A MINIATURIZED FOUR-WHEEL ROBOTIC VEHICLE FOR AUTONOMOUS DRIVING RESEARCH IN OFF-ROAD SCENARIOS

Lucas A. C. de O. Nogueira*, Mauro F. Koyama†, Rafael de A. Cordeiro‡, Alexandre M. Ribeiro*, Samuel S. Bueno‡, Ely C. de Paiva*

*Faculdade de Engenharia Mecânica (FEM) da Universidade Estadual de Campinas (UNICAMP)
Rua Mendenleyv, 200, Cidade Universitária, 13083-860
Campinas, São Paulo, Brasil

†Centro de Tecnologia da Informação Renato Archer - DRVC / CTI
Rod. D. Pedro I, km 143,6 - 13081-970
Campinas, São Paulo, Brasil

‡Instituto Superior Técnico - IDMEC / IST
Av. Rovisco Pais ,1 - 1049-001
Lisboa - Portugal

Emails: lucas.nogueira@fem.unicamp.br, mauro.koyama@cti.gov.br, rafael.a.cordeiro@tecnico.ulisboa.pt, amribeiro@fem.unicamp.br, samuel.bueno@cti.gov.br, elypaiva@fem.unicamp.br

Abstract— This article presents a miniaturized four-wheel robotic vehicle for autonomous driving research. It enables experiments in situations where testing with full-size vehicles becomes dangerous or troublesome. The article describes the physical structure of the vehicle, its actuators and sensing capabilities and the ROS-based software architecture. The vehicle will serve as a platform for future research in the areas of Perception, Planning and Control, particularly for off-road scenario. Validation results are presented which demonstrate that the vehicle is ready to be used in further research applications.

Keywords— Autonomous Navigation, Outdoor Robotics.

Resumo— Esse artigo apresenta um veículo de quatro rodas em escala para pesquisa em navegação autônoma. Ele possibilitará experimentos em cenários onde testes com carros de tamanho regular se tornam perigosos ou problemáticos. O artigo descreve a estrutura física do veículo, seus sensores e atuadores, e sua arquitetura de software baseado em ROS. Ele servirá como plataforma de desenvolvimento para pesquisas nas áreas de Percepção, Planejamento e Controle, em especial para estratégias aplicáveis em ambientes todo-terreno. São apresentados resultados de validação os quais demonstram que o veículo está pronto para ser usado em pesquisas subsequentes.

Palavras-chave— Navegação Autônoma, Robótica de Exterior.

1 Introduction

The first mobile intelligent robot, Shakey (Nilsson, 1984), was built in 1966. It produced research results that are still valid today. Its layered software architecture has become a paradigm in robotics. Since then, the mobile robotics research field has experienced immense growth. In 2018, autonomous cars are starting to be deployed in commercial applications.

Research efforts were fundamentally responsible for the commercial success of these endeavors. The United States Government used the three DARPA Grand Challenge to mobilize the research community. Robotic cars that won the challenges ((Urmson et al., 2008), (Thrun et al., 2006)) became templates for the next decades of research. Key personnel involved in these Challenges went on to lead the development of commercial self-driving cars. In Brazil, research groups have built their own robotic cars. VILMA (Lemos et al., 2016), CaRINA (Fernandes et al., 2014) and VERO (Bueno et al., 2009) are prominent examples. The first two are based on commercially available vehicles for on-road usage, while the latter is a custom-built vehicle intended for off-road applications.

Although the first commercial self-driving vehicles are starting to be deployed, the technology is still far from general use. It is very dependent on optimal road conditions. This leads to the use of geofences that restrict the areas where these vehicles can be used (Waymo, 2017).

More research is needed for better navigation in adverse road conditions, i.e, heavy rain or ice. But testing full-sized robotics cars in such conditions can be dangerous and cost-prohibitive. A solution is to use a miniaturized version of a robotic car, while maintaining the same level of sensing capabilities and processing power as a full-size robotic car. Such a vehicle can serve as a testbed for validating autonomous driving applications (trajectory control, tire-ground force estimation, etc) in dangerous environments such as slippery and highly uneven terrains. One remarkable prototype project is the Helvis III vehicle (Higuti et al., 2016), a small-scale car-like mobile robot currently being successfully used for navigation in agricultural fields.

This article presents a miniaturized four-wheel robotic vehicle for autonomous driving research. Its goal is to allow multiple independent researchers to validate their strategies for Perception, Planning and Control in adverse road conditions. The low-level controller proposed here is designed for indoor and outdoor environments. The outdoor experiment was carried out on an orchard agricultural field and the complete architecture validation is shown in (de Lemos et al., 2018).

The first section presents the robotic car hardware configuration, the second its software architecture. The next sections explain the sensing, control and planning capabilities of the vehicle. The last section contains indoor validation results which demonstrate that the vehicle is ready to be used in further research applications.

2 Robotic Vehicle Structure

The miniaturized vehicle is presented in Figure 1. It is a modified Baja 5B 2.0 (HPI-Racing, 2018) with Ackerman steering and independent traction in the rear wheels. For such, the suspension and steering sets were completely redesigned and two electric motors were installed, replacing the original combustion engine.

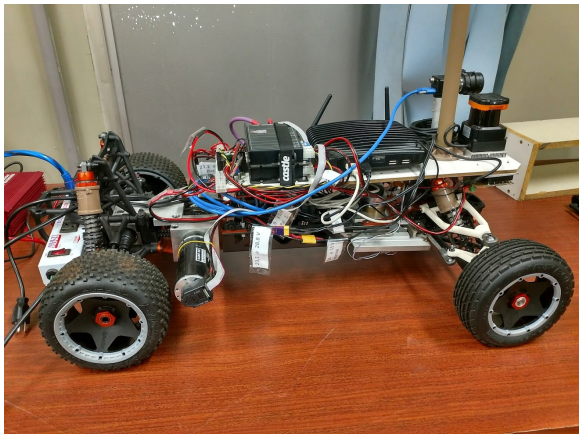


Figure 1: Miniaturized Robotic Car

A Hokuyo UTM-30LX 2D laser is mounted on the front of the vehicle, capable of observing the environment ahead. A PointGrey FL3-U3-32S2C-CS camera is mounted on the side of the laser, equally oriented. These two sensors are responsible for the exteroception capabilities of the vehicle, i.e., perception of the environment. They are important in applications such as mapping and obstacle avoidance.

The remaining sensor is a XSens Mti-G-700 inertial measurement unit (IMU). It contains accelerometers, gyroscopes, barometers, magnetometer and a GPS unit. Its onboard processor performs sensor fusion to provide high-quality position, velocity, acceleration and orientation.

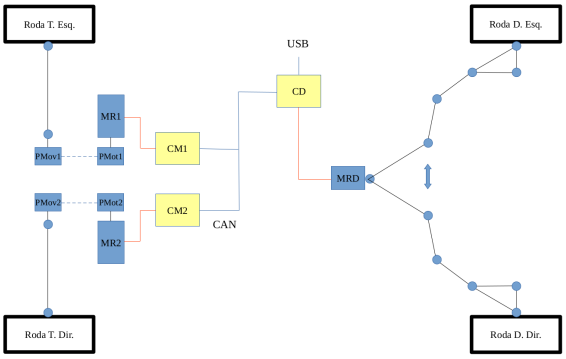


Figure 2: Organization of the car actuators

The actuator set is composed by three electric motors. Two Maxon RE50 motors control each rear wheel independently, using a Maxon EPOS2 70/10 controller. This configuration allows us to implement a electronic differential to distribute different torques and velocities to each wheel. The differential can enhance the vehicle off-road capabilities.

The steering mechanism is activated by a Maxon EC45 motor, together with a EPOS 24/2 controller. All three motors have encoders that can be used for odometry estimation. The actuators organization is shown in Figure 2.

The computing subsystem is composed by an Intense PC with an Intel Core-i7 processor. It runs Ubuntu 16.04 operating system. The IMU, laser, camera and steering motor controller are connected to the computer via USB. The three controllers are connected together via the CAN bus.

The power subsystem uses lithium polymer (LiPo) batteries and is divided in two independent parts. First is the power system of the rear wheel motors. It uses a 4-cell, 14.8V battery. Second is the power system for the computing, sensing and steering components. It uses a 7-cell, 25.9V battery.

3 Software Architecture

The vehicle software architecture is based on Robot Operating System (ROS) Kinetic (Foote, 2016). The possibility of using and sharing code with a global community is a major advantage of using ROS. This allows the development effort to focus on new research ideas, instead of reimplementing existing ones. Another advantage is the built-in modularization. This increases the flexibility for researchers to choose which module they want to use in each experiment.

A ROS system is organized with packages. Each package is related to a specific robot functionality. For instance, each sensor has its own ROS package. It is useful to classify the packages

into five layers, similar to the ones presented in (Thrun et al., 2006). They are:

- **Sensing**: This layer contains all packages that implement the robot sensor drivers. Most of these have been developed and made available by the device's manufacturers.
- **Perception**: Contains the packages related to the extraction of useful information from the sensing data. A classic example is a Kalman Filter used to merge redundant sensor data. Another example is object detection and prediction using images and laser scans.
- **Planning**: This layer is responsible for generating plans that attempt to achieve the robotic car goals, using data from the perception layer and issuing commands to the control layer. It is common to have a user interface for setting these goals.
- **Control**: This layer contains the packages that issue commands to the actuators, usually implementing a feedback loop with data from the perception layer. Common examples are PID controllers.
- **Actuators**: This layer implements an interface with the three motors of the robotic car. It's packages encapsulate the motor's controllers drivers.

Each robotic car application uses a specific set of ROS packages. Different applications can be designed by separate researchers simultaneously. The goal is for researchers to be able to focus on their specialties. Parallel researches can be performed in the areas of Perception, Planning and Control, using the robotic platform for validation. The packages that are developed in this process can be reused later and made available to the research community.

4 Control and Sensing

The Sensing and Actuators layers are the first step in the development of a robotic platform. These layers make it possible for the software to interface with the hardware. Sensor and actuator drivers are publicly available as ROS packages. Camera (Rockey, 2017a) and laser (Rockey, 2017b) interfaces were created by third-parties. For the XSens IMU, a custom driver was developed. The actuator interface encapsulates the EPOS application programming interface (Wills, 2015).

The three motors of the robotic car respond to different command types. The steering motor use a position controller, expressed in encoder units. The rear wheel motors use a velocity command, expressed in rotations per minute (rpm). Two additional interfaces were implemented in the control layer.

- **CarCommand**: Allows the command of independent expected linear speed(m/s) for each rear wheel and an Ackermann steering angle (rad).
- **CmdVel**: Allows the command of a linear speed(m/s) and yaw rate(rad/s). This is a standard interface in ROS environment. It enables easy integration with other packages developed by the research community. Current implementation sends the same speed to both rear motors.

4.1 Steering Control with IMU feedback

It is possible to combine these two interfaces with different control strategies. In one hand, the rear wheel control is straightforward since the gear mechanism does not introduce meaningful deviations. On the other hand, the steering control requires further attention due to the geometry of the steering mechanism. The existence of dead zone and hysteresis makes it hard to map the motor position to a steering angle. One possible solution is to use IMU feedback to control the motor position in order to achieve a desired yaw rate.

Figure 3 shows the control strategy proposed in this paper and implemented in the vehicle. Inputs of the system are: ω_{des}, the commanded yaw rate; v_{des}, the commanded linear velocity; and ω_{meas}, the yaw rate as measured by the IMU. Desired (κ_{des}) and measured (κ_{meas}) curvature are calculated from the inputs. The linear speed of the vehicle is approximated by its desired speed v_{des}. A feedforward component is combined with a PID controller. It determines the equivalent kinematic bicycle model and calculates the desired steering angle command (u_1) from κ_{des} and the wheelbase length L . The PID uses the error e to generate a additional correction command u_2. The sum of both components is used to control the steering motor.

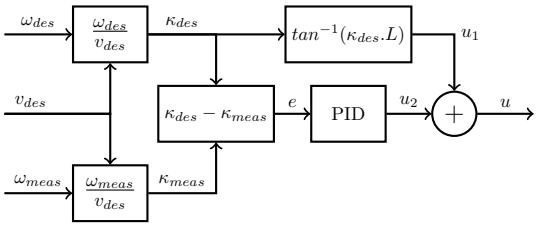


Figure 3: Steering Control Diagram

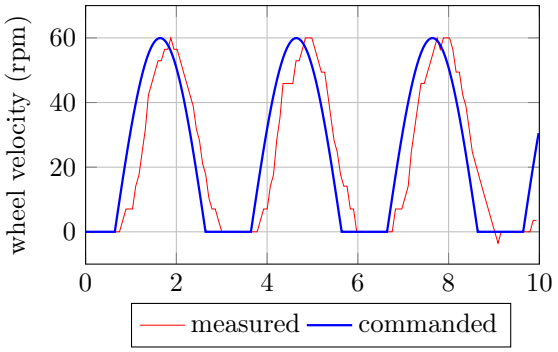
5 Planning

The robotic car utilizes the ROS navigation stack (Marder-Eppstein et al., 2010) to implement the planning layer. This allows the reuse of mobile robotics software. The navigation stack provides a common framework for different planners and

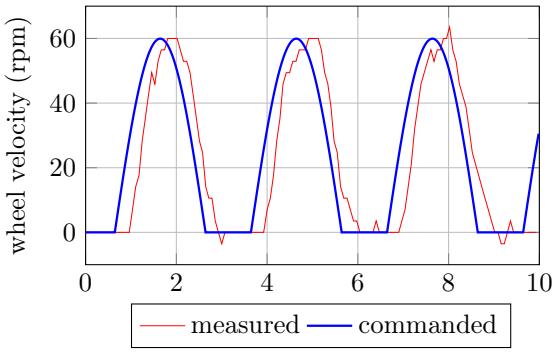
controllers. Some of the features provided by the navigation stack are:

- Obstacle detection using exteroceptive sensing.
- Dynamic and static mapping using SLAM algorithms.
- Path planning in two levels. This allows for a global plan that considers static obstacles and a local plan that can react to dynamic obstacles.
- Path tracking using position estimations and velocity commands.

The navigation stack provides these features to different robotic platforms. It generates **CmdVel** velocity commands. This allows for easy integration with any platform that uses such interface. However, the navigation stack was designed mainly for vehicles with differential drive. Because the miniaturized vehicle has an Ackermann drive configuration, it warrants special attention. The Time-Elastic Band (TEB)(Rösmann et al., 2012) planner solves this problem because it is able to consider the minimum curvature radius, thus creating paths that the vehicle can track.



(a) left wheel control



(b) right wheel control

Figure 4: Linear Velocity Results

6 Platform Validation

A set of experiments were conducted to demonstrate that the vehicle is ready to be used in further research applications. We show that the vehicle responds to the proposed control strategies for linear speed and yaw rate. Then, we end the article by showing a path planning experiment that serves as an integration test for the complete robotic system.

6.1 Linear Velocity Control

The linear and angular speed control were validated separately. Commands were issued to each rear wheel using rotations per minute (rpm) units. The motor encoders provide feedback. The vehicle was commanded a linear speed profile while on the ground. Both the commands and the measurements are produced at a 10 Hz rate. Figure 4 shows the results obtained. They show that the motors can achieve the commanded velocity. There is a reasonable delay that needs further investigations.

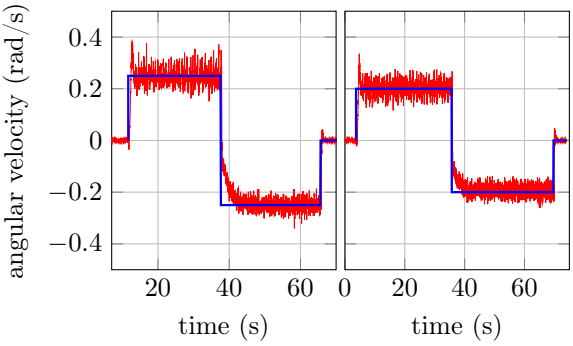
6.2 Curvature Radius Control

The following experiment was used to validate the yaw rate control. The vehicle was positioned approximately in the center of a room. A desired curvature radius was commanded. After it completed a 360 degree turn, the command was inverted so that the vehicle turned to the opposite direction with the same radius. The indoor location was important for the experiment because the laser-based Hector SLAM algorithm (Kohlbrecher et al., 2011) was used as ground-truth for the trajectory of the vehicle. The goal is to verify that the vehicle is able to achieve the desired yaw rate.

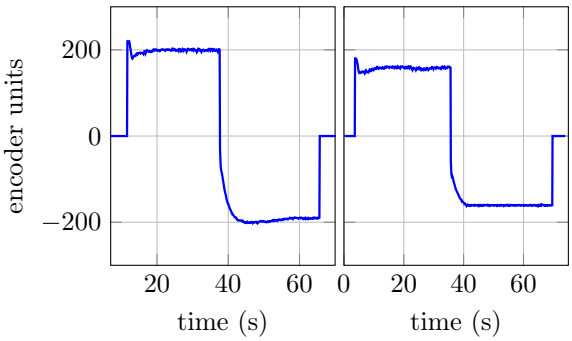
The experiment was conducted for two different curvature radius. In both cases the linear speed was set to 0.4 m/s. The two curvature radius used were 1.6 m and 2.0 m. This results in a desired yaw rate of 0.25 rad/s and 0.2 rad/s, respectively. In Figure 5, results on the left are from the first experiment (1.6m radius). On the right are results from the second experiment (2.0m radius). Figure 5a compares the yaw rate as measured by the IMU with the commanded yaw rate. In both cases the vehicle was able to maintain the desired yaw rate. Figure 5b shows the steering motor control command, in encoder units. For this motor, one full rotation contain 4096 units. In both cases the control effort is shown to be smooth. Figure 5c shows the trajectories for both cases. They show the actual curvature radius was within a reasonable tolerance of the desired one.

6.3 Path Planning and Following

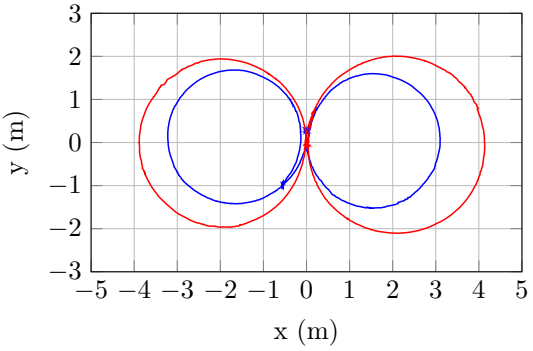
Figure 6 shows a path following experiment. The vehicle is positioned in a room. Initially (6a), the map is incomplete, because of occlusions on the laser field of view. In 6b, the TEB planner inflates the obstacles ahead of the car (blue and pink). In 6c, a goal pose (green arrow) is set. A global plan is created using an A* algorithm (red line). It



(a) commanded vs measured angular velocity



(b) control effort



(c) trajectory

Figure 5: Steering Control Results

does not consider the kinematic constraints of the vehicle. The TEB planner uses this global plan to create a local plan (green line), that does consider the kinematics and controls the car trajectory. In 6f, a previously unmapped obstacle is found. TEB planner reacts accordingly, maximizing the distance to obstacle while trying to reach its goal. In 6l, the vehicle reaches the goal. Figure 7 shows the vehicle in motion.

7 Conclusion and Future Work

In this paper, we presented a new 4-wheel miniaturized autonomous driving research platform. It will enable researchers to validate techniques for challenging scenarios. It contains sensors that allow for a variety of different perception strategies. Its actuators are specially suited to handle adverse road conditions, due to the presence of independent rear wheel motors. We have also proposed a new steering control technique using IMU feedback to control the motor position, yielding better precision in yaw rate control under the presence of actuator nonlinearities and constraints. The computing architecture based on ROS allows researchers to quickly deploy different applications. Multiple research projects that will use this platform are underway in our laboratory.

Acknowledgments

The authors acknowledge the support of CAPES (N. 33003017022P0); INCT - SAC - Sistemas Autônomos Colaborativos (CNPq. N. 465755/2014-3 and FAPESP N. 2014/50851-0), Regular FAPESP Auto_Verde (FAPESP N. 2018/04905-1) and Ph.D FAPESP (FAPESP N. 2018/05712-2).

References

Bueno, S. S., Azevedo, H., Mirisola, L. G. B., de Paiva, E. C. and Ramos, J. J. G. (2009). Uma plataforma para pesquisa e desenvolvimento em robótica terrestre de exterior, *IX Simpósio Brasileiro de Automação Inteligente*.

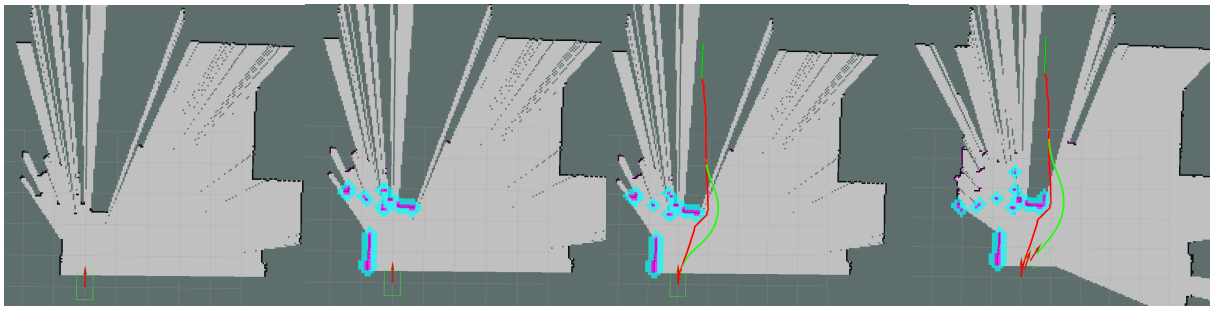
de Lemos, R. A., de O. Nogueira, L. A. C., Ribeiro, A. M., Mirisola, L. G. B., Koyama, M. F., de Paiva, E. C. and Bueno, S. S. (2018). Unisensory intra-row navigation strategy for orchards environments based on sensor laser, *XXII Congresso Brasileiro de Automática*.

Fernandes, L. C., Souza, J. R., Pessin, G., Shinzato, P. Y., Sales, D., Mendes, C., Prado, M., Klaser, R., Magalhães, A. C., Hata, A., Pigatto, D., Branco, K. C., Grassi, V., Osorio, F. S. and Wolf, D. F. (2014). CaRINA intelligent robotic car: Architectural design and applications, *Journal of Systems Architecture* **60**(4): 372–392.

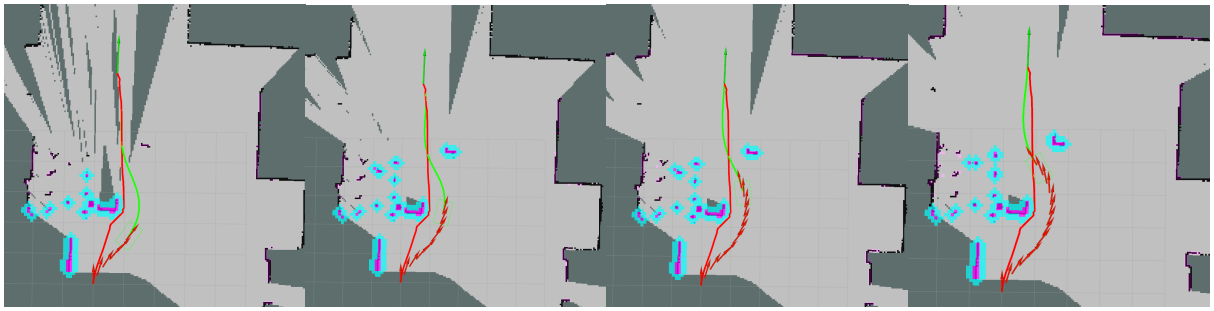
Foote, T. (2016). kinetic - ROS Wiki. `http://wiki.ros.org/kinetic`. Accessed in 26-Feb-2018.

Higuti, V. A. H., Velasquez, A. E. B., Guerrero, H. B., Magalhães, D. V. and Becker, M. (2016). Description of helvis 3 - a small scale car-like robot for precision agriculture, 1° *SiPGEM* - 1° *Simpósio do Programa de Pós-Graduação em Engenharia Mecânica*.

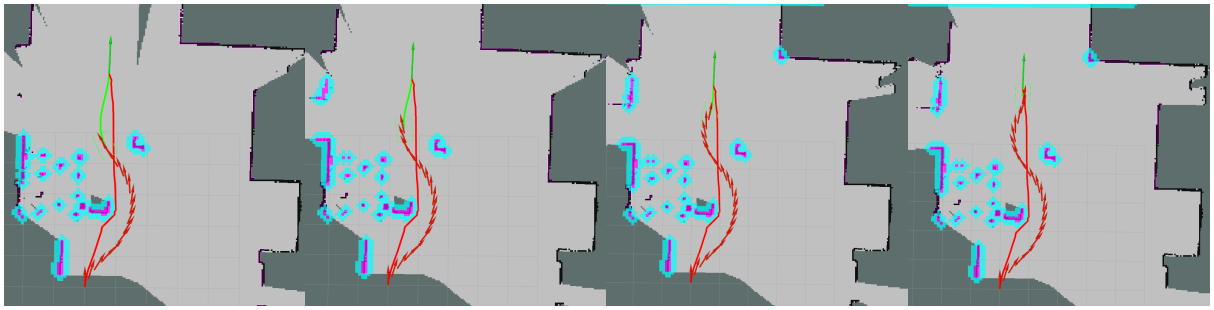
HPI-Racing (2018). 10630 RTR Baja 5B 2.0. `http://www.hpiracing.com/en/kit/10630`. Accessed in 26-Feb-2018.



(a) (b) (c) (d)



(e) (f) (g) (h)



(i) (j) (k) (l)

Figure 6: Path Planning and Tracking

(a) (b) (c)

(d) (e) (f)

Figure 7: A sample of the vehicle movement in path tracking experiment

Kohlbrecher, S., Meyer, J., von Stryk, O. and Klingauf, U. (2011). A flexible and scalable slam system with full 3d motion estimation, *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, IEEE.

Lemos, R., Garcia, O. and Ferreira, J. V. (2016). Local and global path generation for autonomous vehicles using splines, *Ingeniería* **21**(2): 188–200.

Marder-Eppstein, E., Berger, E., Foote, T., Gerkey, B. and Konolige, K. (2010). The office marathon: Robust navigation in an indoor office environment, *IEEE International Conference on Robotics and Automation*, pp. 300–307.

Nilsson, N. J. (1984). Shakey the robot, *Technical report*, SRI International Menlo Park, CA.

Rockey, C. (2017a). pointgrey_camera_driver - ROS Wiki. `http://wiki.ros.org/pointgrey_camera_driver`. Accessed in 26-Feb-2018.

Rockey, C. (2017b). urg_node - ROS Wiki. `http://wiki.ros.org/urg_node`. Accessed in 26-Feb-2018.

Rösmann, C., Feiten, W., Wösch, T., Hoffmann, F. and Bertram, T. (2012). Trajectory modification considering dynamic constraints of autonomous robots, *Robotics; Proceedings of ROBOTIK 2012; 7th German Conference on*, VDE, pp. 1–6.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A. and Mahoney, P. (2006). Stanley: The robot that won the DARPA grand challenge, *Journal of Field Robotics* **23**(9): 661–692.

Urmson, C., Anhalt, J., Bae, H., Bagnell, J. A. D., Baker, C. R., Bittner, R. E., Brown, T., Clark, M. N., Darms, M., Demitrish, D., Dolan, J. M., Duggins, D., Ferguson, D., Galatali, T., Geyer, C. M., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T., Kolski, S., Likhachev, M., Litkouhi, B., Kelly, A., McNaughton, M., Miller, N., Nickolaou, J., Peterson, K., Pilnick, B., Rajkumar, R., Rybski, P., Sadekar, V., Salesky, B., Seo, Y.-W., Singh, S., Snider, J. M., Struble, J. C., Stentz, A. T., Taylor, M., Whittaker, W. R. L., Wolkowicki, Z., Zhang, W. and Ziglar, J. (2008). Autonomous driving in urban environments: Boss and the urban challenge, *Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge, Part I* **25**(8): 425–466.

Waymo (2017). Waymo Safety Report. `https://waymo.com/safetyreport/`. Accessed in 26-Feb-2018.

Wills, M. (2015). epos_hardware - ROS Wiki. `http://wiki.ros.org/epos_hardware`. Accessed in 26-Feb-2018.