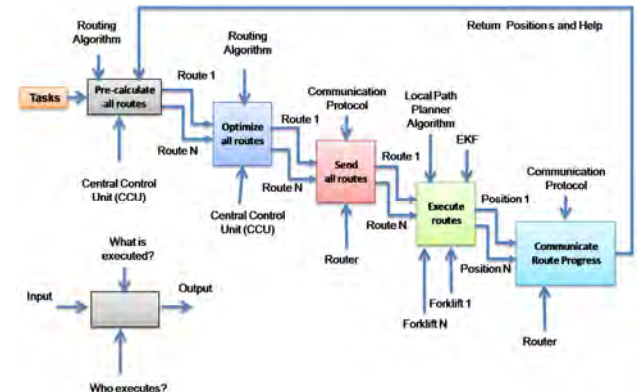


Fig. 4. Interaction of the routing system with navigation, control, and localizations systems.

The routing system verifies each task progress regularly. Each robotic forklift has its own sensor, auto-localization, and local navigation sub-system running independently and informing regularly the global path planner about the status of each task (position and problem found or task finished). Therefore, it is possible to minimize the impact of this local problem in the global system that controls the intelligent warehouse. Table 2 presents a summary of this procedure.

TABLE II
REQUEST DATA, INPUTS AND OUTPUTS OF OUR ALGORITHM

Data	Input	Output
<i>Problems in route execution</i>	Route cannot be concluded.	Inform position and list of the problems found (unexpected collisions, exceeded time) or call operator.

E. Step 5

In this step the algorithm checks the status of the tasks, where: 0 - task finished, 1 - rerouting of the task (unexpected collisions or exceed time), and 2 - call operator and assign tasks of this forklift to others.

If the status is equal 1 to, it is necessary to recalculate the sub-route (Here, the use of sub-routes is very useful as, only

the sub-route is corrupted). Basically the algorithm verifies which robotic forklift presents conflicts and its location (arc or node – see Fig. 5). The reroute process is carried out in Step 6.

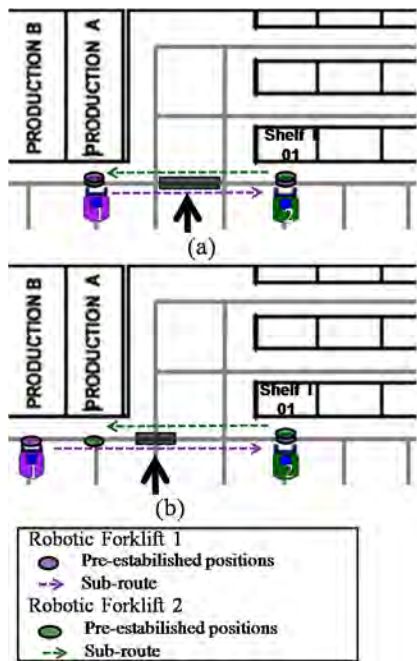


Fig. 5. Illustration of two examples of collision between the robotic forklifts #1 and #2. In (a) a collision in the arc, in (b) in the node.

F. Step 6

Now, the corrupted sub-route is eliminated and the arc where the conflict occurred is blocked. Then, the sub-routes of the robotic forklifts that caused the conflict are recalculated (Fig. 6).

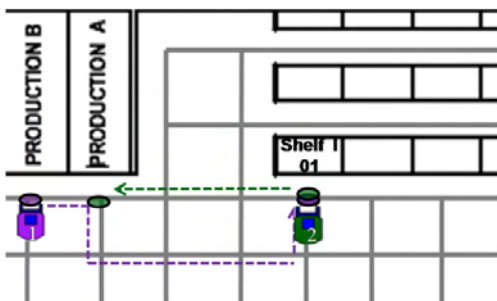


Fig. 6. Result of the rerouting process for the example presented in Fig. 5-b.

To recalculate this sub-route again, the algorithm returns to steps 2 and 3, the origin node receive the position where forklift is and the destination node is the same for this sub-task. These new sub-routes are called conflict-free sub-routes. If after these conflict-free sub-routes there are other sub-routes in the forklift schedule, then it is necessary to verify the conflict-free final time in the destination node. If it differs from the previously calculated (Step 2), then it is necessary to readjust the time windows of this forklift checking if it may cause other conflicts. For instance, let us assume that the robotic forklift #1 in Fig. 5 has more sub-routes scheduled, after its conflict-free sub-route (Fig. 7).

Therefore, it is necessary to readjust the time windows of the subsequent sub-routes by checking the list of available time windows. Then, the algorithm verifies again the presence of conflicts in the route, due to the readjustment of the time windows. If there is any conflict, it returns to the beginning of this step.

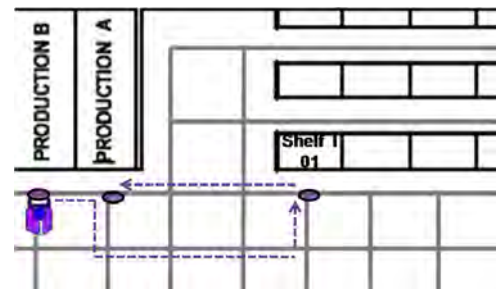


Fig. 7. Sub-routes of the robotic forklift #1.

G. Step 7

Finally, the algorithm verifies if there are more tasks to be executed. If so, it returns to its beginning (step 1). The objective of this step is to verify and validate the execution time of the tasks attributed versus position of the robotic forklifts.

III. INTERFACE

In this work, a graphic interface allows the routing operation to be controlled by an operator, through Qt Creator 4 software [23] and a simulated virtual environment in the Player/Stage developed in previous works [24]. Both were implemented in Linux Operating System.

The graphic interface (Fig. 8) was developed to perform the communication between the user and the routing system (including all system algorithms). It also allowed the configuration of both routing system and simulation environment at a higher level (Fig. 9), guaranteeing the functionality of the routing operation and returning information concerning the process performance in a simple and visual way.

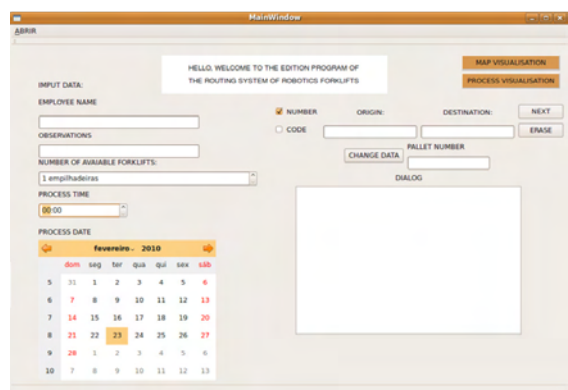


Fig. 8. Graphic Interface

The interface allows the operator to schedule processes of robotic forklifts, setting time, data, quantity of pallets involved, as well as origin and destination, and the quantity of mobilized forklifts. With such information, the operator

can simulate the process. In order to do this, it is possible to configure the simulation environment in the most appropriate and realistic way, providing drivers for the available sensors and the bi-dimensional plan of the modeled warehouse. Thus, the operator can simulate and evaluate with a good accuracy the time and effectiveness of the process without the mobilization of real robotic forklifts.



Fig. 9. Graphic Interface for the simulated virtual environment.

IV. RESULTS

As previously mentioned, this work used a local navigation and auto-localization systems in order to obtain more realistic simulations. The routing system uses the A* algorithm in the local path planning, and applies the Extended Kalman Filter (EKF) algorithm for the auto-localization for each robotic forklift [24][25]. The proposed algorithm is based on Dijkstra's shortest-path method to calculate the routes of the robotic forklift adding heuristic functions to optimize the quantity of maneuvers, and using the method of routing with time-windows to ensure conflict-free routes. Using heuristic functions, it was possible to verify the number of unnecessary curves and make a route optimization reducing the amount of maneuvers to be performed and thus increasing the mobility of the robotic forklift. It is important to emphasize that the optimized route do not exceeds the total cost previously established.

In order to verify the performance of our algorithms, several simulations were carried out using a virtual warehouse (50x30m) and the Player/Stage Simulator. In the simulation 6 robotic forklifts working at the same time in this warehouse were considered (at this moment we did not consider here unknown obstacles like people walking or other vehicles moving in the warehouse). In order to simplify the implementation, we selected in the Player/Stage library the Pioneer mobile robot to represent our robotic forklifts. Maximum cruiser speeds of 1m/s and 5°/s were applied to all robots. The algorithm could solve the conflicts and return the optimized routes. Figure 10 shows the simulations of the tasks attributed for 6 robotic forklifts. This test would have several collisions that can be observed in Fig. 10-a (arrow). Firstly, the deadlock between the robotic forklifts #4 (red) and #5 (yellow) is detected. Due to this deadlock, a new deadlock also involving the robotic forklift #6 (lemon) occurs. The collisions happened because

the robotic forklift #4 needs to occupy the same arc at the same time that robotic forklifts #5 and #6. Unfortunately, as we highlighted before, if the Router did not verify all sub-routes after the first corrupted one detected, more conflicts may occur. In this case, a new deadlock between the robotic forklifts #6 (lemon) and #1 (purple). Figure 10-b presents the proposed algorithm. One may verify that the conflicts identified in Fig. 10-a were eliminated, the final paths provided by the routing algorithm were collision-free and avoided traffic jams. It also shows in detail the optimizations performed by the Routing System.

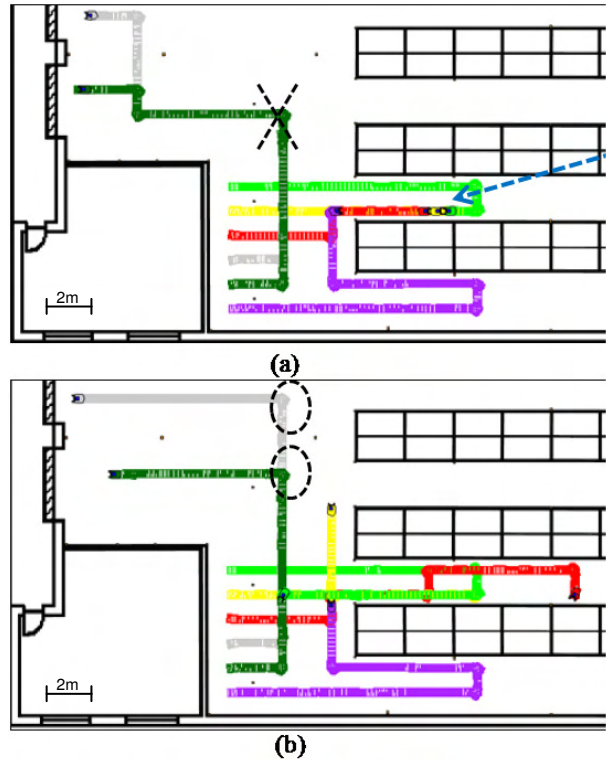


Fig. 10. Comparison between shortest path method and optimized routes of the robotic forklift #3 (grey) and forklift #2 (green). In (a) the route marked with an X represents the maneuver that the robotic forklifts perform using shortest path method; and in (b) the routes marked with circles represent the maneuvers that the robotic forklifts did using the route optimized by our algorithm.

In this scenario, each robotic forklift calculates the trajectory to be executed considering the route sent by the routing system (Fig. 11). Each forklift receives a safety time for the minimum path between two nodes, then navigation

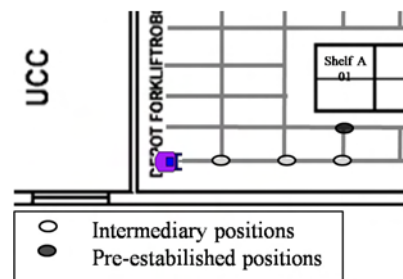


Fig. 11. Illustration of route informed for local planner. The routing system sends the each two points (e. g. forklift position and Intermediary position) the safety time that the local planner has to calculate the path planning.

system can fix both the speed and trajectory to execute the path. The local planner applies time windows to calculate the safe trajectory. In case of disturbances that divert the robot from its planned trajectory, the local planner tries to run the route within the stipulated time, e.g., increasing the speed. If the time is exceeded, the robot sends an error log to the routing system describing the problem found during the route.

V. CONCLUSIONS

Firstly, we described the most important works found in literature that focus on routing systems for AGVs. Next, we presented the approach used to develop our Routing system. It is able to solve traffic jams and collisions. It also guarantees optimized routes before sending the final paths to the robotic forklifts. It also verifies the progress of all tasks. When a problem occurs, the routing system can change the tasks priorities, routes, etc. to avoid new conflicts. In order to verify the algorithm's performance, the robotic forklifts were tested executing tasks that simulate the load and unload of goods using the interface and the virtual environment in the Player/Stage software. The use of the interface was very interesting because it allowed us to control the process simulation at a higher level. Therefore, this program can be used directly in industries at future.

The A* algorithm was a simple way to control the robot maneuvers, and the EKF was sufficient to avoid the position error propagation. Both algorithms were implemented in C. As a result, we could solve many faults and refine the algorithms before embedding them in real robots. We also plan some improvements on the algorithm, for instance, to refine the routing task planner and to take into account more realistic parameters during the simulations (e.g.: velocity changes while maneuvering, load and unload times, etc.). It is important to emphasize that in the current version, the algorithm assigns the time for each path according to the distance, maneuvers and loading and unloading of pallets. The robotic forklift speed during the path is determined by the local planner, which checks the distance to achieve the goal and determines the speed required for the robotic forklift to arrive there in time. One limitation of the algorithm is that collisions are always solved by finding a new route. A possible solution would be the application of stoppages to one of the forklifts during a certain time interval, or the reduction of its speed. Both options will be investigated in future. At this moment our group is finishing the design and construction of mini-mobile robots and a 1:5 scale warehouse to test experimentally our algorithms.

REFERENCES

- [1] A. J. Broadbent, et al. "Free-ranging AGV and scheduling system," *Automated Guided Vehicle Systems*, 1987, pp.301–309.
- [2] L. D. Bodin, et al. "Routing and scheduling of vehicles and crews: The state of the art," *Computers and Operations Research*, vol. 10, no. 2, pp. 63–211, 1983.
- [3] Laporte, G., "The vehicle routing problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, pp.345–358, 1992.
- [4] N. Krishnamurthy, R. Batta, M. Karwan, "Developing conflict-free routes for automated guided vehicles," *Operations Research* vol. 41, pp. 1077–1090, 1993.
- [5] F. Taghaboni-Dutta, J. M. A. Tanchoco, "Comparison of dynamic routing techniques for automated guided vehicle system," *International Journal of Production Research* vol. 33, no. 10, pp. 2653–2669, 1995.
- [6] Fisher, M. L., Jörnsten, K. O. and Madsen, O. B. G. (1997), Vehicle Routing with Time Windows: Two Optimization Algorithms, *Operations Research*, Res, 45, 487–491.
- [7] J. Z. Nanyang, W. Hsu, "Conflict-free container routing in mesh yard layouts", *Robotics and Autonomous Systems*, vol. 56, no. 5, pp. 451–460, May, 2008.
- [8] H. N. Psaraftis, "Dynamic vehicle routing problems", *Vehicle Routing: Methods and Studies*. BL. Golden, AA. Assad, editors. Vehicle routing: methods and studies. Amsterdam: North-Holland., pages 223–48, 1988.
- [9] Ch.W. Kim and M. A. J. Tanchoco, "Conflict-free shortest-time bidirectional AGV routing", *International Journal of Production Research*, vol.29, n. 12, pp.2377–2391, 1991.
- [10] M. Gendreau, F. Guertin, J.Y. Potvin, E. Taillard, "Parallel tabu search for real-time vehicle routing and dispatching," *Transportation Science*, vol. 33, no. 4, pp. 381–390, 1999.
- [11] L. Qiu, W.J. Hsu, "A bi-directional path layout for conflict-free routing of AGVs," *International Journal of Production Research*, vol. 9, no. 1, pp. 2177–2195, 2001.
- [12] S. Maza, P. Castagna, "Conflict-free AGV routing in bi-directional network", in: *Proc. 8th IEEE International Conf. on Emerging Technologies and Factory Automation*, 2001.
- [13] S. Maza, P. Castagna, "Robust conflict-free routing of bi-directional automated guided vehicles", in: *Proc. 9th IEEE International Conf. on Emerging Technologies and Factory Automation*, 2002.
- [14] R. H. Möhring, E. Köhler, E. Gawrilow and B. Stenzel, "Conflict-free Real-time AGV Routing", *Operations Research Proceedings*, vol. 2004, Springer Berlin, 2004.
- [15] M. Klimm, E. Gawrilow, R. H. Möhring and B. Stenzel, (Nov. 2008, 19) "Conflict-free vehicle routing: load balancing and deadlock prevention," Technical Report, 2007. Available: <http://www.matheon.de/preprints/5137_preprint-static-routing.pdf>.
- [16] E. Gawrilow, E. Köhler, R. H. Möhring and B. Stenzel, "Dynamic Routing of Automated Guided Vehicles in Real-time", Technische Universität Berlin, Book: *Mathematics - Key Technology for the future*, pp. 165-177, 2008.
- [17] S. Ravizza. *Control of Automated Guided Vehicles (AGVs)*, diploma thesis, ETH Zurich, 2009.
- [18] I. F. A. VIS, "Survey of research in the design and control of automated guided vehicle systems" in *European Journal of Operational Research*, vol. 170, Elsevier Science, Amsterdam , 2006, pp. 677-709.
- [19] J. Huang, U. S. Palekar, S. Kapoor, "A labeling algorithm for the navigation of automated guided vehicles," *Journal of engineering for industry*, vol. 115, pp. 315–321, 1993.
- [20] E. W. Dijkstra, "A Note on Two Problems in Conjunction with Graphs" *Numerische Math*, 1959, vol. 1, pp. 269-271.
- [21] J. Desrosiers, F. Soumis, M. Desrochers and M. Sauvé. "Methods for routing with time windows," *European Journal of Operational Research*, vol. 23, p. 236-245, 1986.
- [22] P. Serafini, "Dynamic programming and minimum risk paths", *European Journal of Operational Research*, vol. 175, no. 1, pp. 224-237, Nov. 2006.
- [23] Nokia Corporation, (2010, Feb. 10). "Qt creator Whitepaper", 2009 [Online]. Available: <http://qt.nokia.com/files/pdf/qt-creator-1.3-whitepaper>
- [24] K.T. Vivaldini, J. P. M. Galdames, T. B. Pasqual, R. C. Araújo, R. M. Sobral, M. Becker, and G. A. P. Caurin " Robotic Forklifts for Intelligent Warehouses: Routing, Path Planning, and Auto-localization", in *IEEE – International Conference on Industrial Technology*, Viña del Mar – Valparaíso, Chile, Mar. 2010.
- [25] K.T. Vivaldini, J. P. M. Galdames, T. B. Pasqual, M. Becker, and G. A. P. Caurin, "Intelligent Warehouses: focus on the automatic routing and path planning of robotic forklifts able to work autonomously," *Mechatronics Systems: Intelligent Transportation Vehicles*, 2010, submitted for publication.

Coordinating the motion of multiple AGVs in automatic warehouses

Roberto Olmi, Cristian Secchi and Cesare Fantuzzi

Abstract—In this paper an algorithm for planning a coordinated motion of a fleet of Autonomous Guided Vehicles (AGVs) delivering goods in an automatic warehouse is proposed. The AGVs travel along a common segmented layout and a path is assigned to each robot by a mission planner. Coordination diagrams are used for representing possible collisions among the robots and a novel algorithm for efficiently determining a coordinated motion of the fleet is proposed. The coordination approach proposed in the paper is validated through experiments on real plants layouts. We present an example in which the coordinated motion of 10 vehicles is computed in only 12.4 sec. on a common PC.

I. INTRODUCTION

Automated guided vehicles (AGVs) are used more and more in industrial plants and warehouses for transporting goods. In these applications, a central issue is how to plan the motion of the AGVs in order to minimize the delivery time while avoiding collisions and deadlocks. The work presented in this paper is made in cooperation with a company producing AGVs for transporting goods in warehouses. The robots are controlled through a centralized architecture. Given a warehouse, a roadmap along which the AGVs can move is designed. The central system assigns to each AGV a mission, namely a path on the roadmap, that the robot has to track. Our goal is to provide a motion coordination algorithm which is efficient and that avoids collisions and deadlocks between robots.

The problem of coordinating the motion of a fleet of robots with assigned paths has already been tackled in several works. In [1] a mathematical programming formulation in which the task completion time is taken as objective function and where dynamic constraints are considered is given. In many applications, however, considering the average time-to-goal as objective function would be more appropriate since this allows minimize the number of vehicles required to serve a given plant ([2]). Some other strategies apply the so called *coordination diagram* (CD) [3] in order to map the coordination problem into a path planning problem. Given a fleet of robots to be coordinated along assigned paths, the CD is a representation of all configurations of the fleet where mutual collisions might occur. A path through the CD, indicated as *coordination path*, defines a coordinated motion for the robots. In [4] we have presented an incremental coordination algorithm based on the CD. The algorithm determines the coordination action step by step without computing any

preplanning for the coordination. This approach is indicated when a lot of unexpected events could prevent some AGVs from performing the preplanned action. Other approaches try to plan a coordination path by means of standard search algorithms. For an introduction to this topic see [5]. In [6] the cylindrical structure of the CD is exploited in order to give an implicit cell decomposition of the diagram. Then the A* algorithm is applied to find a coordination path. A distributed approach is presented in [7]. In this case the CD is partitioned into a regular grid over which the D* algorithm searches for a coordination path. However, by minimizing the length of the coordination path, the obtained solution does not minimize the average time-to-goal. An algorithm that minimizes the average time-to-goal is presented in [8] where the optimal coordination path is found by applying the dynamic programming principles.

These techniques however require the CD to be partitioned into a grid in which the search algorithm is used in order to find a coordination path. At each iteration step the number of grid elements explored by the algorithm can be exponential in the number of vehicles. To avoid this problem the algorithm proposed in [6] explores only along Manhattan paths, but then a smoothing technique is applied over the output path. Moreover the partition of the entire CD requires an enormous number of cells. As far as the problem instance is simple (i.e. the obstacle within the CD are small and sparsely distributed), the algorithm explores a small subset of cells. Differently the computation time will grow very quickly.

In this paper we propose a new search strategy that explores the CD by incrementally expanding a set of coordination paths. Starting from a null length path, at each iteration step the algorithm selects a path to be extended. This operation generates new paths that are identical to the extended one except for the last added segment. The exploration continues until an optimal path, among all the possible paths that can be explored by the algorithm, is found.

The main feature of our algorithm is that it drastically reduces the directions to be evaluated by explicitly considering the cylindrical structure of the CD. The set of directions used to extend each path is computed by considering two things: the direction of the last segment of the path and the position of the last point of the path compared to the obstacles. The directions will be generated by taking the last direction of the path and modifying only the components that are relevant in order to avoid the obstacles. The second contribution of the paper is the definition of an heuristic estimate of the cost function which takes into account the number of times that a vehicle has to stop and start its motion during the

Authors are with the Department of Sciences and Methods of Engineering, University of Modena and Reggio Emilia, via G. Amendola 2, Morselli building, Reggio Emilia I-42122 Italy

E-mail: {davide.ronzoni, roberto.olmi, cristian.secchi, cesare.fantuzzi}@unimore.it

coordinated motion. This brings two benefits. The first one is that we define a trade-off between time optimality and smoothness of the motion of each vehicle. The second, but more important, benefit is that the number of explored states is further reduced thanks to the fact that the algorithm tends to extend only the smoother paths.

The procedure used for the expansion of a path is similar to that reported in [9] where an exact algorithm for computing Pareto optima paths is given. The input for this algorithm is a collision-free path; the output is the Pareto optimal path homotopic to the input. However the output path is only guaranteed to be a local minimum of the cost function considered. The proposed algorithm does not focus on a particular homotopy class but it searches among all possible homotopy classes.

In order to prove the efficiency of the proposed algorithm an example is shown in which 10 vehicles are coordinated on a roadmap designed to serve a real industrial plant. The coordination has been computed in only 12.4 sec. on a common PC.

The paper is organized as follows: in Sec. II a formal definition of the problem is reported. In Sec. III we present the search algorithm. In Sec. IV the heuristic cost function is defined. In Sec. V an experiment on a real plant is presented. Finally, in Sec. VI some conclusions are drawn and some future work is addressed.

II. OVERVIEW OF THE PROBLEM

A. Coordination Diagram

In the application that we are considering, a centralized planning system plans a set of missions to be executed by the AGVs. We consider N AGVs that are moving in the same environment and that share the same configuration space \mathcal{C} (e.g. $SE(2)$). The path of each vehicle \mathcal{A}_i is computed without considering the presence of other robots and it can be represented as a continuous mapping $\pi_i : s_i \rightarrow \mathcal{C}$. The scalar parameter $s_i \in [0, T_i]$ is the time that the vehicle \mathcal{A}_i would take to reach the position $\pi_i(s_i)$ considering that it travels with a given nominal velocity profile $v_i(s_i)$.

Our coordination strategy is based on the concept of *coordination diagram* (CD). Given N paths $\pi_i(\cdot)$ the CD is given by $\mathcal{S} = [0, T_1] \times \dots \times [0, T_N]$. A point $\mathbf{s} = (s_1, \dots, s_N)$ within the CD represents a possible configuration of the robots along their paths. We denote as $\mathbf{s}_I = (0, \dots, 0) \in \mathcal{S}$ the initial configuration of the fleet and with $\mathbf{s}_G = (T_1, \dots, T_N)$ the goal configuration. For each pair of paths, a *collision region* is defined as:

$$X_{ij}^{coll} = \{(s_1, \dots, s_N) \in \mathcal{S} \mid \mathcal{A}(\pi_i(s_i)) \cap \mathcal{A}(\pi_j(s_j)) \neq \emptyset\} \quad (1)$$

This region defines all the possible configurations of the fleet such that two vehicles collide moving along paths π_i and π_j . Since the collision region depends only on the configuration of two robots, this region can be completely characterized by its 2D projection onto the (s_i, s_j) plane of the CD (that we will denote shortly with CD_{ij}).

A *coordination path* is a continuous map $\gamma : t \rightarrow \mathcal{S}$ where

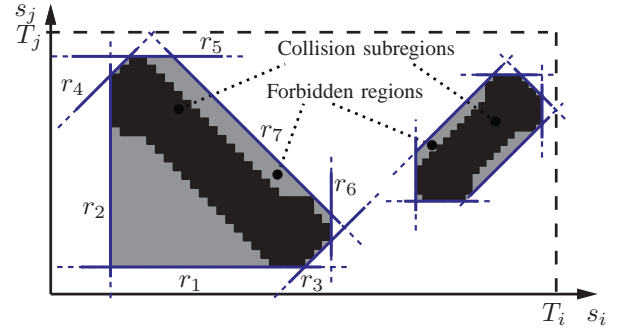


Fig. 1. Two collision subregions and the relative forbidden regions.

$t \in [0, t_{end}]$ is the time, that defines a coordinated motion of the robots along their predefined paths. A path that avoids all collision regions within the CD is said to be *collision-free*. A possible solution to the coordination problem is to find a collision-free path with end points $\gamma(0) = \mathbf{s}_I$ and $\gamma(t_{end}) = \mathbf{s}_G$.

For many industrial applications (e.g. AGVs systems) the paths of the robots are constrained to a specified roadmap. In this case a technique for efficiently computing the CD is presented in [4]. This technique gives a grid representation of each plane of the CD.

The motion of the vehicles is computed by searching for a path within the CD which avoids all the collisions among vehicles. Since the collisions depend only on the configurations of the pairs of vehicles, such a path can be found by considering just all the planes CD_{ij} of the CD.

Note that a collision region X_{ij}^{coll} can be composed of many disjoint subregions. The proposed algorithm exploits an explicit representation of the CD that is obtained by approximating each subregion with a convex polygonal region as shown in Fig. 1. The polygon corresponds to the region enclosed by a set of lines: two horizontal (r_1 and r_5), two vertical (r_2 and r_6), two parallel to the bisector of the plane (r_3 and r_4) and one orthogonal to the bisector of the plane (r_7); all tangent to the obstacle and non coincident with each other. We refer to this polygon as *forbidden region*. This approximation is justified by the observation that frequently, in AGV application, the collision regions have a strip shape ([4]). Thanks to this definition a coordination path that avoids all the forbidden regions is also collision-free. The algorithm that we propose finds a collision-free path by efficiently exploiting the explicit representation of the forbidden regions.

B. Problem formulation

The *collision-free coordination problem* requires to define a coordination path $\gamma : [0, t_{end}] \mapsto \mathcal{S}$ whose components $s_i(t)$ define a motion plan for each vehicle \mathcal{A}_i such that all the paths are completed (i.e. $\gamma(t_{end}) = \mathbf{s}_G, t_{end} < \infty$) without mutual collisions. Our goal is to solve the coordination problem by seeking to minimize the sum of the mission time of all vehicles. We consider that each vehicle can not backtrack along its path and that, at each position s_i ,

the robots are able to switch instantaneously between their nominal speed $v_i(s_i)$ and halting. Therefore the motion of a vehicle can be described using only a binary variable and the coordination paths are piecewise linear curves. The direction of each linear segment of $\gamma(t)$ can be defined by a vector $\mathbf{u} = (u_1, \dots, u_N)$ where $u_i \in \{0, 1\}$ for $i = 1, \dots, N$. Each variation of the direction \mathbf{u} of $\gamma(t)$ corresponds to an instantaneous change of the motion of some vehicles. We denote with n_{acc}^i the number of times that the component u_i switches its value along the path $\gamma(t)$. In other words, n_{acc}^i corresponds to the number of times that the vehicle \mathcal{A}_i changes its speed while executing the coordinated motion $\gamma(t)$. Given a path $\gamma(t)$ with end points at $\mathbf{s}_I = (0, \dots, 0)$ and $\mathbf{s}_G = (T_1, \dots, T_N)$, we denote as t_i^* the time in which the vehicle \mathcal{A}_i reaches its destination (i.e. $s_i(t_i^*) = T_i$). We propose this cost function:

$$C(\gamma) = \sum_{i=1}^N (t_i^* + n_{acc}^i \cdot K_{acc}) \quad (2)$$

where K_{acc} is a parameter that is used to penalize the paths that require many vehicles to be stopped during the execution of their path. Since the measure of t_i^* is computed without considering the vehicle dynamics, K_{acc} can be used to approximately take into account the delay accumulated by the vehicles each time that a component of \mathbf{u} switches its value (considering the same delay for a start or a stoppage).

III. COORDINATION PATH SEARCH

In this section we give an heuristic algorithm for the computation of a near-optimal solution to the coordination problem. The approach consist of building a set of piecewise linear coordination paths by iterative expansions. The algorithm terminates when the best path of the set is better than all those that can be generated. At each iteration step a path is selected from the set and one or more paths are generated by concatenating it with a set of linear segments (for each segment a new path is generated). These segments will be referred as *actions* since they represent a coordinated motion of the fleet from a configuration to another. This process induces a hierarchical structure over the set of paths such that each path can be defined as an extension of another one.

The extension of a path is based on the explicit representation of the CD obtained by defining the forbidden regions. For each forbidden region a set of segments, called *critical segments*, can be defined. Given a plane CD_{ij} , the set of critical segments associated with the forbidden regions of this plane is composed by (see Fig. 2):

- all the boundary edges of the forbidden region but the one coincident with r_7 .
- two segments coincident with r_3 and r_4 outgoing from the boundary of the forbidden region towards the axes of the plane. These are interrupted when they encounter another forbidden region.
- two segments coincident with r_1 and r_2 from the forbidden region towards r_3 and r_4 . These are terminated when they encounter another forbidden region.

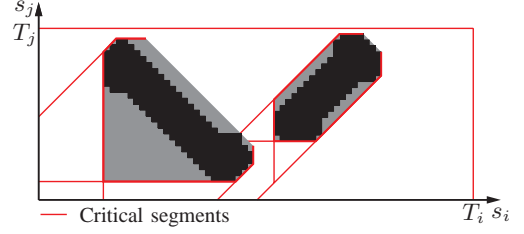


Fig. 2. Critical segments for two forbidden regions

- two segments coincident with the rays $s_i = T_i$ and $s_j = T_j$, from \mathbf{s}_G towards the axes of the plane.

Loosely speaking the subset of critical segments defined in CD_{ij} represents some configurations that are critical for the coordination of the vehicles \mathcal{A}_i and \mathcal{A}_j . This means that when a coordination path $\gamma(t)$ reaches a critical segment on some CD_{ij} , the different alternatives for the coordination of \mathcal{A}_i and \mathcal{A}_j should be explored.

A coordination path $\gamma(t)$ is represented as an object η characterized by some properties (indicated by using the syntax “ $\eta.Property$ ”). The properties are:

- $\eta.s$ denotes the ending point of the path
- $\eta.P$: indicates the parent path
- $\eta.u$ direction of the last action
- $\eta.d$ time duration of the last action
- $\eta.CC$ coordination components
- $\eta.ES$ extension stage

Every path has the starting point at $\mathbf{s}_I \in \mathcal{S}$ while the ending point is defined by the property $\eta.s \in \mathcal{S}$. A path has zero or more children paths. The children represent the paths that are generated, during one iteration of the algorithm, by extending a given path, called the parent path ($\eta.P$). The only exception is the root path, denoted as η_R , for which no parent is defined. The root path is the null length path with which the algorithm is initialized $\eta_R.s = \mathbf{s}_I$. The extension of a path η , generates a new path η' that is obtained by adding to η an action connecting the point $\eta.s$ to a point $\mathbf{s}' \in \mathcal{S}$. The new path η' is the piecewise linear curve connecting the sequence of points $\eta_0.s, \dots, \eta_k.s, \eta_{k+1}.s$ where $\eta_0 = \eta_R$, $\eta_k = \eta$, $\eta_{k+1} = \eta'$ and $\eta_i.P = \eta_{i-1}$ for every $i \in [1, k+1]$. The point \mathbf{s}' is the nearest point at which a ray outgoing from the point $\eta.s$ reaches a critical segment in some CD_{ij} . In Fig. 3 an example of the paths generated after two expansion steps within a two dimensional CD is reported. More formally:

$$\mathbf{s}' = \eta.s + d_{ss'} \mathbf{u} \quad (3)$$

where $\mathbf{u} = (u_1, \dots, u_N)$, $u_i \in \{0, 1\}$, is the vector that defines the direction of the action with which the path is extended. This vector defines the vehicles of the fleet that have to advance in order to reach the new configuration \mathbf{s}' from $\eta.s$. Each component u_i defines whether a vehicle must be moving or not. The scalar value $d_{ss'} > 0$ represents the time required by the fleet to reach the new configuration \mathbf{s}' considering that each vehicle is traveling at its nominal velocity. This parameter is stored as a property of the path,

denoted $\eta'.d$, since it will be used in the next section for the computation of the cost function. A path η can be extended by using a set of directions \mathbf{u} . For each direction in which the path is extended a new path η' is added as a child of η . The direction \mathbf{u} along which the new path is created is stored in $\eta'.\mathbf{u}$. The i -th component of $\eta'.\mathbf{u}$ is referred as $(\eta'.\mathbf{u})_i$. All the pairs of axes (i, j) such that $(\eta'.\mathbf{u})_i = 1$ or $(\eta'.\mathbf{u})_j = 1$ and where the new point $\eta'.\mathbf{s}$ is coincident with a critical segment defined in CD_{ij} are called *coordination components*. These axes are stored in $\eta'.CC$. Note that the point $\eta'.\mathbf{s}$ may lie on a critical segment in more than one plane, thus the cardinality of set is $2 \leq |\eta'.CC| \leq N$. The set of directions \mathbf{u} along which a path can be extended is denoted with U_η . The definition of U_η takes into account the direction $\eta.\mathbf{u}$ of the last action of η and the set of coordination components $\eta.CC$. Instead of all the 2^N possible directions, the set U_η contains only the directions defined by substituting in $\eta.\mathbf{u}$ any possible assignment of the components indicated by $\eta.CC$. Since, generally, $|\eta.CC|$ is only a fraction of N , this definition of U_η leads to a drastic reduction of the search directions. When the point $\eta.\mathbf{s}$ is located on the boundary of some forbidden region, all directions in U_η that enter the forbidden region must be eliminated in order to ensure that the paths extended are all collision free. Moreover, when a path is chosen to be extended the algorithm does not consider all the directions in U_η . The extension of a path is done in different stages, called *extension stages*. This is obtained by defining a partition of U_η so that at each extension stage a different subset $\tilde{U}_\eta \subseteq U_\eta$ is used for the extension. The partition is defined by considering the sum of the components u_i of the coordination components $\eta.CC$. The extension stage of a given path, denoted as $\eta.ES$, indicates which subset of directions must be considered for the next extension of the path. Formally, the subset \tilde{U}_η is defined as:

$$\tilde{U}_\eta = \left\{ \mathbf{u} \in U_\eta \mid \sum_{i \in \eta.CC} u_i = |\eta.CC| - \eta.ES \right\} \quad (4)$$

where $|\cdot|$ denotes the cardinality of the set. Note that $\eta.ES \in \{0, 1, \dots, |\eta.CC|\}$ thus the number of subsets defined by the partition of U_η is equal to $|\eta.CC| + 1$.

The algorithm is summarized in Alg. 1. The paths that are created at any iteration step are inserted into an ordered queue Q . The elements of Q are sorted in ascending order of the cost function defined in the next section. At the beginning of the algorithm Q is initialized with the root path η_R (lines 1-2 in Alg. 1). The algorithm then runs in a while loop (line 4) in which at each step, the first element from Q , denoted as η^* , is considered for the extension. If $\eta^*.ES = |\eta^*.CC|$, all the subset of U_{η^*} have already been evaluated, the path η^* can not be further extended and it is removed from Q (line 5). The function $DirectionSet(\eta^*)$ (line 7) returns the subset \tilde{U}_{η^*} of the directions corresponding to the extension stage reached by η^* . After that, the extension stage of η^* is incremented (line 8). All the directions of \tilde{U}_{η^*} that enter any forbidden region are removed (line 9). For each direction in \tilde{U}_{η^*} , a new path (along with all its properties) is computed

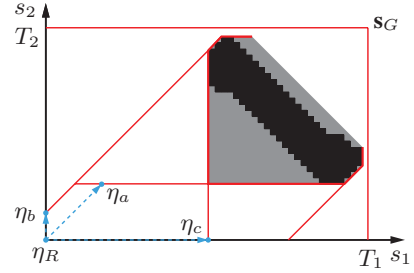


Fig. 3. Coordination paths after two expansion steps in a 2D CD.

(lines 11-13). Then the cost function is evaluated so that the new path can be inserted in the correct position within the ordered queue Q (lines 14-15). When $\eta^*.s \equiv s_G$ the search terminates and the path represented by η^* is the optimal path among all those that can be explored by the algorithm. This path is then used to specify the coordinated motion of the fleet.

Algorithm 1 Coordination path search

Require: Critical segments \mathcal{R} and forbidden regions \mathcal{F}

- 1: $\eta_R.s = \mathbf{s}_I$; $\eta_R.P = \emptyset$; $\eta_R.\mathbf{u} = (0, \dots, 0)$; $\eta_R.d = 0$;
 $\eta_R.CC = \{1, \dots, N\}$; $\eta_R.ES = 0$
 - 2: $Q = \eta_R$
 - 3: $\eta^* = GetFirst(Q)$
 - 4: **while** $\eta^*.s \neq s_G$ **do**
 - 5: **if** $\eta^*.ES = |\eta^*.CC|$ **then** $Q.Remove(\eta^*)$
 - 6: **else**
 - 7: $\tilde{U}_{\eta^*} = DirectionSet(\eta^*)$
 - 8: $\eta^*.ES = \eta^*.ES + 1$
 - 9: $\tilde{U}_{\eta^*} = RemoveCollDir(\tilde{U}_{\eta^*}, \eta^*.s, \mathcal{F})$
 - 10: **for all** $\mathbf{u} \in \tilde{U}$ **do**
 - 11: $\eta'.s = \mathbf{s}' = NewPoint(\eta^*.s, \mathbf{u}, \mathcal{R})$
 - 12: $\eta'.P = \eta^*.P$; $\eta'.\mathbf{u} = \mathbf{u}$; $\eta'.d = d_{ss'}$; $\eta'.ES = 0$
 - 13: $\eta'.CC = FindCoordAxes(\mathbf{s}', \mathbf{u}, \mathcal{R})$
 - 14: Evaluate $f(\eta')$ ▷ See Sec. IV
 - 15: $Q \leftarrow \eta'$
 - 16: **end for**
 - 17: **end if**
 - 18: $\eta^* = GetFirst(Q)$
 - 19: **end while**
-

IV. HEURISTIC COST FUNCTION

In this section we describe the function used by the algorithm to determine the order in which the paths are extended. Thanks to this function the algorithm extends first the paths that are more likely to optimize the objective function defined in (2). When the algorithm terminates the best path among all those that can be created is returned.

Like for the A* algorithm this function, denoted $f(\eta)$, is defined as the sum of two terms. The first, denoted $g(\eta)$, is directly related to the shape of the path while the second, denoted $h(\eta)$, is an heuristic underestimate of the minimum

cost of the remaining path to the goal point $\mathbf{s}_G \in \mathcal{S}$. Formally the cost function is defined by this expression:

$$f(\eta) = g(\eta) + h(\eta) \quad (5)$$

Recall that each path can be described as a sequence of actions (i.e. linear segments). Each new path is obtained from a previous one by adding a new action. Given a path η' the value of $g(\eta')$ is computed by adding to the cost of its parent path $\eta = \eta'.P$ the cost of the new action:

$$g(\eta') = g(\eta) + c(\eta, \eta'), \quad g(\eta_R) = 0 \quad (6)$$

where $c(\eta, \eta')$ is the additional cost of the new added action. This cost is defined as:

$$c(\eta, \eta') = K_{acc} \cdot N_{acc}(\eta, \eta') + N_{run}(\eta) \cdot \eta'.d \quad (7)$$

where, in the first term, $N_{acc}(\eta, \eta')$ is the number of vehicles that have to change their motion at the configuration $\eta.s$ while executing the coordination path η' . This quantity corresponds to the number of non null components of the vector $\eta'.\mathbf{u} - \eta.\mathbf{u}$. In the second term, $N_{run}(\eta)$ is the number of vehicles that have not reached the goal at the configuration $\eta'.s$. The value $\eta'.d$ (defined in Sec. III) represents the time required by the fleet to reach the configuration $\eta'.s$ from the configuration $\eta.s$. The parameter K_{acc} is the additional cost accumulated each time that a vehicle has to change its motion (Sec. II-B). By inflating this value, the costs of the paths that require many changes of velocity increase. Thus the paths that are extended are mainly those that require less changes of velocity due to their lower cost. This focusing of the extension process leads to a reduction of the explored states and thus a reduction of the computational burden.

Thus, the paths that imply less changes of velocities are extended before the others due to their lower cost.

The heuristic estimation of the cost-to-go must be an underestimate of the actual cost in order to obtain an optimal plan ([5]). The estimate of this cost is defined as:

$$h(\eta) = \sum_{i=1}^N (T_i - \eta.s_i) + K_{acc} \cdot S(\eta) + 2K_{acc} \cdot P(\eta) \quad (8)$$

The first term is the sum of the times that each of the vehicles takes to reach the goal position if the collisions with other vehicles are ignored. The last two terms are an underestimate of the amount of costs that will be accumulated each time that a vehicle will have to stop or start its motion. In particular $S(\eta)$ is the number of vehicles \mathcal{A}_i such that $(\eta.\mathbf{u})_i = 0$ and $(\eta.s)_i \neq T_i$. These vehicles will have to start their motion in order to reach their goal. The term $P(\eta)$ represents the minimum number of vehicles that, from the configuration $\eta.s$, will have to stop before reaching the goal. This term is multiplied by two because each time that a vehicle has to stop then it will also have to restart its motion. Consider for example the point $\eta_a.s$ in Fig. 3. From this configuration the vehicles \mathcal{A}_1 and \mathcal{A}_2 will approach a collision if they advance simultaneously. Thus if both are advancing it can be stated that one of the two vehicles will have to stop (thus $P(\eta_a) = 1$). The value of $P(\eta)$ can be

computed by solving a binary integer program (BIP). We denote with z_1, \dots, z_N the variables of the program. The value $z_i = 1$ means that the vehicle \mathcal{A}_i will have to give the way to another vehicle (thus u_i will have to be set to 0) while $z_i = 0$ means that \mathcal{A}_i is stopped or it does not have to give the way to other vehicles. The objective function is $\sum_{i=1}^N z_i$, subject to the constraints defined as follows. For each plane CD_{ij} such that $(\eta.\mathbf{u})_i = 1$ and $(\eta.\mathbf{u})_j = 1$, if the projection of $\eta.s$ onto CD_{ij} is between the rays r_3, r_4, r_5, r_6 and r_7 (Fig. 1), the constraint $z_i + z_j \geq 1$ is imposed. This constraint means that, since the pair of vehicles is approaching a collision, at least one of the two vehicles will have to stop. The minimization of the objective function under these constraints gives an underestimate of the number vehicles that will have to stop in order to avoid all forbidden regions. A solver for the BIP can be found within the Optimization Toolbox of MATLAB.

Given the solution z_1^*, \dots, z_N^* , $P(\eta)$ is computed as:

$$P(\eta) = \sum_{i=1}^N z_i^* \quad (9)$$

Consider the example reported in Fig. 3 for two vehicles in which $\eta_R.s = (0, 0)$, $\eta_a.s = (10, 10)$, $\eta_b.s = (0, 5)$, $\eta_c.s = (30, 0)$ and $\mathbf{s}_G = (60, 40)$. We report the computation of the cost function for $K_{acc} = 10$. By applying (5)-(8):

- For η_R : $g(\eta_R) = 0$, $h(\eta_R) = 120$, thus $f(\eta_R) = 120$. But this path will not further extended because $\eta_R.ES = |\eta_R.CC|$, thus it will be removed from Q.
- For η_a : $g(\eta_a) = 40$, $h(\eta_a) = 100$, thus $f(\eta_a) = 140$.
- For η_b : $g(\eta_b) = 20$, $h(\eta_b) = 105$, thus $f(\eta_b) = 125$.
- For η_c : $g(\eta_c) = 70$, $h(\eta_c) = 80$, thus $f(\eta_c) = 150$.

Thus at the next step the algorithm will extend path η_b .

V. EXPERIMENTS

We have tested our algorithm running a simulation with up to 10 robots in MATLAB on a Intel P8400 2.26 GHz (see Fig. 4). The total time required by the algorithm in order to compute the coordination path is 12.4 seconds. The vehicles are supposed to move on a roadmap used in a real industrial plant. In its real implementation this roadmap is composed both of curvilinear and straight line segments. In order to simplify the simulation program, all the curvilinear segments have been replaced by straight line segments. However, this is only a graphical approximation since the underlying coordination algorithm considers the effective trajectory of the vehicles. The nominal speed with which a vehicle has to cover each segment is defined during the roadmap design and it can be different according to the type of vehicle and the load that it carries. The missions that are assigned to each vehicle are representative of the real working condition of the system. Each mission is defined by four way-points: initial position, pick-up position, drop off position and rest position. In Fig. 4 the test is illustrated. The initial, pick-up, drop-off and rest positions of the vehicles are marked with $S1, \dots, S10$, $P1, \dots, P10$, $D1, \dots, D10$, $H1, \dots, H10$ respectively. See also the attached video of

TABLE I
TIMING FOR EACH VEHICLE.

Vehicle	1	2	3	4	5	6	7	8	9	10
$T_{advance}$ [s]	144.2	118.6	115.2	127	73.4	151	123	107	121.6	136.8
T_{stop} [s]	26.8	0	1.6	0	17	0	0	1.4	0	0

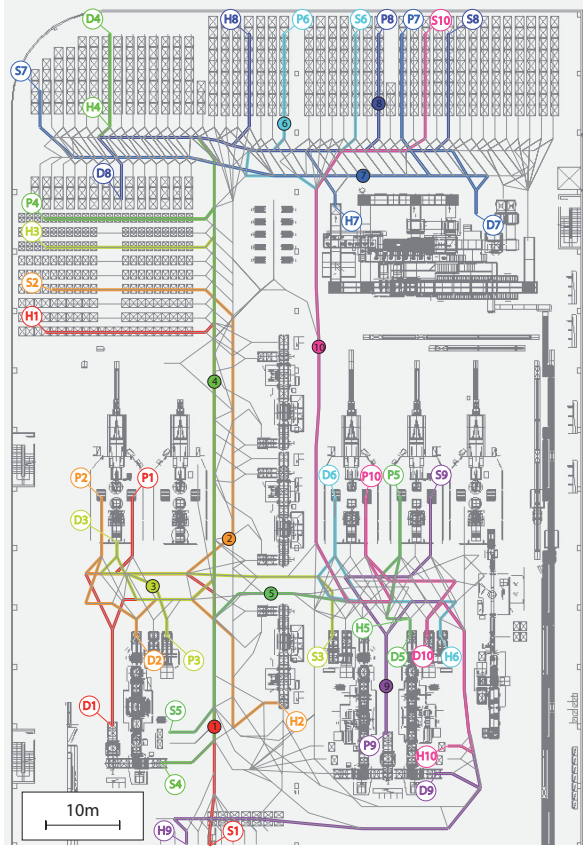


Fig. 4. A snapshot of the simulation with 10 vehicle running in a real industrial plant.

the simulation test. The timings of each vehicle are reported in Tab. I. For each vehicle the total time of advancement ($T_{advance}$) and the waiting time (T_{stop}) are reported. For the proposed experiment we have adopted $K_{acc} = 10$ so that the algorithm extends only the coordination paths that require few accelerations for the vehicles. In particular the solution found will produce a coordinated motion in which all vehicles avoid collisions by stopping their motion only once. Although this could produce a suboptimal solution, inflating K_{acc} allows a substantial reduction of the computation times. In Fig. 5 the planes CD_{ij} , relative to an experiment conducted with 5 vehicles are displayed. For each plane, the projections of the collision-free coordination path are displayed along with the set of paths explored by the algorithm.

VI. CONCLUSIONS AND FUTURE WORK

Future work aims at determining the upper bounds of the computational complexity of the proposed algorithm and studying the relationship between this upper bound and the

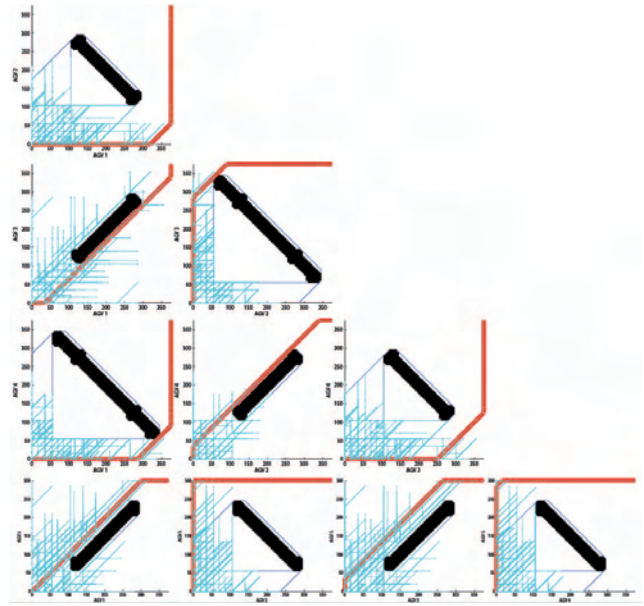


Fig. 5. Projections of the best coordination path within CD_{ij} (red line) and the paths explored by the algorithm (cyan lines). Axis units in 10^{-1} s

parameter K_{acc} . In real world planning problems, time for deliberation is often limited. Future work aims at modifying the algorithm so that it can find a feasible solution quickly and then refine it while vehicles are in motion. We will also study the optimality of the proposed algorithm. This will require to demonstrate that the set of coordination paths extended by the algorithm can contain global optimum of the cost function.

ACKNOWLEDGMENT

The authors wish to greatly acknowledge the company Elettric 80 s.p.a. (www.elettric80.com) which supported this research.

REFERENCES

- [1] J. Peng and S. Akella, "Coordinating multiple double integrator robots on a roadmap: Convexity and global optimality," in *IEEE Int. Conf. on Robotics and Automation*, Apr 2005, pp. 2751–2758.
- [2] S. Berman, E. Schechtman, and Y. Edan, "Evaluation of automatic guided vehicle systems," *Robot. Comput. Integr. Manuf.*, pp. 123–126, 2008.
- [3] P. O'Donnell and T. Lozano-Perez, "Deadlock-free and collision-free coordination of two robot manipulators," in *IEEE Int. Conf. on Robotics and Automation*, vol. 1, May 1989, pp. 484–489.
- [4] R. Olmi, C. Secchi, and C. Fantuzzi, "Coordination of multiple AGVs in an industrial application," in *IEEE Int. Conf. on Robotics and Automation*, May 2008, pp. 1916–1921.
- [5] S. LaValle, *Planning Algorithms*. Cambridge University Press, 2006, also available at <http://planning.cs.uiuc.edu/>.
- [6] T. Simeon, S. Leroy, and J. Laumond, "Path coordination for multiple mobile robots: a resolution-complete algorithm," *IEEE Trans. Robot. Automat.*, vol. 18, no. 1, pp. 42–49, Feb 2002.
- [7] Y. Guo and L. Parker, "A distributed and optimal motion planning approach for multiple mobile robots," in *IEEE Int. Conf. on Robotics and Automation*, vol. 3, May 2002, pp. 2612–2619.
- [8] S. LaValle and S. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Trans. on Robotics and Automation*, vol. 14, no. 6, pp. 912–925, Dec 1998.
- [9] R. Ghrist, J.M.O'Kane, and S. LaValle, "Computing pareto optimal coordination on roadmaps," *Int. J. Robot. Res.*, vol. 24, no. 11, pp. 997–1010, 2005.

ArosDyn: Robust Analysis of Dynamic Scenes by means of Bayesian Fusion of Sensor Data - Application to the Safety of Car Driving

Christian Laugier, Igor E. Paromtchik, Mathias Perrollaz, Mao Yong,
Amaury Nègre, John-David Yoder, Christopher Tay

INRIA Grenoble Rhône-Alpes, 38334 Saint Ismier, France, E-mail: Christian.Laugier@inrialpes.fr

Abstract—The ArosDyn project aims to develop an embedded software for robust analysis of dynamic scenes in urban environment during car driving. The software is based on Bayesian fusion of data from telemetric sensors (lidars) and visual sensors (stereo camera). The key objective is to process the dynamic scenes in real time to detect and track multiple moving objects, in order to estimate and predict risks of collision while driving.

I. INTRODUCTION

Estimation and prediction of collision risks will be mandatory for the next generation of cars. The methods and software being developed in the framework of the ArosDyn project provide to monitor the traffic environment by on-board stereo vision and laser scanners, perform data fusion from multiple sensors by means of Bayesian Occupancy Filter (BOF), as well as estimate and predict risks as stochastic variables by using Hidden Markov Models (HMM) and Gaussian process.

The detection of objects is performed by processing the telemetric and visual data. The local simultaneous localization and mapping (SLAM) algorithm serves for preprocessing the range data from the lidars. The preprocessing of stereo images results in a disparity map. The probabilistic models of a lidar and a stereo camera are used. The environment is represented by a grid, where each cell contains the probability distribution of the cell occupancy and the cell velocity. The data fusion is performed in the BOF with the probability distributions computed from the real data from lidars and stereo vision. The fast clustering-tracking (FCT) algorithm performs clustering of detected objects, data association and tracking of objects. The collision risk estimation results in a probability of risks for a period T ahead. The visual detection and tracking provide to estimate the position of objects relative to the reference frame and to calculate a quality measure of visual tracking.

II. OBSTACLE DETECTION AND TRACKING BY A LIDAR

Perception of the surrounding physical world reliably is a fundamental requirement of autonomous vehicle systems. The major prerequisite for a such system is a robust algorithm of object detection and tracking. We developed a lidar-based moving object detection and tracking scheme. It is composed of three main modules : (1) BOF for environment representation, (2) the FCT algorithm for tracking of objects, and (3) a local SLAM based preprocessing module.

Bayesian Occupancy Filter (BOF)

The BOF operates with a two-dimensional grid representing the environment. Each cell of the grid contains a probability distribution of the cell occupancy and a probability distribution of the cell velocity. Given a set of observations, the BOF algorithm updates the estimates of the cell occupancy and velocity for each cell in the grid. The BOF model is described in [1], [2]. At each time step, the probability distributions of occupancy and velocity of a cell are estimated through Bayesian inference with our model. The inference leads to a Bayesian filtering process, as shown in Fig. 1.

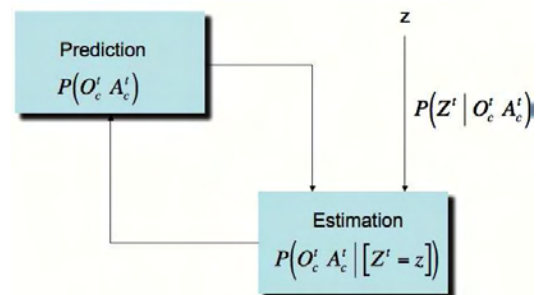


Fig. 1: Bayesian filtering in the estimation of occupancy and velocity distribution in the BOF grid

Fast Clustering-Tracking (FCT) algorithm

Our Fast Clustering-Tracking (FCT) algorithm works at the level of object representation to track the trajectories of objects [3]. The FCT algorithm can be roughly divided into three modules : a clustering module, a data association module, and a tracking and tracks management module.

The clustering module takes two inputs : the occupancy/velocity grid estimated by the BOF, and the prediction of the tracker which provides a region of interest (ROI) for each object being tracked. We then try to extract a cluster in each ROI and associate it with the corresponding object. There could be a variety of cluster extracting algorithms, however, we have found that a simple neighbourhood based algorithm works efficiently and provides satisfactory results. The output of this module leads to three possible cases, as shown in Fig. 2 : (i) no observation, where the object is not observed in the ROI, (ii) ambiguity free, where one and only

one cluster is extracted and is implicitly associated with the given object, and (iii) ambiguity, where the extracted cluster is associated with multiple objects.

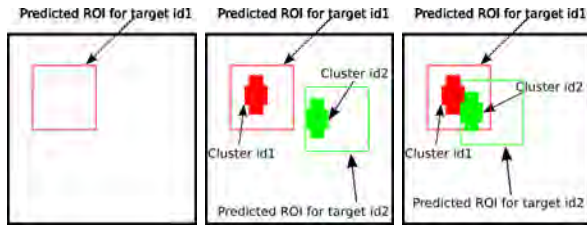


Fig. 2: Cases of the clustering result

In the clustering module, if there exist more than one ROI of the tracked objects which are overlapped, the extracted cluster will be associated with multiple targets. This is the so-called ambiguity data association case. A data association module is then designed to solve this problem. Assume there are N targets associated with a single cluster, where N is a number we know exactly. The causes of the ambiguity are further analyzed as twofold : (i) targets are being too close to each other and the observed cluster is the union of more than one observations generated by N different real objects, (ii) N different targets correspond to a single object in the real world and they should be merged into one.

We employ a re-clustering strategy to the first situation and a cluster merging strategy to the second one. The objective of the re-clustering step is to divide the cluster into N sub-clusters and associate them with the N objects, respectively. Because the number N is known, a natural approach is to apply a K-means based algorithm [4]. To deal with the second case, we follow a probabilistic approach. Whenever an ambiguous association F_{ij} between two tracks T_i and T_j is observed, a random variable S_{ij} is updated which indicates the probability of T_i and T_j to be parts of a single object in the real world. The probability values $P(F_{ij} | S_{ij})$ and $P(F_{ij} | \neg S_{ij})$ are the parameters of algorithm which are constant with regard to i and j . Similarly, the probability $P'(S_{ij} | \neg F_{ij})$ is updated when no ambiguity between T_i and T_j is observed. Then, by thresholding the probability $P'(S_{ij})$, the decision of merging the tracks T_i and T_j can be made by calculating the Mahalanobis distance between them.

Now we arrive at a set of reports which are associated with the objects being tracked without ambiguity. Then, it is straightforward to apply a general tracks management algorithm to create and delete the tracks, and use a Kalman filter [5] to update their states.

Local SLAM based preprocessing module

In the outdoor environment, a large part of the laser impacts comes from the stationary objects, e.g. buildings, trees and parked vehicles. Because the BOF works in the local frame, if we use the laser data directly to build the sensor model for the BOF-FCT framework, the stationary objects

will be detected and tracked which is not our objective. Therefore, we solve a local SLAM problem and use the map built on-line to divide the laser impacts into a stationary set and a moving set. Then, only the laser impacts caused by moving objects are used to build the sensor model.

We solve the local SLAM problem by using an occupancy grid map based algorithm [6]. Let m denote the map, $Z_{0:t} = z_0, \dots, z_t$ be the sensor observations where z_i is the frame of observation at time step i , $U_{1:t} = u_1, \dots, u_t$ denote the odometry data, $X_{0:t} = x_0, \dots, x_t$ be the states of the vehicle. The objective of a full sequential SLAM algorithm is to estimate the posterior $P(m, x_t | Z_{0:t}, U_{1:t}, x_0)$. Though it is already well formulated, the estimation of this posterior is not trivial. However, in our application, we are not concerned with the global precision of the vehicle's states. Instead of building a map of a large cyclic environment, we only need to build a map of a limited area, which moves with the vehicle. The precision of this map can be guaranteed by the relatively accurate laser sensor. Thus, it is feasible to apply a maximum likelihood algorithm rather than a probabilistic algorithm in this case. By estimating the maximum likelihood map and the maximum likelihood state at each time step, the local SLAM problem can be decomposed into a mapping problem and a localization problem.

To solve the mapping problem, we apply a standard log-odds filtering scheme. Let $l(m_i | x_{0:t}, Z_{0:t}) = \frac{P(m_i | x_{0:t}, Z_{0:t})}{P(\neg m_i | x_{0:t}, Z_{0:t})}$ denote the log-odds value of a cell m_i in m . The update formula can be written as :

$$l(m_i | x_{0:t}, Z_{0:t}) = l(m_i | x_t, Z_t) - l(m_i) + l(m_i | x_{0:t-1}, Z_{0:t-1}),$$

where $l(m_i)$ is the prior value which is set to 0, and $l(m_i | x_t, Z_t)$ is obtained from the beam-based *inverse sensor model*.

In the localization step, the maximum likelihood state can be estimated as :

$$x_t = \arg \max_{x_t} P(z_t | m, x_t) P(x_t | x_{t-1}, u_t).$$

To balance the processing time and the accuracy, we employ a sampling based scan matching algorithm in localization. Given the odometry motion model $P(x_t | x_{t-1}, u_t)$, a set of state samples is generated at each time step. From each state sample x_t , we match the current sensor measurement with the map. Under the assumption of the independency of beams, the likelihood probability can be expressed as :

$$P(z_t | m, x_t) = \prod_i P(z_t | m_i, x_t).$$

Then, the sample with the maximum likelihood probability is selected as the current state. Although, in theory, this algorithm may suffer from the local minima problem, our experimental results show that it works fine with an accurate laser scanner in an outdoor urban environment.

Given the occupancy grid map and the current state of the vehicle, the laser impacts generated from stationary object or moving object can be discriminated by applying a threshold of occupancy probability. We then use the impacts generated by moving objects to create the sensor model for the BOF.

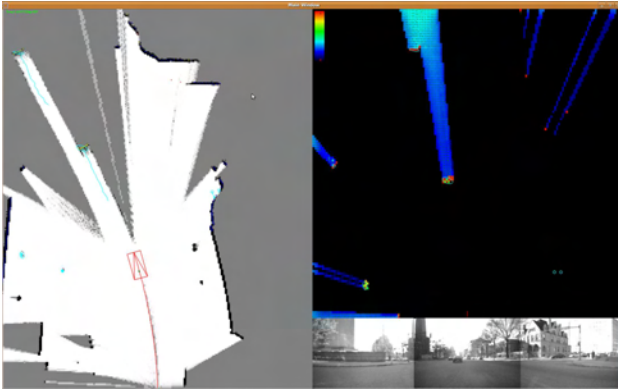


Fig. 3: Results on the CMU dataset

Experimental results

To test the effectiveness of the BOF-FCT framework and the preprocessing algorithm, we use a laser scanner dataset available from the Carnegie-Mellon University (CMU) [7]. The CMU dataset was recorded for the vehicle moving in the urban traffic environment. The maximum detection range of the laser is about 80 meters, the angular range is from 0° to 180° , the angular resolution is 0.5° . The odometry data is also provided. The camera images in this dataset are only used to demonstrate the scenario. The grid resolution of the occupancy grid map is $0.2m$, and the grid resolution of the BOF is $0.3m$. An example is presented in Fig. 3. The left of the figure demonstrates the occupancy grid map view, the right-top shows the BOF view, and the right-bottom is the camera image view. In the occupancy grid map, the rectangle represents the host vehicle, the ellipses represent the mean positions and their covariances of the moving objects, the trajectories of the ego-vehicle and the targets are also shown. It can be seen that most typical moving objects, i.e. cars, buses and pedestrians, are well detected and tracked, while the stationary objects, e.g. poles, parked vehicles and buildings on the roadside, are represented by the occupancy grid map built on-line.

Implementation aspects

The major computational cost of the overall system is the calculation of the BOF grid. It has been shown in [2] and [3] that, compared with the expensive computational cost of the BOF, the cost of the FCT algorithm can be neglected. Thanks to the grid based algorithm of the BOF which provides a way to well parallelizing the computation, we re-implemented it on the GPU. The both versions are implemented in C++ language without optimization. We compared the costs of running the BOF on CPU, on a GPU with 4 multiprocessors (NVIDIA Quadro FX 1700), and on a GPU with 30 multiprocessors (NVIDIA GeForce GTX 280). The BOF on GPU can greatly reduce the processing time which guarantees the feasibility of the proposed framework for real-time applications. In particular, the autonomous vehicle application which is shown in the previous section is capable to run at $20Hz$.

III. OBSTACLE DETECTION BY STEREO VISION

We extend the BOF framework presented in section II to the detection of obstacles by means of stereo vision, which retrieves tri-dimensional data from multiple cameras. We use a stereoscopic sensor equipped with two cameras in a geometrical configuration called "rectified" and shown in Fig. 4, where the image planes are assumed to be perfectly parallel and aligned. In the rectified configuration of the stereoscopic

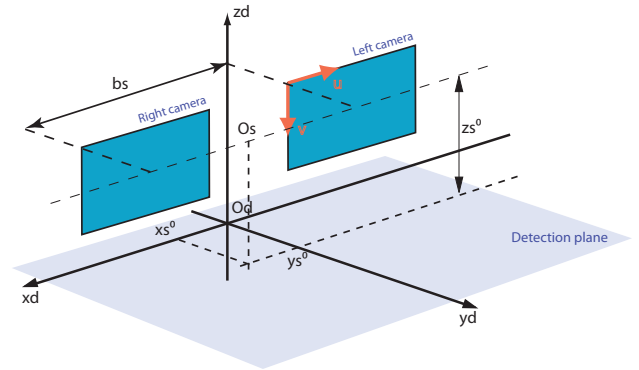


Fig. 4: Geometry of the stereoscopic sensor

sensor, a pixel and its correspondent in the second image are situated on the same image line. Therefore, assuming a matching is performed, one measures the difference of abscisses, named the disparity value. This value is inverse proportional to the distance to the observed object. Thus, the image coordinates and the disparity value for each pixel provide a tri-dimensional measurement. By estimating the disparity value for each pixel of the scene, one can compute a disparity map, corresponding to a partial tri-dimensional representation of the scene, as shown in Fig. 5.

Overview of the detection method

The stereo images are used to compute the occupancy grid. The first stage of the algorithm is to compute tri-dimensional data from the stereo image pair. The corresponding implementation may be in software, or with the dedicated hardware (both approaches are considered in ArosDyn). The occupancy grid is computed from the disparity data. This requires the definition of a probabilistic sensor model. For this purpose we apply a novel approach, which deals with the specific uncertainty of stereo data. The occupancy grid is then used for obstacle detection in the BOF framework. The object level representation of the scene is retrieved from the FCT algorithm. The detection method is explained in [8].

Disparity map computation

To compute a disparity map, we use a local matching method based on the SAD (Sum of Absolute Differences) criterion. The corresponding algorithm relies on the double correlation method [9] used for precise matching over the road surface. The main objective of this method is to provide the instant separation between "road" and "obstacle" pixels.

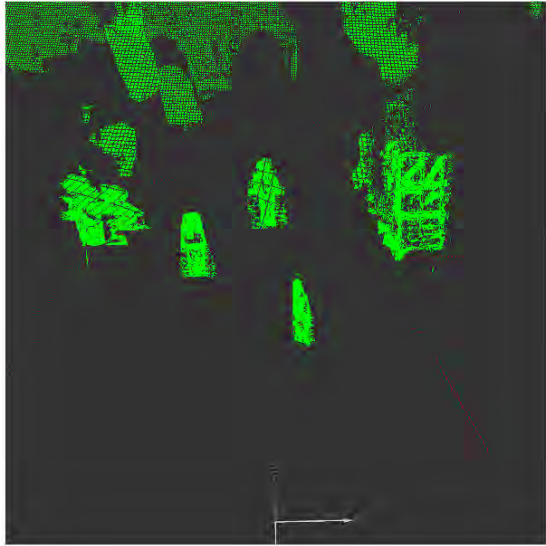


Fig. 5: a) left image from a stereo camera, b) the corresponding disparity image, c) the corresponding 3D representation

The u-v-disparity representation

Our algorithm uses the u-v-disparity approach described in [10]. The idea is to perform projections in the disparity space because this is computationally inexpensive and it allows us to work with simplified data structures. Indeed, it is more convenient to work with images than large point clouds. A significant part of our method uses u-disparity images. These images are computed by projecting disparity images along the columns, with accumulation.

User interface

The input is composed of two images at each time step. The output can be either an occupancy grid for the current

observation, or a list of detected objects, while using the BOF and the FCT algorithm. The user interface is based on the Qt library. Users have access to several parameters related to the filtering of disparity data and to the BOF. The images being processed can be visualized to better understand the computation process. The replay capability is based on the Hugar middleware [11].

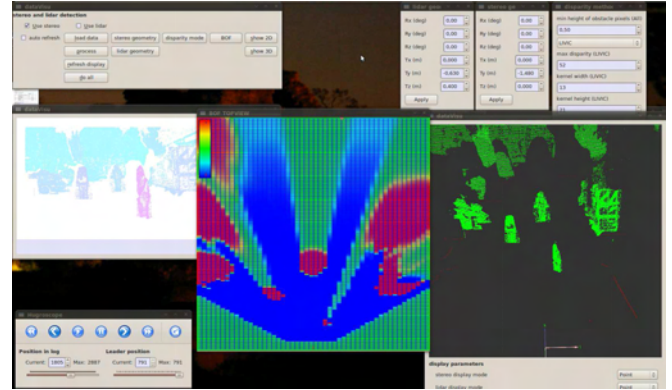


Fig. 6: User interface of the software

IV. CONCLUSION

The developed software modules provide to detect and track multiple moving objects by means of stereo vision and laser scanners. The next step is to improve our algorithms, reduce the computation time, and integrate the software modules for data fusion and risks estimation and prediction in the traffic environment.

REFERENCES

- [1] C. Coué, C. Pradalier, C. Laugier, T. Fraichard, P. Bessière, "Bayesian Occupancy Filtering for Multitarget Tracking : An Automotive Application". *Int. J. Robotics Research*, No. 1, 2006.
- [2] M.K. Tay, K. Mekhnacha, C. Chen, M. Yguel, C. Laugier, "An Efficient Formulation of the Bayesian Occupation Filter for Target Tracking in Dynamic Environments". *Int. J. Autonomous Vehicles* 6(1-2) :155-171, 2008.
- [3] K. Mekhnacha, Y. Mao, D. Raulo, C. Laugier, "Bayesian Occupancy Filter based "Fast Clustering-Tracking" algorithm". *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Nice, 2008.
- [4] C.M. Bishop, "Pattern Recognition and Machine Learning". *Springer*, 2006.
- [5] G. Welch, G. Bishop, "An Introduction to the Kalman Filter", <http://www.cs.unc.edu/~welch/kalman/index.html>
- [6] S. Thrun, W. Burgard, D. Fox, "Probabilistic robotics", *MIT Press*, 2005.
- [7] C.-C. Wang, D. Duggins, J. Gowdy, J. Kozar, R. MacLachlan, C. Mertz, A. Suppe, C. Thorpe, "Navlab SLAMMOT Datasets", *Carnegie-Mellon University*, 2004.
- [8] M. Perrollaz, A. Spalanzani, D. Aubert, "A Probabilistic Representation of the Uncertainty of Stereo-vision and its Application for Obstacle Detection". *Submitted to the IEEE Intelligent Vehicles Symposium*, San Diego, CA, USA, 2010.
- [9] M. Perrollaz, R. Labayrade, R. Gallen, D. Aubert, "A Three Resolution Framework for Reliable Road Obstacle Detection Using Stereovision", *Proc. of the IAPR MVA Conference*, 2007.
- [10] R. Labayrade, D. Aubert, J.-P. Tarel, "Real Time Obstacles Detection on Non Flat Road Geometry through v-Disparity Representation". *Proc. of the IEEE Intelligent Vehicles Symposium*, 2002.
- [11] CyCab Toolkit, <http://cycabtk.gforge.inria.fr/wiki/doku.php?id=start>

Real-Time Detection of Moving Obstacles from Mobile Platforms

Chunrong Yuan and Hanspeter A. Mallot

Abstract—In this paper we present a vision-based algorithm for the detection of moving obstacles in complex and unknown environments. The goal is to find moving objects from images captured by a mobile camera navigating together with a moving platform. One specific feature of our approach is that it does not need any information of the camera and hence works without camera calibration. Another advantage lies in the fact that it integrates motion separation and outlier detection into one statistical framework. Based on sparse point correspondences extracted from consecutive frame pairs, scene points are clustered into different classes by statistical analysis and modeling of the probability distribution function of the underlying motion characteristics. Experimental results based on several real-world video streams demonstrate the efficiency of our algorithm.

I. INTRODUCTION

Vision-based navigation strategies have been applied for a number of autonomous systems. Quite a lot of research effort has been put into camera-based detection of obstacles for the safe navigation of different robot and vehicle systems. For an intelligent vehicle to navigate automatically in dynamic environments, it is critical to provide such a vehicle with the very ability of avoiding moving obstacles.

For road vehicles, there are many possible solutions including the use of camera systems in combination with 3D range sensors. For aerial vehicles, particularly small-size UAVs, it is usually not possible to use many sensors due to weight or upload limit. In some cases, only a single perspective camera can be used. An example is shown in Fig. 1 with an AR-100 UAV. We have used this platform for the development of safe visual navigation system in static environments [1]. In this work, we would like to advance its autonomy so that it can handle moving obstacles as well.

While an imaging sensor is moving in a dynamic environment, the observed image displacements are the results of two different kinds of motion: One is the self-motion of the camera and the other is the independent motion of individual moving objects. Here it is essential to know whether there exist any moving obstacles and eventually to separate the object motion from the motion of the camera.

In the literature, different approaches have been proposed toward solving this problem. Some approaches make explicit assumptions about or even restrictions on the motion of the camera or object in the environment. Under the assumption of constant camera motion, Argyros and Lourakis [2] have

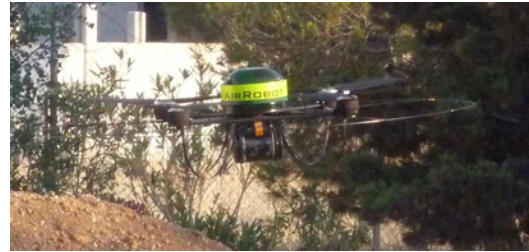


Fig. 1. The AR-100 UAV.

derived a simple constraint extracted from three frames to detect scene points whose 2D motion changes between frames. In the work of Clarke and Zisserman [3], it is assumed that both the camera and the object are just translating. Sawhney and Ayer [4] proposed a method which can apply to small camera rotation and scenes with small depth changes. In the work proposed by Patwardhan et. al. [5], only moderate camera motion is allowed.

A major difference among the existing approaches lies in the parametric modeling of the underlying constraint used for motion detection. One possibility is to use the 2D homography to establish a constraint between a pair of viewed images [6], [7]. Points whose 2D displacements are inconsistent with the homography are classified as belonging to independent motion. The success of such an approach depends on the existence of a dominant plane (e.g. the ground plane) in the viewed scene.

Another possibility is to use geometric constraint between multiple views. The approach proposed by Torr et. al. [8] uses the trilinear constraint over three views. A multibody trifocal tensor based on three views is applied by Hartley and Vidal [9], where the EM (Expectation and Maximization) algorithm is used to refine the constraints as well as their support iteratively. In both approaches, the separation process is only partially automatic. Another inherent problem shared by the two is their inability to deal with dynamic objects that are either small or moving at a distance. Under such circumstances it would be difficult to estimate the parametric model of independent motion, since not enough scene points may be detected from dynamic objects.

So far few of the cited work has been extensively evaluated using different real-world scenarios. With a few exceptions, one or two examples with a couple of images are used to validate the proposed approach. Due to various limitations in the approaches, performance is far from satisfactory.

In this paper, we present a new approach for fast detection of independent object motion under challenging real-world

This work was supported by the European Commission for the research project μ Drones with the contract number FP6-2005-IST-6-045248

C. Yuan and H. A. Mallot are with the Chair of Cognitive Neuroscience, University of Tübingen, Germany, {chunrong.yuan, hanspeter.mallot}@uni-tuebingen.de

situations. Our approach can be applied for arbitrary video sequences taken with a general perspective camera. No camera calibration is required. Unlike most current approaches, where it is already assumed that independent motion exists, it is in our case possible to differentiate automatically whether there is independent motion at all. This is integrated in a general framework which is capable of fully automatic clustering of the motion of scene points into three different classes. They are camera motion, object motion and outliers.

We argue that automatic detection of independent motion requires: (1) Accurate localization of 2D point correspondences in noisy situation. (2) A proper selection of motion constraint. (3) Automatic determination of the membership of each detected scene point by taking into account the probabilistic distribution of the underlying motion characteristics. In section II to IV, we will present our approach by discussing each of these three aspects respectively. In section V, experimental results will be shown. In particular, we evaluate the performance of our approach under various environmental changes, e.g., strong changes of illumination in outdoor environment and motion disturbance of non-rigid background object like swinging leaves of trees.

II. 2D MOTION DETECTION

The goal of 2D motion detection is to find the corresponding scene points between a pair of images. We take a local approach for the measurement of 2D image displacement and evolve from the well-known Lucas-Kanade algorithm [10] into an optimal motion detection approach.

Suppose an image point $\mathbf{p} = [x, y]^T$ is observed at time step t in image \mathbf{f}^t . At time step $t + 1$, a corresponding point \mathbf{q} is observed in image \mathbf{f}^{t+1} . Let's denote the image displacement as a 2D vector $\mathbf{v} = [u, v]^T$, the following relationship holds:

$$\mathbf{q} = \mathbf{p} + \mathbf{v} \quad (1)$$

Let f_x and f_y be the spatial image gradient, and f_t the temporal image derivative, the displacement can be computed as [10]:

$$\mathbf{v} = \mathbf{G}^{-1} \mathbf{b}, \quad (2)$$

with

$$\mathbf{G} = \sum_{(x,y) \in w} \begin{bmatrix} f_x^2 & f_x f_y \\ f_x f_y & f_y^2 \end{bmatrix}, \quad (3)$$

and

$$\mathbf{b} = - \sum_{(x,y) \in w} \begin{bmatrix} f_t f_x \\ f_t f_y \end{bmatrix}. \quad (4)$$

The above solution requires that \mathbf{G} is invertible, which means that the image should have gradient information in both x and y direction in the neighborhood w centered at point \mathbf{p} . For the reason of better performance, a point selection process should be carried out before \mathbf{v} is calculated. Because matrix \mathbf{G} can be diagonalized as

$$\mathbf{G} = \mathbf{U}^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \mathbf{U}, \quad (5)$$

where \mathbf{U} is an orthonormal transform, the following criterion can be used to select point \mathbf{p} from \mathbf{f}^t :

- 1) λ_1 and λ_2 should be big.
- 2) The ratio of $\frac{\lambda_1}{\lambda_2}$ should not be too large.

For the purpose of subpixel estimation of \mathbf{v} , we use an iterative algorithm, updating f_t sequentially. We calculate f_t as follows:

$$f_t = f^{t+1}(x + u, y + v) - f^t(x, y). \quad (6)$$

The initial value of \mathbf{v} is set as $\mathbf{v} = [u, v]^T = [0, 0]^T$. To handle large motion, a further strategy is to carry out the above iterative steps in a pyramidal fashion, beginning with the smallest scale image and refining the estimate in consecutive higher scales. This coarse to fine strategy uses the fact that large image motion in the original images corresponds to smaller motions in a down-scaled version.

Once a set of point $\{\mathbf{p}_i\}$ has been selected from image \mathbf{f}^t and a corresponding set of $\{\mathbf{v}_i\}$ is calculated, we obtain automatically a set of point $\{\mathbf{q}_i\}$ in \mathbf{f}^{t+1} with $\mathbf{q}_i = \mathbf{p}_i + \mathbf{v}_i$. In order to make the estimated 2D displacements \mathbf{v}_i more accurate, we calculate a new set of backward displacement $\hat{\mathbf{v}}_i$ for \mathbf{q}_i from \mathbf{f}^{t+1} to \mathbf{f}^t . As a result we get a backward projected point $\hat{\mathbf{p}}$ with

$$\hat{\mathbf{p}}_i = \mathbf{q}_i + \hat{\mathbf{v}}_i \quad (7)$$

An accurately calculated displacement \mathbf{v}_i should satisfy

$$e_i = \|\mathbf{p}_i - \hat{\mathbf{p}}_i\| = 0. \quad (8)$$

For this reason, only those points whose $e_i < 0.1$ pixel will be kept. In this sense, we have achieved an optimal data set $\{(\mathbf{p}_i, \mathbf{q}_i)\}$ with high accuracy of point correspondences between \mathbf{f}^t and \mathbf{f}^{t+1} .

III. MOTION CONSTRAINT

While the observing camera is moving, the relative motion between it and the surrounding environment gives rise to the perception of image displacement of scene points. The perceived 2D displacement can be caused either entirely by the camera motion, or by the joint effect of both camera and object motion. This means, the displacement vector \mathbf{v}_i can come either from static or dynamic objects in the environment. While static objects remain their location and configuration in the environment, dynamic objects vary their locations with time.

Without loss of generality, we can assume that camera motion is the dominant motion. This assumption is reasonable since moving objects generally come from a distance and can come near to the moving camera only gradually. Compared to the area occupied by the whole static environment, the subpart occupied by dynamic objects is less significant. Hence it is generally true that camera motion is the dominant motion.

As a consequence, it is also true that most vectors \mathbf{v}_i come from static scene points. Under such circumstance, it is possible to find a dominant motion constraint. The motion of static scene points will agree with the dominant motion.

Those scene points whose motions do not agree with the dominant motion constraint can hence be either dynamic points or outliers which result from environmental changes that so far have not been considered during the 2D motion detection process. Motion separation can then be achieved by finding how well each vector \mathbf{v}_i agrees with the dominant motion constraint.

Having gained a corresponding point set $\{\mathbf{p}_i\}$ and $\{\mathbf{q}_i\}$, with $i = 1, \dots, N$, we can find a transform which will explain the 2D motion of n static points between \mathbf{f}^t and \mathbf{f}^{t+1} . We use a similarity transform $\mathbf{T} = \{\mathbf{R}, \mathbf{t}, s\}$ as motion constraint. This means, we can find the transform parameters minimizing the following distance measure:

$$D(\mathbf{R}, \mathbf{t}, s) = \sum_{i=1}^N \|\mathbf{q}_i - (s\mathbf{R}\mathbf{p}_i + \mathbf{t})\|^2 \quad (9)$$

A solution can be found by using a simple linear least-squares minimization method [11].

IV. MOTION SEPARATION

Since the vectors \mathbf{v}_i coming from static scene points are caused by the camera motion, while the \mathbf{v}_i coming from moving scene points are the result of independent motion, the separation of the two kinds of motion is equivalent to clustering the set of mixed $\{\mathbf{v}_i\}$ into different classes. Altogether there are three classes: Static scene points, dynamic scene points, and outliers.

Based on the motion constraint $\mathbf{T} = \{\mathbf{R}, \mathbf{t}, s\}$, we can calculate a residual error for each of the points as

$$d_i = \sum_{i=1}^n \|\mathbf{p}_i + \mathbf{v}_i - (s\mathbf{R}\mathbf{p}_i + \mathbf{t})\| \quad (10)$$

We can expect that:

- $d_i = 0 \Rightarrow \mathbf{v}_i$ is correct (inlier), \mathbf{p}_i is a static point
- d_i is small $\Rightarrow \mathbf{v}_i$ is correct (inlier), \mathbf{p}_i is a dynamic point
- d_i is very big $\Rightarrow \mathbf{v}_i$ is incorrect (outlier)

Now the problem becomes finding two thresholds k_1 and k_2 , so that:

$$\mathbf{p}_i \text{ is } \begin{cases} \text{static point} & \text{if } d_i \leq k_1 \\ \text{dynamic point} & \text{if } k_1 < d_i \leq k_2 \\ \text{outlier} & \text{if } d_i > k_2 \end{cases} \quad (11)$$

This belongs to a typical pattern classification problem, which can be solved by analyzing the probabilistic distribution of the set of distance errors $\{d_i\}$. The most direct way is to quantize the distance d_i into $L + 1$ level, ranging from 0 to L pixels. Following that, a residual distance histogram $h(j)$, $j \in [0, L]$, can be calculated. If $h(j)$ is a multimodal histogram, two thresholds k_1 and k_2 can be found automatically for motion separation.

If there are two classes Ω_0 and Ω_1 , a probability distribution of d_i can be computed as

$$\rho_i = \frac{h_i}{N} \quad (12)$$

Since points belonging to the same class have similar distribution, a threshold k exists for separating Ω_0 and Ω_1 . The threshold k can be computed automatically by going through the following steps:

- 1) Calculate probability density functions of Ω_0 and Ω_1

$$\rho(\Omega_0) = \sum_{i=0}^k \rho_i \quad (13)$$

$$\rho(\Omega_1) = \sum_{i=k+1}^L \rho_i \quad (14)$$

- 2) Calculate the mean of Ω_0 and Ω_1 respectively as

$$\mu(\Omega_0) = \sum_{i=0}^k \frac{i\rho_i}{\rho(\Omega_0)} \quad (15)$$

$$\mu(\Omega_1) = \sum_{i=k+1}^L \frac{i\rho_i}{\rho(\Omega_1)} \quad (16)$$

- 3) Calculate the mean of the whole data set

$$\mu = \sum_{i=0}^L i\rho_i \quad (17)$$

- 4) Calculate the inter-class difference function

$$J(k) = \rho(\Omega_0)(\mu(\Omega_0) - \mu)^2 + \rho(\Omega_1)(\mu(\Omega_1) - \mu)^2 \quad (18)$$

- 5) Threshold is found as

$$\kappa = \underset{k}{\operatorname{argmax}} \{J(k)\} \quad (19)$$

Using the above two-class method, we can first find a threshold k_1 ($0 < k_1 < L$) for separating the points into two classes: Ω_0 with static points and Ω_1 a mixed class of dynamic points and outliers. In case that $k_1 = 0$ or $k_1 = L$, this means there is only a single motion which is the camera motion. If k_1 does exist, then we will further cluster the remaining mixed set of both dynamic points and outliers by calculating another threshold k_2 with $k_1 < k_2 < L$.

V. EXPERIMENTAL EVALUATION

The proposed approach has been implemented using C++ running on a Dell Latitude E4300 laptop. For each pair of frame \mathbf{f}^t and \mathbf{f}^{t+1} , we first detect 3000 features in frame \mathbf{f}^t and try to find their correspondences in \mathbf{f}^{t+1} . Depending on the nature of the input frames, the found number of point correspondences N ranges from a few hundreds to a few thousands. Thanks to the linear methods used, our motion separation algorithm has only a computational complexity of $O(N)$. Consequently, a frame rate of 25-30 frames/s can be achieved for images with a resolution of 640×480 .

For the evaluation of our approach, we have used several video sequences. All the frames are captured by mobile cameras navigating in natural environments. Evaluation is based on the successful detection of moving objects appeared in the visual field of the mobile camera. At the same time, false alarms are computed.

The images in the first video sequence are taken on a sunny day in an outdoor environment, using a camera

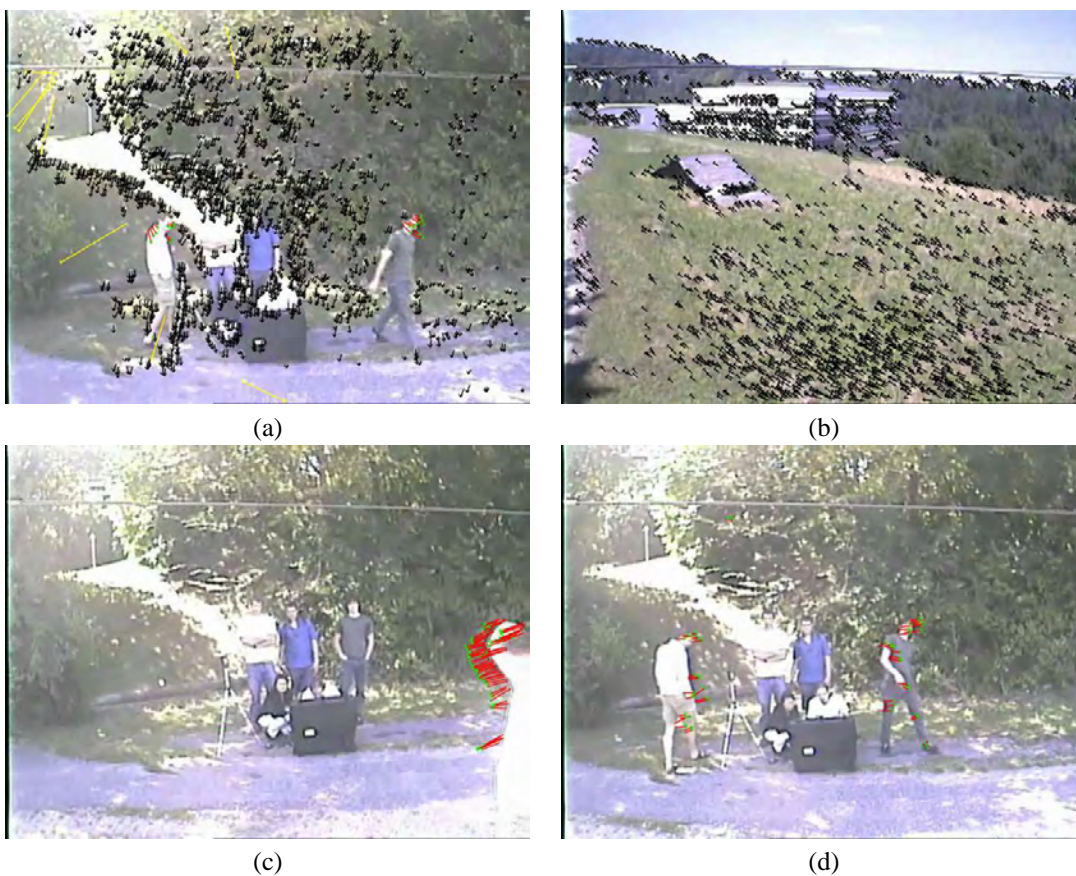


Fig. 2. Examples of detection results in sequence 1.

mounted on the AR-100 UAV. Here the dynamic objects are people moving around the UAV. Altogether there are 3082 frames in the video sequence. Among them, there are 1907 frames where no independent motion occurs. In each of the remaining 1175 frames, there are either one or two objects moving. The total number of moving objects is 1250.

Shown in Fig. 2 (a) is one example image together with the detected 2D displacement vectors. There are a few outliers in the static background as well as on the moving person. This is due largely to illumination variations. Those long vectors (with color yellow) are the outliers. The vectors shown in color black come from the static background. The two moving persons have been identified correctly, as can be seen clearly from the vectors shown in red lines with green tips. Another example is shown in Fig. 2 (b), where neither independent motion nor outlier occurs. In both cases, it is observed that the camera motion consists of both rotation and translation component.

Further examples are shown in the bottom row of Fig. 2. For clearly we show here only the flow vectors from the identified moving objects. In Fig 2 (c), there is only one person moving. In Fig 2 (d), a false alarm has occurred, although the two moving persons have been detected correctly. The false alarm is caused by environmental factors, especially illumination changes and swinging leaves.

Sequence 2 is captured with a hand-held camera. The

moving object to be detected is the AR-100 UAV. There are also some people moving in the background. In Fig. 3 on the left we can see one input frame, where the moving UAV is even difficult to perceive with human eyes. On the right is the result of detection. Three moving objects have been identified. They are the UAV as well as two persons moving behind the tree. It is obvious from this example that our algorithm is capable of detecting moving objects regardless of their speed and distance.

The purpose of performance evaluation with sequence 2 is to find how our algorithm will behave in case the size of the object is very small compared to the visual field of the camera. All together there are 80 frames, with the UAV moving all the time in the scene.

Shown in Fig. 4 are examples of detection results achieved from the third video sequence. The moving object is a single robot car driving on the road. There are 303 frames. Each frame has a single moving object in it.

Performance has been evaluated using all image pairs in the three video sequences. The average detection accuracy is 83.3%. This means, among all those flow vectors detected, 83.3% of them have been correctly classified as coming from either static background or moving objects. The false alarm rate of moving object detection is below 5%. As mentioned already, both video sequence 1 and 2 are quite challenging. Particularly for the images taken in sequence 1,



Fig. 3. Detection of small moving objects in sequence 2.

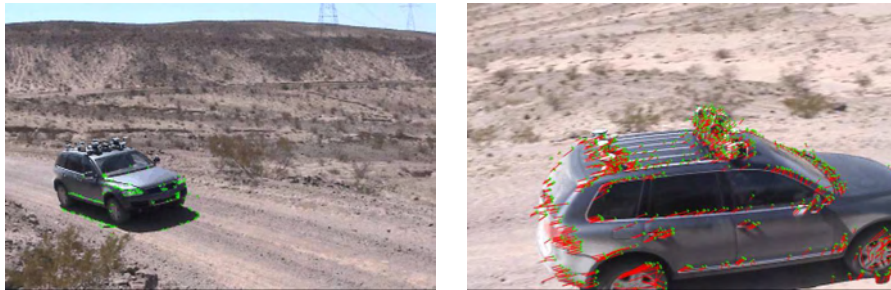


Fig. 4. Detection of a driving car in sequence 3.

there are strong lighting changes due to afternoon sunshine. Also transmission errors occur frequently, since the images are sent online via radio link from the UAV to the ground station laptop, on which our algorithm is running. Another reason lies in the fact that the moving objects are not rigid. A person's motion can be difficult to detect if he or she only changes pose slightly. Further factors which have aggravated the situation are shadows and the swinging of trees due to wind. Considering that the dynamic objects are either small or moving at far distances and that we use only the last and current frame for real-time detection of moving objects in the current frame, the results are quite encouraging.

VI. CONCLUSION

This paper presents a new approach for the detection of objects maneuvering in the visual field of a monocular camera navigating in the real world. Using the approach we have proposed, we are able to establish point correspondences between image pairs despite noisy situations where both independent motion and outliers occur. We do not make any assumption about the motion of the camera and the objects in the scene. The constraint in the form of a similarity transform is simple to implement and requires no camera calibration. Once the transform parameters have been estimated using a linear least-squares method, independent motion and outliers are identified based on how well they agree with the motion constraint. The process of motion separation and clustering is completely automated. Moving objects are identified by analyzing the underlying motion characteristics of each scene point probabilistically. Performance has been evaluated using several challenging video sequences captured

in outdoor environments. We believe that the detection rate can be improved by taking into account the past history of independent motion and combining motion detection and tracking in subsequent frames.

REFERENCES

- [1] C. Yuan, F. Recktenwald, and H. A. Mallot, "Visual steering of UAV in unknown environments," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2009)*, 2009, pp. 3906–3911.
- [2] A. Argyros, M. Lourakis, P. Trahanias, and S. Orphanoudakis, "Fast visual detection of changes in 3d motion," in *IAPR Workshop on Machine Vision Application (MVA'96)*, 1996.
- [3] J. Clarke and A. Zisserman, "Detecting and tracking of independent motion," *Image and Vision Computing*, vol. 14, no. 8, pp. 565–572, 1996.
- [4] H. Sawhney and S. Ayer, "Compact representations of videos through dominant and multiple motion estimation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 18, no. 814–830, pp. 451–467, 1996.
- [5] K. Patwardhan, G. Sapiro, and V. Morellas, "Robust foreground detection in video using pixel layers," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, pp. 746–751, 2008.
- [6] M. Irani and P. Anadan, "A unified approach to moving object detection in 2D and 3D scenes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 6, pp. 577–589, 1998.
- [7] M. Lourakis, A. Argyros, and S. Orphanoudakis, "Independent 3d motion detection using residual parallax normal flow fields," in *Int. Conf. on Computer Vision (ICCV'98)*, 1998, pp. 1012–1017.
- [8] P. Torr, A. Zissermann, and D. Murray, "Motion clustering using the trilinear constraint over three views," in *Workshop on Geometrical Modeling and Invariants for Computer vision*, 1995.
- [9] R. Hartley and R. Vidal, "The multibody trifocal tensor: motion segmentation from 3 perspective views," in *Int. Conf. Computer Vision and Pattern Recognition (CVPR 2004)*, 2004, pp. 769–775.
- [10] T. Kanade and B. Lucas, "An iterative image registration technique with an application to stereo vision," in *Int. Joint Conf. Artificial Intelligence (IJCAI'81)*, 1981.
- [11] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 4, pp. 376–380, 1991.



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Studying of WiFi range-only sensor and its application to localization and mapping systems

F. Herranz, M. Ocaña, L. M. Bergasa, M. A. Sotelo, D. F. Llorca,
N. Hernández, A. Llamazares and C. Fernández

Department of Electronics, University of Alcalá, Madrid (Spain)

Email: {fherranz,mocana,bergasa,sotelo,llorca,nhernandez,allamazares,cfernandez}@depeca.uah.es

Abstract—The goal of this paper is to study a noisy WiFi range-only sensor and its application in the development of localization and mapping systems. Moreover, the paper shows several localization and mapping techniques to be compared. These techniques have been applied successfully with other technologies, like ultra-wide band (UWB), but we demonstrate that even using a much more noisier sensor these systems can be applied correctly. We use two trilateration techniques and a particle filter to develop the localization and mapping systems based on the range-only sensor. Some experimental results and conclusions are presented.

I. INTRODUCTION

For most outdoor applications, i.e. surveillance tasks or vehicle navigation systems, Global Positioning System (GPS) [1] provide enough accuracy. On the contrary, when GPS receiver is in urban environments with high buildings or trees, the signal can suffer multipath fading or even Line-Of-Sight (LOS) blockage. In addition, it is important to remark that GPS signal is not strong enough to penetrate inside buildings, then this problem discards this technique to use it like an indoor localization system.

Vehicle navigation systems use a combination of a previous map with localization information to guide the vehicle through a mesh of connected ways. Maps are usually obtained in a semi-autonomous way process known as mapping [2]. Mapping is based on sensor observations which extract main features of the environment and allow to represent them into a topological or metric map.

Autonomous localization and mapping are two problems with similar features. It is not possible to built a map if the localization process does not work well, and it is impossible to locate a device with high precision without an accurate map. The SLAM(Simultaneous Localization And Mapping) techniques [3] [4] are used to solve these problems simultaneously, because the uncertainty of both processes can be reduced by doing localization and mapping at the same time .

Several systems for localization and mapping have been proposed and successfully deployed for indoor environments. These systems are based on: infrared sensors [5], computer vision [6], ultrasonic sensors [7], laser [8] or radio frequency (RF) [9] [10] [11] [4]. Within the last group we can find localization systems that use WiFi and Ultra Wide Band (UWB) signal level (SL). In order to estimate the vehicle or map feature location, these systems measure the signal strength and then apply a deterministic (i.e. trilateration) or probabilistic (i.e. particle filter) algorithm to infer the estimated position.

In addition, these techniques can be used in the same way in outdoor environments.

While the UWB systems achieve a high accuracy in both systems (localization and mapping), by mean of adding UWB reference beacons in the environment, WiFi technology uses 802.11b/g network infrastructure to estimate a device position without using additional hardware. Unfortunately, signal propagation is affected by reflection, refraction and diffraction in indoor environments. This effect, known as multipath effect, turns the received SL into a complex function of the distance. To solve this problem, several localization systems use a previous map and then, in the estimation phase, the received signal measure from each Access Point (AP) is compared with the map to obtain the estimated position [12] [13] [14]. This last technique is not recommended when the environment is dynamic or when its size increases.

In this work, we use the combination of the WiFi signal measure and a propagation model to obtain a range-only sensor that can be used both indoor and outdoor. We compare two deterministic and one probabilistic techniques to obtain the accuracy of all of them. These techniques are used in the same way for localization and mapping with slightly modifications. This work represents a previous step before obtaining a WiFi range-only SLAM system.

The rest of the paper is organized as follows: section 2 shows propagation models and WiFi signal variations; section 3 shows the localization with propagation model techniques; section 4 shows the mapping process; section 5 describes the results obtained by WiFi localization and mapping systems; and finally, section 6 shows some conclusions and future works.

II. WiFi RANGE-ONLY SENSOR

This section provides an introduction about the WiFi signal measure and its application as a range-only sensor. It is important to highlight that WiFi technology works at 2.4Ghz, a closer frequency to water resonant one, then it can be affected by several variations.

In a previous work [15] authors have throughly studied the main variations that affect to WiFi signal. We identified five main variations that can appear when working with robots. Among this five ones, there are three main variations to take into account when we want to develop WiFi range-only sensor localization and mapping systems:

- **Temporal variations:** when the robot is standing at a fixed position, the signal strength measure can vary over time. SL variations can be up to 2 dBm. These variations are usually due to changes in the physical environment such as people in movement.
- **Small-scale variations:** these variations occur when the robot moves in a small distance, under the wavelength λ . As a result, there are significant changes in the average received SL. For the 802.11b networks working at the 2.4 GHz range, λ is 12.5 cm. This kind of variations are generated by multipath effect. Small-scale variations introduce a high uncertainty in the system. These variations make difficult to estimate the device position because they can be up to 10 dBm for positions around the same location.
- **Large-scale variations:** signal strength varies over a long distance due to attenuation of the RF signal [16]. Large-scale variations can be used to estimate the distance between the robot and reference positions (APs locations).

A propagation model [17] is an empirical mathematical formulation for the characterization of radio wave propagation as a function of frequency, distance and other conditions. A single model is usually developed to predict the behavior of propagation for all similar links under similar constraints. Created with the goal of formalizing the way in which the radio waves are propagated from one place to another, such models typically predict the path loss through link or the effective coverage area of a transmitter.

In our system, we use a propagation model to estimate the distance between the APs and the robot through received SL. Our work is based on Hata-Okumura propagation model, which is studied in [18]. The equation (1) describes this model:

$$d = 10^{\frac{P_{TX} - P_{RX} + G_{TX} + G_{RX} - X_{\alpha} + 20 \log \lambda - 20 \log 4\pi}{10n}} \quad (1)$$

Where:

- d : is the distance between transceiver and receiver.
- P_{TX} and P_{RX} : are the transceiver and the receiver power (dBm).
- G_{TX} and G_{RX} : are the transceiver and receiver antenna gain (dBi).
- X_{α} : represents the error. It is a normal random variable with standard deviation α .
- λ : is the wavelength (12.5 cm).
- n : denotes influence of walls and other obstacles. In outdoors environments with LOS it is defined in the range from 2 to 3. In [19], the authors determine that the variable n must be approximately 2 in outdoor environments.

III. LOCALIZATION WITH RANGE-ONLY SENSORS

Localization is the technique that estimates the position of a mobile device using reference positions and the distance provided by the range-only sensor. In this section we describe the three techniques that we have compared.

A. Spherical trilateration

This technique estimates the robot position using the APs positions and the distances between the mobile and the APs. The algorithm is based on the next constraints:

- The n APs positions are known, and are placed in the coordinates $(x_1, y_1, z_1), (x_2, y_2, z_2), \dots, (x_n, y_n, z_n)$.
- The robot position is defined as (x_r, y_r, z_r) , and it is the position to estimate by the algorithm.
- The distances between the robot and each AP are known r_1, r_2, \dots, r_n .

The trilateration elements are showed in Figure 1.

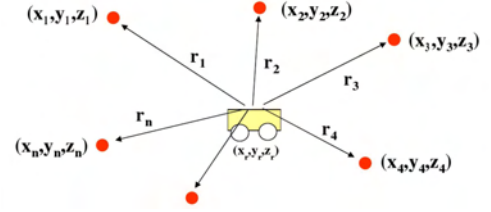


Fig. 1. Trilateration elements

The algorithm is based on these constraints to estimate the robot position using the equations (2), (3), (4) and (5).

$$\begin{aligned} r_i^2 &= (x_r - x_i)^2 + (y_r - y_i)^2 + (z_r - z_i)^2 \Rightarrow \\ x_r^2 + y_r^2 + z_r^2 + x_i^2 + y_i^2 + z_i^2 - 2x_r x_i - 2y_r y_i - 2z_r z_i - r_i^2 &= 0 \\ \forall i &= 1, 2, \dots, n \end{aligned} \quad (2)$$

Where t and S_i^2 are obtained as shown in (3):

$$\begin{aligned} t &= x_r^2 + y_r^2 + z_r^2 \\ S_i^2 &= x_i^2 + y_i^2 + z_i^2 \end{aligned} \quad (3)$$

It is possible to show the equations using the matrix way:

$$AX = B \quad (4)$$

$$\begin{aligned} X &= \begin{pmatrix} x_r \\ y_r \\ z_r \\ t \end{pmatrix} \\ A &= \begin{pmatrix} 2x_1 & 2y_1 & 2z_1 & -1 \\ 2x_2 & 2y_2 & 2z_2 & -1 \\ \dots & \dots & \dots & \dots \\ 2x_n & 2y_n & 2z_n & -1 \end{pmatrix} \\ B &= \begin{pmatrix} S_1^2 - r_1^2 \\ S_2^2 - r_2^2 \\ \dots \\ S_n^2 - r_n^2 \end{pmatrix} \end{aligned} \quad (5)$$

B. Spherical trilateration. Gauss-Newton algorithm

This method is based on the same elements than the previous trilateration (Figure 1). Moreover, a random position is used as initial one to estimate the robot position $emp = (\hat{x}_r, \hat{y}_r, \hat{z}_r)$.

The distance between the robot and the AP_i is defined according to equation (6).

$$r_i = \sqrt{(x_i - x_r)^2 + (y_i - y_r)^2 + (z_i - z_r)^2} \quad i = 1 \dots n \quad (6)$$

Now, we can define the distance between the AP_i and the emp (7).

$$\hat{r}_i = \sqrt{(x_i - \hat{x}_r)^2 + (y_i - \hat{y}_r)^2 + (z_i - \hat{z}_r)^2} \quad i = 1 \dots n \quad (7)$$

This method is based on equations (6) and (7) and Gauss-Newton algorithm. This method is used to solve non-linear least squares problems like this. It makes possible to minimize a sum of squared function values through an iterative way (equation (8)).

$$F(\hat{x}_r, \hat{y}_r, \hat{z}_r) = \sum_{i=1}^n (\hat{r}_i - r_i)^2 = \sum_{i=1}^n [f_i(\hat{x}_r, \hat{y}_r, \hat{z}_r)]^2 \quad (8)$$

Where f_i is obtained as shown (9):

$$f_i = \sqrt{(x_i - \hat{x}_r)^2 + (y_i - \hat{y}_r)^2 + (z_i - \hat{z}_r)^2} - r_i \quad (9)$$

Deriving the equation (9) respect to $(\hat{x}_r, \hat{y}_r, \hat{z}_r)$ it is possible to obtain the equation (10).

$$\begin{aligned} \frac{\partial F}{\partial \hat{x}_r} &= 2 \sum_{i=1}^n f_i \frac{\partial f_i}{\partial \hat{x}_r}; \\ \frac{\partial F}{\partial \hat{y}_r} &= 2 \sum_{i=1}^n f_i \frac{\partial f_i}{\partial \hat{y}_r}; \\ \frac{\partial F}{\partial \hat{z}_r} &= 2 \sum_{i=1}^n f_i \frac{\partial f_i}{\partial \hat{z}_r}; \end{aligned} \quad (10)$$

Equation (10) can be showed using the matrix way, $A \cdot \Delta X = B$ (equation (11)).

$$\begin{aligned} \Delta X &= \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} \\ A &= \begin{pmatrix} \frac{(\hat{x}_r - x_1)}{\hat{r}_1} & \frac{(\hat{y}_r - y_1)}{\hat{r}_1} & \frac{(\hat{z}_r - z_1)}{\hat{r}_1} \\ \frac{(\hat{x}_r - x_2)}{\hat{r}_2} & \frac{(\hat{y}_r - y_2)}{\hat{r}_2} & \frac{(\hat{z}_r - z_2)}{\hat{r}_2} \\ \vdots & \vdots & \vdots \\ \frac{(\hat{x}_r - x_n)}{\hat{r}_n} & \frac{(\hat{y}_r - y_n)}{\hat{r}_n} & \frac{(\hat{z}_r - z_n)}{\hat{r}_n} \end{pmatrix} \\ B &= \begin{pmatrix} (\hat{r}_1 - r_1) \\ (\hat{r}_2 - r_2) \\ \vdots \\ (\hat{r}_n - r_n) \end{pmatrix} \end{aligned} \quad (11)$$

The system can be solved by least squares and it is possible to obtain the algorithm increases according to equation (12).

$$\Delta X = (A^T A)^{-1} A^T B \quad (12)$$

Finally, the estimated robot position is updated using the previous emp and the new increase (equation (13)).

$$emp_{k+1} = emp_k - \Delta X_k \quad (13)$$

The process continues running until the increases ΔX become acceptable by the system.

C. Particle filter

The particle filter is a sequential Monte Carlo algorithm, i.e. a sampling method to approximate a distribution that uses its temporal structure. A "particle representation" of distributions is used. In particular, we will be concerned with the distribution $P(X_{rt}|z_{0:t})$ where $X_{rt} = (x_{rt}, y_{rt}, \theta_{rt})$ is the observed robot state at time t , and $z_{0:t} = (r_1, r_2, \dots, r_n)$ is the sequence of observations from time 0 to time t . The transition and sensor models, $P(X_{rt}|z_{0:t})$ are represented using a collection of N weighted samples or particles, $\{X_{rt}^{(i)}, \pi_t^{(i)}\}_{i=1}^N$ where $\pi_t^{(i)}$ is the weight of particle $X_{rt}^{(i)}$ (equation (14)).

$$P(X_{rt}|z_{0:t}) \approx \sum_i \pi_{t-1} \delta(X_{rt} - X_{rt-1}^{(i)}) \quad (14)$$

The particles are propagated using the movement model $p(X_{rt}|X_{rt-1}, a_t)$ and the verisimilitude $P(z_t|X_{rt})$.

Firstly, the particles are uniformly distributed at the state space. Next, the particles are updated by the previous actions a_{t-1} , the actual observation z_t and the movement model. Finally, the updated particles are weighted, so the density probability function of the particles represents the estimated robot position.

IV. MAPPING WITH RANGE-ONLY SENSORS

Mapping is the process that makes possible to estimate the APs positions using the distance between them and the robot. First of all, to map the positions of the reference is needed to know the trajectory of the mobile, and then estimate the APs positions using this knowledge. This problem is similar to the localization one but with a different point of view, we suppose that the robot position is known and static at different steps, and then it seems like the APs are moving around it.

A. Spherical trilateration and Gauss-Newton Spherical trilateration

These algorithms are used on mapping in a similar way than trilateration algorithms are used on the localization. The main difference between both is the previous knowledge. Localization algorithms know the APs location and estimate the robot position, however, mapping algorithms know the robot trajectory and they estimate the APs position. Figure 2 shows the elements used to estimate the position of one AP.

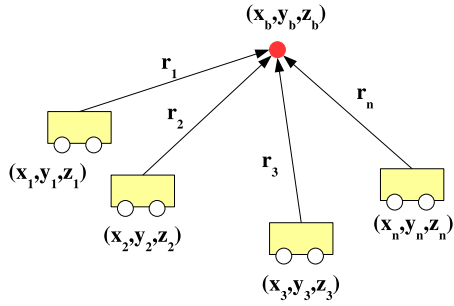


Fig. 2. Mapping elements

Finally, the algorithm is based on the equations (2), (3), (4) and (5), swapping reference (x_i, y_i, z_i) by beacon (x_b, y_b, z_b) positions.

Work [20] puts forward some situations that can make the system fails:

- When the reference positions are align.
- If the beacon position and the reference are on the same plane.
- If the reference position is over one reference.

According to these constraints, close positions of the robot trajectory are useless to map the APs position because they can be aligned. Moreover, it is recommended to design a "zigzag" path for the robot to avoid the alignment of the reference positions.

B. Particle filter

A particle filter like III-C is also used to map the APs. The main difference between both particle filters is the orientation. In this case, only a measurement of distance is obtained and then the AP can be everywhere within a circumference. To adapt the previous filter for mapping some modifications have been performed. These modifications are:

- Measurement vectors Z are the distances between the AP and the robot. The measures depend on the robot location.
- The verisimilitude $P(z_t|X_{rt})$ uses a vectorial space to represent the observations. Thus, we use a circumference equation based on a vectorial space. It is written in parametric form using trigonometric functions as is shown in equation (15). Then, 360 observations (one per angle ϕ) are generated. These observations form a circumference with radius equal to the distance between the AP and the robot.

$$P = X_{r0} + r(\cos \phi, \sin \phi) \quad (15)$$

Where:

- X_{r0} is the actual robot position.
- r is the radius or the distance between the AP and the robot.
- ϕ is the angle.
- P are the observed AP coordinates (x_b, y_b) .

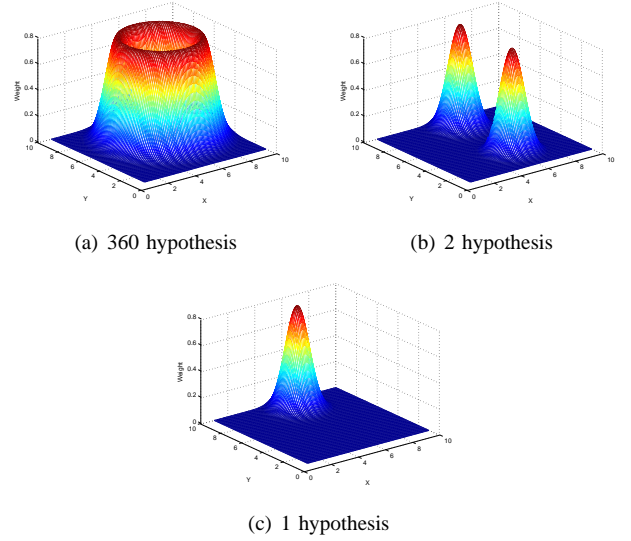


Fig. 3. Mapping with particle filter

Figure 3 shows the particle filter process, at the beginning there are 360 possible positions (one per angle) where the AP can be. Then, the possible positions are less and usually there are only 2 possible positions: the real AP position and the "mirror" one. Finally, only one hypothesis is followed and this position usually corresponds with the real AP position.

It is important to highlight that this algorithm does not need to collect a high number of samples to estimate the AP position. It is an online process and the accuracy is improving when the time is increasing.

V. IMPLEMENTATION AND RESULTS

This section describes some implementation features and the experimental results obtained with the designed tests.

A. Test-Bed Environment

The environment to test the localization and mapping systems is established outdoor and close to the Polytechnic School at the University of Alcalá (UAH).

The environment dimensions are approximately 20x20 metres. Moreover, three APs are used, these APs are located at coordinates (x, y, z) $(5.35, -2.36, 1.70)$, $(14, -2.36, 1.67)$, $(15.10, 7.4, 1.61)$. The WiFi antenna is placed at the mobile robot at 0.71 metres height. The robot trajectory is a pseudo-rectangle, this path is showed in Figure 4. The localization process in this work is calculated in 3D, and it has been necessary to convert the measurements from 3D to 2D to simplify the problem.

The tests have been performed with a laptop using an Orinoco Gold PCMCIA card, Linux Kubuntu 8.04, Wireless Tools v29 and Matlab 2008a. Signal level measure is obtained by the WiFi interface installed in the laptop. This interface scans the APs close to the device. Samples are got at 4 Hz, which is the highest frequency that the interface supports.

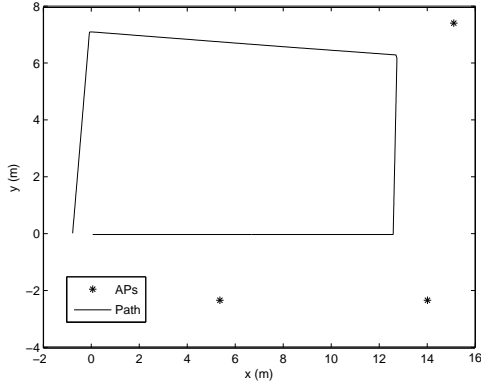


Fig. 4. Real test environment

TABLE I
WiFi RANGE ONLY SENSOR SAMPLES

d (m)	2	4	8	12	16	20
μ_{SL} (dBm)	-44.40	-53.25	-61.44	-66.00	-74.24	-78.04
σ_{SL} (dBm)	-3	-5	-5	-5	-7	-7

B. WiFi range only sensor

To study the WiFi range only sensor a real test has been performed, which consists of measuring the signal level at different distances. The collected samples are processed calculating the mean and the variance of them. Table I shows the mean and the variance values of the samples for each distance. The mean values shows how the SL decreases with the distance, however, the variance increases with the distance. A high variance in the samples, produced by the noise, makes difficult to estimate a distance from a SL value.

Based on Table I values, and paying attention to large-scale variations it is possible to estimate a propagation model. Figure 5 shows an estimated propagation model and a comparison with Hata-Okumura model (HOM) using a set of training data. The propagation model has been estimated obtaining the mean SL of each distance and fitting a polynomial function using a approximation by least squares. The estimated model (EM) obtains better results than HOM because it fits perfectly the training data, however, HOM is a generic model and it can be adapted to new environments.

Table II shows the error for each distance. In both models the error tends to increase with the distance. This error is specially important in the HOM at 12 metres, it is due to a high noise in the training samples. HOM does not fit well to the training samples when they contain a huge noise, however, the EM obtains better results in this case. It is important to remark that in other cases the samples can contain more or less noise and the EM will not obtain good results. Then, the EM is a better choice in an under control environment, however, the HOM is more general and adaptable to new environments.

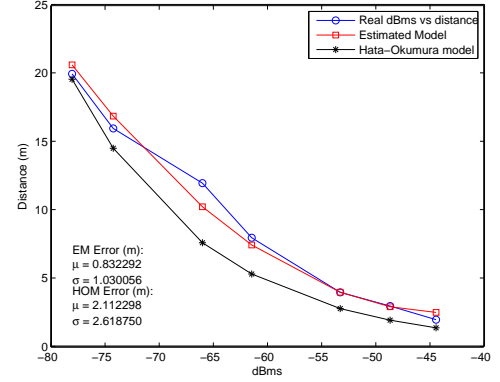


Fig. 5. Large-scale variations on propagation model

TABLE II
PROPAGATION MODELS ERROR

Distance (m)	2	3	4	8	12	16	20
EM error (m)	0.28	0.01	0.00	0.29	3.01	0.83	0.44
HOM error (m)	0.35	1.08	1.41	7.01	19.08	2.09	0.16

C. Localization results

Localization results are showed in Table III. Trilateration algorithm gets a mean error of 9.04 metres, which is the highest error of the three algorithms. On the other hand, Gauss-Newton obtains better results, the mean error is 6.26 metres and the error is more constant than the trilateration one. Both algorithms estimate the robot position without previous information, which decreases the accuracy, but these can be used with a low computational cost. However, the particle filter uses the previous information to estimate the position to avoid high changes in the error. Then, it makes the particle filter the most accurate algorithm, it gets a mean error of 3.16 metres and a maximum error of 9.24 metres.

D. Mapping results

The mapping techniques obtain the following results: trilateration algorithm gets the smallest mean error, 9.7 metres, using 125 reference positions to estimate the AP position. With a lower number of 100 positions it is impossible to estimate the position.

Gauss-Newton algorithm uses 125 robot positions to obtain a mean error of 10.58 metres, this error is higher than the

TABLE III
LOCALIZATION ERROR

Method	Trilateration	Gauss-Newton	Particle filter
Mean (m)	9.04	6.26	3.16
Max (m)	23.48	22.13	9.24
Min (m)	0.56	0.1	0.31

TABLE IV
GAUSS-NEWTON ERROR

Num_pos	5	10	30	50	100	150
Mean (m)	15.92	15.80	15.49	13.48	14.68	9.16
Max (m)	27.10	36.01	41.47	25.40	16.82	9.40
Min (m)	1.99	6.40	5.09	6.13	11.23	8.92

TABLE V
PARTICLE FILTER ERROR

Particles	100	1000	1500	2500	3500	4000
Mean (m)	5.50	4.41	4.10	3.95	3.62	3.51

trilateration one. This method obtains better results in several situations but in some occasions it finds a local minimum and then it does not obtain the optimal solution. Moreover, Table IV shows that this method obtains good results using only 50 robot positions to estimate the AP location. Then, it is necessary to spend only 12 seconds to localize the AP position.

Both methods are affected by the robot trajectory, this problem was previously commented and we have obtained better results using other robot paths in simulation mode.

Finally, the particle filter has been tested varying the number of particles in 100 experiments. Table V shows the mean error obtained in 100 experiments, it has been obtained from the moment that filter converges to the real beacon position. Sometimes, the filter converges to the mirror position due to the high noise, this noise is approximately 10 dBm and it can introduce an error of 10 metres in the observation. Then, in several executions the observation can be near to the mirror position. Results show an error that tends to decrease. The smallest error, 3.51 metres, was obtained using 4000 particles.

VI. CONCLUSIONS AND FUTURE WORKS

In this work has been presented a WiFi range-only sensor and its application to localization and mapping system. In the first time, we have analyzed the main variations of this sensor and we have proposed to use a propagation model to obtain the distance between the robot and a certain reference positions (APs). Three different techniques have been compared to localization and mapping process. Each technique has been configured and performed to obtain the best possible accuracy. We have obtained an accuracy of 3.16 metres to localize the mobile and 3.51 metres to map the environment references. In the future, we have the intention of improving the accuracy of the localization and mapping systems using a WiFi range-only SLAM algorithm and a fusion with an Inertial Measurement Unit (IMU) to improve the movement model.

VII. ACKNOWLEDGEMENTS

This work has been funded by grant S2009/DPI-1559 (Robocity2030 II-CM Project) from the Science Department of Community of Madrid.

REFERENCES

- [1] P. Enge and P. Misra, "Special issue on gps: The global positioning system," in *Proc. of the IEEE*, vol. 87, no. 1, 1999, pp. 3–172.
- [2] S. Thrun, D. Fox, and W. Burgard, "A probabilistic approach to concurrent mapping and localization for mobile robots," in *Machine Learning*, 1998, pp. 253–271.
- [3] K. E. Bekris, M. Click, and E. E. Kavraki, "Evaluation of algorithms for bearing-only slam," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA'06)*, 2006, pp. 1937–1943.
- [4] J. L. Blanco, J. A. Fernandez-Madrigal, and J. Gonzalez, "Efficient probabilistic range-only slam," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'08)*, 2008, pp. 1017–1022.
- [5] R. Want, A. Hopper, V. Falco, and J. Gibbons, "The active badge location system," *ACM Transactions on Information Systems*, vol. 10, pp. 91–102, Jan. 1992.
- [6] J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer, "Multi-camera multi-person tracking for easy living," in *Proc. of 3rd IEEE International Workshop on Visual Surveillance*, 2002, pp. 3–10.
- [7] N. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location support system," in *Proc. of the 6th ACM MobiCom*, 2002, pp. 155–164.
- [8] R. Barber, M. Mata, M. Boada, J. Armingol, and M. Salichs, "A perception system based on laser information for mobile robot topologic navigation," in *Proc. of 28th Annual Conference of the IEEE Industrial Electronics Society*, 2002, pp. 2779–2784.
- [9] V. Matellán, J. M. Cañas, and O. Serrano, "Wifi localization methods for autonomous robots," *Robotica*, vol. 24, no. 4, pp. 455–461, 2006.
- [10] M. Ocaña, L. M. Bergasa, M. Á. Sotelo, R. Flores, E. López, and R. Barea, "Comparison of wifi map construction methods for wifi pomdp navigation systems," *Lecture Notes in Computer Science - Computer Aided Systems Theory*, vol. 4739, pp. 1216–1222, 2007.
- [11] D. B. Jourdan, J. J. Deyst, M. Z. Win, and N. Roy, "Monte carlo localization in dense multipath environments using uwb ranging," in *Proceedings of IEEE International Conference on Ultra-Wideband*, 2005, pp. 314–319.
- [12] A. Howard, S. Siddiqi, and G. Sukhatme, "An experimental study of localization using wireless ethernet," in *Proc. of the International Conference on Field and Service Robotics*, July 2003.
- [13] A. Ladd, K. Bekris, A. Rudys, G. Marceu, L. Kavraki, and D. Wallach, "Robotics-based location sensing using wireless ethernet," in *Proc. of the MOBICOM'02*, 2002.
- [14] M. Youssef, A. Agrawala, and A. Shankar, "Wlan location determination via clustering and probability distributions," in *Proc. of the IEEE PerCom 2003*, 2003.
- [15] M. A. Sotelo, M. Ocaña, L. M. Bergasa, R. Flores, M. Marrón, and M. A. García, "Low level controller for a pomdp based on wifi observations," *Robot. Auton. Syst.*, vol. 55, no. 2, pp. 132–145, 2007.
- [16] M. Youssef and A. Agrawala, "Small-scale compensation for wlan location determination systems," in *Proc. of the 2003 ACM workshop on wireless security*, 2003, pp. 11–20.
- [17] L. Liechty, E. Reifsnider, and G. Durgin, "Developing the best 2.4 ghz propagation model from active network measurements," in *VTC Fall*, 2007, pp. 894–896.
- [18] A. Kotanen, M. Hannikainen, H. Leppakoski, and T. Hamalainen, "Positioning with ieee 802.11b wireless lan," *Personal, Indoor and Mobile Radio Communications, 2003. PIMRC 2003. 14th IEEE Proceedings on*, vol. 3, pp. 2218–2222 vol.3, Sept. 2003.
- [19] A. Bose and C. H. Foh, "A practical path loss model for indoor wifi positioning enhancement," *Information, Communications & Signal Processing, 2007 6th International Conference on*, pp. 1–5, Dec. 2007.
- [20] J. O. Roa, A. R. Jiménez, F. Seco, J. C. Prieto, and J. Ealo, "Optimal placement of sensors for trilateration: Regular lattices vs meta-heuristic solutions," in *EUROCAST*, 2007, pp. 780–787.

Terrain Classification for Improving Traversability of Autonomous Vehicles

Jayoung Kim, Jonghwa Lee, Jihong Lee, *Member, IEEE*

Abstract—One of the requirements for autonomous vehicles on off-road is to move harmoniously in unstructured environments. It is an undeniable fact that such capacity of autonomous vehicles is the most important in an aspect considering mobility of the vehicle. So, many researchers use contact and/or non-contact methods to detect a terrain whether the vehicle can move on or not. In this paper we introduce an algorithm to classify terrains using visual information. As pre-processing, contrast enhancement technique is introduced to improve accurate rate of classification. Also, for conducting classification algorithm, training images are grouped as each material and Bayesian classification recognizes new images as each material using such material groups. Consequently, we can confirm the good performance of classification. Moreover, we can build Traversability map on which autonomous vehicles can predict whether to go or not to go through real friction coefficients which are measured by Load-Cell on surfaces of various terrains.

I. INTRODUCTION

IT is important that autonomous vehicles maintain good mobility performance on off-road terrain to carry out a mission like exploration. The mobility of a vehicle on off-road terrain is known to be strongly influenced by the interaction between the wheels of the vehicle and the terrain. Slip is the result of this complex interaction and, second to tip-over hazards, it is the most important factor in traversing slopes. Slip ratio on a terrain is closely related to friction coefficient of a surface because if friction coefficient on a terrain is high, slip ratio is low. So, to avoid slippery areas for the mobility of the vehicles, many researchers have tried to predict friction coefficient on a terrain using contact and/or non-contact methods for measurement of friction coefficient on a surface of a terrain.

The remainder of this paper is organized as follows. First, in chapter 2, algorithms for terrain classification using visual information is described. In chapter 3, pre-processing method is introduced for classifying terrains in detail. Chapter 4 shows performance of classification algorithm and the results of the experiments are presented with Traversability map by real friction coefficients in chapter 5.

This work was supported in part by Agency for Defence Development and by UTRC (Unmanned Technology Research Center).

Jayoung Kim is with the BK21 Mechatronics Group at Chungnam National University, Daejeon, Republic of Korea (e-mail: hgfdseys@hanmail.net).

Jonghwa Lee is with Mechatronics Group at Chungnam National University, Daejeon, Republic of Korea (e-mail: cnu-ljh@nate.com).

Jihong Lee is with the BK21 Mechatronics Group at Chungnam National University, Daejeon, Republic of Korea (phone: +82-42-821-6873; fax: +82-42-823-4919; e-mail:jihong@cnu.ac.kr).

Finally, conclusions and future work are provided in chapter 6.

II. ALGORITHMS FOR TERRAIN CLASSIFICATION

A. Characteristic Data Extraction in a Terrain

In this section we describe a method to extract characteristic data in a terrain with visual image. Characteristic data includes information of each material in a terrain. Also, such material information is used to make a group of each material (soil, small gravel, big gravel, and asphalt) and classify new terrain as one of groups later.

First step to extract characteristic data is dividing image into homogeneous regions which have homogeneous material. It is almost impossible to determine the material of one pixel. Because of this reason, we tried to divide the image into small region which have meaningful feature and extract that feature from each region for recognition. Hoem *et al.* used over-segmentation method of Felzenszwalb *et al.* for same purpose of us. This over-segmentation method generates good result in merging similar pixel to one segment, although the shape and size of segment is sensitive. However, because the purpose making super-pixel is just to extract material information from homogeneous region, the exact shape of that material does not matter.

Fig. 1 is result when we applied over-segmentation method of Felzenszwalb *et al.* We used Felzenszwalb's source code. In the result image, the pixels of each segment have almost similar color. So, we can assume that each segment is composed of same material. Actually, there are some regions which are divided into different segments even if they are same material. However, it does not matter because we can predict they can be classified into same material at classifying step.



Fig. 1. Over-segmentation result (small gravel terrain).

As shown in Fig. 2, there is a characteristic data set of six materials (sky, soil, small gravel, big gravel, asphalt, and forest) in a segment of over-segmentation image. Data value of each material is calculated by comparing between materials using over-segmentation method. Thus, we can

confirm that a segment is recognized as S. gravel material because the data figure of S. gravel is the lowest in comparison with other materials.

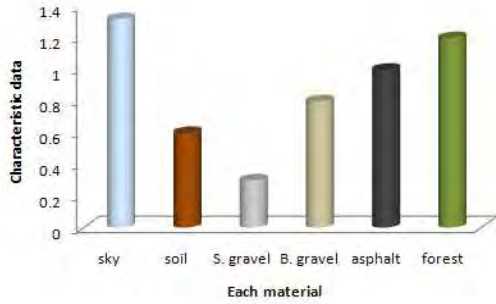


Fig. 2. Characteristic data in a segment of an image.

B. Grouping Method

This section introduces a method to group each material in terrains using characteristic data. First of all we assumed that input data (characteristic data sets) is Gaussian distribution. Considering such Gaussian distribution, representative values are estimated by equation (1) and (2).

$$\hat{\mu}_k = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i \quad (1)$$

$$\hat{\Sigma}_k = \frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i - \hat{\mu}_k)(\mathbf{x}_i - \hat{\mu}_k)^T \quad (2)$$

\mathbf{x}_i : Characteristic data sets

n : The number of segments recognized as same material

T : Transpose matrix

k : The number of groups

Here, $\hat{\mu}_k$ is mean value (4×1 matrix) and $\hat{\Sigma}_k$ is covariance (4×4 matrix) of k group. Consequently, groups of same materials which are decided using over-segmentation method are made by estimated $\hat{\mu}_k$ and $\hat{\Sigma}_k$.

C. Bayesian Classification

To classify new terrains into specified terrains grouped, we use Bayesian theory. Bayesian theory is a fundamental statistical approach to the problem of pattern classification. Thus, we classify new terrains as analyzing pattern of characteristic data of new terrains into specified terrains grouped by using Bayesian classification.

To compute posterior probabilities $\mathbf{P}(\omega_k | \mathbf{X})$ is the heart of Bayesian classification. Here, posterior probabilities $\mathbf{P}(\omega_k | \mathbf{X})$ represent probabilities for classifying segments of new terrains into specified materials groups.

Bayes formula allows us to compute these probabilities from the prior probabilities $\mathbf{P}(\omega_k)$ and the class-conditional densities $\mathbf{P}(\mathbf{X} | \omega_k)$. Here, prior probabilities $\mathbf{P}(\omega_k)$ are the same value as characteristic data of each material in one

segment, because characteristic data is probability of each material in one segment as mentioned above.

The class-conditional densities $\mathbf{P}(\mathbf{X} | \omega_k)$ are calculated by equation (3).

$$\mathbf{p}_k(\mathbf{X} | \omega_k) = \frac{1}{\sqrt{2\pi}^d \sqrt{\det(\Sigma_k)}} \exp\left\{-\frac{1}{2}(\mathbf{X} - \mu_k)^T \Sigma_k^{-1} (\mathbf{X} - \mu_k)\right\} \quad (3)$$

k : The number of groups

T : Transpose matrix

d : Dimension of input data

\mathbf{X} : New terrain information

ω_k : Identifier of k group

If we again let \mathbf{D} denote the set of samples, then we can emphasize the role of the samples by saying that our goal is to compute the posterior probabilities $\mathbf{P}(\omega_k | \mathbf{X}, \mathbf{D})$. From these probabilities we can obtain the Bayes classifier.

Given the sample \mathbf{D} , Bayes formula then becomes

$$\mathbf{P}(\omega_k | \mathbf{X}, \mathbf{D}) = \frac{\mathbf{p}(\mathbf{X} | \omega_k, \mathbf{D}) \mathbf{P}(\omega_k | \mathbf{D})}{\sum_{j=1}^c \mathbf{p}(\mathbf{X} | \omega_j, \mathbf{D}) \mathbf{P}(\omega_j | \mathbf{D})} \quad (4)$$

c : The number of groups

\mathbf{D} : The set of samples (each data of existing groups)

As this equation suggests, we can use the information provided by the training samples to help determine both the class-conditional densities and the prior probabilities. Here, we can separate the training samples by class into c subsets $\mathbf{D}_1, \dots, \mathbf{D}_c$, with the samples in \mathbf{D}_i belonging to ω_k . If we again let class denote an existing group, then a number is given to an existing group by identifier ω_k . When existing or new terrain information, just characteristic data, is inputted to equation (4), posterior probabilities $\mathbf{P}(\omega_k | \mathbf{X}, \mathbf{D})$ is computed by calculating the class-conditional densities $\mathbf{P}(\mathbf{X} | \omega_k, \mathbf{D})$ and prior probabilities $\mathbf{P}(\omega_k | \mathbf{D})$ as comparing existing or new terrain information with existing groups. Thus, characteristic data of segments of terrains is classified into specified material groups as posterior probabilities $\mathbf{P}(\omega_k | \mathbf{X}, \mathbf{D})$ by identifier ω_k .

III. PRE-PROCESSING

We use pre-processing method to improve classification performance. Basically, we use contrast enhancement techniques as pre-processing method.

In Fig. 3 we can know that original image of B. gravel terrain is rather blurry as we see. Actually, if autonomous vehicle is moving on this terrain, it is very difficult that the vehicle exactly recognize whether a surface of this terrain is composed of any materials using this image due to classification errors. So, to compensate this error rate, the

images is enhanced by contrasting a bright color with a dark color in order that two sides are so far apart. Fig. 4 is modified image by contrast enhancement method. We can confirm that modified image is more vivid and brighter than original image.



Fig. 3. Original image (B. gravel terrain).



Fig. 4. Modified image by contrast enhancement.

We applied contrast enhancement algorithm widely known to pre-processing for enhancement. Basic that algorithm turns images into other images by selecting passively contrast values. The range of basic contrast values is from zero (0) to two (2). When contrast value is the zero, the color of entire image is dark. On the contrary, when contrast value is the two, the color of entire image is white. So, for selecting optimum value at which accurate rate of classification is the highest, we tried to make an algorithm which selects automatically optimum value considering accurate rates of classification.

Fig. 5 is flow chart of such an algorithm to select optimum value. As mentioned above, an image is enhanced by specific value in the range of contrast value using image processing. As the next step, characteristic data in the modified image are extracted by over-segmentation method. And then covariance of data which is a measure of how much variables change together in probability theory and statistics is calculated and is stored every contrast values. As a final step, minimum covariance is decided by MIN algorithm. Finally, a value which has minimum covariance is selected as optimum value. The more covariance is close to minimum, the greater

materials are recognized exactly because segments in an image are smaller than before and classification algorithm can compare similar degree between segments in detail.

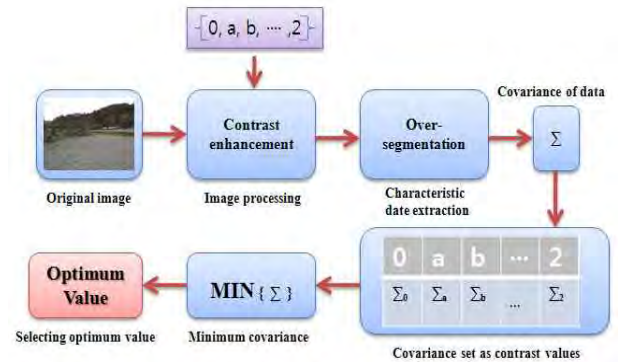


Fig. 5. Flow chart of an algorithm to select optimum value.

IV. PERFORMANCE TEST

A. Review of Entire Classification Algorithm

Fig. 6 shows the entire classification algorithm. First above all, training steps in left side are a process for autonomous vehicles to get information of various terrains like being intelligent as a person gets much knowledge. As a first step of training step, training images which are made with off-line are transmitted to central computer. Training images collected with off-line are separated as small parts. Its small parts are learning images. Learning images are entered to over-segmentation algorithm and each learning image is used as a standard of comparison with other entered images to extract characteristic data of a terrain. As a final step of training step, each material information of training images are grouped using the grouping method which is introduced in chapter II-B.

Classification steps in right side are a process for autonomous vehicles to classify new terrain, not the terrain trained, into specific material. This algorithm is operated with on-line. Once new images through over-segmentation method via pre-processing are entered to Bayesian classification step, new images are classified into material groups which are made in training steps by Bayesian classification. Actually, the number of material groups is determined by the number of learning images. Currently, we have used materials of six categories (sky, soil, small gravel, big gravel, asphalt, and forest) as shown in Fig. 2. However, sky and forest materials are not grouped because sky and forest are the region that a vehicle can't move on. Thus, we use four material groups (soil, S. gravel, B. gravel, and asphalt) in classification algorithm. Consequently, new images are classified into specific groups by Bayesian classification and then Traversability map where a vehicle can move on a surface of a terrain without disturbance is made.

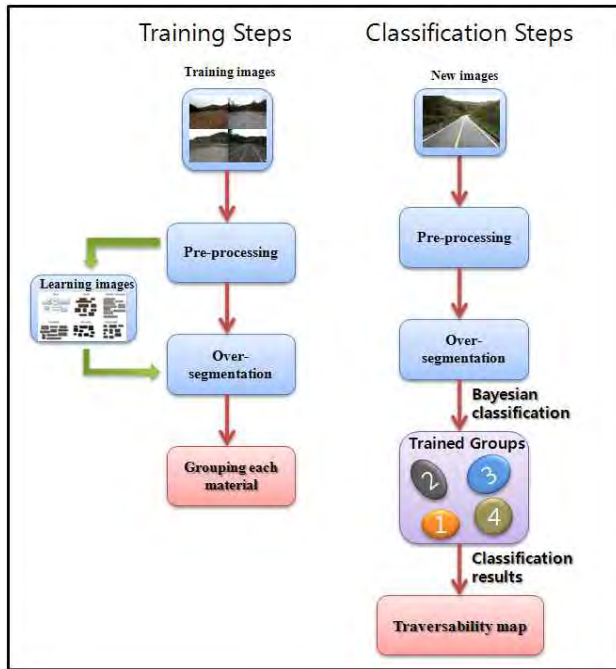


Fig. 6. Flow chart for grouping and classification

B. The Result of Performance Test

In this section we verify the algorithm as accurate rate which is a result to classify a terrain. We use one of training images which are grouped in training steps to confirm accurate rate of Bayesian classification. From a common-sense point of view, the result of classification will naturally be almost exact.

Fig. 7 shows classification result of training image (B. gravel terrain). This training image is composed of big gravel materials on a surface. Thus, we can know that the big gravel area is widely classified. We separate only big gravel region to confirm quantitatively accurate rate of classification result like Fig. 8. The area of big gravel materials is divided by same size mask. Accurate rate of classification is calculated by counting the number of each pixel of the area separated. Consequently, accurate rate of Bayesian classification is 95.05%. It is reasonably good result. As the result, we can confirm that the performance of Bayesian classification is great.

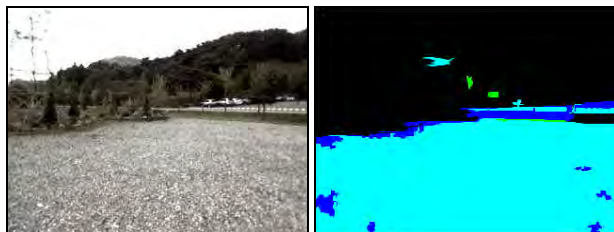


Fig. 7. Classification result of training image (B. gravel terrain).

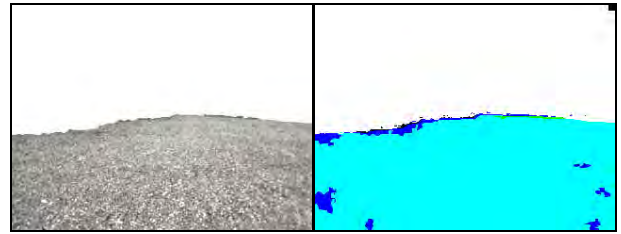


Fig. 8. Separation of only big gravel.

V. EXPERIMENTAL RESULT

A. Friction Coefficient Experiment

We make an experiment on measurement of friction coefficient. We used Load Cell for this experiment. Load Cell is an electronic device that is used to convert a force into an electrical signal. Other equipment is traction car and experimental vehicle. Fig. 9 shows experimental equipment and processes. Traction car pulls experimental vehicle. At the instance when the experimental vehicle starts to slip, we collect the data of Load Cell. The collected data include the information of friction between wheel and ground.



Fig. 9. Experiment to measure friction coefficient

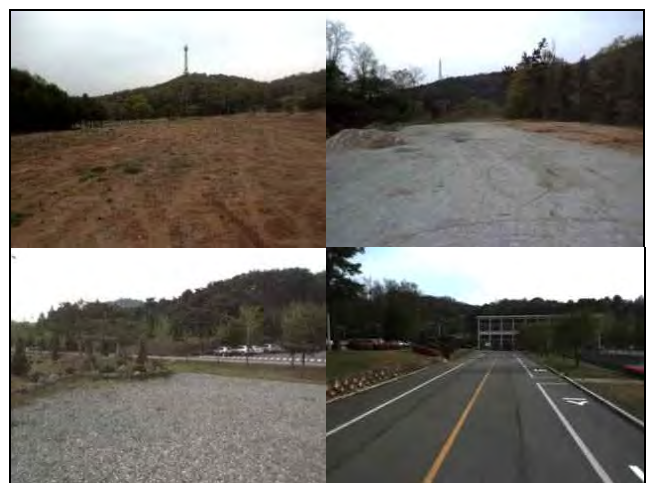


Fig. 10. Experiment terrains. Soil terrain (top left), Small gravel terrain (top right), Big gravel terrain (bottom left), and Asphalt terrain (bottom right).

The experiment is performed in four terrains. As shown in Fig. 10, each terrain is soil, S. gravel, B. gravel, and asphalt terrain. Table I shows friction coefficients calculated on a surface of each terrain. As expected, the friction coefficient of asphalt is the highest among them. Such friction coefficients of each terrain are used for autonomous vehicles to avoid a slippery area and to change kinematic or dynamic parameters to move quickly and slowly for vehicles which have known features on a surface of a terrain. In this paper, we used real friction coefficients to predict whether a surface of a terrain is slippery or not.

B. Classification Result of New Terrain

In this section we explain classification result and traversability map of new image. Fig. 11 is new terrain which is not grouped by training steps. As shown in Fig. 12, green is asphalt materials and sky-blue is big gravel materials. Finally, blue is soil materials. Here, dark color presents sky and forest materials. As mentioned above, sky and forest is a terrain which autonomous vehicles can't move on. So, once a segment is recognized as sky and forest materials, the segment is displayed as dark color.

In chapter V-A, we measured friction coefficient of each material. Therefore, we applied friction coefficient to the result of classification in Fig. 12. Fig. 13 shows Traversability map which autonomous vehicles can predict slippery area on. And the vehicles can plan a path efficiently by predicting a condition of a surface in such a terrain. In Fig. 13 a color is displayed on Traversability map among magnitude of friction coefficients.

TABLE I
FRICTION COEFFICIENT ON EXPERIMENT TERRAINS

TERRAIN	Friction Coefficient [μ]
Soil Terrain	0.9
Small Gravel Terrain	0.74
Gravel Terrain	0.82
Asphalt Terrain	1.2



Fig. 11. New terrain (asphalt terrain)

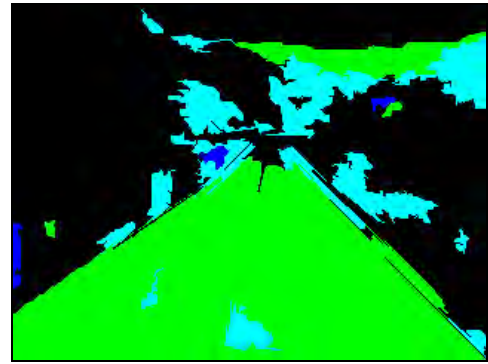


Fig. 12. Bayesian classification result.

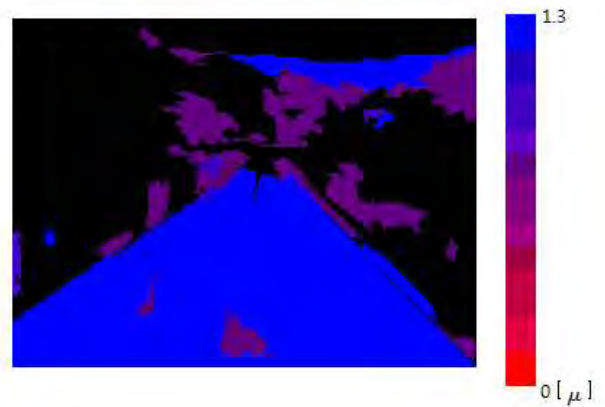


Fig. 13. Traversability map.

VI. CONCLUSION

In this paper we propose an algorithm to classify terrains into each material. As pre-processing, contrast enhancement technique is introduced to improve accurate rate of classification. Also, for conducting classification algorithm, training images are grouped as each material and Bayesian classification recognizes new images, not training images, as each material using such material groups. Consequently, we can confirm the good performance of classification. Moreover, we can build Traversability map on which autonomous vehicles can predict whether to go or not to go through real friction coefficients which are measured by Load-Cell on surfaces of various terrains.

This algorithm can be useful for autonomous vehicles. By using this information, it can make a meaningful decision such as where to go and how to go.

REFERENCES

- [1] Christian Weiss, Hashem Tamini, and Andreas Zell, "A Combination of Vision- and Vibration-based Terrain Classification," IROS, pp.2204-2209, 2008
- [2] Karl Iagnemma, Shinwoo Kang, Hassan Shibl, Steven Dubowsky, "Online Terrain Parameter Estimation for Wheeled Mobile Robots With Application to Planetary Rovers," IEEE Transactions on Robotics and Automation, vol.20, no.5, pp.921-927, 2004
- [3] Pierre Lamon and Roland Siegwart, "Wheel Torque Control in Rough Terrain-Modeling and Simulation," In Proceedings of the IEEE international conference on robotics and automation, pp.867-872, 2005
- [4] R. Manduchi, A. Castano, A. Talukder and L. Matthies, "Obstacle Detection and Terrain Classification for Autonomous Off-Road Navigation," Autonomous Robots 18, Springer, pp.81-102, 2005
- [5] Paul Jansen, Wannes van der Mark, Johan C. van den Heuvel, Frans C.A. Groen, "Colour based Off-Road Environment and Terrain Type Classification," IEEE Conference on Intelligent Transportation Systems, pp. 61-66, 2005
- [6] Morten Rufus Blas, Motilal Agrawal, Aravind Sundaresan and Kurt Konolige, "Fast Color/Texture Segmentation for Outdoor Robots," IROS, pp. 4078-4085, 2008
- [7] Ibrahim Halatci, Christopher A. Brooks, Karl Iagnemma, "Terrain Classification and Classifier Fusion for Planetary Exploration Rovers," In Proc. IEEE Aerospace Conference, USA, 2007
- [8] Anelia Angelova, Larry Matthies and Daniel Helmick, Pietro Perona, "Learning and Prediction of Slip from Visual Information," Journal of Field Robotics, vol. 24, no. 3, pp.205-231, 2007
- [9] T. Leung and J. Malik, "Representing and Recognizing the Visual Appearance of Materials Using the Three-dimensional Texton," IJCV vol. 43, no. 1, 2001
- [10] M. Varma and A. Zisserman, "Classifying images of materials: achieving viewpoint and illumination independence," ECCV, vol. 3, pp.255-271, 2002
- [11] P. Felzenszwalb and D. Huttenlocher, "Efficient Graph based Image Segmentation," IJCV, vol. 59, no. 2, pp.167-181, 2004
- [12] Hanbyul Joo and In-so Kweon, "Terrain Classification Using Texture Recognition for Autonomous Robot," URAI, 2007
- [13] Sethu Vijayakumar, Aaron D'Souza, and Stefan Schaal, "Incremental Online Learning in High Dimensions," Neural Computation, vol. 17, no. 12, pp.2602-2634, 2005
- [14] Richard O. Duda, Peter E. Hart and David G. Stork, "Pattern Classification," 2nd ed, Wiley, New York, 2001



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session V

Multi Robot Control & ITS

- **Keynote speaker: Rüdiger Dillman (Karlsruhe University, Germany)**
Title: Situation Assessment and Behaviour Decision Making of Cognitive Vehicles
- **Title: A global decentralized control strategy for urban vehicle platooning relying solely on monocular vision**
Authors: Pierre Avanzini, Benoit Thuilot, Eric Royer, Philippe Martinet
- **Title: Lyapunov Global Stability for a Reactive Mobile Robot Navigation in Presence of Obstacles**
Authors: Ahmed Benzerrouk, Lounis Adouane, Philippe Martinet



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session V

Keynote speaker: **Rüdiger Dillman (Karlsruhe University, Germany)**

Situation Assessment and Behaviour Decision Making of Cognitive Vehicles

Abstract: Driving an autonomous vehicle on urban and rural environment requires knowledge about the situation on the road. Knowledge about the intension of other vehicles and individuals on the road is required in order to classify the situation and to decide how to behave and to react. This paper addresses the problem of extracting information about the situative traffic environment of a vehicle from ist sensorial observations, its interpretation referencing situative knowledge and an estimation of it's further behaviour. This estimation requires understanding the intension of the other vehicles or agents and a predictive view of further traffic state evolvment. Because of uncomplete observation and uncertainties the estimation and sensor fusion process has an important role. With the help of learning methods in terms of learning from example the vehicle will be able to learn from ist observations which allows the estimation of dangerous situations and a predictive view of its environment which allows the continuation of driving. Furthermore it is necessary to make according the actual situation and drive intension behavioural decisions considering it's effects and results. Also here uncertainty has to be considered to enable predictive driving. A predictive behavioural decision process in combination with a learning process will be presented which allows to enhance the decision performance. A dynamic risk map is used to support algorithms for motion planning of the vehicle. Finally the vehicle should be able to execute maneuvers such as passing a crossing, lane changing, collision avoidance, overtaking and turning off, processing information about the actual situation and a prediction how it may evolve. The work to be reported is part of the collaborative research center SFB/TR 28 Cognitive Automobile which is sponsored by the German Research Agency DFG.

Biography: Ruediger Dillmann holds a chair on Industrial Applications of Informatics and Microsystems at the Faculty of Computer Science and Engineering at the Karlsruhe Institute of Technology – KIT and is spokesman of the Institute for Anthropomatics. He is president of FZI, a Research Center for Information Technology associated with the University of Karlsruhe. At FZI he is also director of the resarch group IDS which stands for interactive diagnosis and service systems. He received his M.S. and Ph.D. degrees from University of Karlsruhe in 1976 and 1980 respectively. From 1976 – 1986 he was research engineer at the Institute of Real-Time Processing and Robotics and worked on the design of autonomous mobile robot systems and research on adaptive and learning systems. From 1986 he was associate professor for Robotics and from 2000 full professor for Computer Science at the University of Karlsruhe. His research interests are in the field of humanoid robots, autonomous robot systems, interactive robot programming and learning by demonstration and observation of human activities and study and design of technical cognitive systems. Dr. Dillmann is the author and coauthor of over 500 technical publications, proceedings, editorials and books. His research interests include mobile autonomous robots, multimodal human-robot interaction and modelling of perception and adaptive controls for robots in varous application. He is coordinator of the collaborative research network SFB 588 on Humanoid Robots at the University of Karlsruhe. He is active member of the IEEE/RAS, GI and EURON and was General Chair of MFI 2001, Co-Chair of CLAWAR 2001, IAS-2000, Humanoids 2003, General Programme Chair of ICRA 2005 and of various other national conference events. Currently he is IARP representative of Germany and engaged in establishing international research cooperation in the field of advanced robot systems.



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

Session V

Multi Robot Control & ITS

- **Title: A global decentralized control strategy for urban vehicle platooning relying solely on monocular vision**
Authors: Pierre Avanzini, Benoit Thuilot, Eric Royer, Philippe Martinet
- **Title: Lyapunov Global Stability for a Reactive Mobile Robot Navigation in Presence of Obstacles**
Authors: Ahmed Benzerrouk, Lounis Adouane, Philippe Martinet



2010 IEEE International Conference on Robotics and Automation
Anchorage, Alaska, May 3-8, 2010

A global decentralized control strategy for urban vehicle platooning relying solely on monocular vision

P. Avanzini^{1,4}, B. Thuilot^{1,4}, E. Royer^{2,4} and P. Martinet^{3,4}

¹ Clermont Université, Université Blaise Pascal, LASMEA, BP 10448, 63000 Clermont-Ferrand, France

² Clermont Université, Université d'Auvergne, LASMEA, BP 10448, 63000 Clermont-Ferrand, France

³ Clermont Université, IFMA, LASMEA, BP 10448, 63000 Clermont-Ferrand, France

⁴ CNRS, UMR 6602, LASMEA, 63177 Aubière, France

Pierre.AVANZINI@lasmea.univ-bpclermont.fr

Abstract—Automated electric vehicles available in free access constitute a promising very efficient and environment-friendly “urban transportation system”. An additional functionality that could enhance this transportation service is vehicle platooning. In order to avoid oscillations within the platoon when completing this task, a global control strategy, supported by inter-vehicle communications, is investigated. Vehicle absolute localization is then needed and is here derived from monocular vision. These data are however expressed in a virtual vision world, slightly distorted with respect to the actual metric one. It is shown that such a distortion can accurately be corrected by designing a nonlinear observer relying on odometric data. A global decentralized control strategy, relying on exact linearization techniques, can then be designed to achieve accurate vehicle platooning. Simulations and full-scale experiments demonstrate the performance of the proposed approach.

Index Terms—automatic guided vehicles, platooning, nonlinear control, observer, monocular vision, urban vehicles

I. INTRODUCTION

Traffic congestion in urban areas, with correlated pollution and waste of time, is currently a serious concern. Automated electric vehicles, available in free access from distributed stations within some given zone, appear as an attractive alternative solution. The large flexibility that can be obtained (commutation at any time and along any route) is definitely a decisive feature which should meet user expectations. An additional functionality of special interest is vehicle platooning, i.e. several automated vehicles moving in a single line. Such a functionality allows to easily adapt the transport offer (via platoon length) to the actual need, and can also ease maintenance operations, since only one person can then move several vehicles at a time (e.g. to bring them back to some station). Moreover, an enhancement in safety and an increase in traffic can be expected from such a cooperative navigation. Platooning is therefore considered in this paper.

Different approaches can be proposed. They can be classified into two categories, according to the information used for vehicle control. The most standard approaches rely on *local strategies*, i.e. each vehicle is controlled exclusively from data relative to the neighboring vehicles. The well-known *leader-follower approach* considers only the immediate front vehicle. For instance, visual tracking has been proposed in [2] and generic control laws have been designed in [12] and [5]. Alternatively, neighboring vehicles (and not only

the preceding one) are taken into account when using *virtual structure approaches*: a structural analogy, characterized by a serial chain of spring-damper, is for instance proposed in [13] and a control law is then derived from the combined front and rear virtual forces.

These strategies present however some drawbacks, the most concerning one being error accumulation: the servoing errors, induced by sensor noises and/or actuator delays, are inevitably growing from the first vehicle to the last one, leading to unacceptable oscillations. Such problems can be overcome by considering *global strategies*, i.e. each vehicle is now controlled from the data received from all vehicles. Most of the *virtual structure approaches* belong to this category. In [4], a mechanical analogy is used to design feedback controllers to achieve straight line motion. A single virtual rigid structure is also considered in [6], relying on graph theory. Nevertheless, these techniques aim at imposing some pre-specified geometric pattern, and not that each vehicle accurately reproduces the trajectory of the first one. In contrast, in previous work [3], a trajectory-based strategy has been proposed relying on nonlinear control techniques: lateral and longitudinal control are exactly decoupled, so that lateral guidance of each vehicle with respect to the same reference path can be achieved independently from longitudinal control, designed to maintain a pre-specified curvilinear vehicle inter-distance.



Fig. 1: Experimental vehicles: a Cycab leading two RobuCab

The potentialities of this last control approach have been demonstrated with the experimental vehicles shown in Fig.1 relying, as a first step, on RTK-GPS receivers for vehicle localization [3]. These sensors are however not reliable in urban applications, since satellite signals can be masked by tall buildings. Cameras appear as more appropriate, since the buildings offer a rich environment from an image processing point of view (in addition, they are definitely cheaper).

Accurate absolute localization can indeed be obtained from monocular vision, relying on a structure from motion approach, but it is then expressed in a virtual vision world, roughly related to the actual metric one via a scale factor. Alas, this scale factor is not perfectly constant, so that the vision world appears slightly distorted with respect to the metric one. This alters noticeably the estimation of inter-vehicle distances, and therefore impairs longitudinal control performances. In previous work [1], the local distortions are estimated from the actual distance between two vehicles, measured with a laser rangefinder. This information is then shared with the whole platoon and longitudinal control performances can actually be improved. However, on one hand the combined use of telemetric and visual data is quite intricate, and on the other hand the corrections are not as accurate as possible, since they are only averaged corrections (related to the inter-vehicle distance). In this paper, a non-linear observer, relying solely on standard odometric data, is designed to correct in an easier way, and more accurately, the distortions of the virtual vision world.

This paper is organized as follows: the platooning control strategy is first sketched in Section II. Then, absolute localization from monocular vision is discussed in Section III. Next, the local correction to the visual world is presented in Section IV. Finally, experiments reported in Section V demonstrate the capabilities of the proposed approach.

II. GLOBAL DECENTRALIZED CONTROL STRATEGY

A. Modeling assumptions

Urban vehicles involved in platooning applications are supposed to move at quite low speed (less than $5m.s^{-1}$) on asphalted roads. Dynamic effects can therefore be neglected and a kinematic model can satisfactorily describe their behavior, as corroborated by extensive tests performed with our experimental vehicles shown in Fig. 1. In this paper, the kinematic tricycle model is considered: the two actual front wheels are replaced by a unique virtual wheel located at the mid-distance between the actual wheels. The notation is illustrated in Fig. 2.

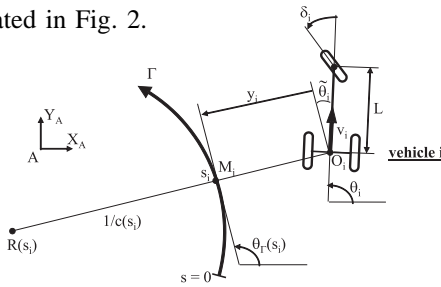


Fig. 2: Tricycle model description

- Γ is the common reference path for any vehicle, defined in an absolute frame $[A, X_A, Y_A]$.
- O_i is the center of the i^{th} vehicle rear axle.
- M_i is the closest point to O_i on Γ .
- s_i is the arc-length coordinate of M_i along Γ .
- $c(s_i)$ is the curvature of path Γ at M_i , and $\theta_\Gamma(s_i)$ is the orientation of the tangent to Γ at M_i w.r.t. $[A, X_A, Y_A]$.
- θ_i is the heading of i^{th} vehicle w.r.t. $[A, X_A, Y_A]$.

- $\tilde{\theta}_i = \theta_i - \theta_\Gamma(s_i)$ is the angular deviation of the i^{th} vehicle w.r.t. Γ .
- y_i is the lateral deviation of the i^{th} vehicle w.r.t. Γ .
- δ_i is the i^{th} vehicle front wheel steering angle.
- L is the vehicle wheelbase.
- v_i is the i^{th} vehicle linear velocity at point O_i .

B. Vehicle state space model

The configuration of the i^{th} vehicle can be described without ambiguity by the state vector $(s_i, y_i, \tilde{\theta}_i)$. The current values of these variables can be inferred on-line by comparing vehicle absolute localization to the reference path. It can then be shown (see [10]) that tricycle state space model is:

$$\begin{cases} \dot{s}_i = v_i \frac{\cos \tilde{\theta}_i}{1 - y_i c(s_i)} \\ \dot{y}_i = v_i \sin \tilde{\theta}_i \\ \dot{\tilde{\theta}}_i = v_i \left(\frac{\tan \delta_i}{L} - \frac{c(s_i) \cos \tilde{\theta}_i}{1 - y_i c(s_i)} \right) \end{cases} \quad (1)$$

Platooning objectives can then be described as ensuring the convergence of y_i and $\tilde{\theta}_i$ to zero, by means of δ_i , and maintaining the gap between two successive vehicles to a fixed value d^* , by means of v_i . It is considered that $y_i \neq \frac{1}{c(s_i)}$ (i.e. vehicles are never on the reference path curvature center). In practical situations, if the vehicles are well initialized, this singularity is never encountered.

C. Control law design

In previous work [3], it has been shown that exact linearization techniques offer a relevant framework to address platoon control: equations (1), as most of kinematic models of mobile robots, can be converted in an exact way into a so-called chained form, see [10]. Such a conversion is attractive, since the structure of chained form equations allows to address independently lateral and longitudinal control.

Steering control laws δ_i can first be designed to achieve the lateral guidance of each vehicle within the platoon w.r.t. the common reference path Γ . In these control laws, v_i just appears as a free parameter. Since conversion of equations (1) into chained form is exact, all nonlinearities are explicitly taken into account. High tracking performances (accurate to within $\pm 5cm$ when relying on an RTK GPS sensor) can then be ensured, whatever initial errors or reference path curvature are. Details can be found in [11].

Control variables v_i can then be designed to achieve longitudinal control. In nominal situation, the objective for the i^{th} vehicle is to regulate $e_i^1 = s_1 - s_i - (i - 1)d^*$, i.e. the arc-length longitudinal error w.r.t. the leader. This control objective is attractive, since the location s_1 of the leader represents a common index for all the vehicles into the platoon, so that error accumulation and inherent oscillations can be avoided. In addition, since it is an arc-length error, this control objective remains consistent whatever the reference path curvature is (in contrast with euclidian inter-distances). Nevertheless, for obvious safety reasons, the location of the preceding vehicle cannot be ignored. Therefore, in previous work [3], the longitudinal control law has been designed to control a composite error: a smooth commutation function

gives the predominance either to the global error e_i^1 or to the local one $e_i^{i-1} = s_{i-1} - s_i - d^*$ according to some security distance. Once more, exact linearization techniques have been used, so that nonlinearities in equations (1) are still explicitly accounted, ensuring high accurate regulation. More details, as well as experiment results carried out with Cycab and RobuCab vehicles (see Fig. 1), relying on RTK GPS sensors for vehicle localization and WiFi technology for inter-vehicle communications, can be found in [3].

III. LOCALIZATION WITH MONOCULAR VISION

The implementation of the platooning control laws presented in previous section requires that some sensors can provide each vehicle with its absolute localization, in a common reference frame (in order that the composite errors could be evaluated). RTK GPS receivers can supply such a localization, with a very high accuracy ($\pm 2cm$). They have successively been used in [3]. However, they are quite expensive sensors, and above all they are not appropriate to urban environments, since satellite signals are likely to be frequently masked by tall buildings. In previous work [8], absolute localization from monocular vision has been alternatively proposed, and satisfactory accurate lateral guidance of a sole vehicle along a given reference path has been demonstrated. An overview of the localization approach is sketched in Section III-A, and its limitations with respect to platooning applications are discussed in Section III-B.

A. Localization overview

The localization algorithm relies on two steps, as shown in Figure 3.

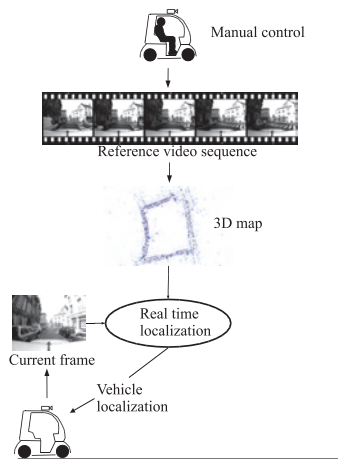


Fig. 3: Localization with monocular vision

First, the vehicle is driven manually along the desired trajectory and a monocular video sequence is recorded with the on-board camera. From this sequence, a 3D reconstruction of the environment in the vicinity of the trajectory is computed. Because only one camera is used, this is a structure from motion problem well-known in the computer vision community. The computation of the reconstruction is done off-line with a method relying on bundle adjustment. The trajectory is thus referred in a non-metric virtual vision world. However, the total covered distance supplied by

on-board odometers, when compared to the same quantity evaluated from vision algorithms, enables to propose a global scale factor such that this virtual vision world is nevertheless close to the actual metric world.

The second step is the real time localization process. Interest points are detected in the current image. These features are matched with the features stored in the visual memory as part of the 3D reconstruction. From the correspondences between 2D points in the current frame and 3D points in the visual memory, the complete pose (6 degrees of freedom) of the camera is computed. Then, the pose of the vehicle on the ground plane is deduced, and finally the vehicle state vector $(s_i, y_i, \tilde{\theta}_i)$ and the curvature $c(s_i)$ required in control laws can all be inferred. More details and localization performances can be found in [9].

B. Distortion in the virtual vision world

Platoon control in urban environment requires vehicle localization to be accurate to within some centimeters. The global scale factor computed from odometric data cannot guarantee such an accuracy: first, odometers cannot supply a covered distance accurate to within some centimeters when the reference trajectory length comes up to few hundred meters. Secondly, the distortion between the two worlds is also varying along the trajectory. These limitations are illustrated in Fig.4: the top graphs show the same vehicle trajectory recorded from monocular vision (Fig.4-a) and from an RTK-GPS sensor (Fig.4-b). The error between the arc-length distances computed from monocular vision and from RTK-GPS data is reported in Fig.4-c: it can be noticed that, on one hand the drift in odometric measurement does not allow a proper evaluation of the global scale factor, so that the total arc-length distance is erroneous in the vision world (the error is $1.72m$, although the trajectory is only $115m$ -long), and on the other hand the distortion between the two worlds is largely varying, since the error comes up to $7.48m$ in the mid-part of the trajectory. The presence of local distortions between the two worlds can also be observed in Fig.4-d, since no global rotation and/or dilatation permits to superpose the two trajectories shown in Fig.4-a/b.

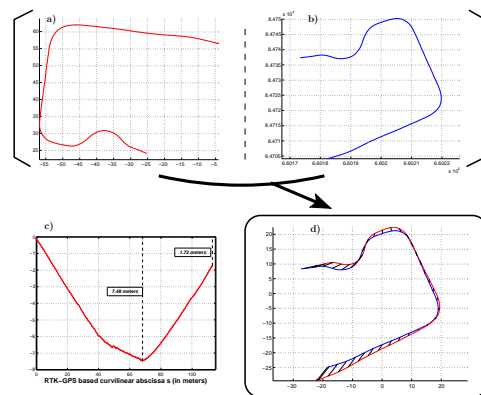


Fig. 4: a), b) resp. vision and RTK-GPS based trajectories
c) error in arc-length distance estimation with vision
d) vision and RTK-GPS based trajectories matching

These distortions in the virtual vision world are not a concern as long as only lateral guidance is considered: since the sign of the lateral and angular deviations y_i and $\tilde{\theta}_i$ supplied by vision algorithms is always correct, these distortions act only as control gain modifications. Asymptotic convergence of y_i and $\tilde{\theta}_i$ to 0 is therefore always guaranteed, and very satisfactory path following results can be obtained, as reported in [8].

The situation is different when longitudinal control is addressed: the distortions in the virtual vision world lead to inaccurate inter-vehicle distance evaluation, and therefore poor longitudinal control performances with respect to the metric world. However, the analysis of experimental results reveals that the distortions are definitely repeatable: lateral guidance along the 115m-long trajectory shown in the upper-left part in Fig.5 has been carried out with several vehicles and with different cameras. For each trial, the set of local scale factors ensuring consistency, on successive 2m-long segments, between the arc-length distance obtained by monocular vision and the actual one supplied by an RTK-GPS sensor, has been computed off-line. Two of these sets are reported in Fig.5. It can be observed that they present a very similar profile, and so do the other sets. More precisely, it can be noticed that the local scale factors are roughly constant in the straight line parts of the trajectory (in magenta), and fast varying in the curved parts (at the beginning and at the end of the cyan segment). As a conclusion, since distortions between the virtual vision world and the actual metric one are clearly repeatable, accurate longitudinal control relying solely on monocular vision appears attainable, provided that the set of local scale factor could be precisely estimated.

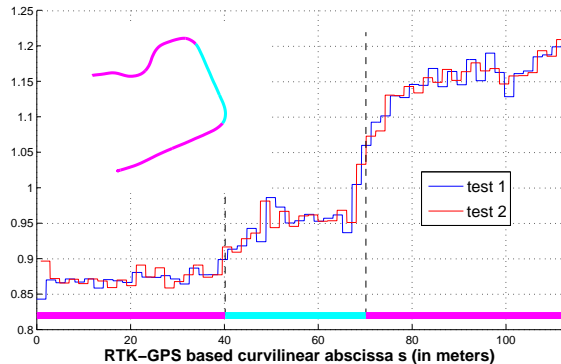


Fig. 5: Off-line local scale factor computation

IV. CURVILINEAR DISTANCE ESTIMATION

Local scale factor estimation requires that some distances in the virtual vision world could also be accurately evaluated in the actual metric world. Very precise measurements in the metric world can be obtained from RTK-GPS receivers. However, these sensors cannot be considered, since on one hand they are not reliable in urban environments due to canyon effects, and on the other hand they are quite expensive when a large fleet of urban vehicles has to be equipped. In previous work [1], it is proposed to rely on a laser rangefinder to obtain a reference measurement in the metric

world: the distance between the leader and the first follower vehicles supplied by this sensor is compared with the same inter-distance derived from monocular vision. The local scale factors can then be inferred and propagated to the rest of the fleet. This approach presents however some drawbacks: first, from a practical point of view, combining telemetric and visual data is quite intricate. But, the major limitation is that distortion corrections thus obtained are necessarily averaged corrections, computed along segments whose lengths are the distance between the two first vehicles, that is to say several meters. The local distortions between the virtual vision world and the metric one might then not be accurately represented, especially in the curved parts of the trajectory, where the local scale factors are supposed to change abruptly, see Fig.5. To relax these limitations, an alternative approach, based on observer theory, and relying solely on standard odometric data, is proposed below.

A. Observer design

In the proposed approach, the reference measurement in the metric world to be used to infer local scale factors is the vehicle linear velocity v_i supplied by the odometers. In the sequel, let us denote $(s_i, y_i, \tilde{\theta}_i)$, $(\dot{s}_i, \dot{y}_i, \dot{\tilde{\theta}}_i)$ and $c(s_i)$ the i^{th} vehicle state vector, state vector derivative and reference path curvature at s_i expressed in the actual metric world, and $(s_i^v, y_i^v, \tilde{\theta}_i^v)$, $(\dot{s}_i^v, \dot{y}_i^v, \dot{\tilde{\theta}}_i^v)$ and $c^v(s_i^v)$ the same quantities expressed in the virtual vision world. Then, in view of the reference measurement to be used, a relevant way to describe the local scale factor at curvilinear abscissa s_i^v is the function:

$$\lambda(s_i^v) = \dot{s}_i / \dot{s}_i^v \quad (2)$$

The distortions in the virtual vision world can realistically be assumed to be locally homogeneous, i.e. the two dimensions in the plane of motion are similarly distorted. Therefore, the following relations can also be written:

$$\lambda(s_i^v) = \dot{y}_i / \dot{y}_i^v \quad (3)$$

$$\tilde{\theta}_i^v = \tilde{\theta}_i \quad (4)$$

$$y_i^v c^v(s_i^v) = y_i c(s_i) \quad (5)$$

Then, injecting relations (2) to (5) into model (1), the vehicle state space model expressed in the virtual vision world can be written as:

$$\begin{cases} \dot{s}_i^v = \frac{v_i \cdot \cos \tilde{\theta}_i^v}{\lambda(s_i^v) \cdot (1 - y_i^v c^v(s_i^v))} \\ \dot{y}_i^v = \frac{v_i \cdot \sin \tilde{\theta}_i^v}{\lambda(s_i^v)} \\ \dot{\tilde{\theta}}_i^v = \dot{\tilde{\theta}}_i \end{cases} \quad (6)$$

Model (6) describes the vehicle motion from the variables actually available, i.e. the vehicle localization in the vision world and its linear velocity in the metric world. The objective now is to design an observer to estimate $\lambda(s_i^v)$ from model (6). Since distortions are the result of a complex and unpredictable optimization process, the time derivative of the variable $\lambda(s_i^v)$ to be observed is completely unknown. Consequently, $\lambda(s_i^v)$ cannot be incorporated into the state vector with the aim to design a standard Luenberger observer.

It is here proposed, just as in [7], to rely on the duality between control and observation to design the observer. More precisely, mimicking the first equation in (6), let us introduce the following observation model:

$$\dot{\hat{s}}_i^v = \frac{v_i \cdot \cos \tilde{\theta}_i^v}{u_i \cdot (1 - y_i^v c^v(s_i^v))} \quad (7)$$

with \hat{s}_i^v the observed curvilinear abscissa in the virtual vision world, y_i^v , $\tilde{\theta}_i^v$ and $c^v(s_i^v)$ measured quantities in the vision world, v_i a measured quantity in the metric world, and u_i a control variable to be designed. Then, the observer principle can be described as follows: if the control variable u_i of the observation model (7) could be designed such that the observed state \hat{s}_i^v converges with the measured one s_i^v , then the control variable u_i would be representative of the local scale factor $\lambda(s_i^v)$ (in view of equations (6) and (7)).

Such a convergence can easily be imposed, by designing u_i straightforwardly as:

$$u_i = \frac{v_i \cdot \cos \tilde{\theta}_i^v}{(\hat{s}_i^v - K \cdot \epsilon)(1 - y_i^v \cdot c^v(s_i^v))} \quad (8)$$

with $\epsilon = (\hat{s}_i^v - s_i^v)$ and K a positive gain to be tuned, since injecting (8) into (7) leads to :

$$\dot{\epsilon} = -K \cdot \epsilon \quad (9)$$

Equation (8) can then be regarded as an accurate estimation of the local scale factor at the curvilinear abscissa s_i^v . If the observer state \hat{s}_i^v is properly initialized, then $|K \cdot \epsilon|$ is largely inferior than $|\dot{s}_i^v|$ (directly related to the vehicle velocity), and observer equation (8) proposes no singularity.

Finally, if $\Gamma(\tau) = (\Gamma_x(\tau), \Gamma_y(\tau))$ denotes the 2D-parametric equations of the reference trajectory Γ in the absolute vision frame, then the corrected curvilinear abscissa at s_i^v can be computed according to:

$$\hat{s}_i = \int_0^{\tau(s_i^v)} \lambda(\tau) \left\| \frac{\partial \Gamma}{\partial \tau}(\tau) \right\| d\tau \quad (10)$$

where $\tau(s_i^v)$ is the parameter value of the 2D-curve $\Gamma(\tau)$ (here, a B-Spline) associated with the curvilinear abscissa s_i^v .

B. Simulations

To investigate the performances of observer (8), a single vehicle has been simulated. The simulation parameters have been tuned in order to be representative of actual conditions:

- The vehicle velocity in the metric world is $v = 1m.s^{-1}$, and the standard deviation of the odometric data is $\sigma_{odo} = 0.015m.s^{-1}$.
- Local scale factors similar to those obtained in Fig. 5 have been generated, thanks to piecewise continuous line segments, see Fig. 6.
- Visual data are provided with a $15Hz$ sampling frequency and two standard deviations $\sigma_v = 0m$ and $\sigma_v = 0.02m$ have been considered.
- Observer gain is $K = 2$, to achieve a compromise between a fast convergence and small oscillations. The observed local scale factor is logically initialized at 1.

The top graph in Fig. 6 shows that observer convergence is achieved within $3m$ (dotted black line). Without any noise on visual data (i.e. $\sigma_v = 0m$), the convergence is very smooth and the average value of the error ϵ is less than $2.4mm$, excepted when the local scale factor changes abruptly (cyan area): then, a very limited $3cm$ overshoot can be noticed. When visual data are corrupted by noise (i.e. $\sigma_v = 0.02m$), the observer error remains inferior than $7cm$, with an average value less than $17.6mm$. Finally, it can be noticed in the bottom graph in Fig. 6 that the observed local scale factor accurately reproduces the simulated one, as desired.

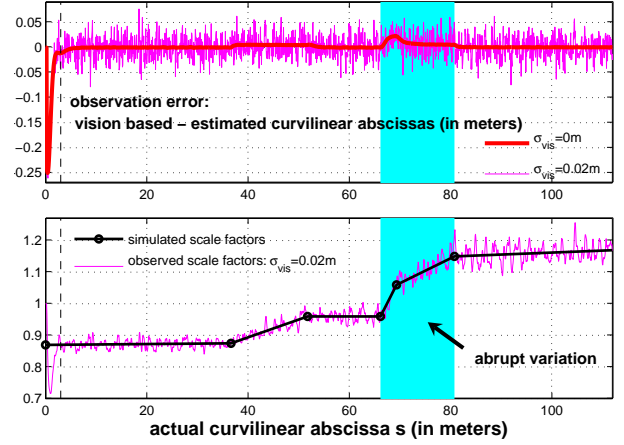


Fig. 6: Simulated scale factor estimation (on-line process)

V. EXPERIMENTAL RESULTS

In order to investigate the capabilities of the proposed approach, several experiments have been carried out in Clermont-Ferrand on “PAVIN Site”, an open platform devoted to urban transportation system evaluation.

1) *Experimental set-up*: The experimental vehicles are shown in Fig. 1. They are electric vehicles, powered by lead-acid batteries providing 2 hours autonomy. Two (*resp. four*) passengers can travel aboard the Cycab (*resp. the RobuCab*). Their small dimensions (length 1.90m, width 1.20m) and their maximum speed ($5m.s^{-1}$) are appropriate to urban environments. Vehicle localization algorithms and platoon control laws are implemented in C++ language on Pentium based computers using RTAI-Linux OS. The cameras supply visual data at a sampling frequency between 8 and $15Hz$, according to the luminosity. The inter-vehicle communication is ensured via WiFi technology. Since the data of each vehicle are transmitted as soon as the localization step is completed, the communication frequency is similar to the camera one. Finally, each vehicle is also equipped with an RTK-GPS receiver, devoted exclusively to performance analysis: its information are not used to control the vehicles.

2) *Experimental results*: The experiment reported below consists in platoon control, with three vehicles, along the 115m-long reference trajectory shown in Fig. 5. The local scale factors computed on-line by the leader vehicle (whose speed is $1m.s^{-1}$) are shown in green in Fig. 7. In order to ease the comparison with the local scale factors computed off-line in Section III-B (and reported in blue in Fig. 7), the

ones obtained on-line have also been averaged on $2m$ -long segments and then shown in red in Fig. 7. It can be noticed that local scale factors computed on-line with observer (8) are as satisfactory as those computed off-line and very close to the actual ones evaluated from RTK-GPS measurements.

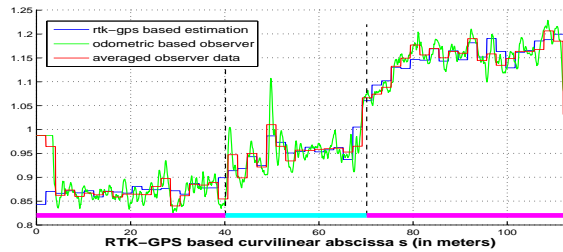


Fig. 7: On-line scale factor estimation

Finally, platoon control performances with corrected vision data are evaluated in Fig. 8. The vehicle inter-distance errors (investigated from RTK-GPS measurements) when longitudinal control relies solely on monocular vision data is as accurate as previously when RTK-GPS data were used to control the vehicles (see [3]): the longitudinal errors remain within $\pm 10cm$. Performances are just slightly depreciated during the abrupt scale factor variation, when $s_1 \in [70, 80]m$. Nevertheless, the inter-distance errors do not exceed *resp.* $14cm$ and $17cm$. If the distortion corrections proposed by observer (8) were not applied to raw localization vision data, then vehicle inter-distance errors would be those displayed in Fig. 9. These large errors (*resp.* $1m$ and $1.7m$) show clearly the significance and the relevance of observer (8).

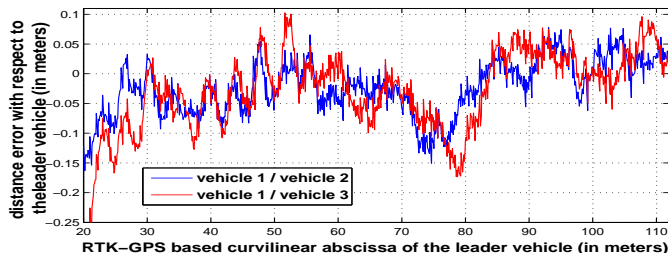


Fig. 8: Vehicle inter-distance errors with corrected vision data

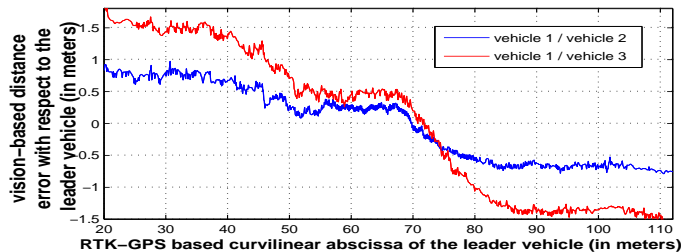


Fig. 9: Vehicle inter-distance errors with raw vision data

VI. CONCLUSION

In this paper, vehicle platooning in urban environments has been addressed. First, a global decentralized control strategy, taking advantage of inter-vehicle communications, has been proposed, in order to avoid error accumulation inherent

to local control approaches. Moreover, nonlinear control techniques have been considered, in order to take explicitly into account the nonlinearities in vehicle models, so that the same high accuracy can be expected in any situation (for instance, whatever the reference trajectory curvature).

Vehicle absolute localization has been derived from an on-board camera, since it is a very appropriate sensor in urban environments. However, it has been pointed out that the localization thus obtained is expressed in a virtual vision world slightly distorted with respect to the actual metric one, and relying on raw vision data would impair platooning performances. A nonlinear observer, only supported by odometric data, has then been designed to estimate on-line local scale factors, and enable accurate platooning relying solely on monocular vision.

Full scale experiments, carried out with three vehicles, have finally demonstrated the efficiency of the proposed approach. Further experiments, involving vehicles led by a manually guided vehicle have to be conducted to emphasize the benefits of on-line corrections when the reference trajectory is being created.

REFERENCES

- [1] P. Avanzini, E. Royer, B. Thuilot, and P. Martinet. A global decentralized control strategy for urban vehicle platooning using monocular vision and a laser rangefinder. In *IEEE Intern. Conf. on Control, Automation, Robotics and Vision (ICARCV)*, Hanoi (Vietnam), 2008.
- [2] S. Benhimane and E. Malis. Vision-based control for car platooning using homography decomposition. In *IEEE Intern. Conf. on Robotics and Automation (ICRA)*, pages 2173–2178, Barcelona (Spain), 2005.
- [3] J. Bom, B. Thuilot, F. Marmoiton, and P. Martinet. A global strategy for urban vehicles platooning relying on nonlinear decoupling laws. In *IEEE Intern. Conf. on Intelligent Robots and Systems (IROS)*, pages 1995–2000, Edmonton (Canada), 2005.
- [4] R.E. Caicedo, J. Valasek, and J.L. Junkins. Preliminary results of one-dimensional vehicle formation control using structural analogy. In *American Control Conference (ACC)*, 2003.
- [5] L. Consolini, F. Morbidi, D. Prattichizzo, and M. Tosques. Leader-follower formation control of nonholonomic mobile robots with input constraints. *Automatica*, 2008.
- [6] W. Dong and Y. Guo. *Formation Control of Nonholonomic Mobile Robots Using Graph Theoretical Methods*, chapter in Cooperative Systems, Springer Verlag, pages 369–386. 2007.
- [7] R. Lenain, B. Thuilot, C. Cariou, and P. Martinet. Adaptive and predictive path tracking control for off-road mobile robots. *European Journal of Control*, 13(4):419–439, 2007.
- [8] E. Royer, J. Bom, M. Dhome, B. Thuilot, M. Lhuillier, and F. Marmoiton. Outdoor autonomous navigation using monocular vision. In *IEEE Intern. Conf. on Intelligent Robots and Systems (IROS)*, pages 3395–3400, Edmonton (Canada), 2005.
- [9] E. Royer, M. Lhuillier, M. Dhome, and T. Chateau. Localization in urban environments : monocular vision compared to a differential GPS sensor. In *Intern. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005.
- [10] C. Samson. Control of chained systems: application to path following and time-varying point stabilization of mobile robots. *IEEE Trans. on Automatic Control*, 40(1):64–77, 1995.
- [11] B. Thuilot, J. Bom, F. Marmoiton, and P. Martinet. Accurate automatic guidance of an urban electric vehicle relying on a kinematic GPS sensor. In *IFAC Symposium on Intelligent Autonomous Vehicles (IAV)*, Lisboa (Portugal), 2004.
- [12] D. Wang and M. Pham. *Unified control design for autonomous car-like vehicle tracking maneuvers*, chapter 8 in “Autonomous Mobile Robots: sensing, control decision-making and applications”, pages 295–329. CRC Press, 2006.
- [13] S.Y. Yi and K.T. Chong. Impedance control for a vehicle platoon system. *Mechatronics*, 2004.

Lyapunov Global Stability for a Reactive Mobile Robot Navigation in Presence of Obstacles

Ahmed Benzerrouk, Lounis Adouane and Philippe Martinet

LASMEA, Blaise Pascal University

24, Avenue des Landais, 63177 Aubière, France.

firstname.lastname@lasmea.univ-bpclermont.fr

Abstract—This paper deals with the navigation of a mobile robot in unknown environment. The robot has to reach a final target while avoiding obstacles. It is proposed to break the task complexity by dividing it into a set of basic tasks: Attraction to a target and obstacle avoidance. Each basic task is accomplished through the corresponding elementary controller. The activation of one controller for another is done according to the priority task. To ensure the overall stability of the control system, especially at the switch moments, properties of hybrid systems are used. Hybrid systems allow switching between continuous states in presence of discrete events. In this paper, it is proposed to act on the gain of the proposed control law. The aim is to ensure the convergence of a common Lyapunov function to all the controllers. This ensures the stability of the overall control. Simulation results confirm the theoretical study.

I. INTRODUCTION

The control of a mobile robot navigating in a cluttered environment is a fundamental problem and is receiving much attention in the robotics community. The purpose is mainly to ensure to the mobile robot a suitable and a safe navigation (avoiding a risk of collision, respecting its structural constraints, etc..)

Some of the literature considers that the robot control is entirely based on the methods of path planning while involving the total or partial knowledge of its environment: Voronoi diagrams and visibility graphs [1] or Artificial potential fields functions containing all the information on the target [2] and the robot environment are among these methods. Another community is interested by the ability of the robot to achieve the control laws according to its constraints (structural constraints, jerk-control, etc.). Even if cognitive methods of path planning and replanning [3], [4], can also be found here, more reactive methods (based on sensors information rather than a prior knowledge of the environment) are more common [5], [6] or [7]. The proposed work falls into the latter approach.

To ensure the robot's ability to accomplish a reactive task, it is proposed to explore behavioral control architectures originally proposed by Brooks [8]. This kind of architecture of control breaks the complexity of the overall task by dividing it into several basic tasks. Each basic task is accomplished with its corresponding controllers. There are two major principles of coordinating them: the action selection [8] and merging actions [9]. In the first, only one controller selected from the basic controllers is applied to the robot at every sample time. In the second case, the control applied

to the robot is a result of merging all or a part of available controllers in the control architecture.

We note that the action selection is more interesting. Indeed, one controller is applied to the mobile robot at a given time. It is then easier to examine individual stability of each controller. However, random switch from one controller to another (avoiding obstacles, follow a trajectory, reaching a target, etc.) may cause instability of the global control law, even if each individual controller is stable [10].

Stability proof of this kind of control architecture has been little explored in the literature: in [5], a merging action node is introduced to the control automaton in order to smoothly switch between the two controllers. The advantage of studying each controller alone is then lost, since we have also to study the merging action node. Controlling a mobile robot to follow a trajectory in presence of obstacles, based on the theorem of multiple Lyapunov functions [10] was established in [11]: A third secondary controller was then introduced to satisfy this theorem. However, this control architecture is not suitable for any cluttered environment.

Finding a common Lyapunov function to the basic systems forming a hybrid system is not a simple task [12]. In this paper, we propose to deal with this problem by ensuring overall stability of our control architecture with a single Lyapunov function. Here we are interested by a mobile robot reaching a target while avoiding obstacles: this task is then divided into two basic tasks: attraction to a target and obstacle avoidance.

The rest of the paper is organized as follows: in next section, the basic controllers and the proposed control law are introduced. The proposed control architecture is exposed in Section III. Simulation results are given in IV. Finally, we conclude and give some prospects in Section V.

II. ROBOT MODEL AND TASKS TO ACHIEVE

Before introducing attraction to the target controller, obstacle avoidance and the proposed control law, we recall that the kinematic model of the used unicycle mobile robot used is expressed by the well-known equations:

$$\begin{aligned}x &= v\cos(\theta) \\y &= v\sin(\theta) \\ \theta &= \omega\end{aligned}\tag{1}$$

With

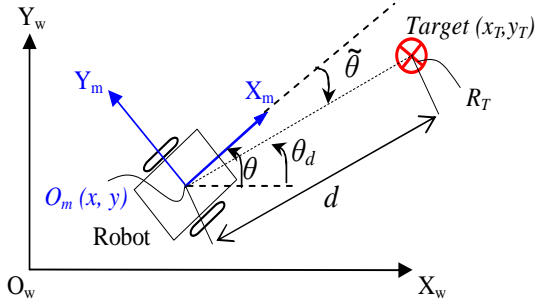


Fig. 1. Controller for attraction to target.

- (x, y) are the world coordinates of the robot axle center O_m (cf. Figure 1).
- θ is the world robot orientation.
- v and ω are respectively linear and angular velocities.

A. Attraction to target controller

The robot has to reach a given target of radius R_T and coordinates center (x_T, y_T) (cf. Figure 1).

Position errors are defined as

$$\begin{aligned} e_x &= x_T - x = d \cos(\theta) \\ e_y &= y_T - y = d \sin(\theta) \end{aligned} \quad (2)$$

d is the distance of the robot to the target and can then be expressed as

$$d = \sqrt{e_x^2 + e_y^2} \quad (3)$$

θ is the orientation error, such that $\theta \in]-\pi, \pi]$ is

$$\theta = \tan^{-1}\left(\frac{y_c - y}{x_c - x}\right) - \theta \quad (4)$$

Its derivative $\dot{\theta}$ is then

$$\dot{\theta} = \left(\frac{e_y}{e_x}\right) / \left(1 + \left(\frac{e_y}{e_x}\right)^2\right) - \omega \quad (5)$$

After computation using the kinematic model (cf. Equation 1) and equations in (2) we obtain

$$\dot{\theta} = \omega_r - \omega \quad (6)$$

Where

$$\omega_r = v \frac{\sin(\theta)}{d}$$

B. Obstacle avoidance controller

The objective of this controller is to control the robot to avoid obstacles that hinder its attraction to the target. To focus on the proposed control architecture, this controller is briefly described. The theoretical details are available in [13].

This controller is based on the limit cycle methods [14], [15]. The differential equations representing the desired trajectory of the robot are given by the following system

$$\begin{aligned} \dot{x}_r &= ay_r + x_r(R_c^2 - x_r^2 - y_r^2) \\ \dot{y}_r &= -ax_r + y_r(R_c^2 - x_r^2 - y_r^2) \end{aligned} \quad (7)$$

With $a = \pm 1$ according to the optimal direction of avoidance (clockwise or counterclockwise direction). (x_r, y_r) are

the relative robot coordinates with respect to the obstacle. The latter is characterized by a circle of radius $R_{cl} = R_o + R_r + \epsilon$ where: R_o is the obstacle radius, R_r is the robot radius and ϵ is a safety margin (cf. Figure 2).

The algorithm for obstacle avoidance is summarized in the following

- The nearest hindering obstacle is detected.
- The direction of avoidance is chosen according to the sensor information.
- The robot avoids the obstacle while following a limit cycle which has a radius $R_c = R_{cl} - \xi$ (attraction phase).
- The robot avoids the obstacle while following a limit cycle which has a radius $R_c = R_{cl} + \xi$ (repulsive phase) (cf. Figure 2). Where ξ is a small value and $(\xi \ll \epsilon)$.

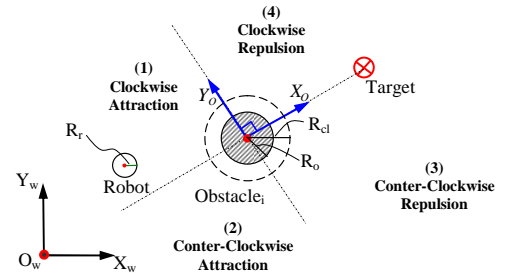


Fig. 2. Obstacle avoidance controller.

C. The proposed control law

It is interesting to notice that only one control law is applied to the robot even if its architecture of control contains two different controllers (attraction to the target and obstacle avoidance). Only the set points change according to the applied controller.

The proposed control law is expressed as follows:

$$\begin{aligned} v &= v_{max} e^{\frac{1}{a} \cos(\theta)} \quad (a) \\ \omega &= \omega_r + k_1 \dot{\theta} \quad (b) \end{aligned} \quad (8)$$

where

- v_{max} is the maximum linear velocity.
- k_1 is a constant such that $k_1 > 0$.
- d is the distance robot-target (cf. Equation 3). The robot reaches the target when $0 < d \leq R_T$ (cf. Section II-A).

To study the stability of the proposed control law, consider the Lyapunov function

$$V = \frac{1}{2} \dot{\theta}^2$$

The control law is asymptotically stable if $V < 0$.

$$V = \theta \dot{\theta}$$

By replacing (6) in (8.b), we get

$$\dot{\theta} = -k_1 \theta \quad (9)$$

and V becomes

$$V = -k_1\theta^2 < 0 \quad (10)$$

and V becomes for every $\theta \neq 0$.

The controller is then asymptotically stable.

Once each basic task and the control law are defined, the proposed architecture of control which coordinates them is given in next section.

III. THE PROPOSED ARCHITECTURE OF CONTROL

Even if each controller is individually stable, it is important to constrain switch between them to avoid instability of the overall system, see [10]. Here, it is proposed to generalize the Lyapunov function previously defined (cf. Section II-C) for the overall control system. Indeed, it was proved (cf. Section II-C) that this function is strictly decreasing. However, the problem arises (as for all hybrid systems) at switching moments where the set point is discontinuous. This means that there is an unavoidable jump of the error θ at these moments. This naturally leads to jumps in the Lyapunov function after the switch and this jump may lead to increasing it.

Hence, It is proposed to adjust the gain k_1 of the control law (cf. Equation 8) at the switch moments so that even if the value of the Lyapunov function increases during the switch, it returns to its value before switch $V(t_{bs})$ in a finite time T_{max} . (t_{bs} is the moment just before switch).

In addition, the robot should not navigate more than a distance d_{max} when ($V(t) > V(t_{bs})$) in order to insure stability criterion as soon as possible. Also, when the robot performs the obstacle avoidance task, it is necessary that ($d_{max} < \epsilon$) (cf. Section II-B) to avoid collision with the obstacle. Notice that ϵ is the minimal distance separating the robot from the obstacle once this one is detected (cf. Section II-B).

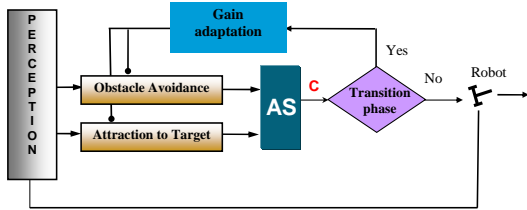


Fig. 3. The proposed architecture of control.

A. Adapting the control law gain

The adjustment of the gain k_1 (cf. Equation 8) is triggered if one of the following events occurs

- The control of the robot switches from one controller to another.
- Obstacle avoidance controller switches from an obstacle to another.
- Obstacle avoidance moves from attraction phase to repulsive phase (cf. Figure 2).

To insure that V decreases in a finite time T_{max} that we can impose, we have to get

$$V(t_s + T_{max}) \leq V(t_{bs}) \quad (11)$$

Where t_s is the switch moment.

The resolution of the differential equation (9) gives the orientation error with respect to time θ

$$\theta(t) = \theta(t_s)e^{-k_1(t-t_s)} \quad (12)$$

Equation (12) allows to easily deduce the Lyapunov function :

$$\begin{aligned} \theta^2(t) &= \theta^2(t_s)e^{-2k_1(t-t_s)} \quad (a) \\ V(t) &= \frac{\theta^2(t_s)}{2}e^{-2k_1(t-t_s)} \quad (b) \\ V(t) &= V(t_s)e^{-2k_1(t-t_s)} \quad (c) \end{aligned} \quad (13)$$

Thus, k_1 is expressed as

$$k_1 = \frac{\ln(V(t)/V(t_s))}{-2(t-t_s)} \quad (14)$$

Note that k_1 is always positive. Indeed, $V(t) \leq V(t_s)$ (cf. Equation 13) and then $\ln(\frac{V(t)}{V(t_s)}) \leq 0$.

The value of k_1 allowing to reach $V(t_{bs})$ in a finite time T_{max} is

$$k_1 = \frac{\ln(V(t_{bs})/V(t_s))}{-2T_{max}} \quad (15)$$

Note that the restriction on T_{max} is necessary especially in the case of obstacle avoidance. Indeed, the stability criterion of hybrid systems (cf. Equation 11) must be satisfied in minimal time. Moreover, the distance achieved during T_{max} has to be ($d_{max} \leq \epsilon$) (cf. Section II-B) to avoid collision with the obstacle. It is easy to see that the minimum necessary time to achieve this distance is

$$t_{min} = \frac{\epsilon}{v_{max}}$$

corresponding to a straight robot navigation to the obstacle center with its maximum linear velocity. (15) becomes then

$$k_1 = \frac{\ln(V(t_{bs})/V(t_s))}{-2t_{min}} \quad (16)$$

Note that k_1 is not defined if $V(t_{bs}) = 0$. The notion of weak stability [16] allows to define a threshold V_{min} such that if $V(t) < V_{min}$, then the system is (weakly) stable without comparing $V(t)$ to $V(t_{bs})$. It means that

$$k_1 = \frac{\ln(V_{min}/V(t_s))}{-2t_{min}} \quad (17)$$

Thus, k_1 is recalculated in this way and replaced in (8.b).

We can then summarize the proposed control architecture as in figure (Fig. 3).

B. The mechanism of the architecture of control

The block *AS* (for *Action Selection*) selects the suitable controller to apply to the robot according to the environment: if no obstacle is detected, Attraction to the target task is accomplished. If there is a discrete event (switching from one controller to another, transition from attraction to repulsive phase, etc.), the block *transition phase* prevents the control from affecting the robot's actuators, until the block *adaptation gain* recalculates the gain k_1 as previously highlighted.

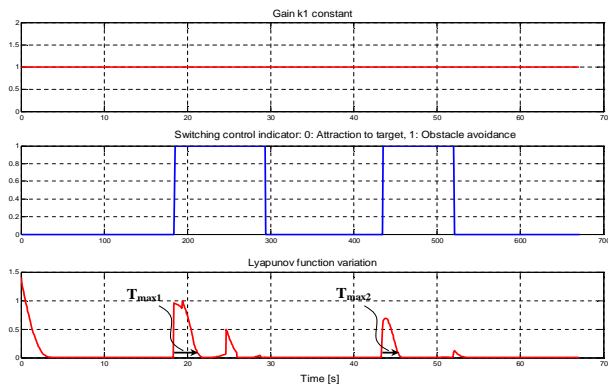


Fig. 4. Variation of the Lyapunov function keeping k_1 constant.

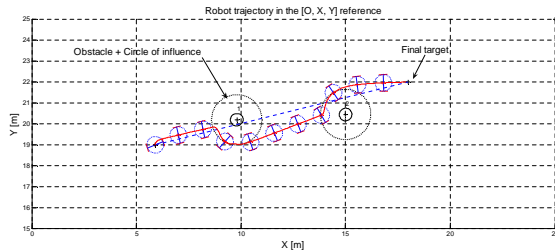


Fig. 5. Robot trajectory in presence of obstacles.

IV. SIMULATION RESULTS

To estimate the relevance of the proposed control architecture, it is proposed to simulate a mobile robot navigation to reach a target in presence of obstacles. Simulation is made twice. In the first case, the used control law has a constant gain during all the navigation ($k_1 = 1$) (there is no gain adjustment in the switch moments). Switching control indicating the active controller can be seen in figure (Fig. 4).

In the second case, the proposed control architecture is implemented on the robot. In the two cases, the robot reaches its target while avoiding obstacles. However, by comparing T_{max1} , T_{max2} which are convergence times for obstacle avoidance controller in figures (Fig. 4) and (Fig. 6), it is noticed that the Lyapunov function of the proposed architecture of control converges faster than the architecture with a constant gain. Evolution of the gain k_1 is given in the same figure (Fig. 6). Note that attraction to the target controller converges fastly in the two cases even if in the proposed architecture, we can see that it is slightly faster.

V. CONCLUSION

A control architecture based on hybrid systems has been proposed. With these systems, it is possible to divide the control architecture into a set of elementary controllers to examine each controller separately. Even if each individual controller is stable, global stability is not necessarily guaranteed. In this paper, the overall stability was established thanks to a single Lyapunov function. The proposed idea is to adjust the gain of the control law in order to accelerate convergence of the Common Lyapunov Function CLF after each switch. The simulation results have confirmed the theoretical study. In future works, it is proposed to introduce the gain k_1 as a

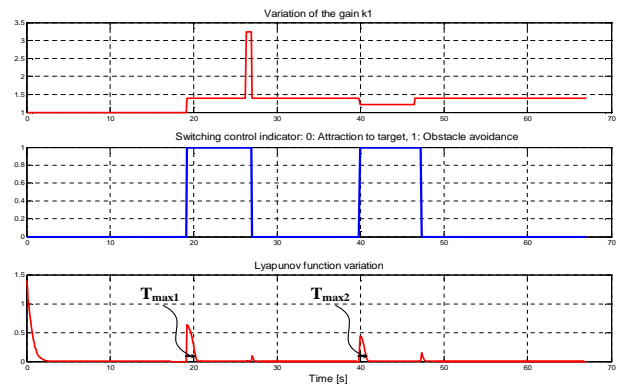


Fig. 6. Variation of the Lyapunov function and the gain k_1 with the proposed architecture of control.

dynamical gain. Thus, once the lyapunov function converges, it returns to its nominal value without disturbing the control.

REFERENCES

- [1] J-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.
- [2] E. Rimon and D. E. Koditschek. Exact robot navigation using artificial potential fields. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, 1992.
- [3] C. Belta, V. Isler, and G.J Pappas. Discrete abstractions for robot motion planning and control in polygonal environments. *IEEE Transactions on Robotics*, 21(5):864–874, 2005.
- [4] D.C. Conner, H. Choset, and A. Rizzi. Integrated planning and control for convex-bodied nonholonomic systems using local feedback control policies. In *Proceedings of Robotics: Science and Systems*, pages 57–64, Philadelphia, USA, 2006.
- [5] M. Egerstedt, K. H. Johansson, J. Lygeros, and S. Sastry. Behavior based robotics using regularized hybrid automata. *Computer Science ISSN 0302-9743*, 1790:103–116, 2000.
- [6] J.M. Toibero, R. Carelli, and Kuchen B. Switching control of mobile robot for autonomous navigation in unknown environments. *IEEE International Conference on Robotics and Automation*, pages 1974–1979, 2007.
- [7] L. Adouane. Hybrid and safe control architecture for mobile robot navigation. In *9th Conference on Autonomous Robot Systems and Competitions*, Portugal, May 2009.
- [8] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2:14–23, 1986.
- [9] R. C. Arkin. Motor schema-based mobile robot navigation. *International journal of robotics research*, 8(4):92–112, 1989.
- [10] M. S. Branicky. Stability of switched and hybrid systems. In *33rd IEEE Conference on Decision Control*, pages 3498–3503, 1993.
- [11] A. Benzerrouk, L. Adouane, P. Martinet, and Andreff N. Multi lyapunov function theorem applied to a mobile robot tracking a trajectory in presence of obstacles. In *European Conference on Mobile Robots*, 2009.
- [12] D. Liberzon. *Switching in systems and control*. Birkhäuser, 2003.
- [13] L. Adouane. Orbital obstacle avoidance algorithm for reliable and on-line mobile robot navigation. In *9th Conference on Autonomous Robot Systems and Competitions*, May 2009.
- [14] D. Kim and J. Kim. A real-time limit-cycle navigation method for fast mobile robots and its application to robot soccer. *Robotics and Autonomous Systems*, 42:17–30, 2003.
- [15] M.S. Jie, J.H. Baek, Y.S. Hong, and K. Lee Woong. Real time obstacle avoidance for mobile robot using limit-cycle and vector field method. *Knowledge-Based Intelligent Information and Engineering Systems*, pages 866–873, October 2006.
- [16] B. Brogliato, S. Niculescu, and Orhant. On the control of finite dimensional mechanical systems with unilateral constraints. *IEEE Transactions on Automatic Control*, 42(2):200–215, 1997.