CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering

BACHELOR THESIS

Veronika Pěčonková

Motion planning for autonomous car manipulator

Department of Computer Science

Thesis supervisor: **Tomáš Krajník**

May, 2021

PRAHA 2021

BACHELOR'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Pěčonková Veronika** Personal ID number: **481770**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Cybernetics and Robotics**

II. Bachelor's thesis details

Bachelor's thesis title in English:

Motion Planning for Autonomous Car Manipulator

Bachelor's thesis title in Czech:

Plánování pohybu pro autonomní mobilní manipulátor vozidel

Guidelines:

The aim of this work is to design, implement and experimentally test methods for planning the movement of a mobile manipulator. The manipulator must be able to quickly and precisely load a car, move it to the intended location and unload it there. This work will focus on a key component of the system, which is to implement the (un-)loading maneuver. This method will be integrated into the Robotic Operating System (ROS) and experimentally verified on a real platform.

1) Familiarize yourself with the Lipraco Phoenix vehicle and its control system.
2) Familiarize yourself with the Robotic Operating System (ROS).
3) Get acquainted with the motion planning methods used in mobile robotics.
4) Analyze the vehicle handling problem and determine key parameters.
5) Select suitable methods and perform their preliminary tests in the simulator and on a real platform.
6) Based on the results of preliminary experiments, select a suitable motion planning method and implement it for the Lipraco Phoenix platform.
7) Integrate the method into the navigation system of this platform.
8) Design a set of experiments and perform them.

Bibliography / sources:

[1] Yoshiaki Kuwata et al.: Motion planning for urban driving using RRT. In IROS 2008
[2] Summers, T. : Distributionally Robust Sampling-Based Motion Planning Under Uncertainty. In IROS 2018
[3] Costa M.M. et al.: A Survey on Path Planning Algorithms for Mobile Robots. In ICARCS 2019

Name and workplace of bachelor's thesis supervisor:

doc. Ing. Tomáš Krajník, Ph.D., Artificial Intelligence Center, FEE

Name and workplace of second bachelor's thesis supervisor or consultant:

Date of bachelor's thesis assignment: **06.01.2021** Deadline for bachelor thesis submission: **21.05.2021**

Assignment valid until: **30.09.2022**

doc. Ing. Tomáš Krajník, Ph.D. Supervisor's signature	prof. Ing. Tomáš Svoboda, Ph.D. Head of department's signature	prof. Mgr. Petr Páta, Ph.D. Dean's signature

III. Assignment receipt

The student acknowledges that the bachelor's thesis is an individual work. The student must produce her thesis without the assistance of others, with the exception of provided consultations. Within the bachelor's thesis, the author must state the names of consultants and include a list of references.

. Date of assignment receipt	Student's signature

Prohlášení

I hereby declare that I have completed this thesis independently and that I have used only the sources (literature, software, etc.) listed in the enclosed bibliography.

In Prague on............................ ....

Acknowledgements

I would like to thank my supervisor doc. Ing. Tomáš Krajník, Ph.D. for opportunity and patience.

Abstract

The work deals with loading the car with an autonomous loader with Ackermann's chassis. Three maneuvers are designed, each of them can load a car. The first is based on the correct setting of the correct position and orientation of the loader. During the second maneuver, the correct position and orientation are set at the same time. The third maneuver is based on driving a suitable circle trajectory. Finally, the speed and space requirements of all three maneuvers are measured. From the proposed maneuvers, driving a circle trajectory turned out to be the most accurate, fastest, and least space-consuming.

Abstract

Práce se zabývá naložením automobilu autonomním nakladačem s Ackermannovým podvozkem. Jsou navrženy tři manévry, každým je možné naložit automobil. První je založen na seprátním nastavením spravné pozice a orientace nakladače. Během druhého manévru se správná pozice a orientace nastavuje zároveň. Třetí manévr spočívá v jízdě po vhodné kružnici. Nakonec je změřena rychlost a prostorová náročnost všech tří manévrů. Jako nejpřesnější nejrychlejší a nejméně prostorově náročná se z navrhnutých manévrů ukázala jízda po kružnici.

Contents

List of Figures

1 Prolog

"There's a lot of automation that can happen that isn't a replacement of humans but of mind-numbing behavior."

— Stewart Butterfield

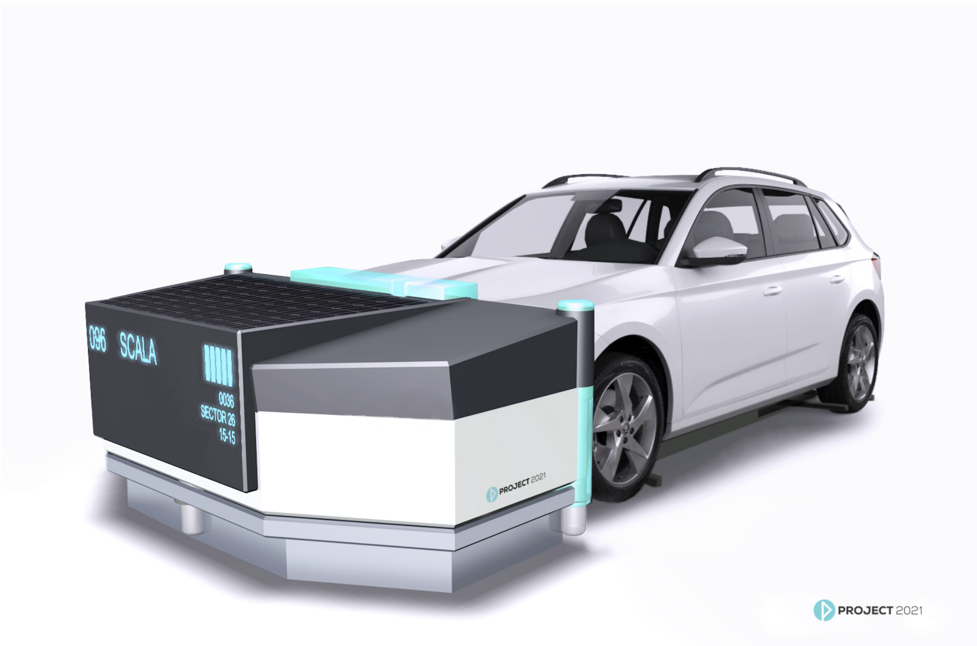

Figure 1: Vision of logistic at parking lots

2 Introduction

2.1 Motivation

This work describes the autonomous loading of a vehicle with a mobile manipulator, which is part of the project for autonomous vehicle storage in large outdoor parking lots. You can read about a successful solution of a similar project from Stanley Robotic here [1]. Stanley Robotics has introduced its robot, Stan, as the first outdoor valet parking robot. Before we dive into the analysis of the topic of the work, we will first describe its context, then the actual project of the unmanned and solution for car warehouse, and finally, we will present the role that loading maneuver plays in the project.

A context of the project is automation. Automation replaces humans in some tasks, for example, to increase work efficiency. Automation is being implemented in various industries (e.g., series production) and in households (e.g., robotic vacuum cleaners). Efforts to automate are today's trend. People are trying to get rid of some of their responsibilities, and companies are trying to produce more efficiently and cheaper.

Some issues are resolved, others remain challenging.

This work falls under the logistics industry, which in 2021 is still a challenge for robotics. The greatest successes in this industry are in the solution of autonomous warehouses, where robots mainly operate with products, place them in the shelf. They often work in an environment with the presence of a human.

First law

The robot must not harm a human being or, by inaction, allow a human being to be harmed.

Second Law

The robot must obey the commands given to it by humans, except in cases where these commands would be contrary to the first law.

Third law

The robot must protect its own existence unless such protection is contrary to the first or second law. - Isaac Asimov

The environment of warehouses is mostly unchanging and organized, products are normally stored on standardized pallets. Situations in warehouses are often well predictable. Navigation can be realized based on the detection of placed markers. The problem's low complexity is perhaps why automation in warehouses is further than the next problem we will write about.

Autonomous vehicles are a topic to which significant attention is paid today. The leading company that manufactures self-driving cars is Tesla. The task is difficult for several reasons:

- vehicles move in an outdoor variable environment
- occurs on roads together with human drivers
- runs at high speed (error has great potential to kill)
- The use of autopilots is prohibited by laws

Our project is the service in an outdoor car park, where the path to it is similar to a public road and the orientation and transport in the car park itself is similar to the service in a warehouse.

There is a car factory, the cars produced are transported to the car park by the drivers, and the drivers are then transported back from the car park to the cars waiting to be stored. There are several benefits to replacing drivers with an autonomous manipulator. Space is saved because vehicles can be parked next to each other because there is no need to open their doors. In addition, robots do not need lighting to operate; they can save energy and help reduce light pollution.The manipulator is expected to be as fast as possible at each stage of its movement, while preventing a collision. The solution must be as close as possible to the optimal space and time efficiency.

Robot motion planning should be robust to changes in the environment. The environment and its attributes where the robot works do not have to be static. The surface on which the robot works can change from slippery to less slippery due to the weather. The same control inputs do not always lead to the same movement. Objects, where the robot is working can change their position. When moving around the car park, using GPS is sufficient. When moving near cars, a more accurate location is needed.

This work deals with the loading maneuver, specific to the limited operating space and requires precise localization. This maneuver is a critical element of the whole project. It has to be fast and faultless, and it is a challenge, especially in a limited workspace. It relies on sufficiently accurate localization based on information from lidar sensors.

2.2 Aim

This thesis focuses on a precise loading of a car by an autonomous non-holonomic polygonal mobile manipulator with an Ackermann-steer. The manipulator loads the car by driving its platform under the car. The motion must be precise and fast.

Maneuver has to minimize the duration and space requirements. These parameters will be measured.

The algorithm will be implemented in simulation, and its performance will be verified on a real platform.

3 State of the art

In this chapter we define used terms, then we briefly describe motion planning for mobile robots.

3.1 Terms and definitons

3.1.1 Localization

Localization is used for getting information about robot position. We have passive and active localization. Passive is about receiving data and just using them but, active is about actively looking for data, e.g., turning the camera or moving with the whole robot.

We can also divide localization as local and global. Local needs to know the previous position for further position estimation. The robot must get information about its initial position. Change in its position is measured and integrated. An example of localization of this kind is odometry.

Global localization is the method when a robot knows its position without knowing its previous position. The robot knows where it is after "waking up" and also after "being kidnapped," those are known problems (kidnapped robot problem, wake-up robot problem), which can not be solved using the local technique. An example of a global localization method is the GPS [2].

3.1.2 Mapping

Mapping is modeling an environment. Robot position is known [3].

Different types of maps are used. A topological map gives information about relations between places, objects, or features. A metric topological map adds information about the distance between items.

A Metric map provides the location of features according to some reference.

Cell-based maps divide an environment into areas. An example of a cell map is a grid. Using this map is based on discretizing a space. Objects are in that approach modeled this way: if an object is present in a cell, the whole cell is marked as occupied. The smaller cells are, the more accurate the environment's model is, but become less memory efficient, and computational time arises when navigating through the environment. Compromise between smaller cells and the bigger cell makes cells smaller near obstacles—so-called adaptive cell decomposition.

A semantic map gives places or objects meaning. For example, neural networks classify objects using visual data, therefore express the environment's semantic.

3.1.3 Configuration Space

Configuration space helps answer a question: "What if the robot is not a point?" asked by [4].
If the robot is not a point, usually is not, then it is convenient to define robot's configuration space.
Configuration of a robot is a complete specification of all its points. Values of the robot's generalized coordinates can describe the configuration. Generalized coordinates are independent parameters that define the state of the system uniquely. For example car on a 2D plane has three generalized coordinates – position (x, y) and orientation. Those parameters are independent since it is not possible to define one as a combination of others.
Configuration space is the space of all configurations. For example, a car's configuration space on a plane is all positions and orientations that a car can reach.
The dimension of a robot's configuration space is equal to a count of the robot's degrees of freedom. For example car on a plane has three degrees of freedom – 2 for a position and one for orientation. When the configuration space is defined, the configuration space's point defines a state of a robot.

3.1.4 The autonomy of robot

We can describe the level of robot's autonomy as teleoperated, semi-autonomous or autonomous.

An operator guides teleoperated robots; these robots can also be called telerobots. They are remotely controlled via wired and/or wireless communication networks. They find their applications in medicine or mediate access for humans to inhospitable environments, e.g., underwater and space [5].

Semi-autonomous can perform some actions autonomously without being guided.

Entirely autonomous robots are rare. Usually, users can influence an robot's behavior in some way. An example of an autonomous robot is a space probe; it operates far so that commands would have a considerable delay.

3.1.5 Mobile robots

Mobile is a robot that can change its location in space. It can move on the ground, in the air, or in the water. Robots moving on the ground can be, for example, wheeled or walking.
Considering directions in which a robot can move, we divide robots into non-holonomic and omnidirectional. Omnidirectional robots can move in every direction. Examples of non-holonomic wheeled robots are robots with a differential drive or Ackermann steer.

3.1.6 Differential Drive

Common steer system for mobile robots is differential drive its geometry is described in fig. 3.

3.1.7 Ackermann steer

Cars have Ackermann steer system. See fig. 2 describing the geometry of Ackermann steering.

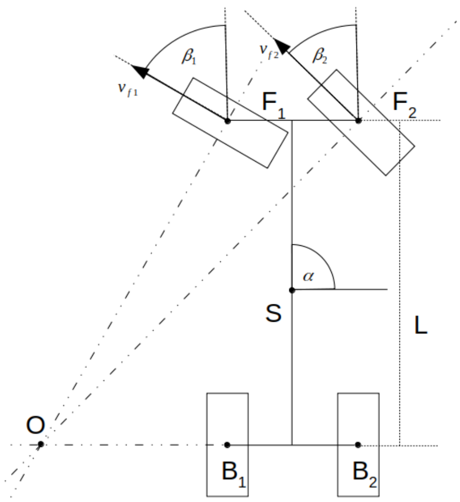

Figure 2: Ackermann steering

F - center of front wheel
B - center of rear wheel
S - center of line FB
O - center of rotation
α - orientation
β - steering angle
$\mathbf{v_f}$ - velocity of wheel's roll

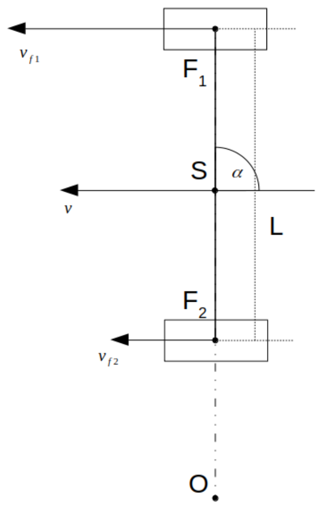

Figure 3: Differential Drive

F - center of wheel
S - center of mass
O - center of rotation
α - orientation
$\mathbf{v_f}$ - velocity of wheel's roll

3.2 Motion planning for mobile robots

Mobile robots can change their location and are usually equipped with a number of sensors. Those robots must be able to perceive the environment and adapt their actions based on their observations.

Mobile robots' motion planning consists of two steps:

1. path planning,
2. execution of the plan.

Both steps are usually done repeatedly to react to changing environments and keep plans and actions up to date.

The fact that the robot has a shape and often also kinematic constraints (e. g.: it can not move to the side or has limited acceleration) complicates the motion planning.

The approach, taking into account the robot's shape, is based on a definition of the robot's configuration space.

Algorithms searching for a path leading to a goal are of various types:

- Grid-based,
- Randomized sampling-based,
- Roadmap based,
- Artificial Potential Field Method,
- Algorithms designed for a specific type of robot.

Grid-based methods use Configuration space represented by a grid. The grid discretized the representation of space. The robot can move to adjacent cells and look for a path using graph search algorithms (e. g.: breadth-first search, A*). Path planning for large environments using these methods is computationally inefficient.

Randomized sampling-based methods rely on randomized sampling of configuration space. [6] introduced PRM algorithm, which is easy to implement, computationally efficient, and can be directly used for every holonomic robot. Another algorithm belonging to the same group is RRT, which was firstly published by [7]. This algorithm is suitable for path planning under non-holonomic constraints.

The Artificial Potential Field approach was presented for the first time by [8] and is based on defining a field of forces where obstacles act repulsively, and goal attracts.

Algorithms designed for a specific type of robot are, for example, Dubin's curves [9] which are suitable for a robot moving forward with a constrained steering angle. The planned path consists of arcs and straight lines. Reed and Shepp [10] added driving backward to Dubin's approach.

4 System description

4.1 Task definition

Mobile manipulator has to load a car. We placed origin of global coordinate system to the position of a car such that orientation of a car is 0 degrees. We assume initial orientation of the robot to be 90 degrees and initial position of the robot to be somewhere near a car and in second quadrant of global coordinate system and steering angle 0 degrees., see fig. 4.

Goal is to find set o control inputs in order to get from a initial configuration q_i to a goal configuration q_g.

Configuration of robot is defined as

$$q = (x, y, \beta, \alpha) \tag{1}$$

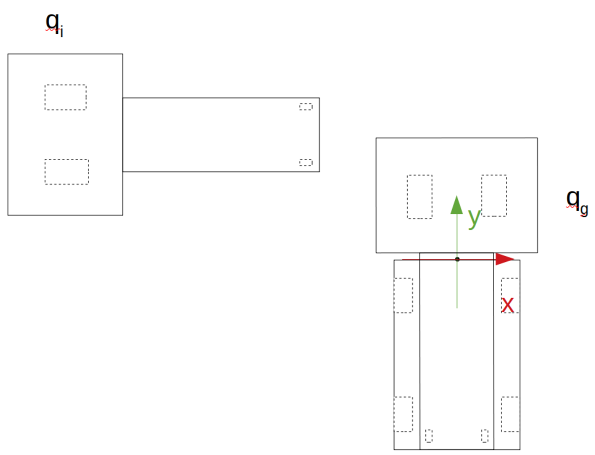

Figure 4: Initial position q_i and goal position q_g

4.2 Robot description

In this section geometry of the manipulator, its sensors, mathematical model and a control unit will be described.

4.2.1 Geometry and function

Fig. 5 displays Phoenix. Robot has a block in front, in the block are components: controlled wheels, motor, batteries and control unit. Back part is a low ramp. At the end of the ramp are small wheels (they are not visible in the picture 5 which are not controlled and can roll freely. On each longer side of the ramp are four rolls which are designed to connect (load) a car to the robot. The rolls are paired,one roll of the pair is controlled and can change a position from closed to opened.

Figure 5: Phoenix - side

Car is loaded following way: The robot reverse so as its ramp gets under the car. When each wheel of the car is between two rolls, controlled rolls begins closing process. Closing of the rolls causes that the vehicle is being lifted up.

When the rolls are fully closed the car is connected to the robot and it can be manipulated with it (loaded car is in fig. 6.

Chassis and geometry is described in a chapter 4.2.4.

4.2.2 Sensors

The robot is equiped whith sensors which are listed in table 1, their positon is shown in fig. 7

4.2.3 Control unit

There is a NUC i7 with an operating system Ubuntu 18.04 which hosts the communication between sensors, motion planning module and control.

Figure 6: Car loaded

Sensor	Type
front and two of the side lidar's	MICS3-CBUZ40
back middle lidar	LMS111
camera	Intel® RealSense™ Depth Camera D435

Table 1: List of the sensors

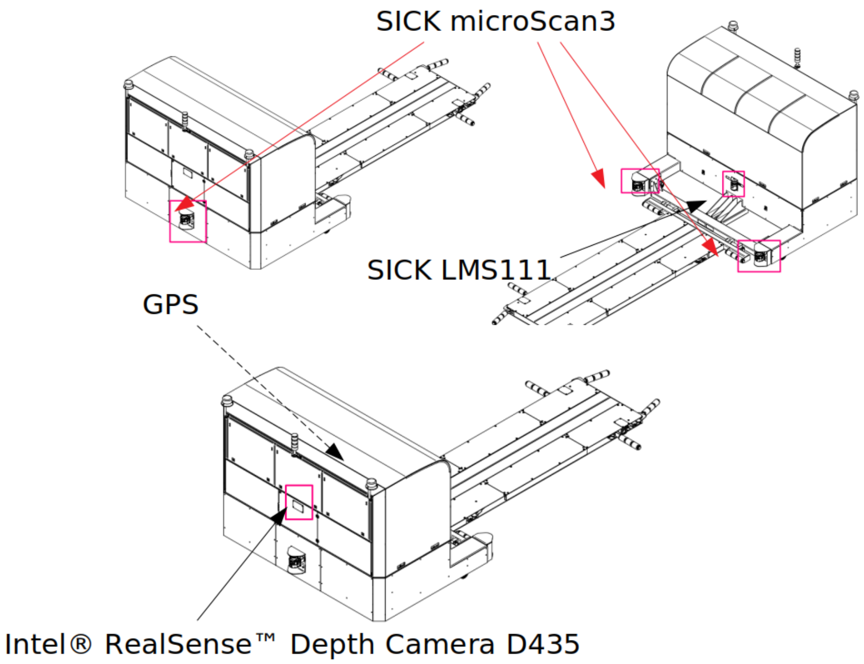

Figure 7: Position of the sensors

The communication architecture is built on ROS Melodic.

Robotic Operating System(ROS) is framework which is free open-source and provide tools and libraries for programming robots, it runs on Ubuntu.

4.2.4 Mathematical model of the robot

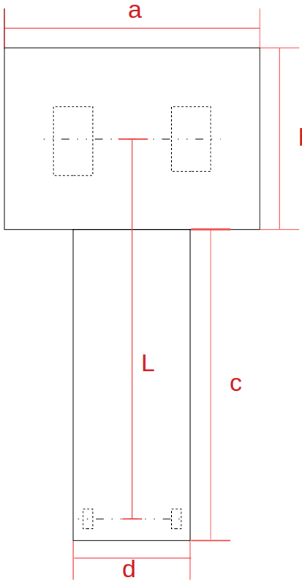

Figure 8: Shape of robot

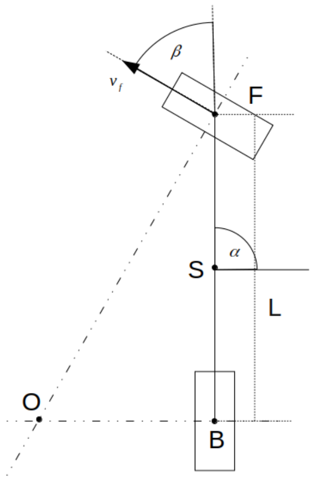

Figure 9: Bicycle model
F is center of front wheel

name	length [m]
a	2.6
b	5
c	3
d	2
L	4.2

Table 2: List of proportions

The manipulator has a polygonal shape, see fig 8. Its chassis has an Ackermann steering geometry. We used a bicycle model, see fig. 9, as chassis model to compute kinematic paths.

Available control inputs are described by

$$u = (v_{set}, \beta_{set}), \tag{2}$$

v_{set} is a target speed of front wheel's roll, β_{set} is a target steering angle.

Commands have transport delay d

$$v_{set}(t-d) \tag{3}$$
$$\beta_{set}(t-d) \tag{4}$$

.

The state of the robot is described by

$$q = (x, y, \alpha, \beta) \tag{5}$$

(x, y) is position of the robot, α is its orientation, and β is steering angle of the wheels.

We know that $\dot{v_f}$ and $\dot{\beta}$ are functions of control inputs and time;

$$\dot{v_f} = f_1(v_{set}, t) \tag{6}$$
$$\dot{\beta} = f_2(\beta_{set}, t) \tag{7}$$

, v_f is the speed of the wheel's roll.

Combining the bicycle model with definitions of f_1 and f_2 we get a simplified mathematical model of the system.

$$\begin{aligned} \dot{x_f} &= v_f \cdot \cos(\alpha + \beta) \\ \dot{y_f} &= v_f \cdot \sin(\alpha + \beta) \\ \dot{\alpha} &= \frac{v_f}{L} \cdot \sin(\beta) \\ \dot{v_f} &= \operatorname{sgn}(v - v_{set}) \cdot c_1 \\ \dot{\beta} &= \operatorname{sgn}(\beta - \beta_{set}) \cdot c_2 \end{aligned} \tag{8}$$

c_1 and c_2 are constants, $(x_f,\ y_f)$ is position of point F, v_f is speed of front wheel's roll, α is orinetation of robot, β is steering angle, L is distance between axis of front and rear wheel, v_{set} is set speed, β_{set} is set steering angle.

4.2.5 System identification

transport delay (d)	150 ms
steering speed (c_2)	$40°s^{-1}$
rolling acceleration (c_1)	$0.7\ m \cdot s^{-2}$
max rolling speed	$\pm\ 3\ m\cdot s^{-1}$
max steering angle	$\pm$ 115°

Table 3: System identification

5 Loading maneuvers

This section describes proposed loading maneuvers. We chose static maneuvers which are designed for our specific task and robot. For planning we had just location of a car available.

5.1 Simple maneuver

Firstly, the robot drives backward until the sharpest possible rotation's center lies on a car's longitudinal axis. Distance from the line is proportionally controlled.

Then, wheels steer is performed in order to set demanded center of rotation.

Later, the speed of the wheels roll is set non-zero. Proportional control will ensure zero angular shift between the car and the manipulator.

After the previous completed, the wheels turn to be parallel with the robot. If the car's and the manipulator's angular and traversal reciprocal shift remained zero, the robot finally reverses to under the car. In the fig. 10 is shown flowchart for the algorithm, fig. 10 illustrates the maneuver.

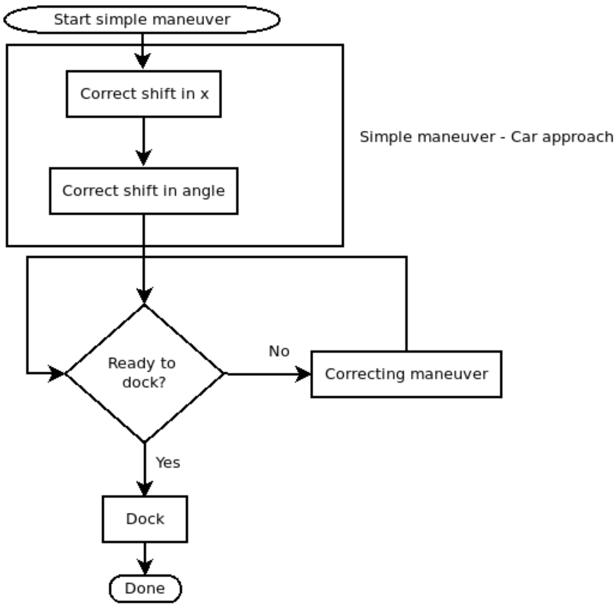

Figure 10: Flowchart: Simple maneuver

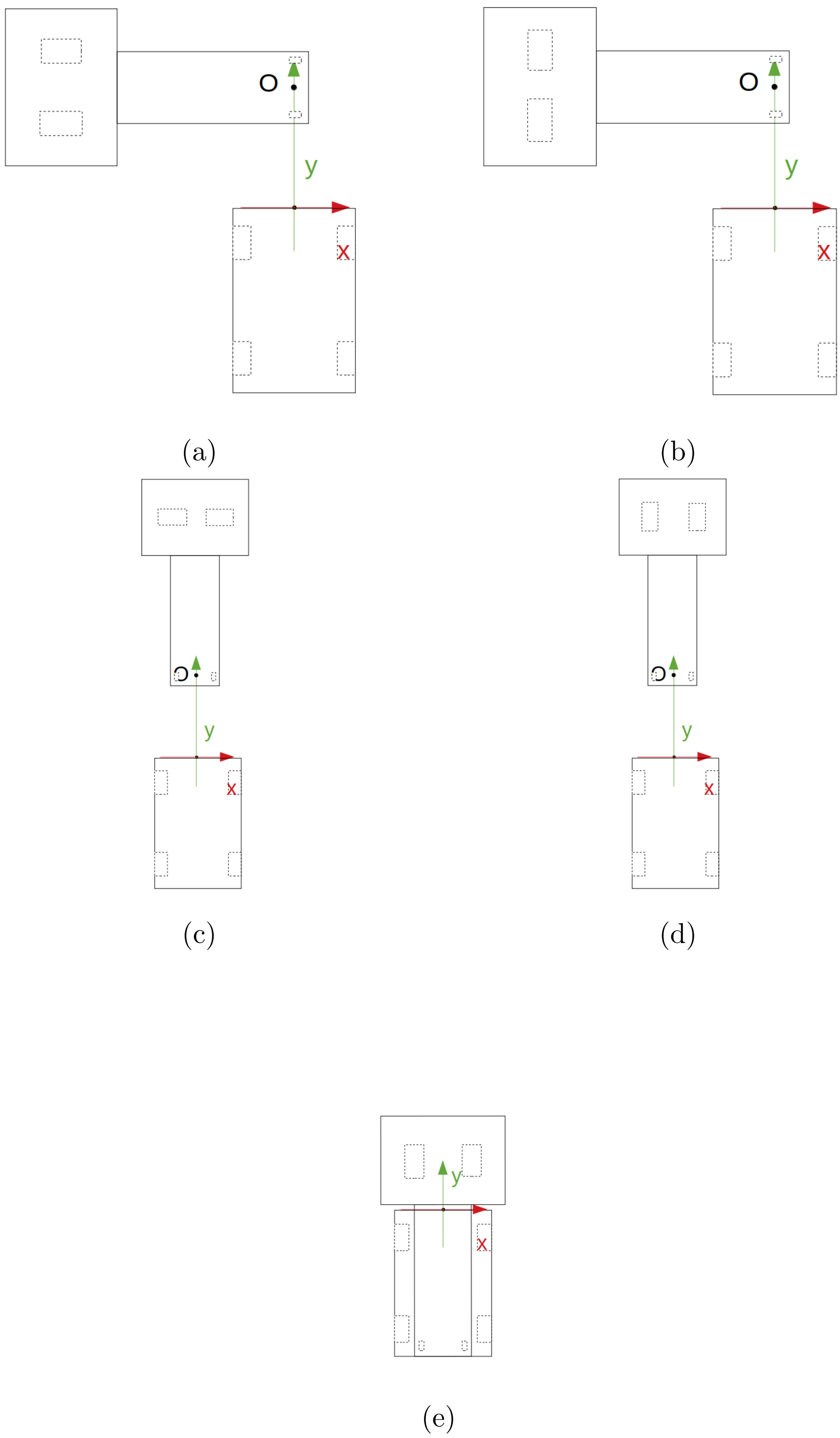

Figure 11: Diagram: Simple maneuver

5.2 Correcting maneuver

Algorithm of correcting maneuver is described in fig. 12

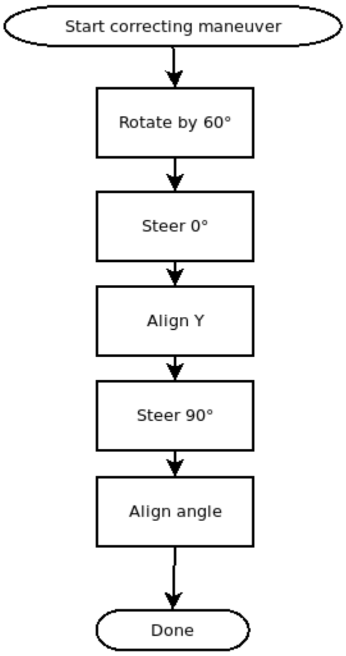

Figure 12: Flowchart: Correcting maneuver

5.3 Continuous maneuver

This maneuver is based on simultaneous reduction of shift in angle and position. Flowchart of the maneuver is in the fig. 13.

Continuous maneuver Car approach terminates if the manipulator is too close to the car.

It was expected that this maneuver will be faster than Simple maneuver.

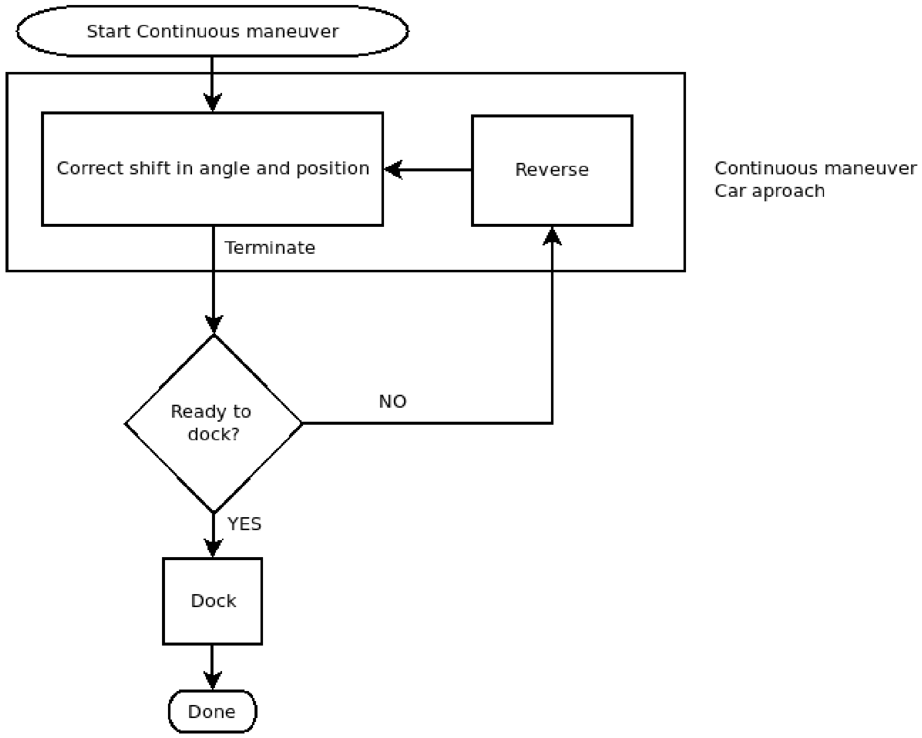

Figure 13: Flowchart: Continuous maneuver

5.4 Circle maneuver

Flowchart of the Circle maneuver is in the fig. 14.

Steering angle is computed followingly (for diagram see fig 15):

1. we take longitudinal axis of a robot and a car,
2. we design a circle which is tangent to robot's longitudinal axis and car's longitudinal axis. The rear wheels of the robot must be tangent to the circle.
3. Steering angles of the front wheels are given by center of rotation, which is the center of the designed circle.

Manipulator drives and steering angles are being updated until following terminating conditions are reached:

- the point between rear wheels of the manipulator has reached longitudinal axis of the car,
- the point between rear wheels of the manipulator is too close to the car.

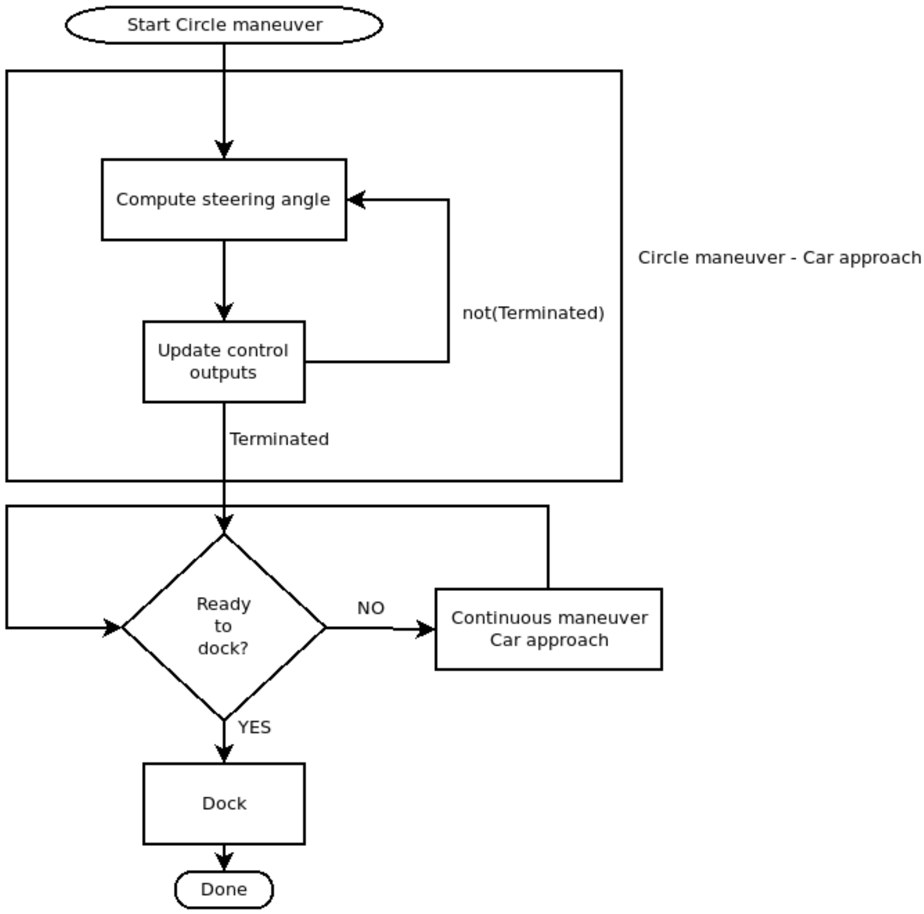

Figure 14: Flowchart: Circle maneuver

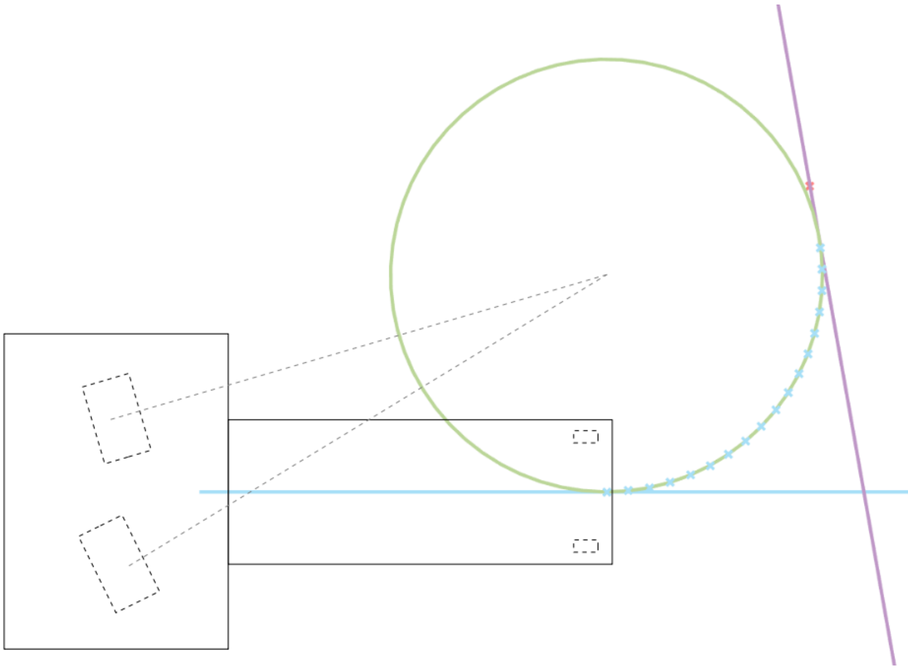

Figure 15: Circle maneuver

position of a car is the red cross

6 Experiments

Experiments will be described in this section a their results are shown in section 7.

First tests were performed in a 3D robotic simulator Gazebo. The simulator includes physics engine and it is able to simulate sensors. It is integrated with ROS, which allows to run the same code in simulation and on the real robot. Gazebo is free open-source, which reduces cost of developing and testing robots and algorithms.

Phoenix Lipraco motion planning module runs in ROS and for localization during loading it uses laser scanners. We created an simulated environment with robot and car. Nodes for car sensing with laser scanners where created but in our particular task for developing motion planning module we just worked with global coordinates of car and robot which are provided by Gazebo itself. We simulated just kinematic model and did not use physic engine since real parameters such as weight of manipulator were not known. Kinematic model was implemented according to mathematical model of robot (8) and system identification experiments made in section 4.2.5.

6.1 Loading maneuvers

The implementation procedure was as follows:

1. Preparation in Gazebo simulation
2. Transfer code to an real robot
3. Improving the algorithm based on experiments with the real robot

6.1.1 Simple maneuver

First, we implemented a Simple approach maneuver. There were several problems with the implementation to the real robot:

- Both front wheels of the car are not clearly visible if there is sharp angle between the car and the manipulator. Therefore, the correct position of the point in between the wheels is not available.
- The axis of rotation with a minimal radius does not lie in the middle between the rear wheels of the robot, because of robot design.
- Steering the wheels on the spot causes significant wear and tear.

The first problem in the list solved by rotating the robot until both wheels were seen properly. To determine the position of the center of rotation with minimal radius, we performed following experiment:

1. Steer wheels to maximum
2. Drive till orientation is changed by 180
3. Drive opposite direction to reach initial orientation
4. Repeat from step 2

During the experiment the position of middle of the rear wheels was tracked, see the fig. 16. In the fig. 16 can be seen, that the position of tracked point oscillates according to direction

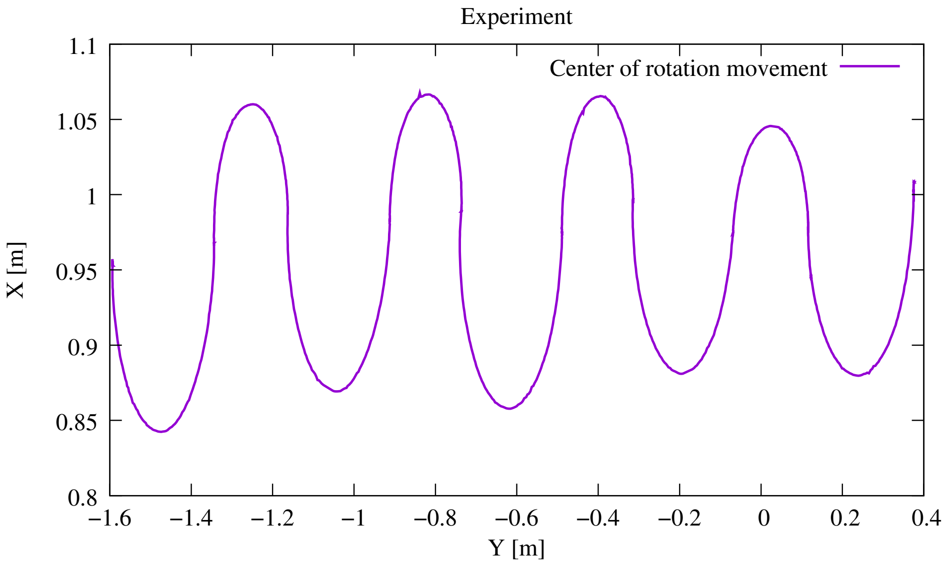

Figure 16: Position of middle of rear wheels of manipulator when turning with minimal radius

of Y axis and at the same time moves towards positive direction of X axis. The movement to the right appeared because the road had slight declining slope to that direction.

It often happened on a real robot, that error between estimated center of rotation and real center of rotation was big. After finishing align angle phase there was significant lateral shift between car and the robot and the robot could not load the car by simple reversing. In that situation the Correcting maneuver was used.

After solving the problems, time and space required were measured.

6.1.2 Continuous maneuver

A real robot with a Continuous maneuver algorithm made several attempts to load a car. We made an average of duration of the maneuver and logged its space requirements.

6.1.3 Circle maneuver

The simulation showed that the robot does not follow the circle accurately enough when reference speed is too high and does not have the correct orientation at the end of the maneuver. It was necessary to add a speed controller, which took into account a difference between set steering angle and real steering angle.

There was a problem with steering angle setting on the real robot. The control unit of the steering angle caused that the set angle overshot reference angle significantly before settling at a correct value, see fig. 17. We solved that problem by sensing the actual position of the wheels and starting the maneuver after the settling time had elapsed.

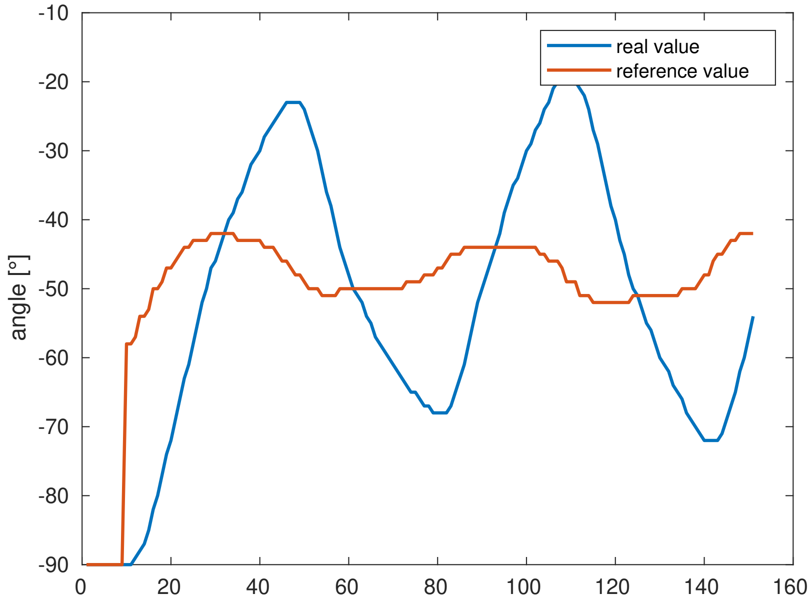

Figure 17: Overshoot

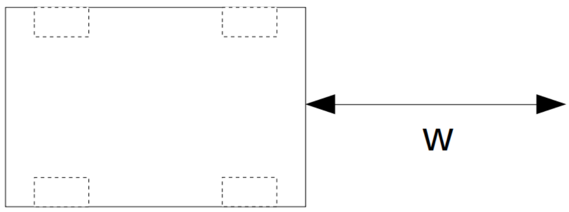

Figure 18: Aisle width

7 Results

In this section is described how much space and time proposed maneuvers required.

7.1 Space requirements

Following table shows width of a aisle (for w, see fig. 18) needed for loading a car.

Maneuver	w in simulation [m]	w in reality [m]
Simple	7	8
Continuous	7	8
Circle	5.6	6

Table 4: Space requirements

7.2 Time requirements

Tables 5, 6, 7 show duration of phases of the maneuvers implemented on the real platform. Table 8 compares duration of all three maneuvers.

	Phase	Phase duration [s]	Note
1.	Simple maneuver car approach	31	
2.	Rotate by 60°	14	Correcting maneuver
3.	Steer 0°	5	Correcting maneuver
4.	Align Y	12	Correcting maneuver
5.	Steer 90°	5	Correcting maneuver
6.	Align angle	12	Correcting maneuver
7.	Steer 0°	5	
8.	Move under car	8	
9.	Close rolls	15	
10.	Move away	13	

Table 5: Time requirements: Simple maneuver phases

	Phase	Phase duration [s]	Note
1.	Continuous maneuver car approach	30	
2.	Move under car	8	
3.	Close rolls	15	
4.	Move away	13	

Table 6: Time requirements: Continuous maneuver phases

	Phase	Phase duration [s]	Note
1.	Circle maneuver car approach	9	
2.	Continuous maneuver car approach	30	executed if approach failed
3.	Move under car	6	
4.	Close rolls	15	
5.	Move away	13	

Table 7: Time requirements: Circle maneuver phases

Maneuver	Minimal duration [s]	P
Simple	120	0.5
Continuous	66	0.3
Circle	43	0.1

Table 8: Time requirements: minimal and average duration

P - probability of unsuccessful Car approach or Correcting maneuver or Continuous maneuver

7.3 Discussion

Fastest, most space efficient and accurate maneuver was Circle maneuver.

8 Conclusion

The work dealt with the design of a loading maneuver for an autonomous manipulator. The maneuver is a critical component in the autonomous transport of cars to the storage car park. We designed three maneuvers, which we tested in a simulation and on a real platform. The maneuvers were compared concerning space and time requirements. The proposed maneuver - Circle maneuver, was evaluated as the fastest and least space-consuming.

Due to significant differences in simulation and reality and the failure rate of the real robot, we were not able to further develop the motion planning module. We would like to test the proposed maneuver further and compare it with a more robust motion planning approach.

Bibliography

[1] Philip Polack, Louis-Marie Dallen, and Aurélien Cord. Strategy for automated dense parking: how to navigate in narrow lanes*. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9196–9202, 2020.

[2] Marek Skalka. *Srovnání lokalizačních technik*. PhD thesis, Univerzita Karlova v Praze, Matematicko-fyzikální fakulta, 2011.

[3] Terry Payne. Robotics and Autonomous Systems - Lecture 7: Maps and mapping (2017), 2017.

[4] Howie Choset. Ch. 03: Configuration Space - Robot Motion Planning, 2007.

[5] Sarmad Mehrdad, Fei Liu, Minh Tu Pham, Arnaud Lelevé, and S. Farokh Atashzar. Review of advanced medical telerobots. *Applied Sciences (Switzerland)*, 11(1):1–47, 2021.

[6] Lydia E. Kavraki, Petr Švestka, Jean Claude Latombe, and Mark H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[7] Steven M. LaValle. Rapidly-Exploring Random Trees: A New Tool for Path Planning. 1998.

[8] O.Khatib. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. pages 500–505, 1986.

[9] L. E. Dubins. On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents. *American Journal of Mathematics*, 79(3):497, 1957.

[10] J. A. Reeds and L. A. Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, 145(2):367–393, 1990.