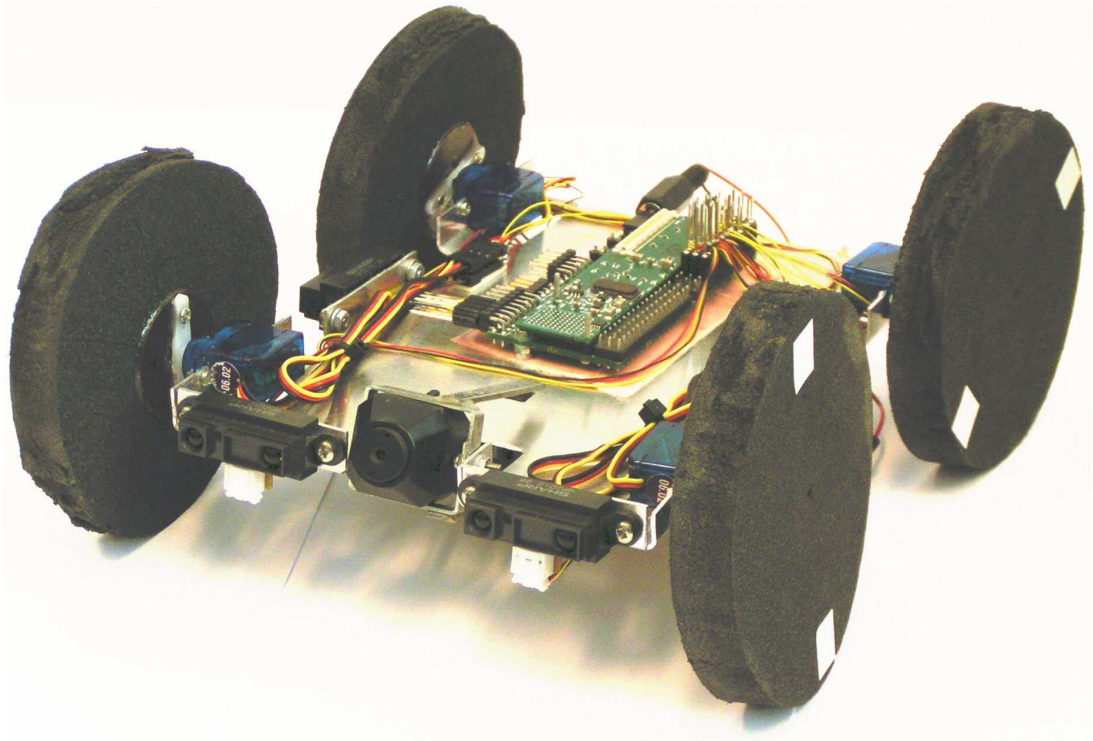
Path Finding Autonomous Car, Designed for Room Mapping



Group members of IAS10-1032d

Kristian Borup Pedersen
Thore Svejgaard Wienike

Fall 2006 - Spring 2007

Aalborg University
Institute of Electronic Systems
Department of Control Engineering

Institute of Electronic Systems
Department of Control Engineering
Aalborg University

TITLE:

Path Finding Autonomous Car, Designed for Room Mapping

PROJECT PERIOD:

IAS9-10,
September 4th, 2006 -
June 7th, 2007

PROJECT GROUP:

IAS10-1032d

PROJECT PARTICIPANTS:

Kristian Borup Pedersen
Thore Svejgaard Wienike

SUPERVISOR:

Anders la Cour-Harbo

COPIES: 4

NUMBER OF PAGES: 131

ATTACHMENT: 1 CD-ROM

ABSTRACT:

Abstract:
This project deals with the design and construction of an autonomous car able to video map an arbitrary room. The main topics is developement of a mapping pattern, construction of the physical car, and designing a controller for the car.
First the patterns are discussed and the better is found. The best pattern depends on driving alongside the walls and making a sweep with the camera every two meters and in the corners.
The car has to be able to be carried by an autonomous helicopter and the weight of every hardware component is considered.
Two controller are designed, a P and a PI. The P controller is by simulation found to be the best of the two. The controller is furthermore used to compensate for feedforwards used in the mapping pattern.

Preface

This report, dealing with autonomous robotics, is documentation of the work of group 1032d, doing a master thesis on intelligent automomous system in control engineering. The report is written as a part of the Civil Engineer education (M.Sc.) in Intelligent Autonomous Systems at Aalborg University. The project work took place from Sebtember 4th 2006 to June 7th 2007, with Anders la Cour-Harbo as supervisor.

The report is divided in two parts; the main report and the appendix. The main report has six chapters: Chapter one describes the project and outline the problem formulation. Chapter two uses Structured Program Development (SPD)[Biering-Sørensen et al, 1994, page 69-96] to make the functionality requirements, and the accepttest specification. Chapter three discusses the hardware system and its components. Chapter four discusses the developement of a controller. Chapter five ends the main report with a discussion and a conclusion of the project.

SPD is a method developed for software developement and is originally directed at software developers and project leaders. In this project it will be used for full system analysis, as the method leads to a testable way of making a specification of requirements.

Citations throughout the report are indicated by numbers and optional page or chapter, e.g. [Biering-Sørensen et al, 1994, page 69-96].

The enclosed CD contains the report in PDF, the Matlab source code, developed C software, and datasheets included in the litterature list.

Kristian Borup Pedersen

Thore Svejgaard Wienike

Contents

Chapter 1

Introduction

Contents

In everyday life, the use of robots is increasing. Both consumers and the industry call for this evolution, which also increase the demand for more advanced robots. To meet these demands, large investments, in developing new more advanced robots, are made
[The Danish Ministry of Science, Technology and Developement, 2006].

The robots developed today are, by many, considered to have some sort of intelligence. For this to be true the robots will, at least, have to be autonomous and have cognitive capabilities.

This project is based on developing an autonomous robot. The main purpose is to make a video map of a room based on algorithms using sensor inputs and PWM outputs for the motors. The robot developed is a car, which is designed and contructed specifically for use in this project.

1.1 Motivation

What do we do if we want to see places we cannot, or will not, go to? Places that may be dangerous for humans to enter or places where other humans are causing a threat.

One thing to do is to send a robot with an on-board camera to film the place. This robot can be controlled remotely or maybe even better, drive around autonomously mapping the area in question.

The motivation for this project is based on thoughts such as these. More specifically an example scenario of such a situation is used to help define the perspective of the project.

Association for Unmanned Vehicle Systems International (AUVSI) has a competition in creating autonomous vehicles. And for a part of this competition three example missions are created. One, a hostage rescue, two, a nuclear power plant melt down, and three, a biological disaster. The competition is called International Aereal Robots Competiotion (IARC). The idea and rules of IARC will be used in this project.

The main topics of this project are pattern planning, controller design, and simulation of situations the robot should be able to handle. The pattern algorithms are part of what makes the robot autonomous, and the foundation of the robots movement. A pattern using feedback control will propably work well. Using a controller opens up for the posibility to see if it can compensate for the errors the feedforward algorithms introduces. Simulations are used to see how the robot will react in different situations, for instance when encountering an object.

1.2 Issue Domain

1.2.1 Mapping Environment

According to the rules of IARC the car is to be send into a room through a window. This means that the car will land in a random spot on the floor, pointing in a random direction, and that it is possible it turns upside down. These three factors are unknown about the car's position in the room. The room itself is likewise unknown and as such a controlled test environment is made. This has to include the possible obstacles that can appear. The only thing known about the room is that it is in a house and therefore, some assumptions can be made. The two issues mentioned, obstacles and assumption, are described in the following two sections.

Room Obstacles

Two obstacles are considered, doors and furniture legs. The doors are considered to have two positions only, either closed or fully opened, in which case an open door can be considered as a gap in the wall and a closed door can be considered as part of the wall. This is illustrated in Figure 1.1.

Furniture legs can be from tables, chairs, sofas etc., which means that there will always be four or more legs from each piece of furniture. Moreover, all legs are considered to be taller than the height of the car, so that the car can drive beneath the furniture.

Room Assumptions

The room is within a house and as such it can only assume certain dimensions. We chose to defined these for the test environment along with the sizes of doors and obstacles.

Shape: The room will have nothing but 90° corners. These can be both open and closed.

Walls: all walls are each between one and six meters long and the height at least 2.50 m, so that people are able to walk around upright.

Furniture legs: The dimensions of these are approximately 5×5 cm and the height is more than the height of the car.

Doors: In Denmark the standard door width is 82.5 cm, which therefore will be the width of the doors in the test environment.

In Figure 1.1 some examples of different rooms are given, all of which the car should be able to map.

1.2.2 International Aerial Robotics Competition

IARC is a competition in designing and constructing autonomous vehicles that are able to sense the surrounding environment. It is divided into four levels of completion. The first two levels are carried out by en autonomous helicopter, carrying the car from this project. Level one and the first part of level two is illustrated in Figure 1.2. The helicopter has to fly from its start position to a cluster of buildings. This is a flight of approximately three kilometers. When the buildings are located it has to find a specific building marked with the competition logo.

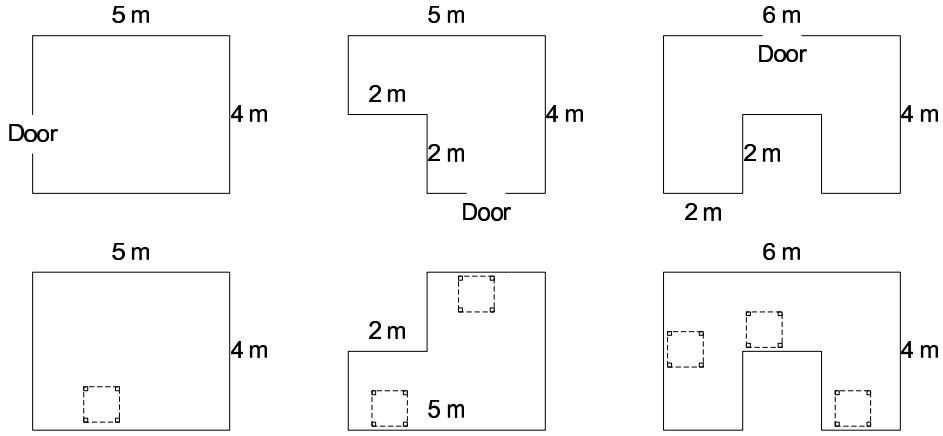


FIGURE 1.1: The mapping environments, with an example of the two types of obstacles considered.

The final part of level two to primo level three is shown in Figure 1.3. Once the building is found, the car has to be send into a room in the building in order to commence its mapping procedure.

For the car to be eligible for the competition there are rules and boundaries which it will have to comply with. It must...

1. be fully autonomous
2. not be independent of the helicopter to complete the competition.
3. be launched from the ground or air under command of the helicopter.
4. fly or be carried the 3 km to the cluster of buildings.
5. be started before the helicopter is converted to automatic control.
6. be able to map the room in less than 15 minutes.

1.3 Problem Formulation

Taking the three aspects in the preveous section, Issue Domain, into considerasion, the project's main problem can be expressed:

- *How can we design and construct an autonomous car able to send a video map of an arbitrary room to an autonomous helicopter, without the use of image recognition and with the extra aspects that the car has*

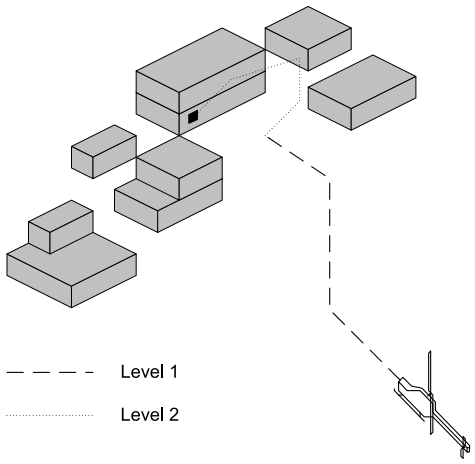


FIGURE 1.2: To complete level one, the autonomous helicopter has to fly from the ground station to the town. Within the town it has to find a specific, marked, house to complete level two.

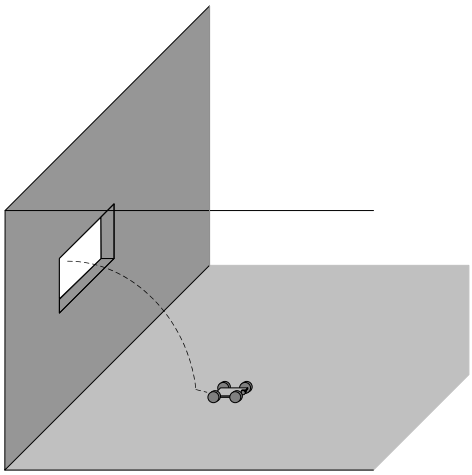


FIGURE 1.3: The last part of level two is sending the car into the house, through a window. Level three is completed when the entire room is mapped and either video or pictures of the entire room is send back to the ground station.

to be light enough for the helicopter to carry it and robust enough in its physical construction to survive the impact with the floor after being passed through an open window.

Chapter 2

Analysis

Contents

The analysis takes the perspective of the car, as there is no influence from any user, once the car is activated. That is a view on how the car will interact with its surroundings and how it needs to communicate with the helicopter.

The method used to analyse is based on the SPD method to find the functionality requirements.

2.1 General Description

2.1.1 System Description

The system is the autonomous car. An overview of the system is given in the deployment diagram in Figure 2.1. With this in mind, the components of the system will be described.

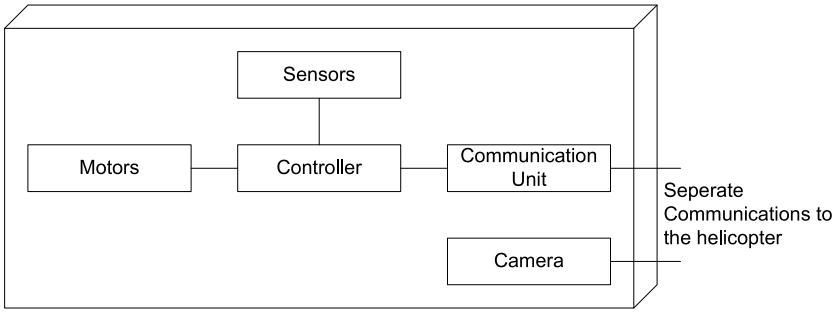


FIGURE 2.1: Deployment diagram of the system.

Controller

This component controls the timing of the inputs and outputs, regulates the speed of the motors, calculates the path, and has a signal send to the helicopter, when the mapping is done.

The only component the controller does not interact with is the camera, as it transmits its pictures independant of the rest of the system.

Motors

The motors of the car are used solely for the car propulsion. That is, there are no motor controlled movable parts mounted on the car except for the wheels.

Sensors

There are four obstacles the car can encounter, as described in 1.2.1. To detect these, the car has four sensors mounted; two sensors pointing forward and one pointing to either side.

Furthermore there is a need for the car to know whether it has the bottom of it facing the floor, as it can flip over during the flight through the window,

described in 1.2.2 International Aereal Robotics Competition. Due to this a sensor is placed facing downwards.

Communication Unit

The car control is activated from the begining of the IARC mission. The only communication necesary from the helicopter to the car is a start signal. The communication needed for the car is to send a signal when it has completed mapping.

Camera

The camera is working independently of the rest of the system. It is activated, along with the rest of the system, at the begining of the IARC mission. The only communication the camera performs is continously transmitting pictures to the helicopter.

2.1.2 Functionality

The car has some main functionalities, like a motor control to handle the communication to the motors. The main functionalities are shown in Figure 2.2. Also in this figure the directions of communication between them are outlined.

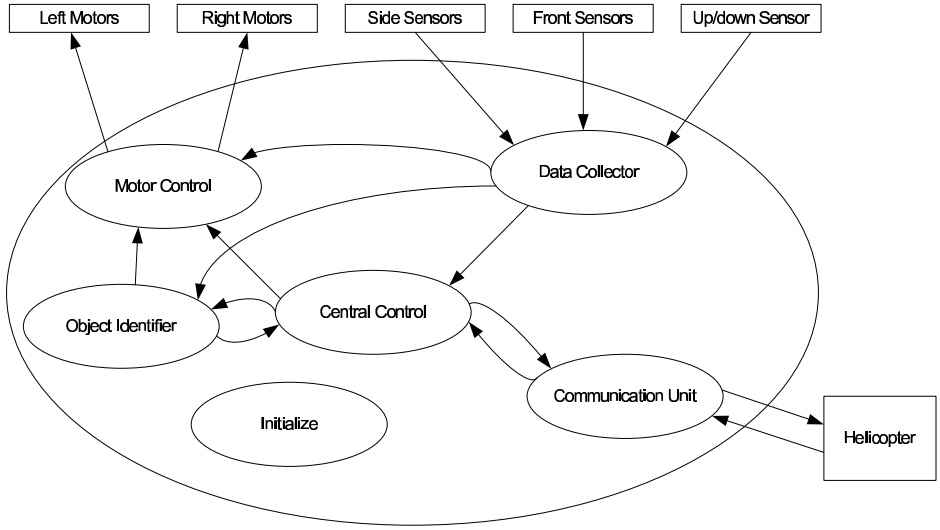


FIGURE 2.2: The main functionallities and communications internally.

The car has to drive in a straight line forward, following the walls. To do this the measurements from the side sensors are used. The car will have to take measurements from the side sensors with a suitable frequency, which is chosen to be every five centimeters, as it assumably leaves enough time for the motor driver functionality to compute the response to changes.

The Motor Control should take care of all the basic functionalities, like drive forwards, drive backwards, rotate, and stop. While the Central Control should take care of the more advanced issues, like where to go next, when encountering an object. The Central Control has furthermore been divided into two extra functionalities, the Object Identifier and the Communication Unit, to point out the importance of these.

The Object Identifier is used when an object is detected, for instance a wall or a piece of furniture, to identify what sort of object it is to help the central control determine how to react.

The car acts on its own, without influence from the helicopter or other external device, during the mapping process. This means, the only data-communication taking place is a transmission from the helicopter to the car when the mapping should start and a transmission from the car to the helicopter when the mapping is complete. This functionality is provided by the Communication Unit.

2.1.3 Limitations

The car is develloped in order to handle:

- Rooms with 90°.
- Corners and no holes in the floor, e.g. stairs down.
- Doors that are 82.5 *cm* wide, which is one of the commonly used door width i Denmark [SWEDOOR-KILSGAARD, 2007].

2.2 Mapping Considerations

For the car to efficiently map the room, the pattern will have to be predefined. To find an efficient pattern some considerations are made. These considerations are the topic of this section.

To choose which pattern is the better one, six parameters are used.

1. Room coverage, by the camera.

2. Complexity as function of how much online calculation is needed.
3. Handling of unprecise hardware.
4. Complexity of handling obstacles.
5. Addaptability to different shape rooms.
6. Time to map the room as a function of room size.

The following patterns are described by a normal case scenario, with no obstacle, whereafter the handling of exceptions are described. The considered exceptions are, objects, open doors, and different shape rooms. Ending each pattern description is a discussion of the pattern and how well it live up to the parameters.

At the end of the mapping considerations all patterns are held up against each other. To conclude on which is the best, the patters will be given point, 0,1, and 3, depending on wheter it is the best to fulfil the parameter, where 3 points are given to the best pattern. The overall best pattern will be the one with the highest score.

2.2.1 Mapping by Continous Driving

The basic idea with this pattern is to have the car drive forwards in countinous motion. This means the car have to make turns when encountering walls, as shown in Figure 2.3. The pattern on the figure is the normal case scenario for this pattern.

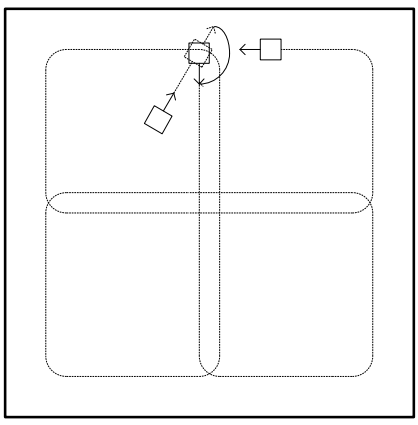


FIGURE 2.3: Continous Driving example, normal case scenario.

Normal Case Scenario

1. The car lands and stabilizes in a random possition in the room.
2. The initial direction of driving is directly forward from the position the car stabilizes in.
3. When detecting a wall with both front sensors, the car stops and corrects the direction by rotating until the driving direction is ortogonal to the wall, pointing away from the wall.
4. The car drives forward until the front sensors detect a wall and then makes a 90° arc turn to the left with a radius of 200 mm. The width of the room is now known by the car.
5. Same procedure as previous step. This distance is likewise stored.
6. Using the width of the room, the car drives half that distance, including a 90° arc turn to the left.
7. The car drives forward until the front sensors detect a wall and then makes a 90° arc turn to the left. The length of the room is now known by the car.
8. The rest of the mapping is based on calculating the drive distances from the three known distances.

Exceptions

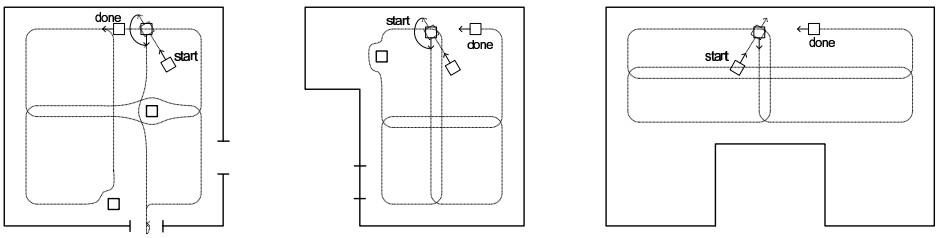


FIGURE 2.4: Exception for the mapping by continous driving pattern.

Objects: As this pattern is based on the car's ability to drive continously, the handling of objects is as dynamic as the normal case scenario. This means the car will drive around any object and estimate the distance

driven corresponding to the current forwards direction. The handling of objects is shown in Figure 2.4. There is one problem associated with the handling of objects, which is likewise relatted to the no-stop strategy. The car will have to distinguish between walls and objects while moving. This means the car will be close to the walls before it determines that it is a wall, which leaves less time for the onboard micro-computer to calculate the following actions.

Open doors: This exception will require the car to stop or at least change the rotation direction of the wheels, which will seem like a stop-and-go motion. This will only happen if an open door is encountered when the car uses its front sensors, but as most of this pattern is based on calculation on how far the car needs to drive in a certain direction it will only be relevant in the beginning of the mapping process. Examples of this exception is shown in Figure 2.4.

Different shape rooms: This pattern is based on mapping rectangular rooms and as such cannot be used to map rooms which differs from the rectangular room, as shown in Figure 2.4.

Camera Coverage

The coverage of a rectangular room is shown in Figure 2.5. Though the pattern is not suited for rooms of different shapes, it can cover certain shapes of rooms. These are also shown in Figure 2.5.

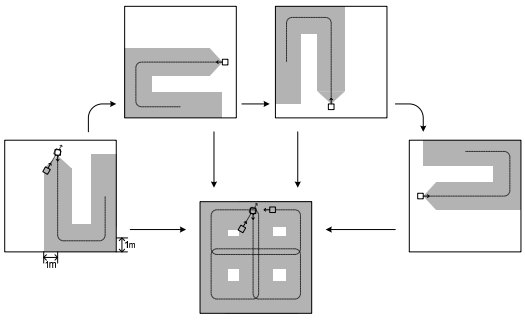


FIGURE 2.5: The coverage by the camera of the rectangular room.

In the figure, two tints of gray are used; the darker on is what the camera can film within 2 m, the lighter is what part of the room the camera can cover, though without getting a good view of the walls. Furthermore the parts of the walls, which at some point has the camera pointing directly at it, at a

distance of about 2 m is marked with a broader line than the rest of the wall. This way of showing what part of the room is covered how well, will be used for the other patterns as well.

Discussion of the Pattern

This pattern can be used in a rectangular room and a limited number of other room shapes. A disadvantage is that some walls in the rectangular room is covered only by driving alongside them, which means that the camera never will be poining directly at all walls seperately.

The complexity of the normal case scenario is low as only a few online calculations are needed and all of these are based distances driven.

While following walls, some control of the distance to the walls can be implemented, which means the car has a way to compensate for unprecise hardware, but as this pattern also takes the car across the room with no walls within sensor distance, some estimations of how far the car has driven in the intended, and in the not intended, direction will have to be made.

The handling of exceptions with this pattern, might bring the car too close to the walls, as it has to identify them firstly, and when encountering a door it can affect the idea of continous driving adversely.

The time it takes to map the room is a function depending on the width and length of the room. The number of turns is constant for a room within the limitations given in 1.2.1, that is $turns = 11$, which is multiplied with the time it take to make a turn. The speed of the car is estimated to be $1\ {}^{km}/_{h} \approx 0.28\ {}^{m}/_{s}$. The time estimation equation for this pattern is given in equation 2.1. The distance, 1 m, subtracted from the width and length is the esteimated distance the car will drive from the wall plus the distance needed to make a turn, as illustrated in Figure 2.6.

$$\begin{aligned} t &= \frac{4 \cdot (lenght - 1\ m)}{0.28\ {}^{m}/_{s}} + \frac{4 \cdot (widht - 1\ m)}{0.28\ {}^{m}/_{s}} + \frac{11 \cdot \left(\frac{0.2 \cdot 2\pi}{4}\right)}{0.28\ {}^{m}/_{s}} \\ &= 14.3(length + width) - 16.2 \end{aligned} \tag{2.1}$$

2.2.2 Mapping by Grid

This pattern is based on getting a direct view of all the walls at a distance of approximately two meters. This also means that it depends more on rotations than turns and that the car will display more of a stop-and-go pattern, than

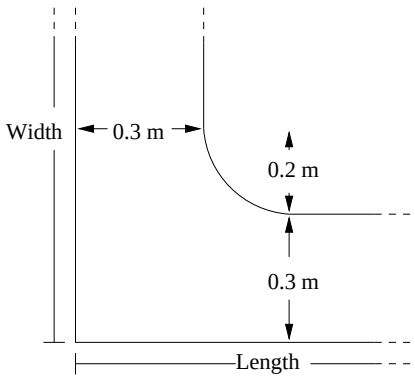


FIGURE 2.6: When calculating the time it takes to map a room, the distance to the walls and the turn radius are subtracted from the length and width.

with mapping by continous driving. This pattern's normal case scenario is shown in Figure 2.7.

This normal case scenario does not use the width and length of the room, but relies on making a grid as it moves around in the room. Like with Mapping by Continous Driving, this is described through steps.

In this scenario the filming distance from the camera to the wall is defined to be 2 m.

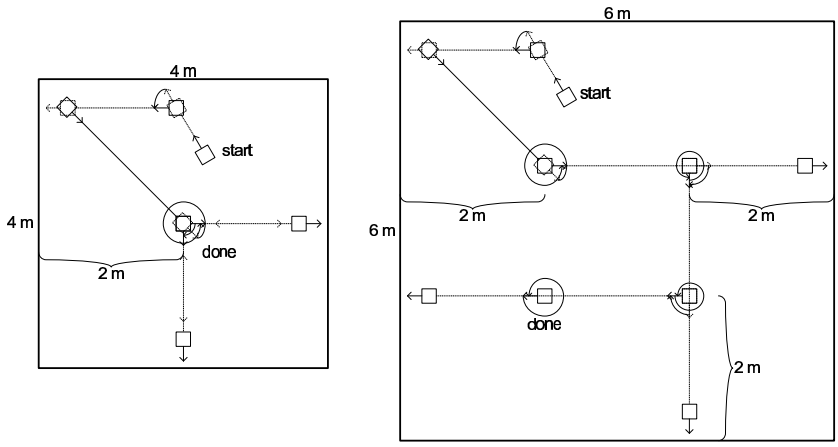


FIGURE 2.7: Mapping by grid used in a 4×4 m room and a 6×6 m room.

Normal Case Scenario

1. The car lands and stabalises in a random possition in the room.
2. The initial direction of driving is directly forward from the position the car stabalises in.
3. When detecting a wall with both front sensors, the car stops and corrects the direction by rotating left until the driving direction is parallel to the wall.
4. The car drives forward until the front sensors detects a wall, at a 300 mm distance. The car then rotates left 135°.
5. At the first grid spot, the center of the car is 2.1 m from the two walls so far detected. This means from the previous step the car will drive 2.4 m and be at the first grid spot.

Exceptions

Objects: Using this pattern, objects can either be in the vicinity of a grid point and/or in the drive path. The vicinity of a grid point is defined as being within ≈ 30 cm of a grid point. When encountering an object the car must first identify whether it is an object or a wall. This is done by stopping and making a rotation to the left, and to the right if something is detected to the left, as illustrated in Figure 2.8. If the object is in the vicinity of a grid point the car will make the grid point where it is and drive around the object. Otherwise it will just drive around the object.

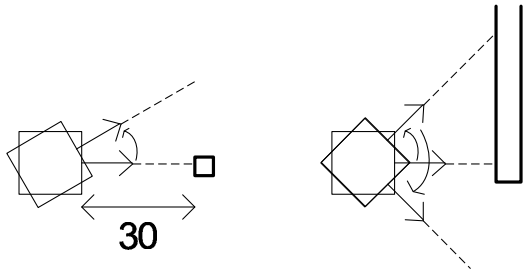


FIGURE 2.8: Identifying objects by rotating left and right if necessary.

Open doors: If the car drives through a doorway, the side sensors shall detect it and the car will rotate 180° and drive back. A door will

furthermore mark the end of a grid row or if detected while driving between grid rows, the door mark the bottom wall and the car will comence making the last grid row.

Different room shapes: As with mapping by continous driving, this pattern is mainly tuned to cope with rectangular rooms, but as this pattern expands its grid points continously and is not based on the length and width of the room, it can cope with certain room shapes besides the rectangular room. Some examples of rooms the car can map, and cannot map, using this pattern is shown in Figure 2.9.

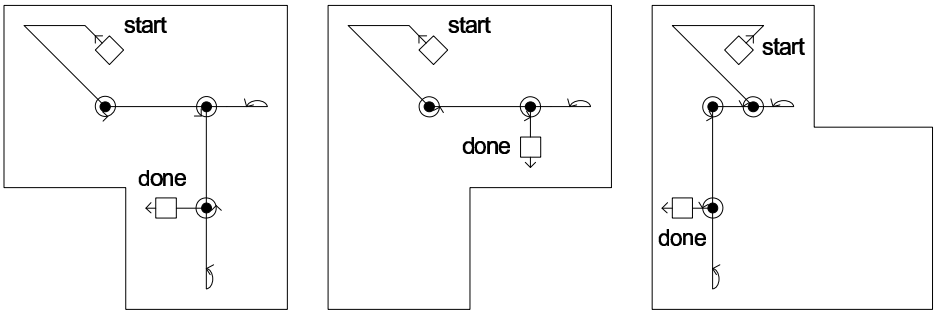


FIGURE 2.9: Room shapes which the car can map and cannot map.

Camera Coverage

Though made for mapping a rectangular room, this pattern can cover the room of all rooms with only one open corner. By covering the room is meant that a clear vision of the walls may not be given, but things in the room is covered. The coverage of different room shapes is illustrated in Figure 2.10.

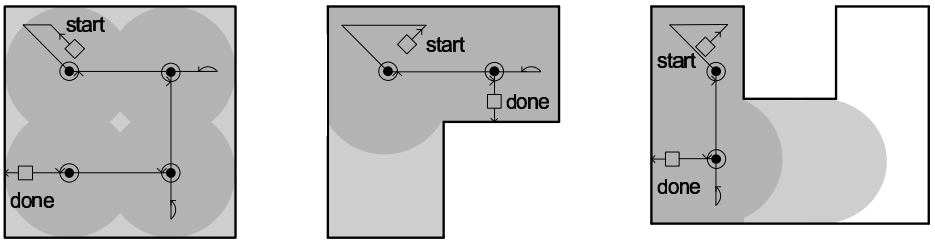


FIGURE 2.10: Mapping by grid can cover rooms one open corner, but might not be suitable if the room has more than one open corner.

Discussion of the Pattern

This pattern gets a clear vision of most of the rectangular room, depending on the size. When it comes to rooms with one open corner this pattern lack the ability to get a clear vision of the walls, but will in all cases with one open corner, get a well enough angle to get a clear vision of what is in the room. When it comes to rooms with more than one open corner, this pattern will in most cases be insufficient.

This pattern is based on calculating where to put the grid point in a room the car has not measured the dimensions of, which means that the load of online calculations will increse with the size of the room.

The imprecision of the hardware is not dealt with by means of feedback control in this pattern. As the car will spend almost the entire time moving around the room with the walls out of sensor distance, the control used for this pattern will be feedforward and thereby be sensitive to disturbanses, like imprecise hardware. The effect of having a 1 % error between the speed of the wheels on either side of the car is shown in Figure 2.11.

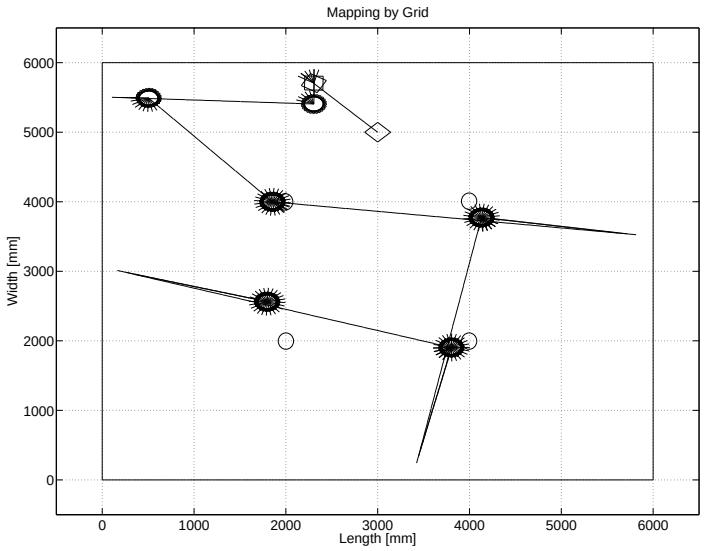


FIGURE 2.11: With a 1 % deviation of the left side wheels compared to the right side wheels the car will end up deviating about 0.5 m.

The handling of exceptions is for objects of low complexity, as the car will either have to rotate and drive around or just drive around the object, depending on whether the object is in the vicinity of a grid point or not. The

handling of doors is of low complexity as well, as the car will drive backwards and continue its normal case mapping process.

The time it takes to map a room is a function depending on four variables; length, width, number grid points per row, $\#gp$, and number grid rows, $\#gr$. Within the wall length limits given in 1.2.1 this pattern is only usable from 2 to 6 m, as the car has to be close to 2 m from the wall it is filming, that is minimum 1.7 m like the definition of being in the vicinity of. $\#gr$ depends on the width:

$$\#gr = 1 \ for \ 3.5 \leq width \leq 4.5, \text{ and } 2 \ else$$

$\#gp$ depends on the length and $\#gr$:

$$\#gp = 1 \cdot \#gr \ for \ 3.5 \leq length \leq 4.5, \text{ and } 2 \cdot \#gr \ else$$

The equation for calculating the time to map is then dependant on three variables: length, width and $\#gp$. The relationship between the width and the distance the car needs to drive is linear:

$$distance_w = width - 0.6$$

The same linear equation count for the lenght, though it is also dependant on $\#gr$:

$$distance_l = (length - 0.6) \cdot \#gr$$

The rotation speed of the car is estimated to be 45° per 0.1 s and the speed while driving forwards and backwards is the same as for mapping by continous driving, 0.28 $^m/_s$.

The expression for the time it takes to map a room dependant on $distance_w$, $distance_l$, and $\#gp$ of the walls is given in equation 2.2.

$$t = \frac{distance_w}{0.28} + \frac{distance_l}{0.28} + \frac{360°}{45°} \cdot 0.1 \cdot \#gp \tag{2.2}$$

2.2.3 Mapping with Sweeps

This pattern is based on driving alongside the walls, using feedback control to keep at a constant distance to the walls. At every 2 m and in the corners the car makes a "sweep," which means it rotates 180° left and back right to drive alongside the wall. The idea of this pattern is illustrated in Figure 2.12 and it is shown how the car handles open corners while using "Mapping with Sweeps."

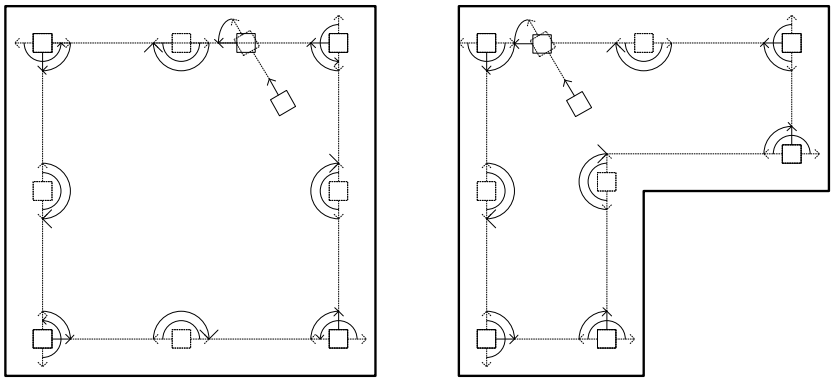


FIGURE 2.12: Mapping of a square room, and handling of open corners using "Mapping with Sweeps".

Normal Case Scenario

1. The car lands and stabilizes in a random position in the room.
2. The initial direction of driving is directly forward from the position the car stabilizes in.
3. When detecting a wall with both front sensors, the car stops and rotates 45° and initiates the feedback control.
4. The car drives alongside the wall for two meters or until a wall is detected.
   (a) If the car has driven 2 m it rotates 180° left and back. Whereafter it continues alongside the wall, repeating step 4.
   (b) If a wall is detected the car rotates 180° left and 90° back, and follows the new wall.
5. The car continues like this until it reaches the first corner again.

The idea on how to detect when the car is at the first corner again, is to count the corners.

Exceptions

Objects: As the car is driving alongside the walls the object avoidens for this pattern is done by stopping, rotating to the left, to determine whether it is an object or a wall. If it is an object, the car will coninue

around it and use the feedback control as soon as it estimates that it has passed the object.

Open doors: Doors are handled as exceptions to open corners. The idea is to use the feedback control to turn at the open corners, but if the left side sensor detects something while doing this, a door has been encountered and the car stops, and rotates 90° left and continues with the feedback control.

Different room shapes: As mentioned, the closed corners are counted to handle the extra closed corners, that will be introduced if the room has open corners. The corner counter will decrese when encountering an open corner and increase when encountering a closed corner, which means that when the car reaches the first corner again, the corner counter will increase to five.

Camera Coverage

In rectangular rooms with walls longer than two meters, the main part of the walls will be filmed while driving alongside them. In rooms with open corners the camera will cover the room better, as illustrated in Figure 2.13.

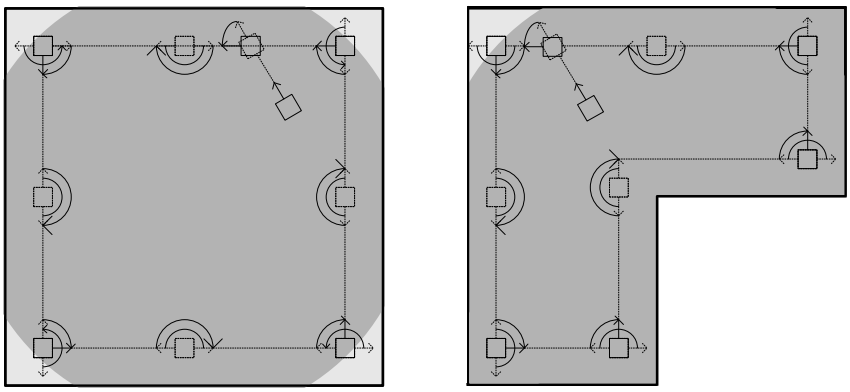


FIGURE 2.13: The coverage of the room is better if the room has open corners.

Discussion of the Pattern

As long as the room has 90° corners this pattern will film the room. As with the mapping by continous driving will it cover the walls mostly by driving alongside them if the room is rectangular,but in rooms with more corners will this pattern give a better coverage.

The complexity of this lies mostly in designing the feedback control, as a good design of this should be able to compensate for the feedforwards needed to handle closed corners and exceptions.

Having the feedback control also makes the car less affected by imprecise hardware. The worst impreciseness would be an inaccurate right side sensor, as the controller will use this for feedback, but an offset from it will just propagate to an offset in the distance from the wall.

The handling of objects will be performed by displacing the car to the left and when the object is passed the controller should take over again. This idea goes for all exceptions and the corner handling as well. When encountering something a feedforward sequence is activated and when that is performed, the controller should be able to get the car to the right distance to the wall.

The time to map the room will be based on rooms within the limits acording to wall lengths, but also include a room shape none of the other two patterns can cover. The room shapes are illustrated in Figure 2.14. The factors in the equation needed to express the time to map are:

Four corners: The time to make the sweeps a closed corner is the total angle divided with 45° times 0.1 s, which is the time to turn 45°: ${}^{270°\cdot 0.1\ s}/_{40°} = 0.6\ s$. The total for four corners is

$$0.6 \cdot 4 = 2.4\ s$$

.

Sweeps between corners: The number of sweeps between the corners is $\approx {}^{(2\cdot length+2\cdot width)}/_{2} - 4$, where the subtracted 4 four corners the roo will always have. If the walls are less than 2 m there will not be any sweeps between the corners. The time per sweep is calculated like the sweeps in the corners, though the total angle is 360°.

$$\left(\frac{360° \cdot 0.1\ s}{40°}\right) \cdot \left(\frac{2 \cdot length + 2 \cdot width}{2} - 4\right) = 0.8(length+width)-3.2$$

Around the rectangular room: The speed of the car is 0.28 ${}^{m}/_{s}$. The distance to the wall the car is driving alongside is estimated to be

0.3 m. This means to cover the length and width of the room the expression is:

$$\frac{2(length - 0.6) + 2(width - 0.6)}{0.28} = 7.1(length + width) - 8.53$$

.

First open corner: An open corner will always add a closed corner to the room, and as such the time added for the first open corner includes a closed corner. The car will aproximately turn at the open corner with a radius of 0.3 m.

$$\frac{0.3 \cdot 2 \cdot \pi}{4 \cdot 0.28} + 0.6 = 2.3 \ s$$

Second open corner: Again the a closed corner is added, but there will furthermore be added the displacement shown in Figure 2.14 times two.

$$\frac{0.3 \cdot 2 \cdot \pi}{4 \cdot 0.28} + \frac{2 \cdot displacement}{0.28} + 0.6 = 2.3 + 7.1 \cdot displacement$$

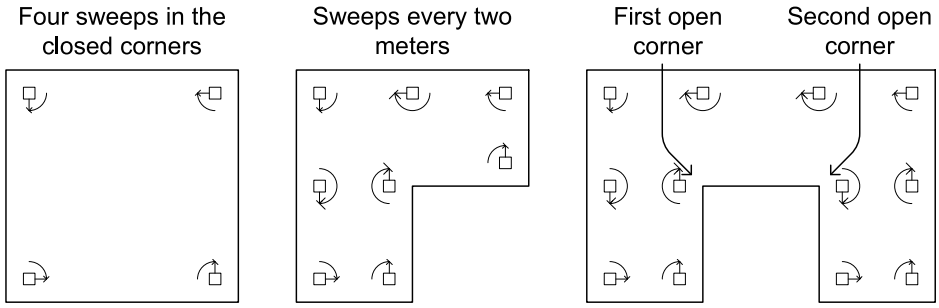


FIGURE 2.14: The time to map using mapping by sweep is dependant of the variables mentioned on these figures.

The total time to map a rectangular room:

$$\begin{aligned} t &= 2.4 + (0.8(length + width) - 3.2) + 7.1(length + width) - 8.53 \\ &= 7.9(length + width) - 9.32 \end{aligned} \quad (2.3)$$

The total time to map a room with two open corners is:

$$\begin{aligned} t &= 2.4 + (0.8(length + width) - 3.2) + 7.1(length + width) - 8.53 \\ &\quad +2.3 + (2.3 + 7.1 \cdot displacement) \\ &= 7.9(length + width) + 7.1 \cdot displacemnt - 4.72 \end{aligned} \quad (2.4)$$

2.2.4 Discussion of the Three Patterns

This discussion is based on the six parameters outlined in the beginning of the mapping considerations. At the end of each discussed parameter points are given depending on which pattern is the best. The point are given according to the description in the beginning of mapping considerations.

Room Coverage: All three patterns film the rectangular room acceptably, but Mapping by Grid does this best, as it has the camera pointing directly at all areas of the walls. If other room shape is non-rectangular it is Mapping with Sweeps that is the best though, as it covers all the wall like Mapping by Grid does in the rectangular room.

Mapping by Continous Driving: 0

Mapping by grid: 1

Mapping with Sweeps: 3

Normal Case Complexity: The complexity vary from light calculations with Mapping by Continous Driving to the more difficult continous expansion of the grid in the room with Mapping by Grid. Mapping with Sweeps need a controller, but when that is designed the pattern is less complex than Mapping by Grid, but requires more online calculations than Mapping by Continous Driving.

Mapping by Continous Driving: 3

Mapping by grid: 0

Mapping with Sweeps: 1

Handlin Unprecise Hardware: Both Mapping by continous Driving and Mapping with Sweeps uses a controller to drive alongside the walls. This gives them an advantage compared to Mapping by Grid. And as Mapping with Sweeps does not cros the room, but always has at least one wall within sensor distance, this is deemed best.

Mapping by Continous Driving: 1

Mapping by grid: 0

Mapping with Sweeps: 3

Complexity of Obstacle Handling: As with the preveous parameter it is estimated that having feedback control when dealing with room obstacles is an advantage. And as such, the priorities is the same as for the preveous parameter.

Mapping by Continous Driving: 1

Mapping by grid: 0

Mapping with Sweeps: 3

Different Room Shapes: The only pattern designed specifically for room shapes other than the rectangular is Mapping with Sweeps, and as such that is the best at it. Of the other two the Mapping by Grid is the best as it expands its view of the room as it expands its grid and for some room shapes other than the rectangular the Mapping by Grid can map the room acceptably.

Mapping by Continous Driving: 0

Mapping by grid: 1

Mapping with Sweeps: 3

Time: To find the best calculation on the smallest room all three patterns can map is made and likewise for the largest room all three can map. Equation (2.1), (2.2) and (2.3) are used to calculate the times.

Pattern	$length = 2,\ width = 2$	$length = 6,\ width = 6$
Continous Driving	41	155.4
Grid	18.2	61.1
Sweeps	22.3	85.5

Mapping by Continous Driving: 0

Mapping by grid: 3

Mapping with Sweeps: 1

The overall best pattern is the one with the highest score.

Mapping by Continous Driving: 5

Mapping by grid: 5

Mapping with Sweeps: 14

2.3 Specific Requirements

The first part of this chapter is based on finding the requrements for the functionalities in Figure 2.2. The second part is illustrating these. This is made by use of timing diagrams.

2.3.1 Functionally Requirements

Motor Control. This functionality controls the four motors used for propulsion. These are described in Appendix E. The communications of the motor control functionality is illustrated in Figure 2.15. An option is to implement some cognitive capabilities, and thereby make it able to drive adapting the motors speeds.

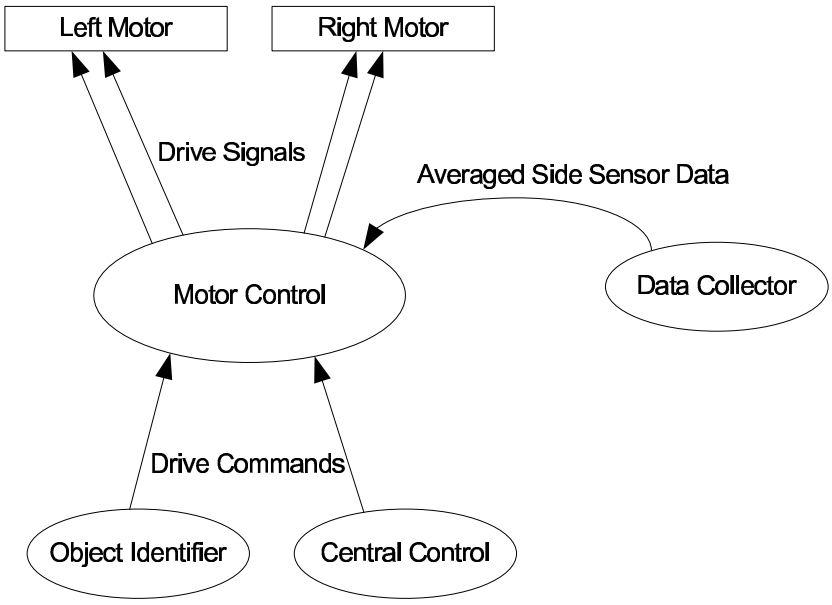


FIGURE 2.15: The Motor Control uses feedback from the side sensors to drive straight. The drive command from the Central Control overwrites the feedback control.

Input: Three inputs are given to this functionality, an average of the side sensor measurements and drive commands, which are given from both the Central Control and the Object Identifier.

Side sensor averages: The input from the data collector is read every 0.1 s, and has values ranging from $10 - 60$.

Drive commands: These have a number from $0 - 5$ to represent the action required.

Function: With the motors chosen, see Appendix E, the motor control's update rate is 10 Hz, which corresponds to the 5 cm side sensor update rate stated in 2.1.2. The input from the Data Collector will be used as feedback to the update algorithm for the front left motor. This feedback shall ensure the car's ability to

drive alongside the walls. The feedback will only directly act on one motor to simplify the control algorithm. The way this should be done is by having the back left motor dependend on the front left motor and have the two right motors driving at constant speeds. The feedback control shall not be in effect, when a new drive command is given by the central control, and should in general only be active, when the car is driving either forwards or backwards.

Output: There are four outputs from this functionality, one for each motor.

Front left motor: This is the only motor directly affected by the feedback from the Data Collector.

Front right motor: This has three speeds, 100, 0, and $-100\ rpm$.

Back left motor: The update algorithm for this motor shall be dependant on the updated algorithm of the front left motor, which means that, indirectly, this one is also affected by the feedback control.

Back right motor: As with the front right motor this one has three speeds, 100, 0, and $-100\ rpm$.

Data Collection. The purpose of this functionality is to collect sensor measurement data, verify them, and make them usable for the motor control and central control functionalities. A diagram of this functionality is shown in Figure 2.16. An option is to have this functionality use fault detection or sensor information fusion.

Input: This functionality has five inputs, from two side sensors, two front sensors, and a sensor pointing downwards. The distance range and digital resolution for the side and front sensors are the same, as described in Appendix D, and will henchforth be refferred to as the distance sensors.

Distance range: $10 - 60\ cm$

Digital resolution: 6 bit (1 cm resolution)

The sensor mounted at the bottom of the car is found in Appendix C and is used to detect the floor. As such, its distance range is indicating the low and high output range used.

Distance range: Low: $0.5 - 1.5\ cm$, high: $1.5 +\ cm$

Digital resolution: 1 bit (low:0, high:1)

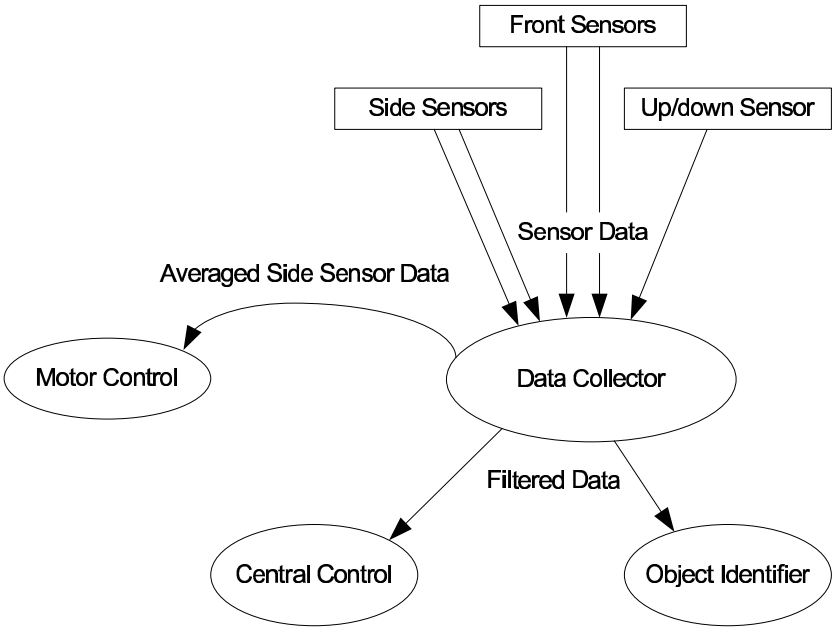


FIGURE 2.16: The Data Collector functionality taking inputs from the sensors and filtering them for the motor control and central control.

Function: For each of the distance sensors the moving average will be computed, and for this two ring buffers are used, one with the real data and one with the average. The sensor at the bottom of the car will only be used once in the mapping process, and the data used will be the average of ten measurements.

Output: This functionality has three outputs, one to the motor control, one to the central control, and one to the Object Identifier.

Motor control: The averaged data from the side sensors are made available for the mortor control. The motor control runs with 10 Hz and will therefore have a sensor update every 0.1 s.

Central control and Object Identifier: All averaged data are send stored and thereby the central control can use it. The ring buffer with the average sensor data will be in a block of memory accessible by the central control.

Central Control. This is the functionality, which controls the mapping process. It estimates and counts the number of corners, estimates how much of the room has been mapped, determines what actions to take

when encountering an object in the drive path, and makes sure a signal is send when the mapping is done. This functionality is illustrated in the diagram in Figure 2.17.

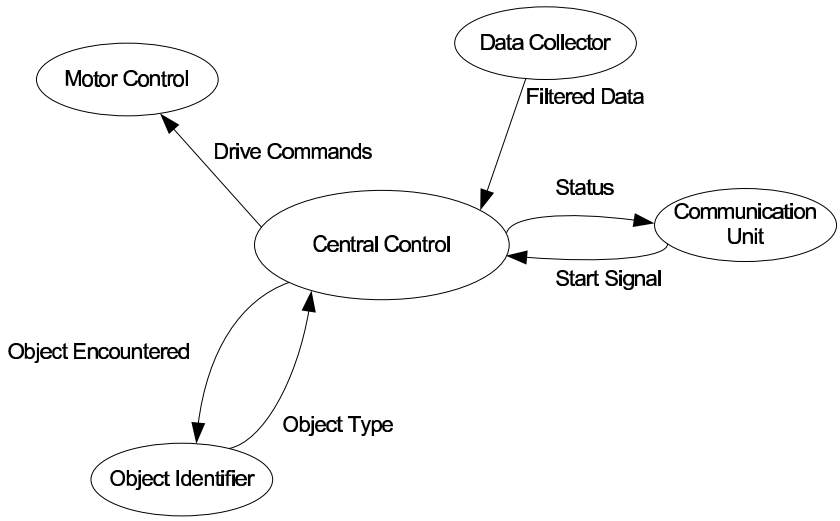


FIGURE 2.17: Diagram of the system's Central Control.

Input: The Central Control has three inputs, Filtered Data, Start Signal, and Object Type.

Filtered Data: This is the filtered data from all the sensors. Values from the distance sensors are ranging from 10 to 60 and the data from the bottom sensor have a range of 0 to 1.

Start Signal: The signal from the helicopter to the car to commence mapping.

Object Type: This shall indicate what sort of object the car has encountered. Depending on how many types of object that can be encountered this should be ranging from 0-9.

Function: The Central Control takes command of the car when an abnormality is encountered. That is, when the front sensors detect an object closer than 30 *cm*, or on the right side sensor measurement has an increment of more than 10 *cm*.

In both cases the Central Control will send a stop command to the Motor Control and call upon the Object Identifyer to identify the type of abnormality. The Object Identifyer can give one of four replies, wall, object, door, or no door.

Wall: If the car has encountered a wall it is detected by the front sensors. The right side sensor is then read, and if it measures less than 60 *cm* it is estimated to be a corner, and the Central Control calculates a command sequence for the Motor Control in accordance to the mapping considerations in section 2.2. The process of dealing with a closed corner is shown in Figure 2.18.

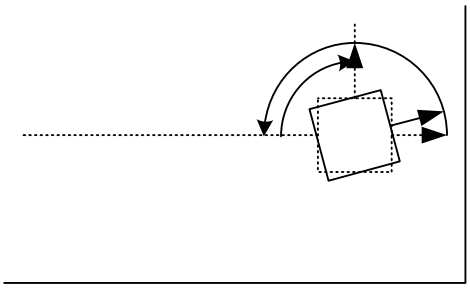


FIGURE 2.18: When the car encounters a closed corner it makes a sweep to film inwards in the room.

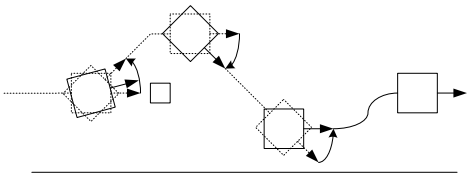


FIGURE 2.19: When encountering an object, the car drives left around it.

Object: In this case the Central Control finds a command sequence taking the car to the left side of the object, and when the object is no longer detected by the right side sensor, the Motor Control's feedback control takes over again. An illustration of how the car passes an object is illustrated in Figure 2.19.

No door: This reply means there is an open corner on the car's right and the Central Control sends the command to the Motor Control.

Door: If a door is detected the car is in the doorway, and the Central Control makes the car rotate left until the front sensors are pointing at the door frame. Then Central Control gives the command to Object Identifier, as if an object is in the drive path.

When the mapping is done the Central Control passes a done signal to the Communication Unit.

Output: The Central Control has three outputs, drive commands to the Motor Control, status to the Communications Unit, and object encountered to the Object Identifier.

Drive Commands: These range from 0-5 and contains the basic drive functionalities, stop, forwards, backwards, rotate left, and rotate right.

Status: This contains an error and a finished mapping signal.

Object Encountered: With this the Object Identifier functionality is called.

Object Identifier. This functionality is called everytime an abnormality is encountered by the car. There are two possible abnormalities considered, when the front sensor measurements decreases to less than 20 cm, and when the side sensor mesurements show an increse of more than 10 cm. A diagram of the Object Identifier's interactions with other functionalities is shown in Figure 2.20.

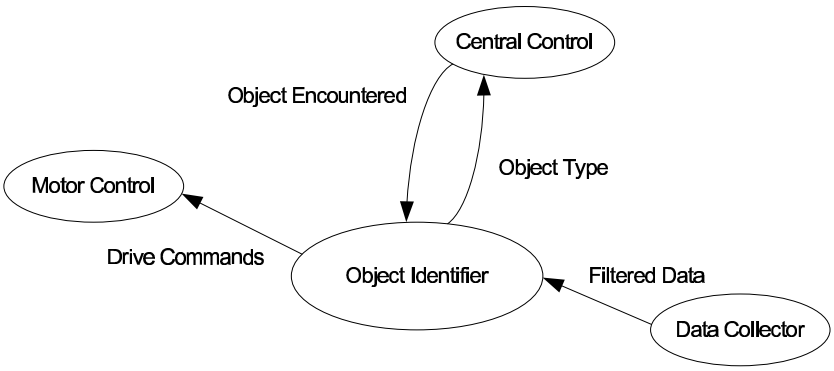


FIGURE 2.20: When the Object Identifier functionality is called, it can give drive commands to the Motor Control.

Input: The inputs for this functionality are object encountered from the Central Control and the filered data from the Data Collector.

Object Encountered: The input can contain one of two values, 1 or 2.

Filtered Data: This is the filtered data from all the sensors. Values from the distance sensors are ranging from 10 to 60 and the data from the bottom sensor have a range of 0 to 1.

Funtion: If the input is 1, the car has met either a wall or an object in front of it. To determine which, the Object Identifier sends a rotate left command to the Motor Control and reads the change in the front distance mesurements. If, during the rotation, one of the front sensors shows significantly more than the other, and

more than the initial distance, the car has encountered an object. Otherwise it is a wall. In both cases the information is sent to the Central Control.

If the input is 2, the car has either a door or an open corner on the car's right. To determine which, the Object Identifier should find a command sequence that brings the car drive around the open corner, or through the door way, and at the same time poll on the left side sensor. If there is a reaction on the left side sensor a door is identified. If there is no reaction on the left side sensor withn a certain time, a open corner is detested. In both cases the information is sent to the Central Control. The door encounter is shown in Figure 2.22.

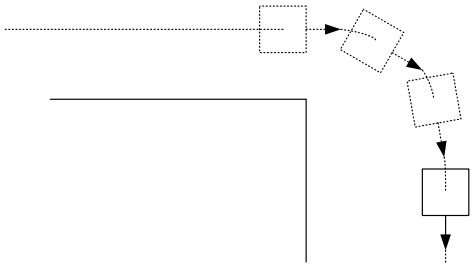


FIGURE 2.21: The car encountering an open corner.

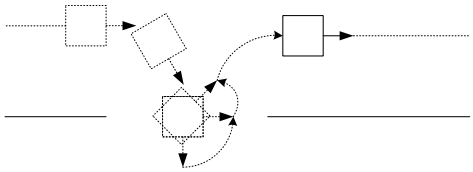


FIGURE 2.22: When encountering a door, the car rotates left until the front sensors are pointing on the door frame, and runs the Object Identifier.

Output: The Object Identifier has two outputs.

Object Type: This is a number, send to the Central Control, in the range 0-9 send to the Central Control, which indicates which type of abnormality the car has encountered.

Drive Command: As with the drive command from the Central Control to the Motor Driver. It ranges from 0-5 and has a basic drive functionality associated with each of the numbers.

Communication Unit. The purpose of this functionality is to receive the start signal from the helicopter and send the done mapping signal to the helicopter when done.

Input: There are two inputs, one from the helicopter and one from the Central Control functionality.

Status: This is the done signal from the Central Control.

Start: This signal is send from the helicopter to the car.

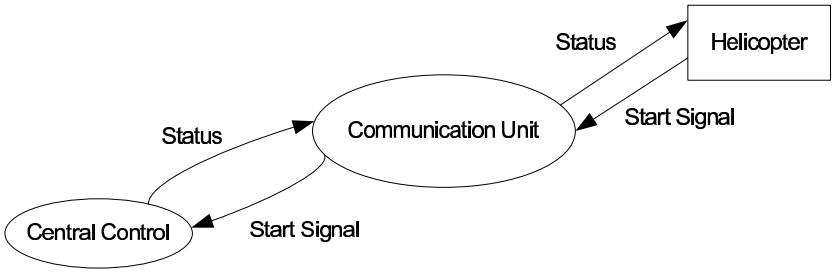


FIGURE 2.23: The Communication Unit controls the transmissions and data reception to and from the helicopter.

Function: The WLAN communication gets activated when the system starts up. Upon receiving the start signal from the helicopoter, the Communication Unit have to shut down the WLAN to avoid interruption from elements outside the system. When the car is done mapping the Communication Unit shall activate the WLAN again and start transmitting the done mapping signal.

Outputs: The outputs are a done signal to the helicopter, transmitted via WLAN, and a start signal to the Central Control.

Initialize. This functionality shall not interact with the rest of the functionalities and, as such, does not have any inputs or outputs. As the name implies it is the start up functionality.

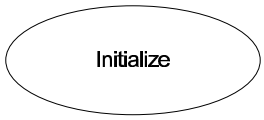


FIGURE 2.24: Initialize is responsable for the system start up.

Function: To start up the system, activate the WLAN, the motor driver and the sensors.

2.3.2 Timing Diagrams

The timing diagrams in this part is based on the environment descriptions in section 1.2.1 in the introduction.

Initialize: When the system is first activated the Initialize functionality starts up the Motor Control and the Central Control. The Motor Control starts looping and is initially set to stop. The Central Control

calls the Communication Unit, which starts pending for a start signal to pass to the Central Control. The intialization process is shown in the timing diagram in figure 2.25.

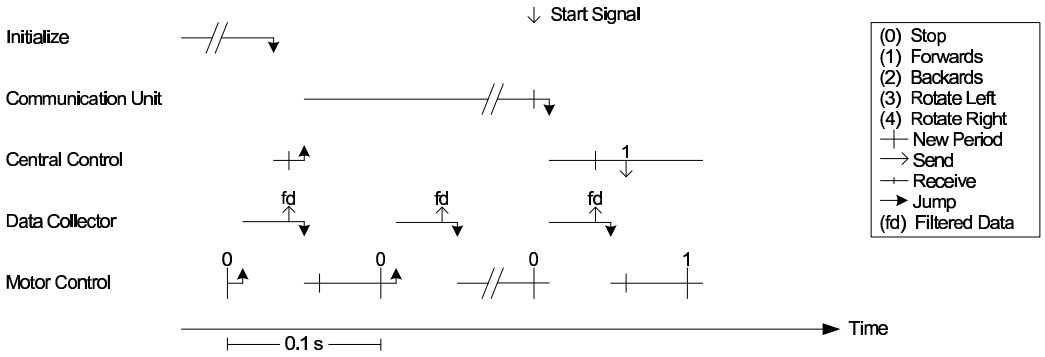


FIGURE 2.25: Timing diagram for the initialization process.

Closed Corner Handling: The car drives alongside a wall using feedback control. Measurements from the front sensors of less than 30 cm, passed from the Data Collector to the Central Control, means that the car has encountered either a wall or an object. The Central Control will take over, send a stop signal to the Motor Control, and call upon the Object Identifyer to determine which kind of abnormality the car has encountered.

The Object Identifier sends a rotate left signal to the Motor Control. The information, that it is a wall, is passed to the Central control and computes a command sequence, which is send with 1 Hz to the Motor Control. The given command sequence makes the car rotate left to point in its former driving direction and 90° back, pointing in a new direction along the new wall encountered. When done the control is once again given to the Motor Control and the feedback control is resumed. This rutine is illustrated in Figure 2.26.

Object Handling: As with the closed corner handling, an abnormality is encountered in front of the car and, the feedback control in the Motor Control stops, the Central Control takes over, and the control is passed further on to the Object Identifier. The Object Identifier sends a rotate left signal to the Motor Control. The information, that it is an object, is passed to the Central Control, which then computes a command sequence to take the car to the left side of the object. Once on the side of the object the control is given back to the Motor Control and

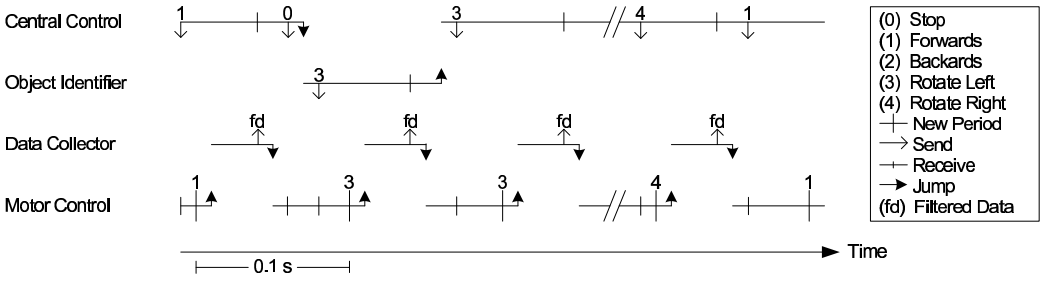


FIGURE 2.26: The timing diagram for handling a closed corner.

the feedback control is resumed. The timing diagram for this routine is shown in Figure 2.27.

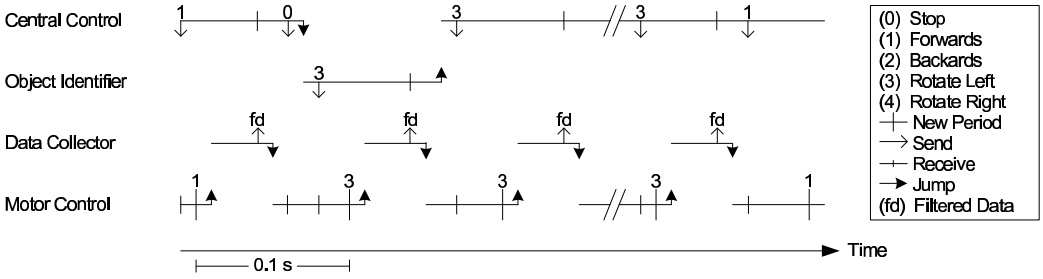


FIGURE 2.27: The timing diagram for handling an object blocking the drive path.

Open Corner and Door Handling: The car drives alongside a wall and the data from the side sensors sent to the Central Control indicates a more than 10 *cm* increment. The Central Control sends a drive command to the Motor Controller and notates that a corner is incountered. Furthermore data from the left side sensor is read, to distinguish an open corner from a door. When the identification is made the Central Control acts acording to the path finder description in 2.2.3. The timing diagram for when encountering a door or an open corner is much alike, as with the object and closed corner timing diagrams, and as such, only one timing diagram covering both are made. This is illustrated in Figure 2.28.

Mapping Done: When the last corner is reached, a signal needs to be sent. The Central Control sends a stop command to the Motor Control and

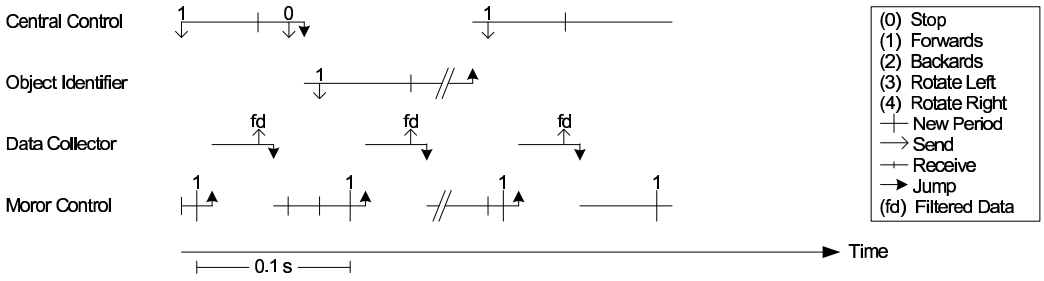


FIGURE 2.28: Timing diagram for encountering an open corner or a door.

calls the Communication Unit and it sends the done signal. After that is done the control is given back to the Central Control. This routine is illustrated in Figure 2.29.

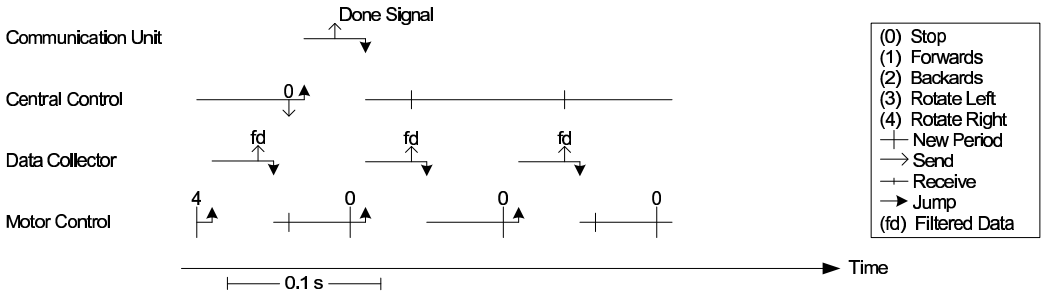


FIGURE 2.29: Timing diagram for when the car is done mapping.

2.4 Accepttest Specification

The main goal of the project is to have the car make a video map of an arbitrary room. This task is divided into smaller tasks. Each of these tasks is described from a test point of view, which means that each task will have to be testable. The tasks can furthermore be divided into two groups

- the tasks concerning the car variables, for instance start position,
- and the tasks concerning the room variables, for instance room shape.

When both these groups are tested, a test including both is performed.
In all tests the car is expected to map the room and stop when it is done.

2.4.1 Test Setup: Car Variables

The test setup for tasks concerning the car variables is shown in Figure 2.30. In this line of tests the room has constant dimension, $4\ m \times 6\ m$. There are no objects in the room and there are only closed doors.

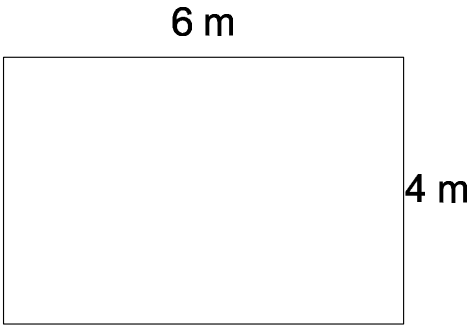


FIGURE 2.30: Room test setup.

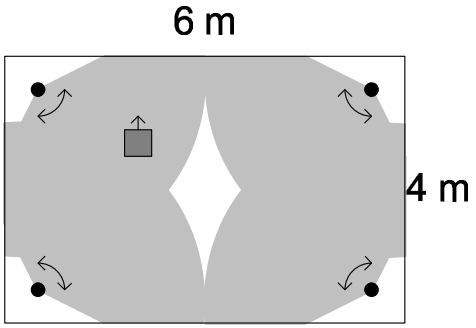


FIGURE 2.31: Cameara coverage from the corners in test one.

The car variables are:

Start position: Where in the room does the car land.

Start direction: In which direction does the camera point.

Up/down: Which side of the car is up.

Test 1: All Fixed

This is a test of whether the car can map the room starting from the position and with the direction shown in Figure 2.31. The top of the car is facing upwards. The expected result is to have a video map of the entire room, and what is not covered by the rotational movement in the corners, is covered when driving between the corners.

Test 2: Random Start Position

Different start positions are randomly generated to test whether the car car start at a random start position. These positions are shown in Figure 2.32. The start direction is the same for all three positions. and the top of the car is facing upwards.

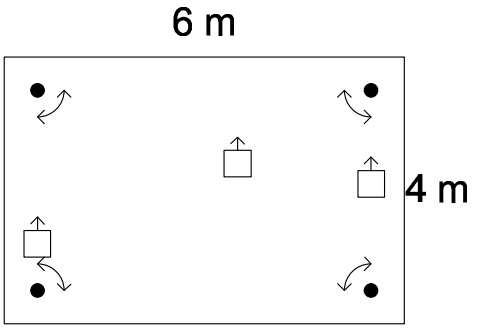


FIGURE 2.32: Test setup for test 2. Three different start positions.

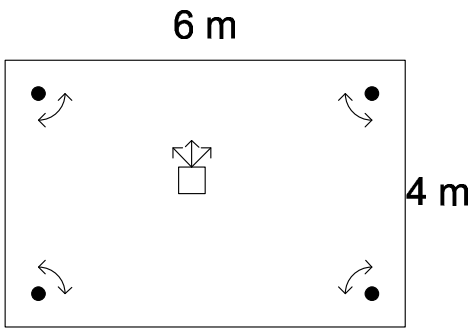


FIGURE 2.33: Test setup for test 3. Three different start directions.

Test 3: Random Start Direction

Here the random start direction is tested by randomly generating different directions, as shown in Figure 2.33. The start position is fixed. The top of the car is facing upwards.

Test 4: Random Side Up

The bottom of the car is facing upwards, which means the motors will have to run backwards to move the car forwards. The start position and direction is predetermined to be the same as in test 1. Figure 2.34 shows how the car turns.

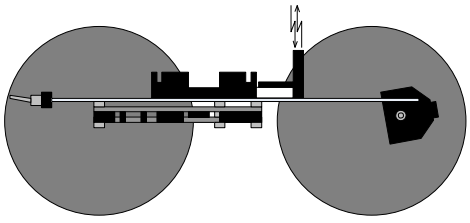


FIGURE 2.34: Test setup for test 4. Turning the car upside-down.

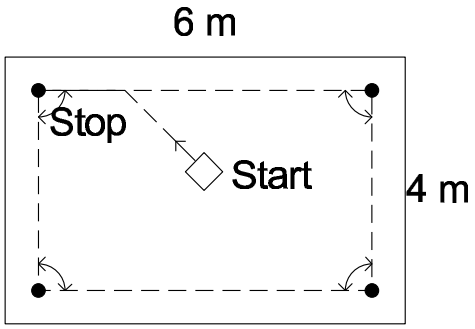


FIGURE 2.35: Test of whether the three previous tasks can be performed simultaneously.

Test 5: All Random

To round up the first line of test, a test is made where start position and direction is random and which side of the car is upwards is random. The specific test setup is illustrated in Figure 2.35.

2.4.2 Test Setup: Room Variables

For this line of tests the car starts in the same position as in test one, with the same direction and the same side up.
The room variables are:

Room shape: The number of corners and length of the walls.

Doors: Doors placed where the car should detect a wall.

Objects: The objects placed on the floor, specifically in the drive path of the car.

Test 6: Random Shape

This is a test of the car's ability to map regardles of the number and kind of corners and regardless of the legths of the walls. The test setup is a wall where the car will first encoiunter a closed corner, then an open corner and lastly another closed corner.

Test 7: Door

An open door is introduced in this test. The door is placed in a wall and will appear to the simulated car after one closed corner encounter.

Test 8: Object in Drive Path

An object is placed in the drive path of the car. The object's dimensions are: Width= 10 mm, depth= 10 mm, and height= 200 mm. The specific test setup is a ten meter wall with 3 objects placed in teh drive path.

Test 9: All Random

To round up this second line of tests, the mapping process is performed with the length of the walls varying from 4 m to 6 m, one open door, two objects

of the same size as in test 8 and 9, one placed in the drive path and one in a grid point. The specific test setup is illustrated in Figure 2.36.

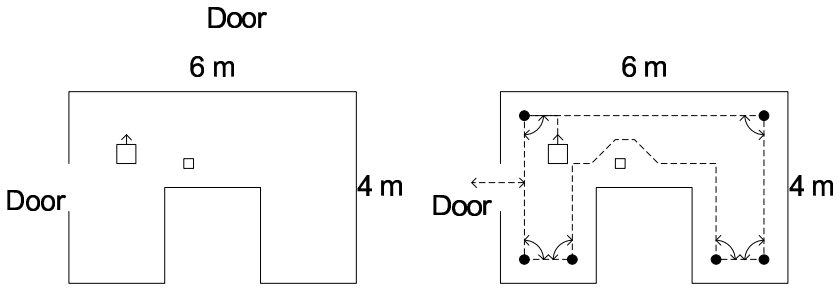


FIGURE 2.36: A test of all the room variables combined.

Chapter 3

Hadware Design

Contents

The main issue in the design of the hardware is that it has to be light weight, have a minimum of mechanical components, and still fulful the task of mapping the abitrary room. For instace, will a compromise have to be made when seleting motors. Should the car have one for propulsion and one for driving to lessen the weight, or should it have four motors to lessen the mechanics.

This and issues concerning all the other components of the car will be discussed in this chapter.

3.1 Hardware Components

3.1.1 μ-computer

The chosen μ-computer for the car is a Gumpack from Gumstix inc., containing a gumstix and a robostix. This has the advantege of two processors, which is required to fulful the timing presented in 2.3.2, one processor for the Motor Control and the Data Collector, and one for the rest of the functionalities.

The Gumstix is mounted on top of the Robostix, which has the connectors to the ports.

Ports used on the robostix.

Port B This port is the PWM output port, used for the motors.

Port F This port is the ADC port used for the sensors.

3.1.2 Up/down Sensor

This sensor is placed facing downward initially and has to be able to detect the floor. The sensor chosen, an opb704, see Appendix C, has a destinct difference in output, as shown in Figure 3.1, depending on whther it is facing the floor, a 0.15 V output, or the ceiling, a 4.8 V output.

3.1.3 Distance Sensors

There are four of these sensors mounted on the car, two facing forwards and one facing either side. The sensors used are four GP2D12, which has a theoretic range of $10 - 60$ cm. These were mainly chosen due to availability and the test of them can be found in Appendix D. The test result of the distance to output voltage relation is shown in Figure 3.2, which shows that the range is closer to $10 - 70$ cm for the front sensors and $10 - 60$ cm for the side sensors, but still, this is acceptable.

3.1.4 Motors

In choosing the motors several aspects are taken into consideration. As mentioned in the introduction to this chapter a compromise will have to be made concerning the wheight or the mechanics. From looking at what types

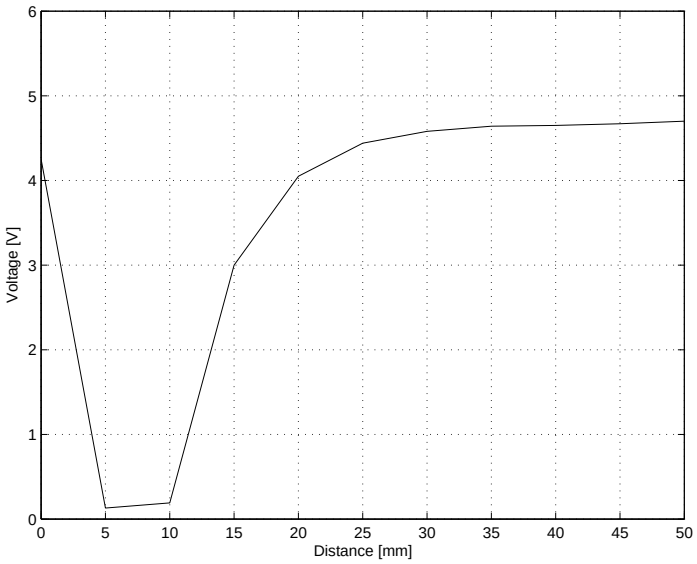


FIGURE 3.1: The test of the opb704 showed a distinct difference in output depending on whther it is facing the floor or not.

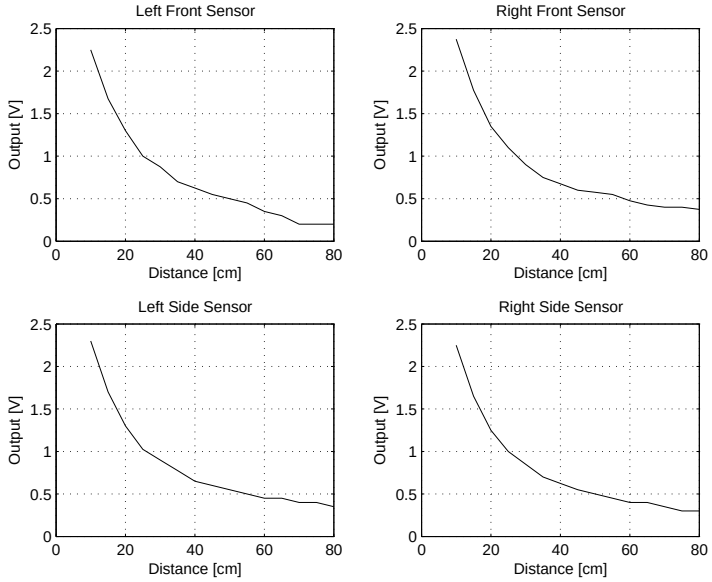


FIGURE 3.2: Distance to output voltage relation for the GP2D12 distance sensors used.

of motors that are available one type made the compromise easier to make. The motor type chosen is μ-servo. These have the advantage of being light weight and still maintain a speed which can be compared to that of the standard servoes. As found in Appendix E the power-to-weight ratio for a 0.5 kg car is high enough even for the Hitec HS-50, which has the lowest torque and the highst weight of the two motors considered. Due to this another factor is taken into consideration, the speed. The motor with the highest rpm, 110, is the Hitec HS-50, which is why it has been chosen for this project.

Concerning the number of motors, we deem that the avoidens of mechanics has greater value that adding the weight of two motors, 10 g. This means that the car has a four motor setup, one for each wheel.

3.1.5 Wheels

The wheels has a composite of two materials. The main part of the wheels is made of foam rubber and the part connecting it to the motors are made of wood. The wheel is fixed to the motors using two screws for each wheel. The wheel design is illustrated in Figure 3.3. The wheels has a radius of 4.5 cm, which, with the motor rpm in mind, gives the car a maximum speed of 0.52 $^{m}/_{s}$.

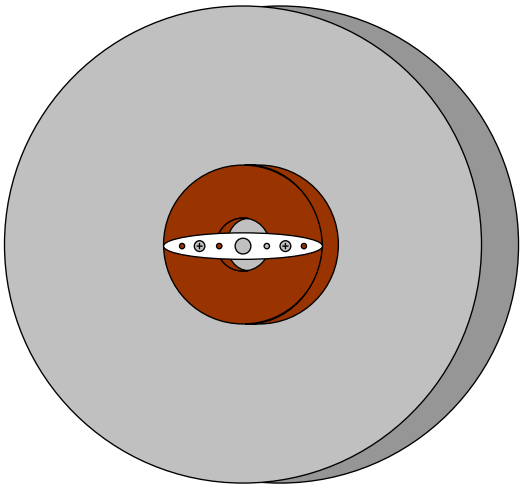


FIGURE 3.3: The foam rubber wheels are fixed to the motors with a wodden disc and two screws each.

3.1.6 Camera

The camera is mounted on the front end of the car pointing forwards. In choosing the camera the internet was browsed and a spycam was found. The camera chosen is a Wireless 811T.

It is frequency modulated, has a transmission frequency of $2.400 \approx 2.483\,GHz$, and an undisturbed transmission range of 100 m.

Its view angles are $\approx 44°$ vertical and $\approx 61°$ horizontal.

To enhance the used view angles of the camera a tilt mechanism has been designed, shown in Figure 3.4. The design is described in Appendix F.

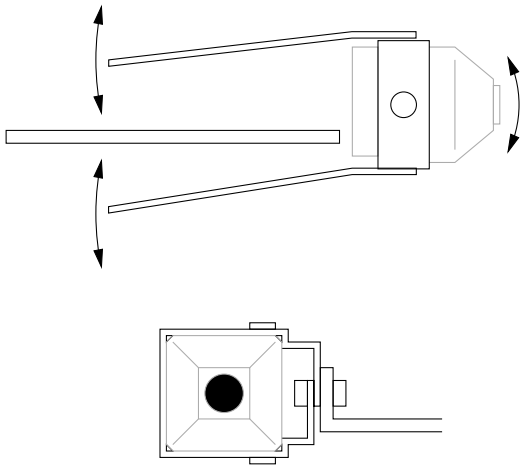


FIGURE 3.4: The tilt mechanism, which mounts the camera to the foundation plate.

3.1.7 Power Supply

The supply for the car's systems has been divided in three loads, the camera, the motors, and the Robostix, which supplies the Gumstix and sensors. The specific current of each load has been tested and the results of these tests are found in Appendix G. Each of the following load currents are when they use the most.

The Camera needs an 8 V supply and uses 125 mA.

The Motors needs a 6 V supply and uses 1550 mA.

The Robostix needs a $5 - 9$ V supply and uses up to 325 mA.

To accomidate these needs six AAA 1.5 V batteries are used. The batteries has a charge of 1200 mAh. As shown in Figure 3.5 the voltage supply is taken at 6 and 9 V and an 8 Ω resistor is placed in serial with the 9 V supply and the camera.

The total current passing through the system is 2000 mA, which means that with a battery charge of 1200 mAh, the car can be active for approximately 0.6 $h = 36$ $minutes$.

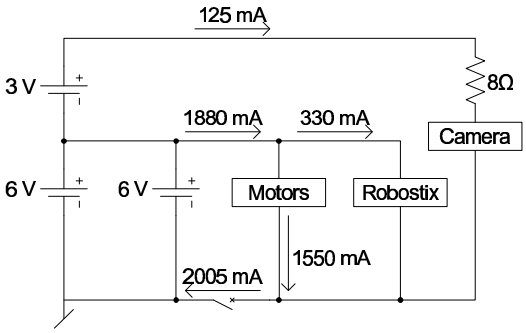


FIGURE 3.5: the circuit for the car's power supply.

3.1.8 Foundation Plate

The material used for the foundation plate is aluminium. A car layout is illustrated in Figure 3.6. The plate weights 54.8 g.

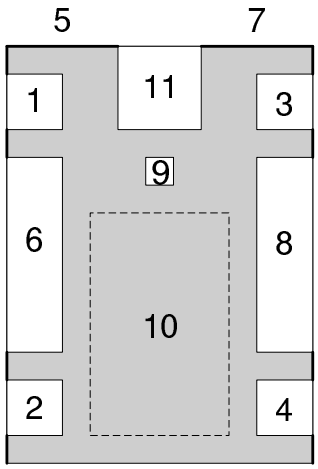


FIGURE 3.6: Spaces 1-4) Motors, 5-9) Sensors, 10) Robostix and Gumstix, and 11) Camera.

3.2 Discussion

The way the car ahs been designed, hardware wise, has minimized the mechanics such that the only movable parts are the motors, and the camera tilt device. The final layout of the car is shown in Figure 3.7. Furthermore, has the weight been kept low. The total weight of the car is 280 g, which is acceptable.

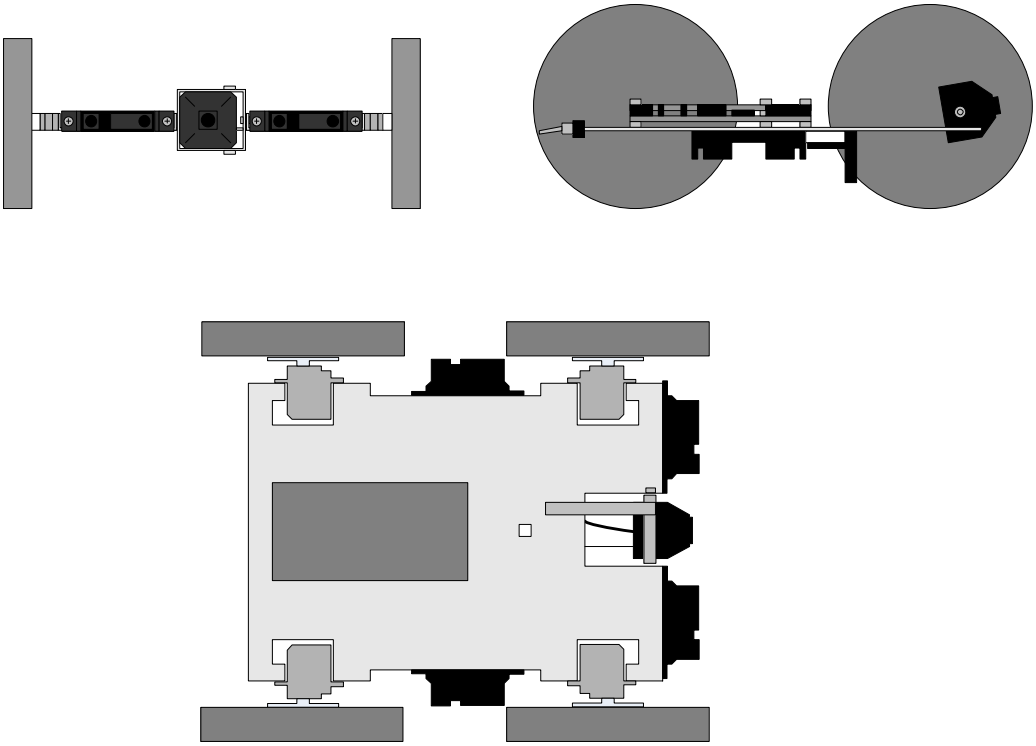


FIGURE 3.7: The final layot of the car seen from the front, side and top.

The speed of the car is 0.52 $^m/_s$. At that speed the car will, in the time frame defined in 1.2.2, be able to map a room that is approximatly 300 m when following the walls and having $^1/_3$ of the time to make the video sweeps in the corners.

Chapter 4

Controller

Contents

In this chapter a controller is found, such that the car is able to drive alongside a wall at a given distance. First the general feedback principle is descriped, then the system description is made, and finally different types of controllers are considered.

4.1 Scope of Use

The car needs to drive alongside the walls at a distance of 30 cm. As this will not always be the case a controller is designed to ensure this. The limits of the controller is based on the limits of the sensor used for the feedback. The sensor used is the right side sensor, which has a measuring range of $10 - 60$ cm. This means the maximum offset of ± 20 cm corresponding to a range of $10 - 50$ cm from the right side of the car to the wall. In Figure 4.1 the wanted behaviour is illustrated.

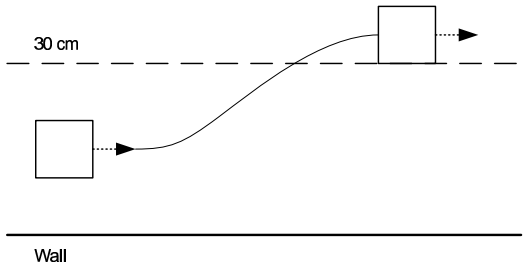


FIGURE 4.1: The wanted behaviour of the controller is to have the car drive smoothly to a distance of 30 cm from the wall it is driving alongside.

When the maximum offset occurs it is wanted to reach the reference within 1 m. From this information the risetime can be derived. Risetime t_r is defined as when the curve rises from 10 % to 90 % of a unit step, in this case the step size is 20 cm, the length to reach this is 1 m which is equal to 100 cm, and the velocity of the car is 23 cm/s. This gives the following equation, and are illustrated in Figure 4.2.

$$t_r = \frac{0.8}{20} \cdot \frac{100}{23} \approx 0.174 \ s \tag{4.1}$$

For faster risetime an overshoot M_p of 10% is allowed, which corresponds to 2 cm when the maximum offset is 20 cm.

To find the maximum angle the car is permitted to use when settling, a corner is considered. The car is driving with the maximum angle θ, and the front sensors have a value equivalent to a distance of 30 cm. When rotating 45° left to identify the corner, the front sensors have a maximum value equivalent to 60 cm to define a corner. This is illustrated in Figure 4.3, and two equations with θ can be written.

$$\cos(\theta) \quad = \quad \frac{a}{30} \tag{4.2}$$

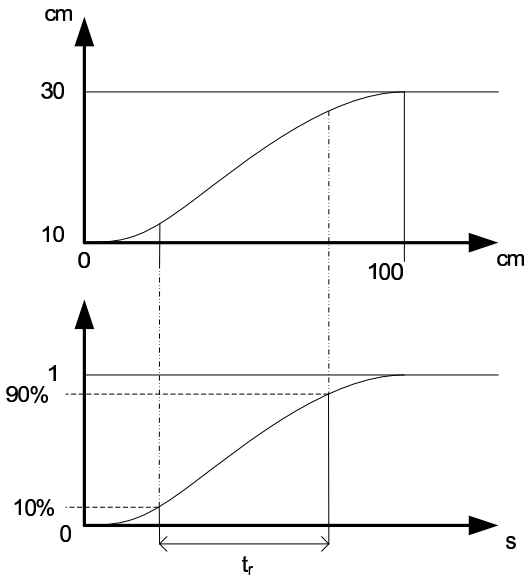


FIGURE 4.2: This illustrates the risetime.

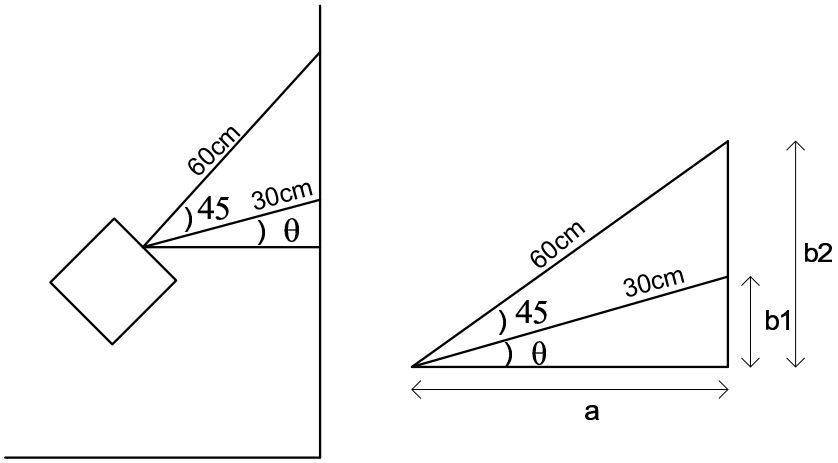


FIGURE 4.3: Illustration of the variables in a corner, when the maximum angle is calculated.

$$\cos(\theta + 45^\circ) = \frac{a}{60} \Rightarrow a = \cos(\theta + 45^\circ) \cdot 60 \tag{4.3}$$

$a = \cos(\theta + 45^\circ) \cdot 60$ is now inserted in (4.2), and θ is found.

$$\cos(\theta) = \frac{\cos(\theta + 45) \cdot 60}{30} \Rightarrow \tag{4.4}$$

$$\theta = 16.3249 \approx 16^\circ \tag{4.5}$$

This maximum θ is when rotating left, when rotating right, a case where an obstacle is avoided is considered. When finding the wall after the obstacle , the right side sensor has to locate the wall before the front sensor is less than 30 cm from the wall. This case is illustrated in Figure 4.4, and the maximum θ when rotating right is then -45°.

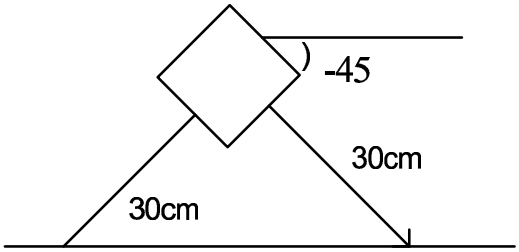


FIGURE 4.4: Illustration of the maximum angle when rotating right.

To make sure that these angles never are violated, a symmetric rate limitter is inserted with the most strict requirement, which is the smallest angle. This gives a requirement for the controller of $\theta < \pm 16^\circ$.

4.2 Requirements

The controller have to live up to the following requirements, which are given for a unit step:

1. Rise time: $t_r = 0.174\ s$.
2. Overshoot: $M_p = 10\%$.
3. Maximum rotate angle: $\theta < \pm 16^\circ$

4.3 Controller Structure

A standard feedback control setup is used, shown in Figure 4.5. The input to the controller r_{dif} is the difference between the distance reference r_{ref} and the measured distance r, in other words the distance error. The controlled variable is the angle, θ, the car need to deviate from the driving direction parallel to the wall. Figure 4.6 illustrates the variables.

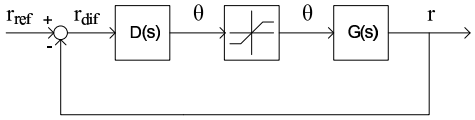


FIGURE 4.5: The used controller structure.

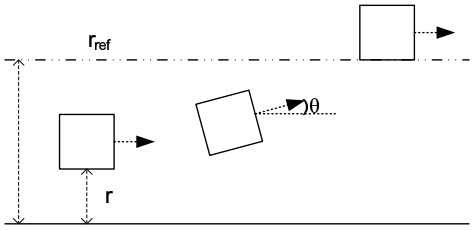


FIGURE 4.6: The feedback variables.

4.4 The System

The system is a model of how θ affects the distace to the wall per second. The car drives at a speed of $23^{cm}/_{s}$. With $\theta = 1°$ the car has displaced itself 0.4 cm away from the wall per second.

The reference to the control loop is a distance in voltage, since the output from the distance sensors are voltage, then the system needs to describe how the measured distances relates to the output voltage of the sensors. The sensor used is the right side sensor, which has been tested, see Appendix D for the test. The distance to voltage relation is shown on the graph in Figure 4.7 together with an approximatet function, which will be the system in the control loop.

The approximated distance to voltage function is

$$V_{out} = 3 \cdot e^{(-0.04 \cdot r)} \tag{4.6}$$

and together with the angle to distance realation $r = 0.4 \cdot \theta$ as shown in Figure 4.7, the system expression becomes:

$$g(r) = 0.4 \cdot 3 \cdot e^{(-0.04 \cdot r)} = 1.2 \cdot e^{(-0.04 \cdot r)} \tag{4.7}$$

The maximum deviation of the model is 0.239 V at 10 cm, which is acceptable, since a ratelimitter is used, to keep det maximum angle below 16, which

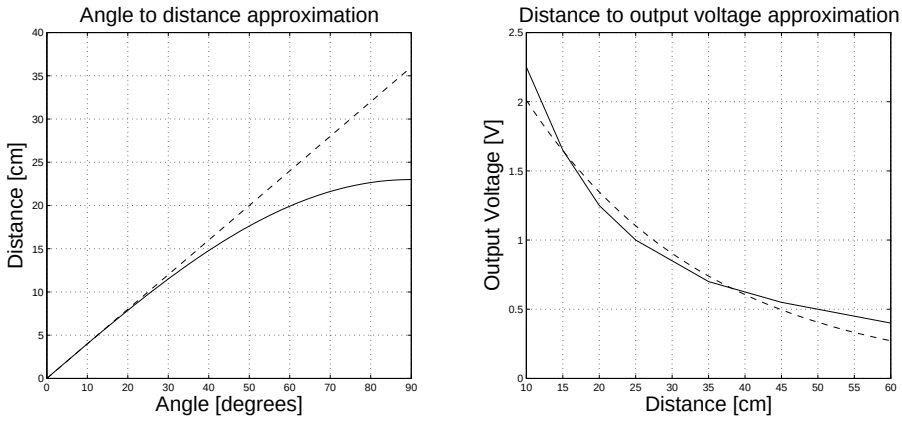


FIGURE 4.7: The calculated sine of the angle to distance test result and distance to voltage. Fitted functions are shown with dotted lines.

corresponds to a distance lower than the 10 cm.

4.4.1 The System in the s-domain

The system $g(r)$ is transformed to the s-domain and is thereby expressed as in equation 4.8.

$$G(s) = L[1.2 \cdot e^{(-0.04 \cdot r)}] = \frac{1.2}{s + 0.04} \tag{4.8}$$

The systems step response is shown in Figure 4.8.

4.4.2 Voltage Reference

With the system in place, the reference voltage is found. According to the measurements it should be 0.85 V, but as the system is described by the exponential function, the voltage reference is:

$$V_{ref} = 3 \cdot e^{(-0.04 \cdot 30)} = 0.904 \ V \tag{4.9}$$

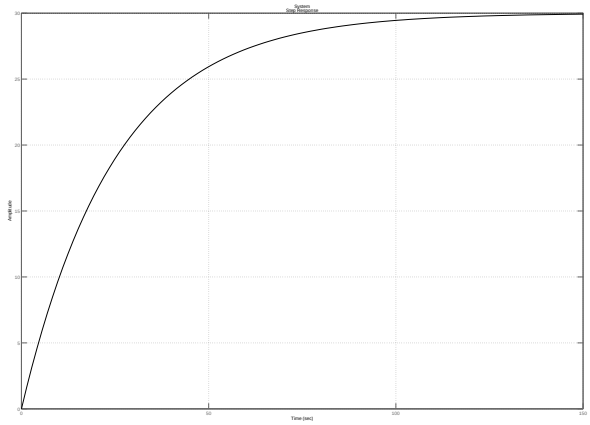


FIGURE 4.8: The systems step response.

4.5 The Controller

4.5.1 Control Setup

The control setup for the found system is shown in Figure 4.9.

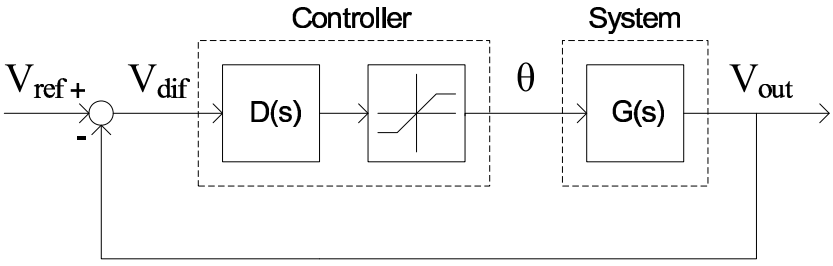


FIGURE 4.9: The setup for the controller.

The rate limitter was inserted to insure that θ stays within $\pm 16°$, $G(s)$ was found in the previous section, now the controller $D(s)$ can be found.

The controllers considered are proportional controller (P), proportional controller with integrator (PI), and a proportional controller with integrator and derivator (PID).

4.5.2 P Controller

The P controller is bassicly an amplifier of the error between the reference and the output of the system.

Designing the P Controller

The system that has to be controlled is:

$$G(s) = \frac{1.2}{s + 0.04} \tag{4.10}$$

and a P controller is a constant $D(s) = k_p$, which gives the following open loop equation:

$$\textit{open loop} = D(s) \cdot G(s) = k_p \cdot \frac{1.2}{s + 0.04} = \frac{1.2 \cdot k_p}{s + 0.04} \tag{4.11}$$

Now the closed loop is found, to locate the pole placement. Since the controller structure has a unity feedback the following equation is setup.

$$\begin{aligned} \textit{close loop} &= \frac{D(s) \cdot G(s)}{1 + D(s) \cdot G(s)} = \frac{\frac{1.2 \cdot k_p}{s+0.04}}{1 + \frac{1.2 \cdot k_p}{s+0.04}} \\ &= \frac{1.2 \cdot k_p}{s + 0.04 + 1.2 \cdot k_p} \end{aligned} \tag{4.12}$$

k_p is now found, such that the pole lies in the left half plane on the real axis.

$$s + 0.04 + 1.2 \cdot k_p = 0 \Leftrightarrow s = -0.04 - 1.2 \cdot k_p \tag{4.13}$$

This means that $s < 0$ for $k_p > -0.033 \approx 0$. k_p is now found such that the risetime for the close loop fulfills the requirement of $t_r = 0.174\ s$. The gain is found to be $k_p = 11$, which fulfills the requirement with a risetime of $t_r = 0.167\ s$. k_p is now inserted in the close loop which yields:

$$\textit{close loop} = \frac{1.2 \cdot k_p}{s + 0.04 + 1.2 \cdot k_p} = \frac{13.2}{s + 13.24} \tag{4.14}$$

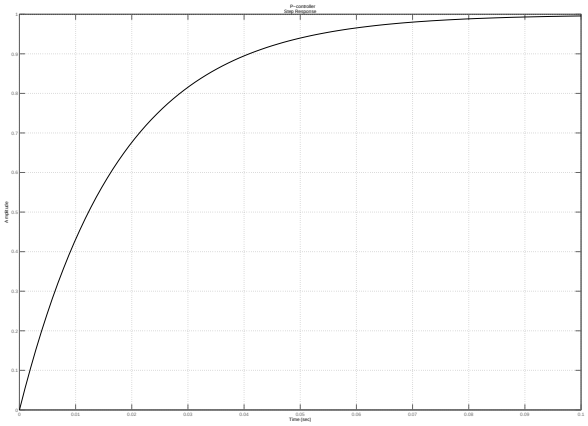


FIGURE 4.10: Step response of the P controller.

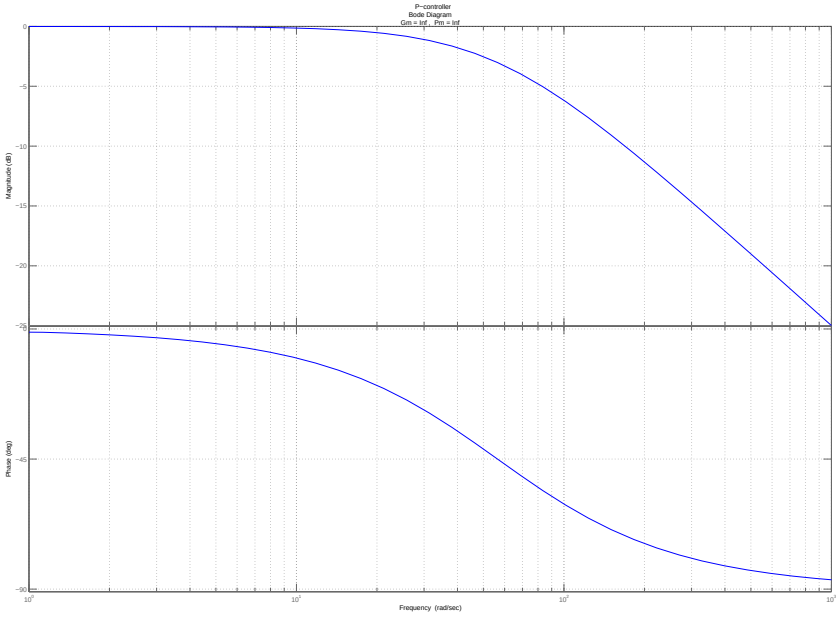


FIGURE 4.11: Bode plot of the P controller.

The step response of the closed loop with the P controller is shown in Figure 4.10, where it can be seen that the P controller does not have any overshoot. A bode plot is made, to see whether the system is stable.

The bode plot shows that the system have a infinite phase margin, which means that the system is stable. But there is an offset of 0.45% which is equivalent to a distance of 0.162 cm at 30 cm.

The car with the P controller is now simulated driving alongside a wall of 10 m with different offsets, to verify that the controller behaves as expected.

Simulation Results

The results of the simulation where the car with the P controller drives alongside a wall with different offsets are shown in Figure 4.12. The simulation shows that when the center of the car starts with an offset of 60 cm from the wall, and have to settle at a distance of 40 cm from the wall, the system will settle after 383 cm. The simulation results are listed below:

t_r: 15.7 s

M_p: 0%

e_{ss}: 0.125% $\approx$ 0.025 cm

$\theta[k]$: $11 \cdot r_{dif}[k]$

The rise time requirement from the step respons is $t_r = 0.174$ s which corresponds to $t_r = 3.48$ s when a step of 20 cm is applied. The P controller have a too slow rise time, but no overshoot. The steady state error e_{ss} is corresponding to a distance error of 0.025 cm, which is considered without significanse, due to the sensors resolution. The complexity of the controller is simple, because it is only an amplification of the error, and thereby only relying on the current measurement, which gives the following structure $\theta[k] = 11 \cdot r_{dif}[k]$.

Because of the slow settling time, the P controller is tuned.

Tuning of the P Controller

The P controller is tuned to have a rise time of $t_r = 3.48$ s which is the requirement for a step of 20 cm. The results of the tuned P controller simulation is shown in the lower graph of Figure 4.12. To get this rise time k_p is found to be $k_p = 47$. This gives the following results.

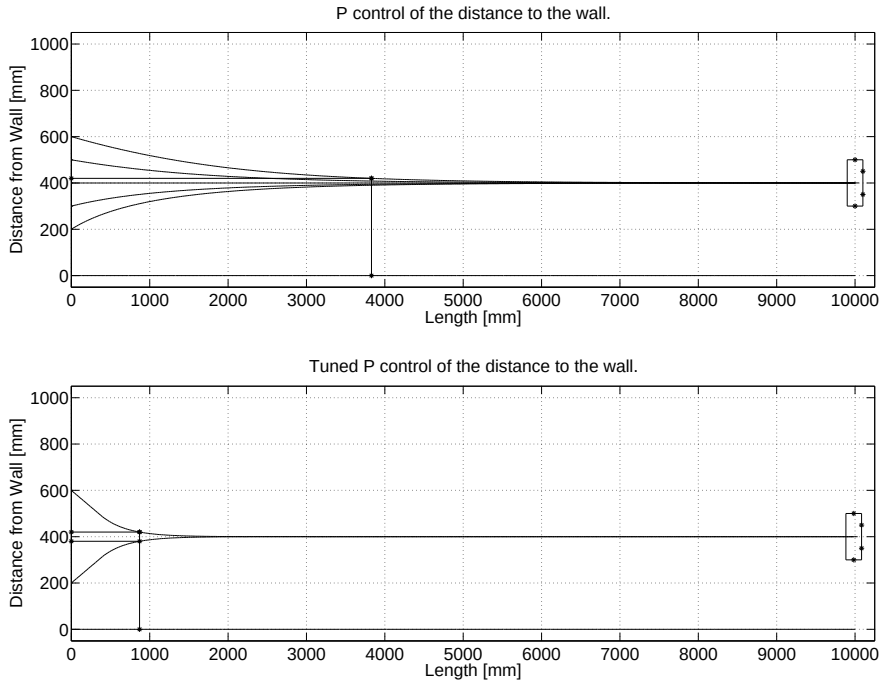


FIGURE 4.12: The P controller is tested driving alongside a wall with different offsets.

t_r: 3.48 s

M_p: 0%

e_{ss}: 0%

$\theta[k]$: $47 \cdot r_{dif}[k]$

The simulation of the tuned P controller shows that the controller does not have either overshoot or steady state error. Furthermore the controller fulfills the rise time requirement. A step response is made for the tuned P controller, the closed loop is as follows.

$$close\ loop = \frac{56.4}{s + 56.44} \tag{4.15}$$

The step response of the tuned P controller gives the following results:

t_r: 0.039 s

M_p: 0%

e_{ss}: $0.07\% \approx 0.0007\ cm$

$\theta[k]$: $47 \cdot r[k]$

Discussion of the P Controller

The controller have three requirements for a unit step, $t_r \leq 0.174\ s$, $M_p \leq 10\%$, and $\theta \leq \pm 16°$.
The risetime for the tuned P controller is lower than the requiremen, which is then approved. It was permitted to have an overshoot of 10%, but the controller does not have overshoot, this requirement is also approved. The maximum angle of $\pm 16°$ is complied with the use of a rate limitter. The simulation showed that the steady state error was small enough to be neglected.

4.5.3 PI Controller

The PI controller is a controller with a proportional gain, with an integrator added.

Designing the PI Controller

A PI controller has the general form of:

$$D(s) = k_p + \frac{k_i}{s} = \frac{k_p \cdot s + k_i}{s} \tag{4.16}$$

The open loop for the system $G(s) = \frac{1.2}{s+0.04}$ with the PI controller:

$$\begin{aligned} \textit{open loop} &= D(s) \cdot G(s) = \frac{k_p \cdot s + k_i}{s} \cdot \frac{1.2}{s + 0.04} && (4.17) \\ &= \frac{1.2 \cdot s \cdot (k_p + \frac{k_i}{s})}{s^2 + 0.04 \cdot s} = \frac{1.2 \cdot (k_p + \frac{k_i}{s})}{s + 0.04} && (4.18) \end{aligned}$$

The closed loop is found for the same system with the PI controller:

$$\begin{aligned} \textit{close loop} &= \frac{D(s) \cdot G(s)}{1 + D(s) \cdot G(s)} = \frac{\frac{1.2 \cdot (k_p + \frac{k_i}{s})}{s+0.04}}{1 + \frac{1.2 \cdot (k_p + \frac{k_i}{s})}{s+0.04}} && (4.19) \\ &= \frac{1.2 \cdot (k_p + \frac{k_i}{s})}{s + 0.04 + 1.2(k_p + \frac{k_i}{s})} && (4.20) \end{aligned}$$

The tuned P controller performed well, therefore the same k_p is used in the PI controller. This gives:

$$\begin{aligned} \textit{close loop} &= \frac{1.2 \cdot (47 + \frac{k_i}{s})}{s + 0.04 + 1.2(47 + \frac{k_i}{s})} = \frac{56.4 + 1.2 \cdot \frac{k_i}{s}}{s + 56.44 + 1.2 \cdot \frac{k_i}{s}} && (4.21) \\ &= \frac{56.4 \cdot s + 1.2 \cdot k_i}{s^2 + 56.44 \cdot s + 1.2 \cdot k_i} && (4.22) \end{aligned}$$

For stability the poles have to be in the left half plane, which means that $s^2 + 56.44 \cdot s + 1.2 \cdot k_i < 0$. The solution to this equation is:

$$\frac{-56.44 \pm \sqrt{56.44^2 - 4 \cdot 1.2 \cdot k_i}}{2} \tag{4.23}$$

For s to be in the negative half plane $0 < k_i < 663.64$. k_i is now found such that the overshoot of the close loop fulfills the requirement of $M_p = 10\%$.

$k_i = 450$ fulfills the requirement with a overshoot of 10%. k_i is inserted in the close loop which yields:

$$close\ loop = \frac{56.4 \cdot s + 540}{s^2 + 56.44 \cdot s + 540} \tag{4.24}$$

The step response of the closed loop with the PI controller is shown in Figure 4.13, where it can be seen that the PI controller have a faster risetime than the P controller, but an overshoot of 10%. A bode plot is made to check for stability. The bode plot is shown in Figure 4.14, and it can be seen that the closed loop has a phase margin $Pm = 148°$, which is considered enough for the closed loop system to have stability.

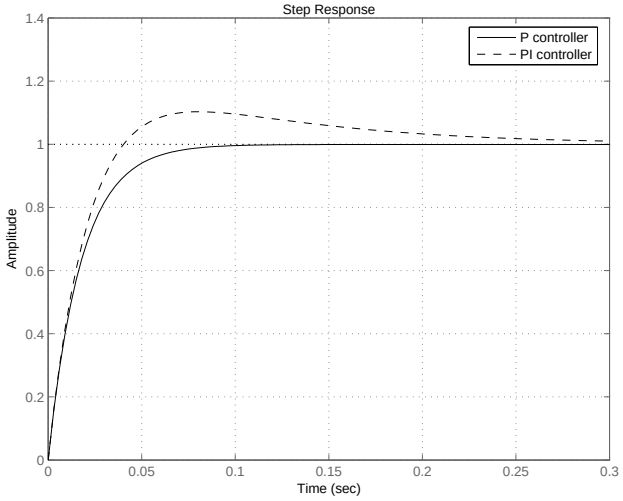


FIGURE 4.13: Step response of the PI controller.

The PI controller does not have any steady state error, and the controller is simulated as the P controller driving alongside a wall, with different start offsets.

Simulation Results

The simulation shows that when the center of the car starts with an offset of 60 *cm* from the wall, and have to settle at a distance of 40 *cm* from the wall, the system will settle after 383 *cm*. The simulation is shown in Figure 4.15, and the simulation results are listed below:

t_r: 15.9 *s*

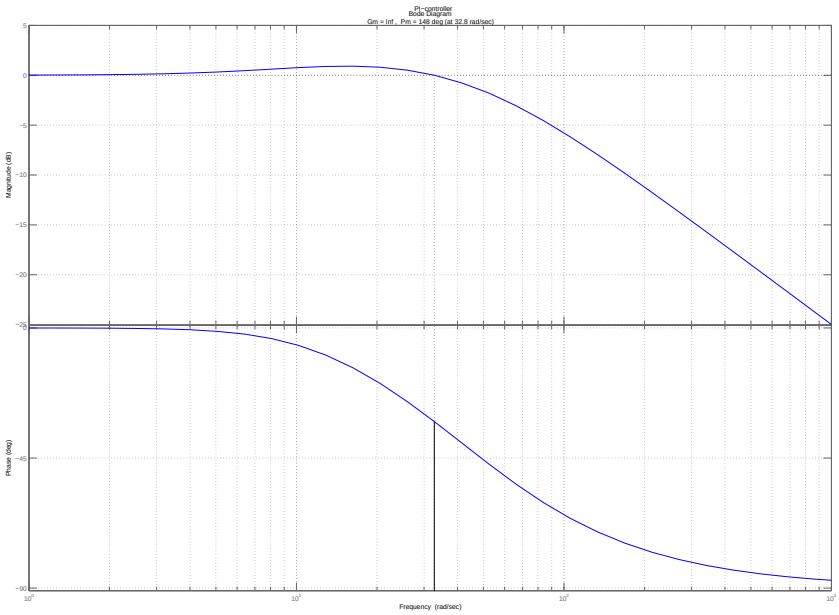


FIGURE 4.14: Bode plot of the PI controller.

M_p: 5%

e_{ss}: 1 $cm \approx 5\%$

$\theta\ [k]$: $\theta\ [k-1] + 69.5 \cdot r[k] - 24.5 \cdot r[k-1]$

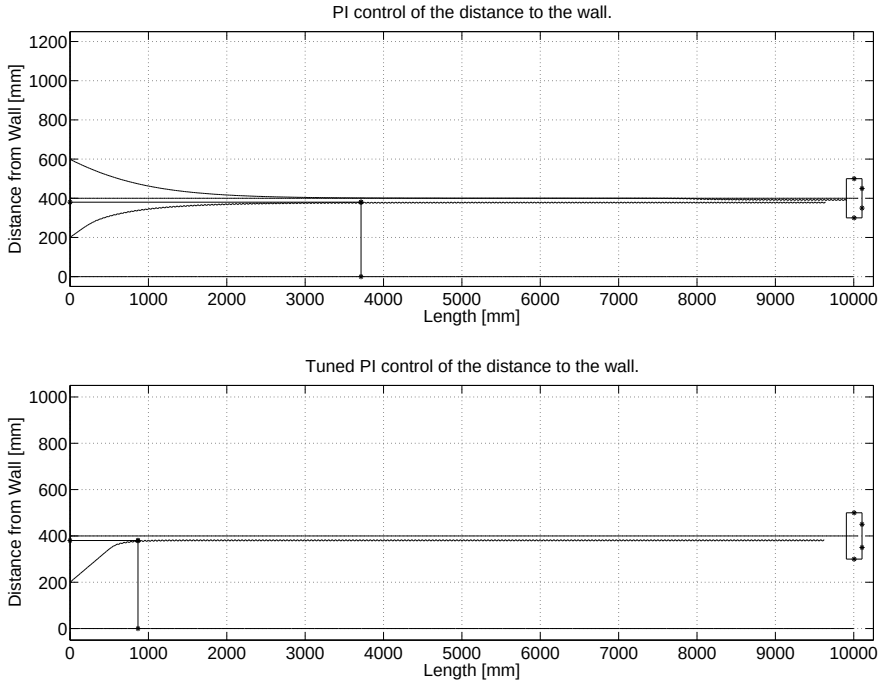


FIGURE 4.15: The PI controller is tested driving alongside a wall with different offsets.

The risetime is about the same as the untuned P controller, which is too slow, so a tuning of the PI controller is needed too. The controller was designed to be fast, and have an overshoot of 10%, but the simulated overshoot were only 5%, which means that it is possibly to tune the PI controller to be faster, and still be under the limit of 10%. The complexity of the controller is greater than the complexity of the P controller, since the PI controller is dependent of the previous measurement. The PI controller have a steady state error $e_{ss} = 5\%$. This is shown in Figure 4.16.

The steady state error is because the car does not drive perpendicular to the wall that is followed, which then gives an error in the measured distance. When the controller tries to correct the error the distance deviate from the actual distance, and keeps oscillating.

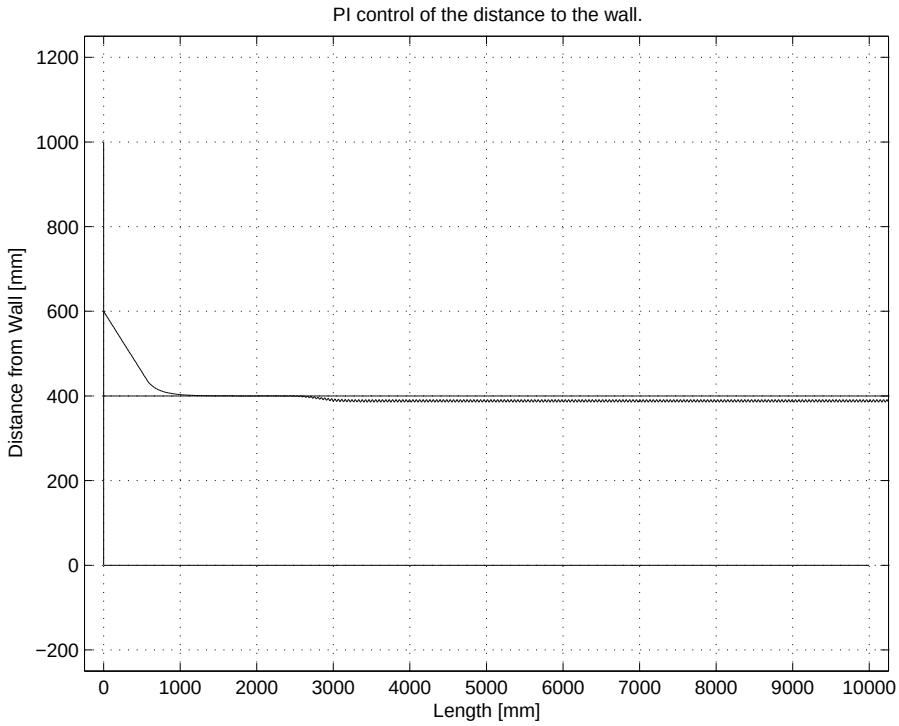


FIGURE 4.16: Wall test of the PI controller, where a stead state error occurs.

Tuning of the PI Controller

The PI controller is tuned such that the risetime is under the required risetime. The result of the simulation of the tuned PI controller can be seen in the lower graph in Figure 4.15, and are listed below.

t_r: 3.47 s

M_p: 7.5%

e_{ss}: 2 $cm \approx 10\%$

$\theta\ [k]$: $\theta\ [k-1] + 185 \cdot r[k] - 15 \cdot r[k-1]$

The risetime is now acceptable, and the overshoot is under 10% which is acceptable too. There is still a steady state error, which has been doubled. The complexity of the controller is the same as for the untuned PI controller.

Discussion of the PI Controller

In general the PI controller performs worse than the P controller. The improvement of risetime is not worth mentioning as it is only 0.01 s better than the P controller, when a stepsize of 20 cm is imposed to the system. Furthermore the PI controller have an overshoot that is permitted, but it does not improve the risetime significant. Compared to the P controller that had a unsignificant steady state error, the PI controller with a steady state error of 10% is too much, to chose the PI controller instead of the P controller. The complexity of the PI controller is greater than the complexity of a P controller.

4.5.4 PID Controller

The PID controller has the advantage of adding a differentiator of the error. But since the PI controller performed worse than the P controller, even with an overshoot permitted, there is no need for a PID controller. This is due to the ratelimitter, that makes sure that the driving angle never exceeds $\pm 16°$ in respect to the followed wall. This means that a PI or a PID controller can not be faster than a P controller with a large gain. The PID controller is chosen not to be made.

4.6 Extended Use of the Controller

With the controller designed we will now look at some extended uses of it. As the mapping pattern is based on "Mapping with sweeps" from section 2.2, there are four things that the feedback control can be used for, which would otherwise be handled by feedforward or not dealt with at all. The first thing is the handling of closed corners and sweeps. The car has to rotate a total of 270° in the corners and 360° when making a sweep after driving two meters without encountering a corner. The second, is when encountering an object, the altenative to the feedback control is to have the a feedforward take the car all the way around the object, but with the feedback control, the use of feedforward can be lessened, as will be described in 4.6.2. The third and fourth uses are when encountering either an open corner or a door.

4.6.1 Closed Corner and Sweeps

When the car rotates, it is expected to experiance some displacement of its center. This displacement is, for simulation purposes, estimated to be about 0.5 *cm* per 45°. At sweep is the total rotation angle 360°. This means that the center of the car can be displaced about 4 *cm* after makin a sweep. This displacement shall the controller correct when the car start driving forwards again. As the displacement falls within the controllers maximum offset of 20 *cm*, it should be able to get the car center back to the reference distance at 40 *cm*.

To determine whether the controller can live up to this, a simulation is performed with the car driving alongside a wall towards another six meters away. The car is expected to make two sweeps before reaching the corner and teh another at the corner and start driving forwards along the second wall and end at the reference distance.

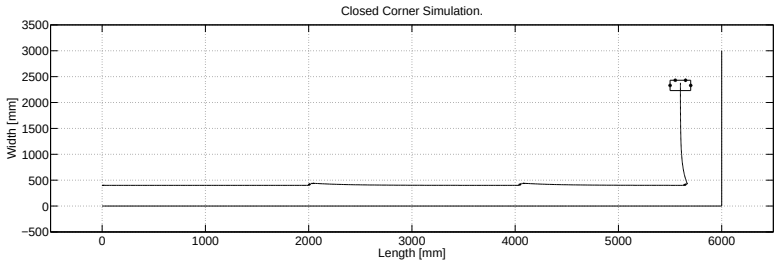


FIGURE 4.17: A simulation of how well the controller handles the dispecements experienced while rotating.

The results of the simulation is shown in Figure 4.17. The simulation showed that the controller is able to compensate for teh displacements caused by the rotations of the car.

4.6.2 Avoid Object

The car can encounter an object in front of it or drive past it. In the first case the car will enter a feedforward control state that will make the car drive to the feft until it is on the side of the object. How far the car need to drive is predetermined and estimated from some geometric considerations. The facts of these considerations and the wanted displacement are illustrated in Figure 4.18.

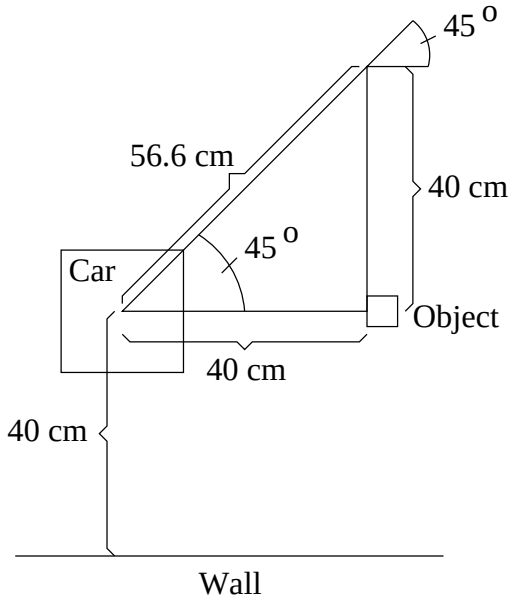


FIGURE 4.18: The wanted behaveour and geometrics of the feedforward implemented to bring the car to the side of an object in the drive path.

When at the side of the object the feedback control is activated again and this is expected to bring the center of the car to the reference distance of 40 *cm* again.

The object avoidans of an object in the drive path was simulated with the controller and the feedforward. The simulation result is illustrated in Figure 4.19. The simulation was performed to show both the reaction to one object and to two objects placed one meter apart.

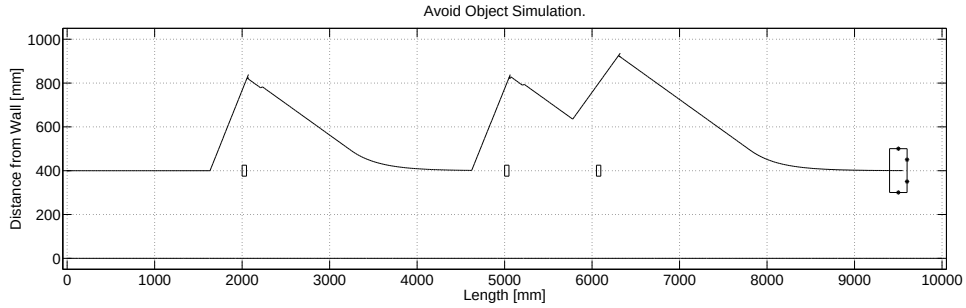


FIGURE 4.19: Simulation of object avoidens, with on and two objects.

As the simulation indicates can the car's feedback control compensate for object in the drive path when it is at the reference distance and also while the controller is currently compensating for a preveous object.

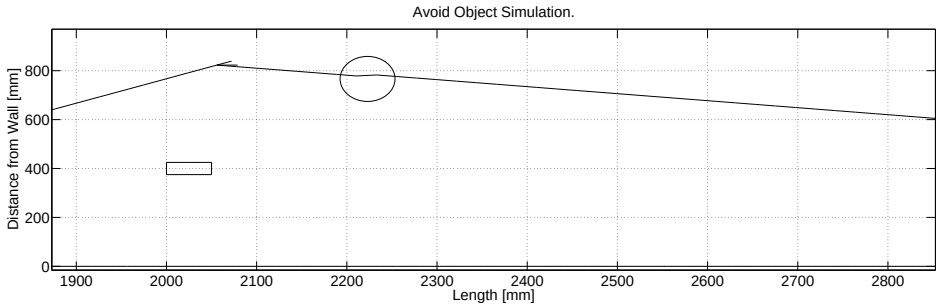


FIGURE 4.20: The spike in car possition appears due to the the right side sensor is detecting the object for one update.

At the first object there is a spike in the car position while the controller is running. Figure 4.20 is an enlarged illustration of that area. The reason for the spike is that the right side sensor detects the object for one update, corresponding to 2.3 cm. As sensing the object results in a spike in the sensor output voltage the controller tries to compensate for the new distance to what it believes is the reference distance. After this one object, the controller is back to control acording to the wall again.

Passing an Object

This is only relevant if the object is between the car and the wall it is driving alongside. To see the controllers response a simulation was performed with three objects, placed at three different distances from the wall. The simulation result is shown in Figure 4.21.

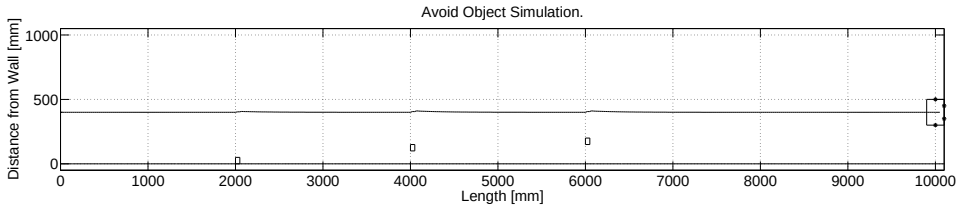


FIGURE 4.21: The result of the simulation with three objects placed between the wall and the car.

An object between the car and the wall it is driving alongside can be compared to increasing the reference poin for 5 *cm*, which is the width of the object. This means that the car shall commence moving further away from the wall during those 5 *cm* and afterwards move the center of the car towards teh real reference, 40 *cm* away from the wall. The result from the simulation shows that car does display this line of actions, which means that the controller acts correctly.

4.6.3 Open Corner

Open corners can be handled by a feedforward, having the car drive in a specific pattern until the corner is passed, but having the feedback control perform this action would be preferable.

A simulation is performed to see wheter the controller can handle this task. The expected result is that the car will detect that the output voltage of the right side sensor will drop, which means the distance to the wall has incresed. The controller is then expected to make the car compensate for the "offset" in distace and move around the corner. The simulation result is shown in Figure 4.22 which shows that the controller is capable to complete the task of driveng the car around an open corner.

4.6.4 Open Door

Initially will a door look like an open corner to the car, as it is just detected as a gap in the wall, like open corners are. The car will commence the routine for open corners, which is the feedback control, and at the same time start reading the sensor output from the left side sensor. If the sensor output changes during the motion around the open corner the car will assess that it has reached a door instead of an open corner. The actions taken are a 90° rotation followed by the same action performed then responding

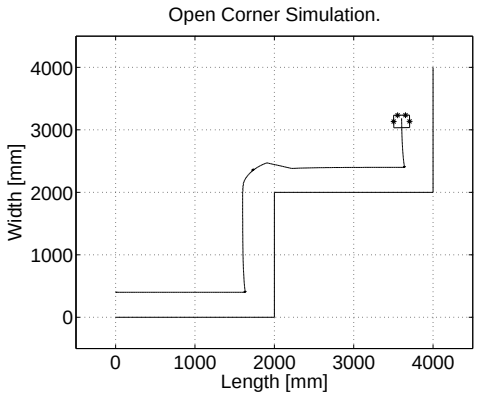


FIGURE 4.22: The simulation result for haveng the controller handle the open corner.

to objects. This means the car should drive out of the doorway and within sensor distance of the wall on the other side of the door.

To see whether this is doable by the car a simulation is performed. The simulation is expected to show that the car makes and arce towards following the open corner around, detect the door and drive back into the room again.

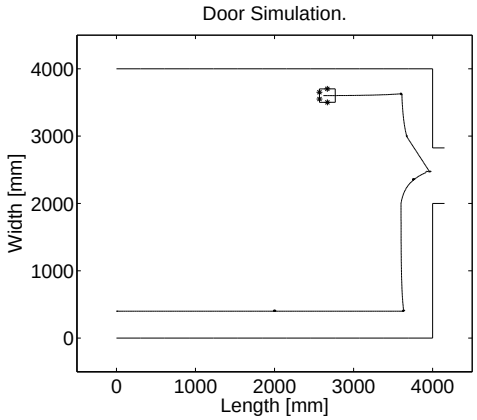


FIGURE 4.23: The simulation of how the car reacts to open doors.

As seen in Figure 4.23 does the car show the expected response to the open door and continous the mapping according to the normal case scenario, when it has passed the door. This means that the feedback control is suitable for handling the task of getting the car back on track after it has passed a door.

4.7 Discussion of the Controller

The main use of teh controller is to bring the center of the car to a distance of 40 *cm* to the wall it is driving alongside. The wall is on the car's right and as such is the right side sensor used for feedback. The requirements were that the step response rise time of around 0.174 *s*, an overshoot of maximum 10% and a maximum angle the car can have to the wall of $\pm 16^{\circ}$. For simple one state systems the commonly used controller is PID, which was initially chosen for this controller as. The design of the controller were performed in steps as two aspects was considered. Teh first aspect was that the controller have to live up the the requirements and the second aspect was that if two controller both lived up to the requirements, the most simple of the two would be chosen. The first controller designed adn tuned was a P controller, which proved to be capable of fulfilling the requirements. Though the P controller did well, a PI controller was designed to see whether it would do even better. The PI controller proved to do worse than the P controller as it had a steady state error, caused by the changing angle to the wall, which mean that, seen from the PI controllers, was the reference moved closer to the wall. As the PI controller is worse thatn the P controller a PID was not designed, as the D part would make it more sensitive to disturbances adn it would make the controller more complex than the P controller.

Besides being able to drive alongside a wall, it would be prefered if the controller can be used in other aspects of the mapping with the Mapping by Sweeps pattern. Four such aspects were simulated; the sweeps, where the car experiances up to a 4 *cm* dispacement, objects, where the car is dispaced with about 40 *cm*, open corners, which are solely handled by the controller, and doors, which are an extension to the handling of open corners and also uses the dispacement made in the handling of objects. For all four of these aspects the simulations implied that the controller is capable of handling the situation after a feedforward has been used.

Chapter 5

Accepttest

Contents

The tests are simulated in accordance to the accepttest specification in section 2.4. The acceptest are divided into car variable and room variables like in the specification. Each test will be discussed and at the end of this chapeter will the conclusion contain a table showing whether the simulations were a succes or not.

5.1 Test 1: All Fixed

The simulateion results are shown in Figure 5.1 and was ran ten times to ensure a higher credability.

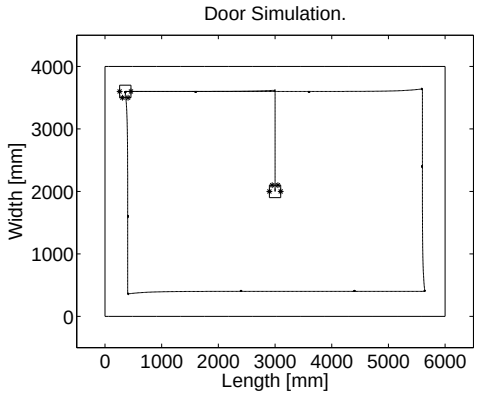


FIGURE 5.1: Simulation of the mapping with all cariables fixed.

The simulation showed the same behaviour all ten times with a mapping time of 100.1 s.

Test	Result	Comments
1: All Fixed	Success	No deviations

5.2 Test 2: Random Start Position

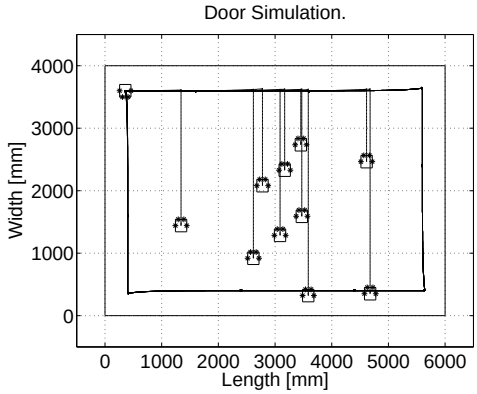


FIGURE 5.2: Simulation of the random start position test.

Ten simulations were ran and the results are shown in Figure 5.2. The start positions were randomly generated and the times to complete was:

Mean: 103 s

Maximum: 114 s

Minimum: 95.2 s

Test	Result	Comments
2: Random Start Position	Success	

5.3 Test 3: Random Start Direction

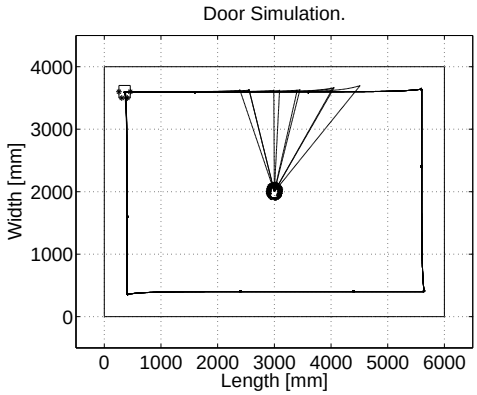


FIGURE 5.3: The simulation of the random start direction test.

As with the two preveous test, were ten simulations ran. The start direction was randomly generated. The times to complete the mapping, shown in Figure 5.3, were:

Mean: 102 s

Maximum: 109 s

Minimum: 97.9 s

Test	Result	Comments
3: Random Start Direction	Success	

5.4 Test 4: Random Side Up

This test has not been performed as this would require implementation of the simulated code in C and on the micro computer.

Test	Result	Comments
4: Random Side Up		Not conducted

5.5 Test 5: All Random (Car Variables)

For this test the simulation was ran 100 times to cach any errors. The simulation shown in Figure 5.4 is for a simulation with ten runs. When the 100 runs simulations was conducted no errors occured, but one simulation indicated that the car can come too close to the walls when driving towards the first wall. This is acceptable as the motors are deemed strong enough to keep the forwards movement, even if the wheels are dragging alongside the walls.

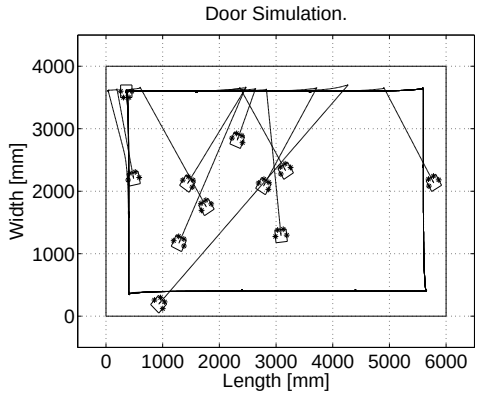


FIGURE 5.4: The simulation of the all random test, for the car variables.

The times to map were:

Mean: 101 s

Maximum: 118 s

Minimum: 85.5 s

Test	Result	Comments
5: All Random (Car Variables)	Success	Might have the car dragging the wheels alongside the walls.

5.6 Test 6: Random Shape

For this test it was simulated whether the car can react correctly to both open and closed corners. This test was described and discussed in **??** and is for its purpose as accepttest deemed acceptable behaviour. The simulation is shown in Figure 5.5

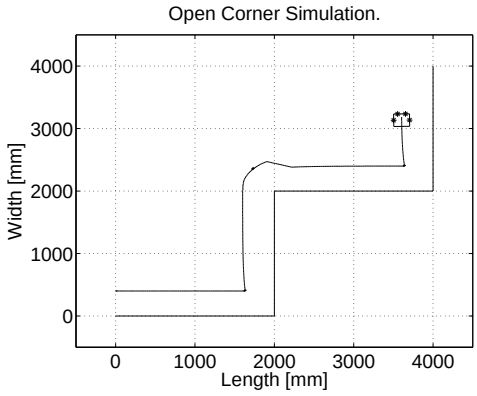


FIGURE 5.5: Simulation of room shapes test.

Test	Result	Comments
6: Random Shape	Success	

5.7 Test 7: Door

As with test 6, this was performed when testing to see if the car's feedback control can handle exception. This test is discussed in 4.6.4 and showed an acceptable beaviour, as illustrated in Figure 5.6.

Test	Result	Comments
7: Door	Success	

5.8 Test 8: Object in Drive Path

This test has been conducted in 4.6.2 and showed that the controller can compensate for more than the 20 *cm* offset it is designed to handle, as long as it is away from the wall and not closer to it. The result of teh simulation is shown in Figure 5.7.

Test	Result	Comments
8: Object in Drive Path	Success	

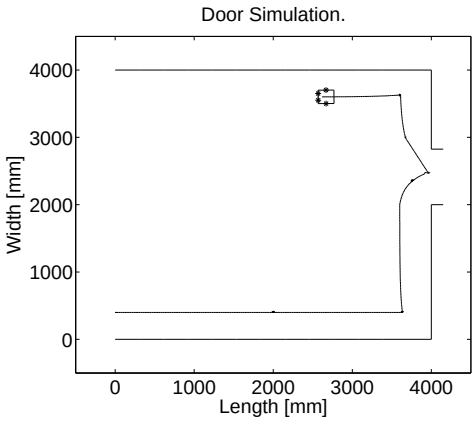


FIGURE 5.6: Simulation of the Door test.

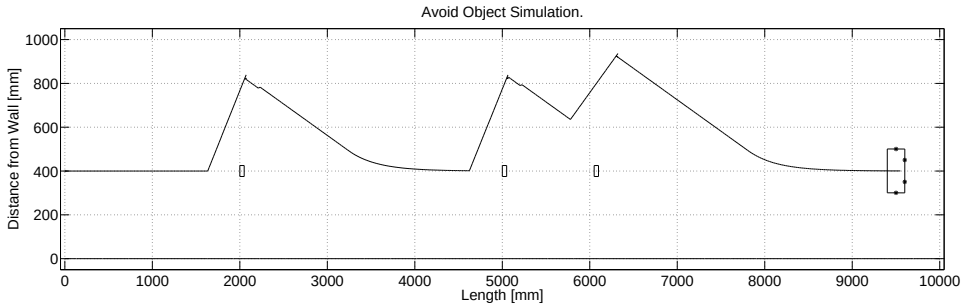


FIGURE 5.7: Simulation of the Object in Drive Path test.

5.9 Test 9: All Random (Room Variables)

This test was not conducted.

Test	Result	Comments
9: All Randoms (Room Variables)		Not Conducted

5.10 Accepttest Conclusion

Test	Result	Comments
1: All Fixed	Success	No deviations
2: Random Start Position	Success	
3: Random Start Direction	Success	
4: Random Side Up		Not conducted
5: All Random (Car Variables)	Success	Might have the car dragging the wheels alongside the walls.
6: Random Shape	Success	
7: Door	Success	
8: Object in Drive Path	Success	
9: All Randoms (Room Variables)		Not Conducted

Chapter 6

Conclusion

Contents

This chapeter concludes the main report and as such is the major topics of the report discussed here. After the discussion the conclusion of the project is presensted.

6.1 Discussion

This report is based on designing and constructing a ligthweight robot in form of a car. In the introduction two parameters which is the minimum requrements for artificial intelligence is mentioned. The first is that for a robot to have artificial intelligence it will have to be autonomous, and the second is that it shall have some cognitive capabilities. The car designed and constructed in this project has only one of those two. It is an autonomous vehicle designed to map an arbitrary room.

The report mainly deal with how to plan the mapping to get a full video map of the room, how to contruct a car which is ligth wight and still got the needed sensors and actuator, designing a controller and consider how widely it can be used by the car, and simulations of the pattern deemed the better and of how the controller act when the car is confronted with different obstacles.

Three mapping patterns were considered. Two of these were meant for rectangular rooms only and as the discussion of these tell, these does not give a clear view of the entire room if it should be non-rectangular. The third pattern, Mapping with Sweeps, discussed were designed with two main purposes; it should be able to map room shapes other that the rectangular and should have feedback control for controlling the distance to the wall, which it will drive alongside. This pattern is better suited for non-rectangular rooms, as it might not get a good view of the walls in the rectangular room. Still Mapping with Sweeps is the better choise, as it, due to its controller, has good obstacle avoidans, as it for every feedforwards has the controller to compensate for any offsets the feedforwards can introduced.

The main issuses when designing the car's physical systems was the weight and the robustness. These two issues arise from the needs of being carried by an autonomous helicopter and be able to survive landing on the floor after being send in through a window. All hardware components were selected mostly due to their advantage in low weight, which means that the car is less robust than wanted. When it comes to the robustness of the car the weakest points are considered to be the wheel bearings. Should the wheel bearings survive the landing the four μ-servos is the strongest parts of the construction. These are powerful enough to propel the car if placed against a wall where the drag of the wheels are considered the highest.

Two controller were designed, a P and a PI. For both controllers was a rate limiter used, as to large angle can result in an inability to detect corners. The P controller was tuned to fulfil the requirements regarding rise time, and overshoot. For the PI to be better it would have to be faster and keep

within the 10% overshoot allowed. A problem was encountered during the simulation and the tuning of the PI controller, though. It had a steady state error. This is due to the way the controller relates to the sensor. Normally the sensor measures the distance perpendicular to the wall or surface, but as the car changes its angle to the wall depending on controller output, the PI will miscalculate the distance to the wall and thereby get a steady state error of about 5%. The P controller however is less sensitive to the skew angles as the angles of with the P controller are lower than for the PI controller when closing in on the reference. As such, the P controller does not introduce any steady state error and as it has no overshoot it can get the car from an offset of $+50$ cm to its reference.

To verify the mapping pattern and the controller, simulations were made. These cover the issues, such as response to objects, discussed in the analysis. The simulations of the car driving according to Mapping with Sweeps indicated that the car will detect and react correctly to obstacles and that the controller can be used for more than just having the car drive alongside a wall.

6.2 Conclusion

An battery driven car has been design and contructed. Before the implementation of software can take place the autonomity and intelligence of the system is considered and simulated. The chosen mapping pattern, Mapping with Sweeps, was found best of the three considered in 4 out of 6 parameters. Looking at the scores for these parameter, the chosen pattern had almost thrice the points the other two had.

For every component of the physical system a low weight was considered of high importance. The total weight of the car is 280 g which is acceptable.

The controller is designed for the car to have it driving distance of 30 cm to the wall. It is a P controller with a rate limiter. The use of the controller has been simulated and it works as expected, with a rise time of 3.48 s for a step of 20, corresponds to adistance driven alongside the wall of 80 cm, no overshoot and a steady steady state error of 0.07%.

Using Mapping with Sweeps, it was simulated whether the controller could do more than just make the car follow a wall. The findings through simulations was that the controller could handle open corners witout interruptions in form of feedforwards. Other obstacles like the displacement of the car center that takes plece when the car is rotating, did the controller compensate for as well. In other words the controller compensated for every obstacle.

Appendix A

PC to Gumstix

Contents

There are different ways to communicate with the Gumstix, and it depends on the task that is going to be made, which type of communication is necessary. If for some reason the file system has to be changed on the Gumstix a serial communication line has to be used. This is the only way to get access to the file system. But if a program is running on the Gumstix, and some data is wanted to be read, there can be used a range of communication types depending on the hardware that is connected to the Gumstix for instance LAN, WLAN, Bluetooth. In this project serial and LAN is used. How to get connected with the two different network types are described in the following when using a Linux system.

A.1 PC to Gumstix - Serial

To log on to the Gumstix a console is opened and cKermit is run with the following arguments:

```
$ kermit -l /dev/ttyS0
```

where ttyS0 is the serial port that is connected to the Gumstix. Then the following configuration is set up in the serial cKermit console:

```
C-Kermit> set speed 115200
C-Kermit> set reliable
C-Kermit> fast
C-Kermit> set carrier-watch off
C-Kermit> set flow-control none
C-Kermit> set prefixing all
```

Then connect to the port by typing:

```
C-Kermit> connect
```

Now turn on the power to the Gumstix, then it will boot and be ready to log in to. If the Gumstix cant boot automaticly it can be booted manually, by interrupting the automatic boot by hitting a key when the Gumstix counts down for boot. Then type the following:

```
GUM> fsload a2000000 boot/uImage
GUM> bootm a2000000
```

To make these changes permanent, they have to be saved in the boot command (bootcmd) in the following way:

```
GUM> setenv bootcmd 'fsload a2000000 boot/uImage; bootm a2000000'
GUM> saveenv
```

A.2 PC to Gumstix - LAN

When the Gumstix is running, open a console and type:

```
$ ssh root@"IP address"
```

where "IP address" is the Gumstix's IP address to log on to the Gumstix. From a console it is now possible to copy files and programs on to the Gumstix by using the scp command in the following way:

```
$ scp "filename" root@"IP address"
```

Now the file or program can be run on the Gumstix by typing ./"filename".

A.3 PC to Robostix

When a connection to the Gumstix is running, a library is made on the Gumstix, when the Robostix and Gumstix are connected for the first time. This is done in the following way.

```
GUM> cd /rs/
GUM> insmod robostix_drv.ko
```

In this case "rs" is the library the Robostix program is plased in, robostix_drv.ko is the driver that power up the Robostix, and "insmod" is the command to load the driver. When this is done the Robostix is ready to use. The program is transferred to the rs library and to activate and remove it from the Robostix, uisp commands are used in the following way.

```
GUM> uisp --upload "filename"
GUM> uisp --erase
```

Here the "filename" is the program that is wanted to be run on the Robostix. Programs to the Robostix are .hex files.

Appendix B

Micro Computer

Contents

B.1 Introduction

To control the car a microcomputer is needed. The requirements for the microcomputer are listed below.

- PWM port to control the μ-servos
- ADC to get data from the analogue sensors
- Interface to a PC

B.2 Gumpack

The chosen microcomputer for this project is a Gumpack, which contains two micro computers, a gumstix motherboard and a robostix microcontroller. This solution is small in size, and light weight, which is preffered. Furthermore this solution has the ports needed to interact with the motors and sensors, PWM and ADC respectively.

The two micro computers have the following specifications, found [Gumstix inc., 2007] and [Gumstix inc., 2007].

B.2.1 Gumstix connex 200xm

The technical specifications of the gumstix are shown below.

Processor: Intel XScale PXA255

CPU speed: 200 MHz

Flash memory: 4 MB

Connections: 60 pin Hirose I/O connector, 92-pin bus header

This is an older version of the gumstix, the new version has 16 MB flash memory instead of 4 MB. The 60 pin Hirose I/O connector is used for basix-side expansion board , in this case to connect to the robostix. The 92-pin bus header is used for connex-side expansion boards like cfstix.

B.2.2 Robostix R341

The technical specifications of the robostix are shown below.

Processor: ATMega128 (an Atmel AVR processor)

CPU speed: 16 MHz

Flash memory: 128 KB

Connections: 60 pin Hirose I/O connector, I2C bus

Other functions: FFUART, ADC, PWM, timers, interrupts

The robostix can run programs as a stand alone microcontroller, or be connected to the gumstix by the 60 pin Hirose I/O connector to maximize the programming capabilities.

To get data from the sensors, the ADC pins 0-4 at Port F on the robostix are used. The sensors are connected directly to this port, from where the they get their power supply.

The PWM output pins 1A and 1B at Port B are used to control the μ-servos. The μ-servos get their power from the power supply, and the signal wire is connected directly at the port on the robostix.

When the gumstix is mounted on the robostix with the 60 pin Hirose I/O connector, the FFUART pins can be used with a RS232 circuit to connect the gumstix with a PC. To activate this communication port, pin 1 is connected to pin 4, and pin 5 to pin 8 on the UART port on the robostix. The ports used are shown in Figure B.1.

There are three timers/counters on the robostix. One timer is used to generate the PWM output, another is used to control the input rate from the sensors, and the last timer is then usable for the robostix program.

The only circuit that has to be designed is the RS232 circuit. Interrupts are not used at this point.

B.3 MAX232A

As mentioned above the only external circuit needed is for communicating with a PC. To do this a MAX232A linedriver from Maxim is chosen due to its availability.

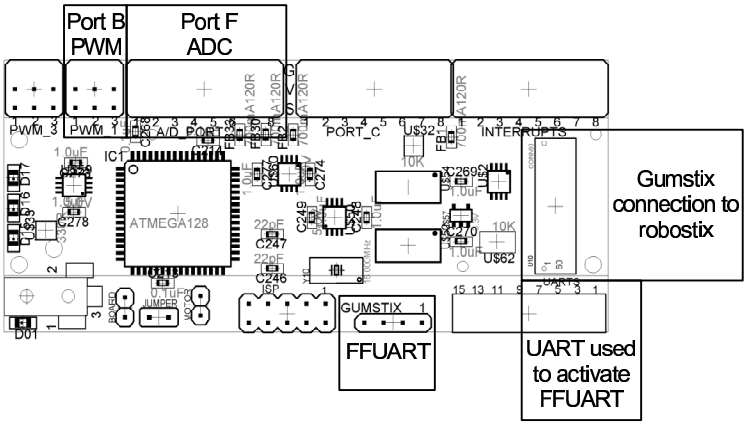


FIGURE B.1: The ports used on the robostix.

B.3.1 RS232 Communication

The MAX232A circuit is constructed as the datasheet[MAX238.pdf] describes. Figure B.2 shows the circuit diagram of MAX232A, with the used components. The circuit is constructed on its own Veroboard, and are connected to a PC with leads and a DB9 female connector, and the robostix with leads and a four pin connector.

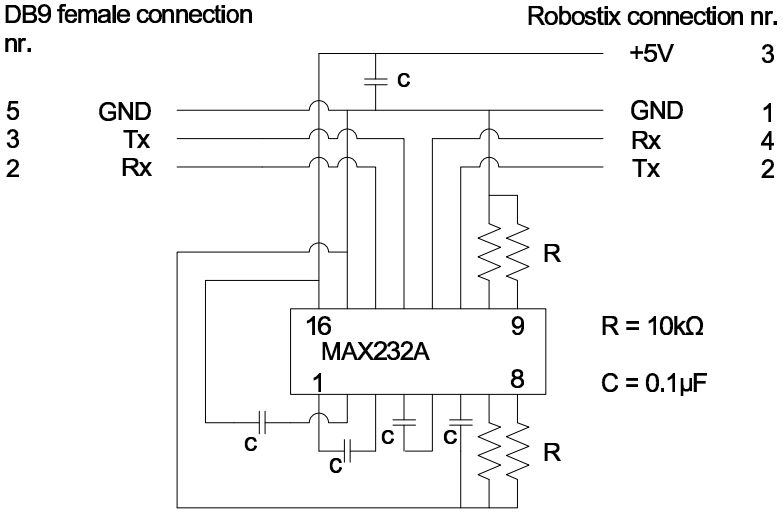


FIGURE B.2: Diagram of MAX232A circuit.

Discussion

The circuit has been succesfully used to transfer programs from a PC to the gumstix. This indicates that the circuit works as wanted, and therefore will be used in the communication between a PC and the gumstix.

B.4 Conclusion

The gum pack fulfills the requirements, and in combination with the RS232A circuit it can be used directly in this project. The PWM port (1A-B) is used to control the motors, the ADC port (0-4) is used to get data from the sensors, and the FFUART port is used to communicate with a PC through the RS232A circuit.

Appendix C

Up/Down Sensor

Contents

C.1 Introduction

During its fall, the car may flip and thereby land upside down. Therefore it has to identify which side is up. To do this a sensor is placed facing downwards, initially, as illustrated in Figure C.1.

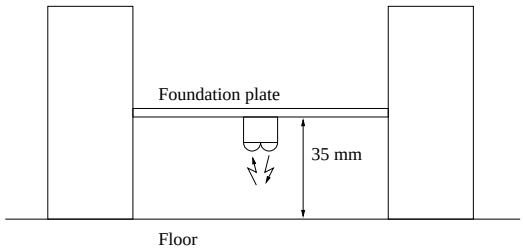


FIGURE C.1: The sensor, used to detect the floor, is initially facing downward.

The idea is to measure the distance from the bottom of the car to the floor and thereby determine whether the car has flipped. In practice a threshold is defined and if the distance is greater, the car has flipped.

C.2 Requirement Specification

Distance measuring: From the bottom of the car to the floor is an estimated 35 mm due to the wheel diameter. The difference in sensor output, depending on distance, has to be distinct.

Weight: As with the design of the rest of the car, the sensor has to be light weight.

C.3 Possibilities

Of the sensors available, two has been chosen. These are tested with respect to the two requirements.

SHARP GP2D15. This is an IR sensor. It has an internal distance measuring IC and a connector, with supply, ground, and output signal pins. These features eases the integration with other hardware components. The output is digital, which means that the output, while closer than a predefined distance, is high and low, when not.

The distance interval of the GP2D15 is $10 - 80$ cm.
The weight of the GP2D15 is 3.6 g.

Optek OPB704. This is also an IR sensor, but it consists only of a LED and a phototransistor in a casing. To control the current through the LED, an external circuit will have to be designed.
The distance interval of the OPB704 is $5 - 20$ mm.
The weight of the OPB704 is 1.4 g, plus any extra components needed.[OPB704-extra.pdf]

C.4 Test of a GP2D15

The test of the GP2D15 is conducted to determine whether, or how, suitable it is for the purpose of detecting whether the car is turning upside down.
The following is used for the test:

	Manufacturer	Model	Lab.No.
Power Supply	HAMEG	HM7042-3	52755
Oscilloscope	Agilent	54621D	52772

GP2D15 circuit: According to the datasheet, the GP2D15 needs a pull up resistor of 12 $k\Omega$. The circuit is illustrated in Figure C.2.

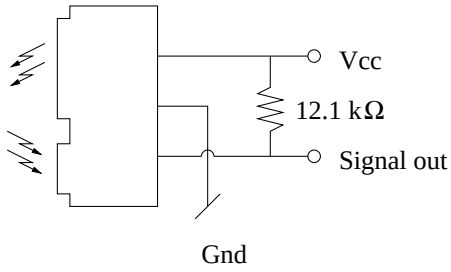


FIGURE C.2: GP2D15 circuit.

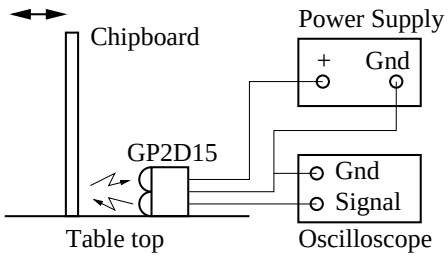


FIGURE C.3: GP2D15 test set-up.

C.4.1 Test Set-up

The test was conducted by connecting the GP2D15, supply and ground, to the power supply and the GP2D15, signal and ground, to the oscilloscope. The sensor was then placed on a table with the detection direction parallel to the table. A chipboard was used to act as a floor and was placed at 16 different distances from the sensor, starting with 0 mm and increased with

a 50 mm interval. The test was performed three times at different times in the day, at 9am, 1pm, and 4pm. Figure C.3 illustrates the test set-up.

The test was expected to show that above a threshold, $\approx 240\ mm$, the output signal is low and below the threshold the output signal is high.

C.4.2 Results

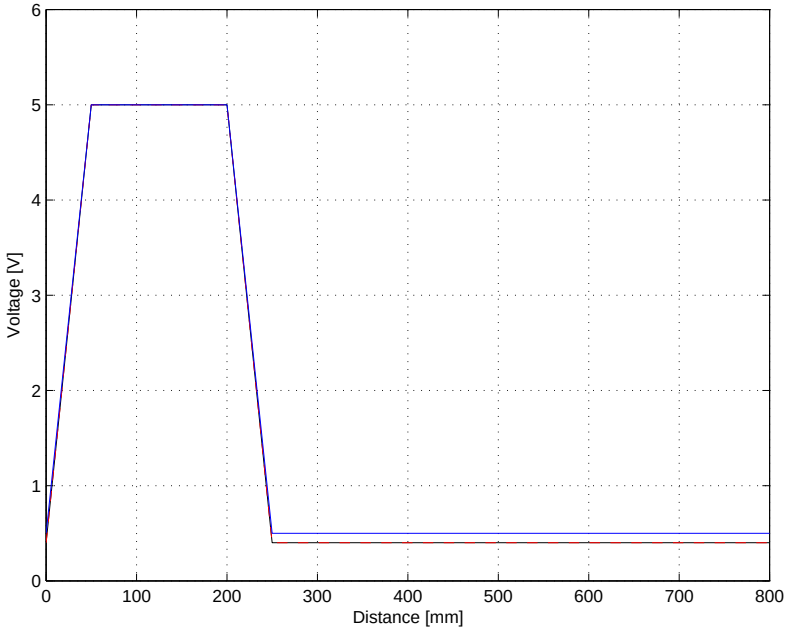


FIGURE C.4: Test of the GP2D15 IR digital distance sensor.

C.4.3 Discussion

The test result, illustrated in Figure C.4, shows that the sensor acts as expected within its range of operation, $10 - 80\ cm$. Furthermore the sensor is working acceptable consistent at varying light effects. At 0 cm the GP2D15 sensor detects no object though, just as it would if an object was at a distance above the threshold. Therefore an additional test were performed where the chipboard started at 0 mm and moved away from the sensor 10 times, with an interval of 5 mm. The test results from this is shown in Figure C.5.

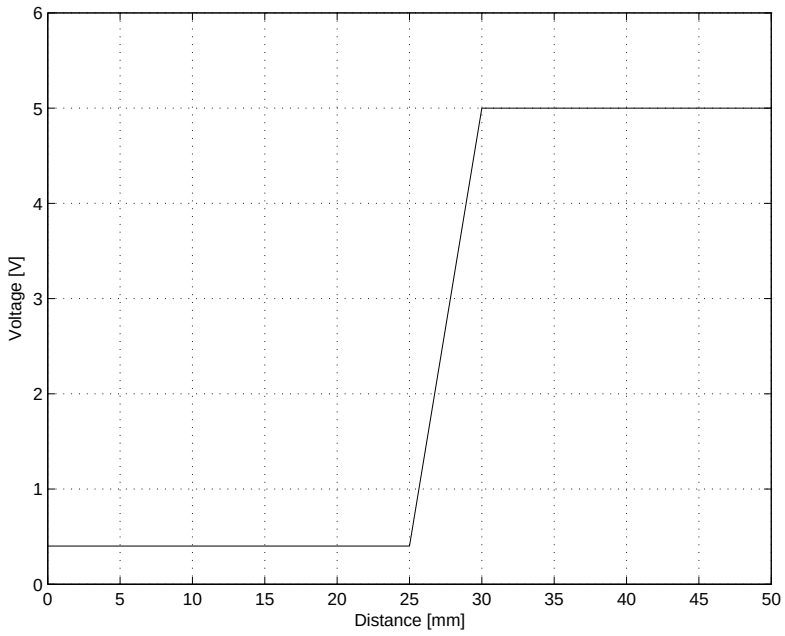


FIGURE C.5: Result of the additional test of the GP2D15 sensor.

The results of the additional test shows that the output signal is low at 25 mm, which is the same as at maximum distance (35 mm minus the sensor height) and therefore the GP2D15 sensor is deemed not usable.

C.5 Test of a OPB704

This test were conducted to find the variation in the output signal depending on distance.

The following test equipment were used:

	Manufacturer	Model	Lab.No.
Power Supply	HAMEG	HM7042-3	52755
Oscilloscope	Agilent	54621D	52772

OPB704 circuit: To gain a useful output from the sensor an external circuit is needed. The circuit used, is illustrated in Figure C.6[1]. The basic idea is to have a low output when the sensor is detecting the floor and high otherwise. Therefore a resistor connects the output with Vcc.

Furthermore there is a need to control the current through the LED. The current wanted is 20 mA. A resistor is serial connected from Vcc to the anode on the LED. The voltage drop across the LED is 1.6 V. Resistor dimensioning:

$$R = \frac{(U_{Vcc} - U_{LED})}{I} = \frac{5\ V - 1.6\ V}{20\ mA} \approx 180\ \Omega$$

The emitter of the phototransistor and the cathode of the LED is both connected to ground.

C.5.1 Test Set-up

The test was conducted by connecting the power supply and the oscilloscope to the circuit. The sensor was then placed on a table with the detection direction parallel to the table. A chipboard was used to act as a floor, as seen in Figure C.7, and was placed at 11 different distances from the sensor, starting with 0 mm and increased with a 5 mm interval.

The test was expected to show that above a threshold, $\approx 15\ mm$, the output signal is high and below the threshold the output signal is low.

[1]The circuit was found on the Internet, at http://www.roborugby.org/optical.html

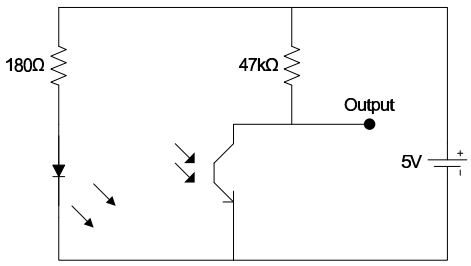


FIGURE C.6: OPB704 circuit.

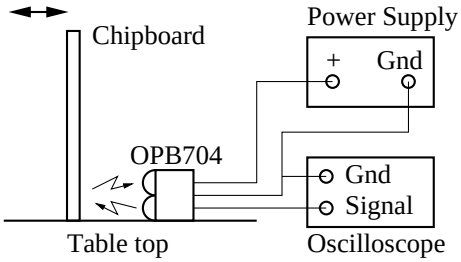


FIGURE C.7: OPB704 test set-up.

C.5.2 Results

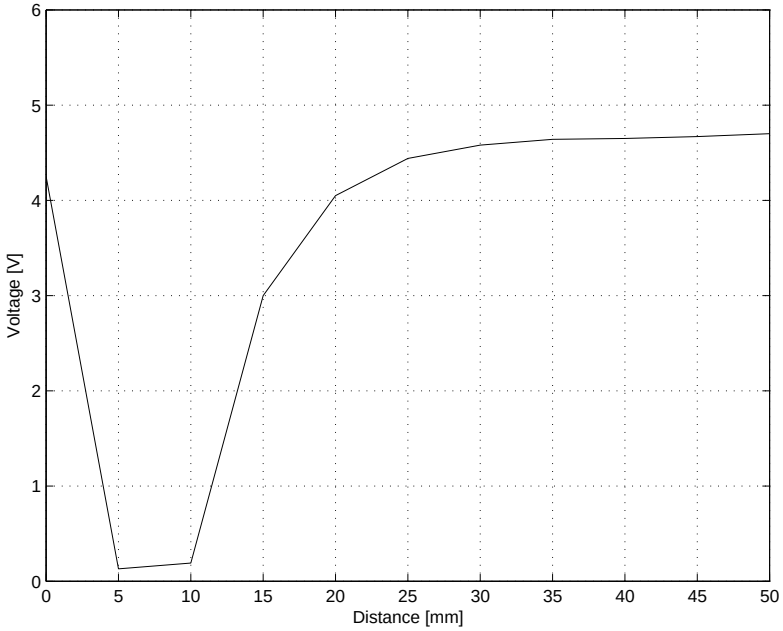


FIGURE C.8: Test results from OPB704.

C.5.3 Discussion

The test results, illustrated in Figure C.8, show that there is a difference in the signal output depending on whether the distance is 15 mm or more than 20 mm. The difference is not as distinct as preferred. This means that the sensor will have to be lowered $5 - 10$ mm from the bottom of the car.

The total weight of the OPB704 and circuit is 2.4 g, which is acceptable.

The OPB704 sensor is deemed usable.

C.6 Conclusion

The tests of the two sensors were conducted to find the better one and with the requirements stated in the beginning of this appendix, only one of the sensors are usable, the OPB704. Another fact to back up this choice is that the OPB704 is the most light weight of the two sensors.

Appendix D

Distance Sensors

Contents

To find its way in the room, the car has four distance sensors mounted. Two pointing forwards and one pointing to either side, as shown in Figure D.1.

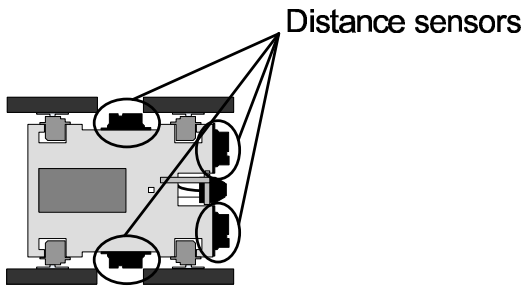


FIGURE D.1: The sensors are needed for the car to identify doors, objects, and corners.

The side sensors will furthermore be used for feedback control to have the car drive alongside the walls.

D.1 Requirements

Distance: The sensor have to be able measure distances up to at least 50 cm. This will give the car time to respond to what the sensors measures.

Weight The distance sensors has to be light weight as with the rest of the components of the car.

D.2 Possibilities

SHARP GP2D12. This is an IR sensor. It has an internal distance measuring IC and a connector with supply, ground, and signal pins. It is easy mountable with two screws. Its output is an analogue signal of $0.6-4.7\,V$ wich corresponds to the distance range $10-80\,cm$.[GP2D12.pdf] The weight of the sensor is 3.6 g.

Other types of sensors has been looked into, but as the GP2D12 should live up to the requirements and it is availeble, it was chosen to be used.

D.3 Test of the GP2D12

The sensor type was already estimated usable for this project, so what needed testing is the exact realionship between the distance and the output voltage.

The car four sensors mounted and each of these were tested individually while mounted. The sensors used the power supply from the Robostix, as they were already mounted on the car.

To read the output voltage a multimeter were used:

	Manufacturer	Model	Lab.No.
Multimeter	UNIGOT	A43	08097

D.3.1 Test Set-up

The GP2D12 was connected to the Robostix and the multimeter to the output and ground. the car was placed on the floor with the sensor being tested 10 *cm* from the wall. The car was the moved five centimeters at the time to a distance of 80 *cm*.

This was repeated twice for each sensor to ensure the reliability of the test.

The expected result was that the output would decrease the furhter away the sensor got from the wall and that the difference in output voltage would be significantly enough, depending on distance, to have a resolution of at least five centimeters.

D.3.2 Results

D.3.3 Discussion

The mminimum measureing distance for all four sensors is 10 *cm*, as Figure D.2 implies. The two front sensors has a difference in output voltage at distances up to 70 *cm*, as seen from the two top graphs in Figure D.2. The two side sensors is hava a lesser range of up to 60 *cm*. This difference might be caused by interuptions from the wheels, which are place less than a centimeter from the side sensors. None of teh sensors live up to the 80 *cm* stated in the datasheet, which can be the cause of otehr light conditions, and a different surface of the object, to which the distance is measured.

For this project the most used sensor ranges will be about $10 - 50$ *cm*. Therefore does the lack of $10 - 20$ *cm* maximum measureing distance not interfere with the car performing its tasks depending on the sensors.

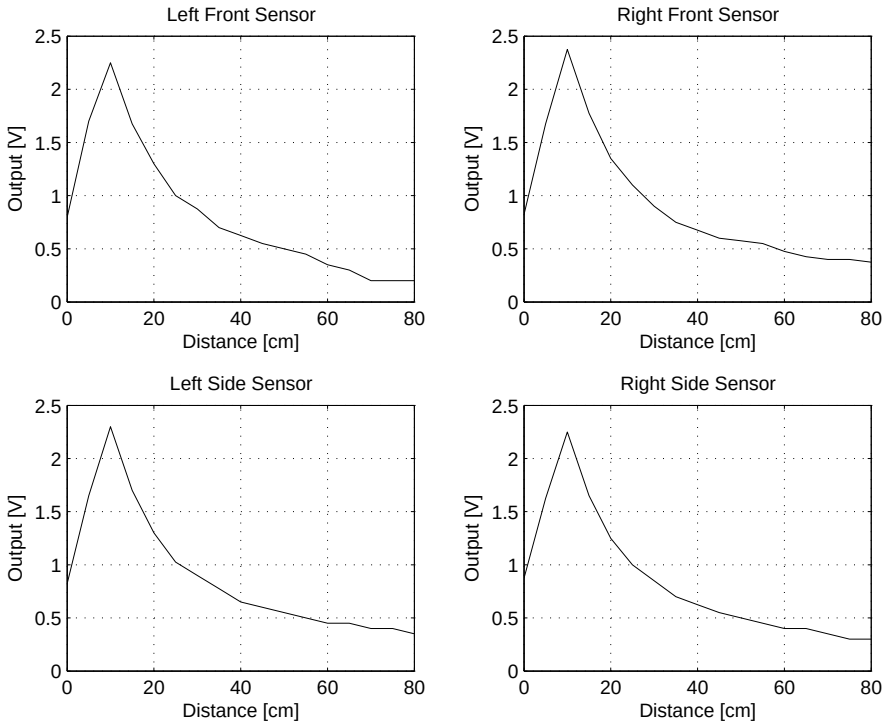


FIGURE D.2: The graphs shows the average of the two test conducted for each sensor.

D.3.4 Conclusion

The test of the GP2D12 showed that they all live up to the requirement of having a maximum range of more than 50 cm, as it has a measuring range of $10 - 60$ cm. The weight is 3.6 g per sensor, which is demmed an acceptable low weight.

Appendix E

Motors

Contents

The electric propulsion of the car is chosen to be provided by servo motors, and in particular μ-servos due to the light weight and small size. The μ-servo candidates were found on the internet, from a danish store the university have used before.

E.1 Introduction

Weight is an issue in the design of the car, which means that the power-to-weight ratio considerations are of high importance. The car has to sustain a fall and, as such, it is deemed neccesary to minimize the mechanics in the design of the car. As a result of this, the car will have an engine for each wheel to avoid the use of gears and drive belts, which could result in mechanical failure as a consequence of the car hitting the floor at the end of its flight. Choosing to use four motors has the disadvantage of making the car heavier. The motor set-up is illustrated in figure E.1.

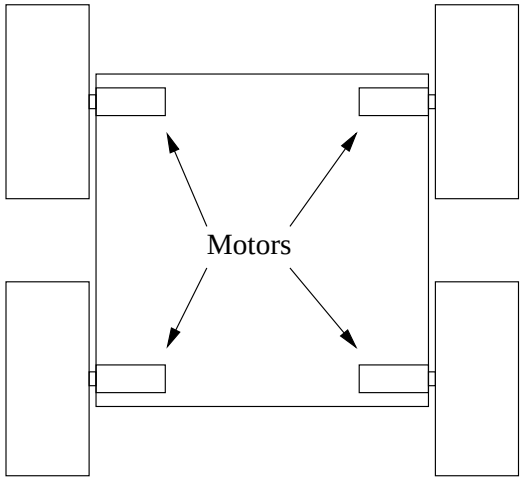


FIGURE E.1: The four motor set-up is chosen to lessen the use of mechanical components.

Two factors is considered in the choice of engine: Weight, and speed.

Weight, m: Addressing the weight factor, no maximum weight limit is specified. Instead, a specific type of motor is chosen. μ-servos have the advantage of low weight and enough torque to drive low weight vehicles and are even usable for small RC helicopters. Most μ-servos weigh less than 10 g.

Speed, v: Time is a factor, and as such is the speed of the car a factor as well. The speed of μ-servos is typically $58 - 111\ rpm$.

E.2 Possibilities

Of the μ-servos studied, two were considered to be best suitable for the car. One, which has the lowest weight and one, which has the highest speed. In Table E.1 the technical data for the two μ-servos is specified.

Data	Contraction	Hitec HS-50	Blue Bird BMS-303	Units
Weight	m	5	3.4	[g]
Speed	v	111	90.9	[rpm]
Torque	τ	0.0586	0.0777	[Nm]

TABLE E.1: Best suitable engine choises for the car. The lightest and the fastest of the μ-servos studied.

To find the maximum weight the two μ-servo types can drag, the power P is found.

$$P = \tau \cdot \omega = \tau \cdot \frac{rpm \cdot 360^\circ}{60} \tag{E.1}$$

$$P_{HS-50} = 0.0586 \cdot \frac{111 \cdot 360}{60} = 39\ W \tag{E.2}$$

$$P_{BMS-303} = 0.0777 \cdot \frac{90.9 \cdot 360}{60} = 42.4\ W \tag{E.3}$$

The total power of four μ-servos of each type yields:

$$P_{4xHS-50} = 156\ W \tag{E.4}$$

$$P_{4xBMS-303} = 169.5\ W \tag{E.5}$$

The maximum weight the μ-servos is able to drag, is found on the following way, where f_t is the total friction force, f_a is the friction force of air, and f_r is the force of rolling friction. Due to the size of the car, the friction force of air is neglected.

$$P = f_t \cdot v = (f_r + f_a)(\frac{rpm \cdot C}{60}) \quad \text{(E.6)}$$

$$= (\mu \cdot m \cdot g)(\frac{rpm \cdot C}{60}) \quad \text{(E.7)}$$

Here is the rolling friction coefficience assumed to be about the same as for a car tire on a road which is $\mu = 0.015$, m is the mass in [kg] which is wanted to be found, g is the force of gravity, and C is the circumference of the wheel. Now the equation for m is found:

$$m = \frac{P \cdot 60}{\mu \cdot g \cdot rpm \cdot C} = \frac{P \cdot 60}{0.015 \cdot 9.82 \cdot rpm \cdot 0.28} \quad \text{(E.8)}$$

Now the maximum weight the two μ-servo types can drag can be found.

$$m_{HS-50} = \frac{156 \cdot 60}{0.015 \cdot 9.82 \cdot 111 \cdot 0.28} \approx 2044\ kg \quad \text{(E.9)}$$

$$m_{BMS-303} = \frac{169.5 \cdot 60}{0.015 \cdot 9.82 \cdot 90.9 \cdot 0.28} \approx 2713\ kg \quad \text{(E.10)}$$

This shows that the two μ-servo types both are more than strong enough for the purpose of moving a small car like the one in this project. This means that it now depends on the weight and speed of the two μ-servos, and since they both are light weight the speed is considered most important.

The Hitec HS-50 has been chosen, as it is faster.

E.3 Test

The test is done by giving the μ-servo a PWM signal, and measure the rpm of the wheel. To measure the rpm a digital tachometer is used, which detects reflecting light. To reflect the light on the wheel two reflectors are put on the wheel opposed to each other to make the measurement more reliable. The test setup is shown in Figure E.2, and the test equipment is listed in the following Table.

	Manufacturer	Model	Lab.No.
Tachometer	Shimpo	DT-205	40158703

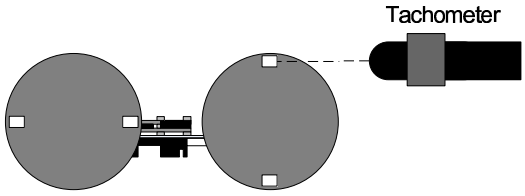


FIGURE E.2: Test setup for calibrating the pulsewidth with a speed of 100 rpm.

μ-servo nr.	RPM	Clockwise [μs]	Counter Clockwise [μs]
1	100	2745	370

TABLE E.2: Results from PWM test of one μ-servo, with a frequency of 50 Hz.

One μ-servo was tested at a rpm of 100 without load, and the pulsewidth of the signal required to reach this were found. The frequency used is 50 Hz which is common for servo's. The results are shown in Table E.2.

When testing another of the four μ-servos it did not give the same rpm, which indicate that the reference point for the μ-servos are placed at different places. The PWM signal were found for each μ-servo to reach exactly 100 rpm. The results can be seen in Table E.3.

μ-servo nr.	RPM	Clockwise [μs]	Counter Clockwise [μs]
1	100	2745	370
2	100	2775	350
3	100	2700	350
4	100	2780	350

TABLE E.3: The results from PWM test of all four μ-servos, with a frequency of 50 Hz.

The 100 rpm is chosen, because it is still faster then the alternate μ-servo type, and gives a velocity of the car on 1.68 km/h, which is sufficient to map a room of the given size. Furthermore it gives the possibility to calibrate the velocity of one side of the car up and down, if for some reason one side does not drive as fast as the other one, and thereby still obtain a velocity of the car on 1.68 km/h.

E.4 Discussion

The test showed that the HS-50 μ-servos that have been chosen have different reference points, and therefore needed to be calibrated seperately. It also showed that the μ-servos can hold a rpm of 100 as wanted, and thereby keep a velocity of 1.68 km/h of the car with the chosen wheels, but the test were conducted without load. This means that it can be expected that the rpm will decrease by a small amount, when the car is put down on the ground, but not enough to make a significant difference.

E.5 Conclusion

The HS-50 μ-servos were chosen because it was the fastest, and the test showed that it works as wanted for the purpose of this project. With this μ-servo the car can drive with a velocity of 1.68 km/h, which is sufficient for the roomsize given in this project. Therefore the conclusion is that the HS-50 μ-servo is used in this project as driving engines.

Appendix F

The Camera

Contents

The on board camera was found browsing the Internet for spy cameras, as we assumed these has the functionality, and the light weight, we need. The camera deemed best, judging from the specifications given, is the Wireless Camera GP-811T. The receiver for this camera connects to the USB-port in a computer and the camera is online right after installing the software that comes with the camera on a cd.

To use the camera in an optimal way we had to make modifications to its power supply cord, as it had a connector almost the size of the camera itself. Also the view angles of the camera had to be found to define how far from the walls we need to be to film the faces of people in the room. Furthermore, a device was constructed to mount the camera on the car.

F.1 Transmission

Transmission Frequency: ISM-2.400 $\approx$ 2.483 GHz

Modulation Type: FM

Bandwidth: 18 MHz

Undisturbed Transmission Range: 100 m

F.2 Supply Cord

The camera uses an eight voltage supply which is assured by a transformer, connected to the mains. To use it on the car, the supply cord was cut and connected to the car batteries instead. The connection is shown in Figure F.1.

F.3 View Angles

The desired camera coverage is to be able to see approximately two meters up at a distance of two meters from the wall. This means the camera will need to have a horizontal view angle of $\sim 45°$. For the chosen camera the horizontal view angle is $\sim 44°$, which, by itself, is acceptable. And when taking into consideration that the camera is elevated 5 or 5.7 cm depending on which side of the car face upwards, the actual view height is approximately 1.98 m at a two meters distance to the wall. The vertical and horizontal view angles is show in Figures F.2 and F.3 respectively.

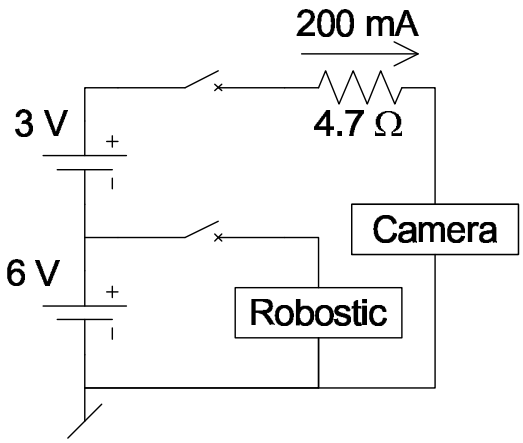


FIGURE F.1: The camera circuit.

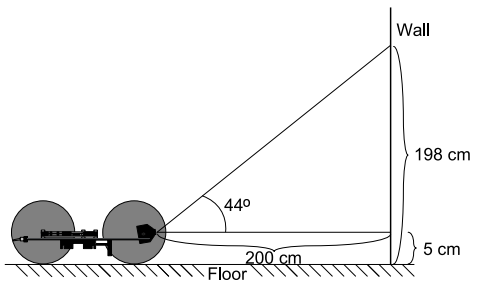


FIGURE F.2: The vertical view angle of the camera.

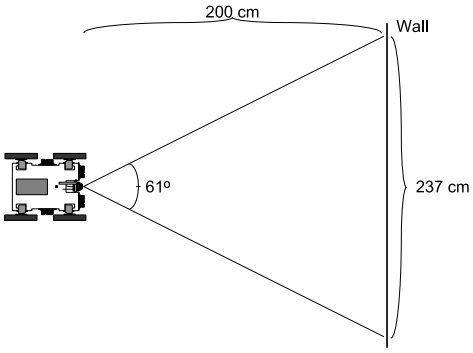


FIGURE F.3: The horizontal view angle of the camera.

F.4 Mounting

To film the room the car has a mounted camera pointing forwards. For full use of the camera's scope, a tilting mechanism is designed to have the camera film as little of the floor as possible, measured at its operating distance. The tilt mechanism is shown in Figure F.4 and its mount on the car is illustrated in Figure F.5.

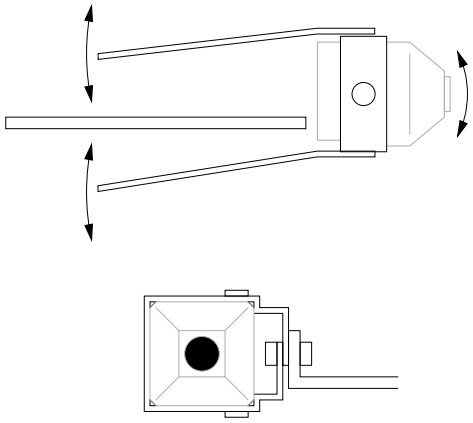


FIGURE F.4: The tilt mechanism.

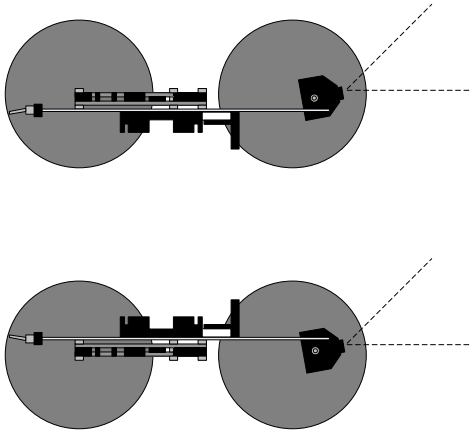


FIGURE F.5: The mounted tilt mechanism.

Appendix G

Power Supply

Contents

The circuit for the car's electrical power supply is illustrated in Figure G.1. A test to find the current through each components, The Robostix, the motors, and the camera, was performed. The general test setup is shown in Figure G.2 and the laboratory equipment used was:

	Manufacturer	Model	Lab.No.
Power Supply	HAMEG	HM7042-3	52755

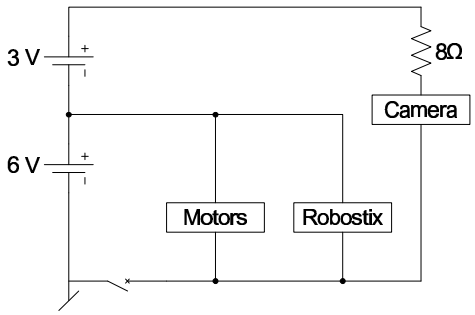


FIGURE G.1: The Power supply circuit for the car's systems.

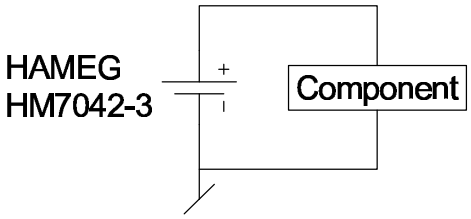


FIGURE G.2: The general test setup for the tests of the components in the supply circuit.

G.1 Robostix, Gumstix, and Sensors

To find the maximum current through the car system the first test conducted was for the Robostix, which also functions as supply for the Gumstix and the sensors. The following five tests were conducted.

When idle:	215 mA
With uploaded Robostix driver and idle:	263 mA
When data is send from a PC the Gumstix:	325 mA
When the Gumstix sends data to the Robostix	300 mA
When the both the Robosix and Gumstix are active	330 mA

G.2 Motors

Three test where performed on the μ-servos, when idle, when active with no load, and when active and driving on the floor (with load).

When idle:	43 mA
With no load:	400 mA
With load:	$1450 - 1550$ mA

G.3 Camera

As the camera is activated from the beginning of the mission and it only has one active state, and only one test was needed. The test of the active camera showed that the current through it is 125 mA. The camera needs to have 8 V and uses 125 mA, so a resistance of 8 Ω is inserted in series with the camera.

G.4 Discussion

The tests showed that the maximum total current through the three components is 2005 mA. The 6 V supply has to deliver this charge, while the less loaded 3 V supply has to deliver a charge of 125 mA.

The batteries used are Varta No.4003. Which are AAA batteries with a voltage supply of 1.5 V and a charge of 1200mAh.

This means that the 6 V supply can last for ${}^{1200\ mAh}/_{2005\ mA} = 0.6\ h \approx 36\ minutes$ and the 3 V supply can last for ${}^{1200\ mAh}/_{125\ mA} = 9.6\ h = 576\ minutes$.

With the circuit shown in Figure G.1 we encountered a problem though. It seems, the batteries in the 6 V supply cannot keep the voltage when the motors are activated by the Robostix. The voltage drop to 2.2 V which is not enough for the Robostix to keep active. To accommodate this problem an extra 6 V voltage supply, consisting of four extra batteries of the same kind, are added to the circuit in parallel to the first 6 V supply. The circuit with a total of three voltage supplies is illustrated in Figure G.3. The maximum charges are also included in this figure.

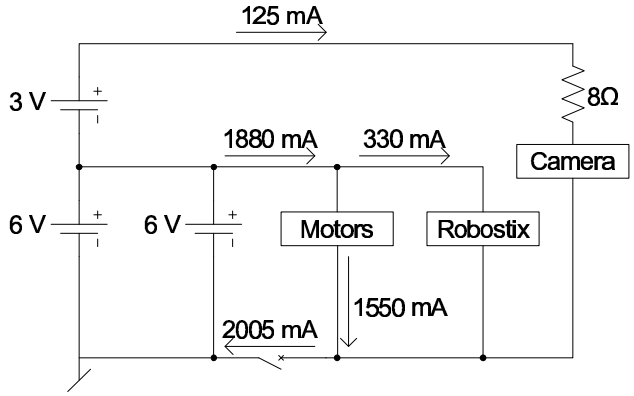


FIGURE G.3: The final power supply circuit.

As with the preveous circuit a test were perfomed on this circuit and the result was that it is able to keep the voltage and thereby keep the system active. The double 6 V supply can therefore keep the system active for 72 *minutes*.

G.5 Conclusion

The maximum system current load is 2005 mA which is supplied by use of 10 AAA batteries. The entire system can be kept active for 72 *minutes*.

List of Tables

List of Figures

Bibliography

[Biering-Sørensen et al, 1994] Stephen Biering-Sørensen et al. *Håndbog i Struktureret Program-Udvikling*. Teknisk Forlag, 1994, 1st. udgave. ISBN: 87-571-1046-8.

[The Danish Ministry of Science, Technology and Developement, 2006] Teknologiens fremsyn - om kognition og robotter (the vision for technology - on cognition and robots), April 2006. The Danish Ministry of Science, Technology and Developement.

[GP2D12.pdf] Sharp. *GP2D12 datasheet*.

[Gumstix inc., 2007] Gumstix inc. *Gumstix specifications*, 2007. URL: `http://gumstix.com/store/catalog/product_info.php?products_id=135`. Downloaded 2007-06-06.

[Gumstix inc., 2007] Gumstix inc. *Robostix specifications*, 2007. URL: `http://gumstix.com/store/catalog/product_info.php?cPath=31&products_id=%139`. Downloaded 2007-06-06.

[MAX238.pdf] Maxim. *MAX232A datasheet*, 2004.

[OPB704-extra.pdf] Optek. *OPB704 datasheet*, 1996.

[SWEDOOR-KILSGAARD, 2007] SWEDOOR-KILSGAARD. *Måleskema*, 2007. URL: `http://www.swedoor.dk/glatte_formpressede_maalskema.pdf`. Downloaded 2007-05-15.