BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Electrical Engineering
and Communication

BACHELOR'S THESIS

Brno, 2018 Michal Granát

BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF CONTROL AND INSTRUMENTATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

MOBILE ROBOT CONTROL AND GUIDANCE USING COMPUTER VISION

ŘÍZENÍ A NAVÁDĚNÍ KOLOVÉHO ROBOTA POMOCÍ POČÍTAČOVÉHO VIDĚNÍ

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR
AUTOR PRÁCE

Michal Granát

SUPERVISOR
VEDOUCÍ PRÁCE

Ing. Štefan Mišík

BRNO 2018

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

Bachelor's Thesis

Bachelor's study field **Automation and Measurement**
Department of Control and Instrumentation

Student: Michal Granát ***ID:*** 186068
Year of study: 3 ***Academic year:*** 2017/18

TITLE OF THESIS:

Mobile robot control and guidance using computer vision

INSTRUCTION:

The goal of this project is to design control algorithm for mobile robot to follow the path marked on the ground. Use camera attached to single board computer (e.g. BeagleBoard, Raspberry PI) to detect desired direction of the robot.

1. Research mobile robot control methods
2. Design testing track for the mobile robot
3. Identify features in the image taken by robot's camera, which can be used to control the robot
4. Use computer vision tools (e.g. OpenCV) to obtain these features from the image
5. Feed the features obtained from the image into controller(s)
6. Design several controllers (possibly controlling for different aspects of the robot's movement simultaneously, e.g. steering, velocity) and compare results using some suitable metric

RECOMMENDED LITERATURE:

[1] LAGANIÈRE, Robert. OpenCV 2 computer vision application programming cookbook: over 50 recipes to master this library of programming functions for real-time computer vision. Birmingham, U.K.: Packt Open Source Pub., 2011.

Date of project specification: 5.2.2018 ***Deadline for submission:*** 21.5.2018

Leader: Ing. Štefan Mišík
Consultant: Arben Cela

doc. Ing. Václav Jirsík, CSc.
Subject Council chairman

ABSTRACT

Products of theory of control are almost everywhere around us beginning with temperature controller in our houses through modeling of market behavior to spaceship control. The reason of its widespread is its performance and beauty hidden in mathematical approach. Wheeled mobile robots are also widely used in industry because of their loading capability which is necessary. High accuracy path-tracking is very important for the mobile robots to precisely follow the designed path. One of the easiest way how to mark the path is to use line with different optical features than background. There are several methods to detect line and most powerful and efficient is camera usage. Subsequently, there is need to use digital image processing which demanding for computational performance but there is possibility to obtain huge amount of data. Although this principle could look obsolete (guide line) it is still used in many solutions because of its simplicity and cheapness. What is more, using camera to navigate robot global (without using line) is sort of modern technology and it is still in development. Objective of this work is to assembly small mobile robot, create its mathematical model, design several controllers and use camera and computer vision for robot guidance using guide line.

KEYWORDS

Wheeled mobile robot, differential drive, digital image processing, line following, path tracking, openCV, raspberry pi, controller design, motion control

ABSTRAKT

Teória riadenia je aplikovaná takmer všade okolo nás, počnúc reguláciou teploty v našich domoch cez modelovanie správania sa trhu až po riadenie vesmírnych lodí. Dôvod, prečo je tomu tak je, je jej výkonnosť a krása, obe skryté v použitých matematických aparátoch. Kolesové mobilné roboty sú taktiež veľmi rozšírené, hlavne v priemysle, kvôli ich obrovskej nosnosti, ktorá je nevyhnutná. Veľmi presné detekovanie trasy je dôležité kvôli dôslednému riadeniu robota po jeho ceste. Jedna z najjednoduchších možností ako vynačiť trasu je použitie čiary, ktorá má iné optické vlastnosti ako jej podklad. Na detekovanie takto určenej trasy existuje niekoľko metód, ale najvýkonnejšie a najefektívnejšie je použitie kamery s následným digitánym spacovaním obrazu, ktoré je síce náročné na výpočtovú silu, ale je ním možne získať obrovské množstvo dát. Hoci táto metóda môže vyzerať zastaralo (vodiaca čiara) stále sa využíva v niektorých projektoch práve kvôli jej jednoduchosti a nízkej cene. Skutočné použitie kamery pre navigáciu všeobecne (nie pomocou čiary) je dokonca zatiaľ oblasť, ktorá je stále vo vývoji. Cieľom tejto práce je zostaviť malý mobilný robot, vytvoriť matematický model, navrhnúť niekoľko regulátorov a riadiť robot po vyznačenej čiare za použitia camery a počítačového videnia.

KĽÚČOVÉ SLOVÁ

Mobilný kolesový robot, diferenčný podvozok, digitálne spracovanie obrazu, sledovanie čiary, hľadanie cesty, openCV, raspberry pi, návrh regulátora, riadenie pohybu

GRANÁT, Michal. *Mobile robot control and guidance using computer vision*. Brno, 2018, 63 p. Bachelor's Thesis. Brno University of Technology, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Advised by Ing. Štefan Mišík

Vysázeno pomocí balíčku `thesis` verze 2.63; http://latex.feec.vutbr.cz

DECLARATION

I declare that I have written the Bachelor's Thesis titled "Mobile robot control and guidance using computer vision" independently, under the guidance of the advisor and using exclusively the technical references and other sources of information cited in the thesis and listed in the comprehensive bibliography at the end of the thesis.

As the author I furthermore declare that, with respect to the creation of this Bachelor's Thesis, I have not infringed any copyright or violated anyone's personal and/or ownership rights. In this context, I am fully aware of the consequences of breaking Regulation § 11 of the Copyright Act No. 121/2000 Coll. of the Czech Republic, as amended, and of any breach of rights related to intellectual property or introduced within amendments to relevant Acts such as the Intellectual Property Act or the Criminal Code, Act No. 40/2009 Coll., Section 2, Head VI, Part 4.

Brno ....

author's signature

ACKNOWLEDGEMENT

I would like to thank my supervisor for my bachelor thesis Mr. Štefan Mišík, Ph.D., for professional guidance, consultations, patience and inspirational suggestions for work.

Rád by som sa poďakoval vedúcemu mojej bakalárskej práce, pánovi Štefanovi Mišíkovi, Ph.D., za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

Brno . . . .

author's signature

CONTENTS

LIST OF FIGURES

1 INTRODUCTION

Guide a small robot using camera is not quite new idea, but there are still many domains that can be improved. Digital image processing has already been developing for more than fifty years and we can recognize faces and objects, but I would say we are not even in the middle of it capability. As always, limiting factor is computing power, but there is also place for optimization and new ideas. My first objective is to create algorithm that would obtain data from image but would be enough efficient and fast. Basically I need data for two controllers: first would control forward speed, second would control angular speed (rotation). If that would work I will try to obtain some more data, e.g. obstacle presence, line interruption, line crossing and more.

In my bachelor thesis I want to try to solve this classical problem more mathematically. That means that firstly I have to create mathematical model that would describe my robot and its behaviour. The best case would be to find linear model, but I am not sure if it would be possible with my chassis and linear motors. After that I will design number of controllers of different types using several standard methods like ITEA criteria. In order to reach the best speed of regulating I want to apply anti wind-up methods.

Code modularity and recycling is well known idea and one of my aims is to apply it. I want to create three main blocks of code, each in its own file: one for image processing, one for controllers and one for motor control. Each of them would have in advance defined interface so they could communicate with each other. When I will have to change chassis, robot type, controller or way of getting feedback I just simply change one of these modules.

2 RESEARCH OF MOBILE ROBOT CONTROL METHODS

Whatever you want to develop, it would by silly to start without checking history of your topic. That is the reason, why research is so important at the beginning of every study. In this chapter, I will try to find out something about methods of mobile robot control, that are already developed.

First I have to specify the exact type of chassis. For different kinds of drives there are, naturally, different methods of controlling and mathematics of control. For purposes of my thesis I am going to use robot chassis with differential drive. The exact type is TI-RSLK (Robot System Learning Kit developed by Texas Instruments). [7]

2.1 Differential drive

This means two independently powered wheels. Usually there is also one small omnidirectional wheel for support that is not powered. Rotation of robot is reached by difference in angular velocity of individual wheels. It is really simple but effective concept. This image in simplicity shows, how does differential drive work Figure 2.1. I am going to explain it more detailed in next chapter.

Some of most known uses of differential drive in industry are robot vacuum cleaner and some kinds of bagels. Tanc driver is also kind of differential drive, but it has belts instead of wheels.

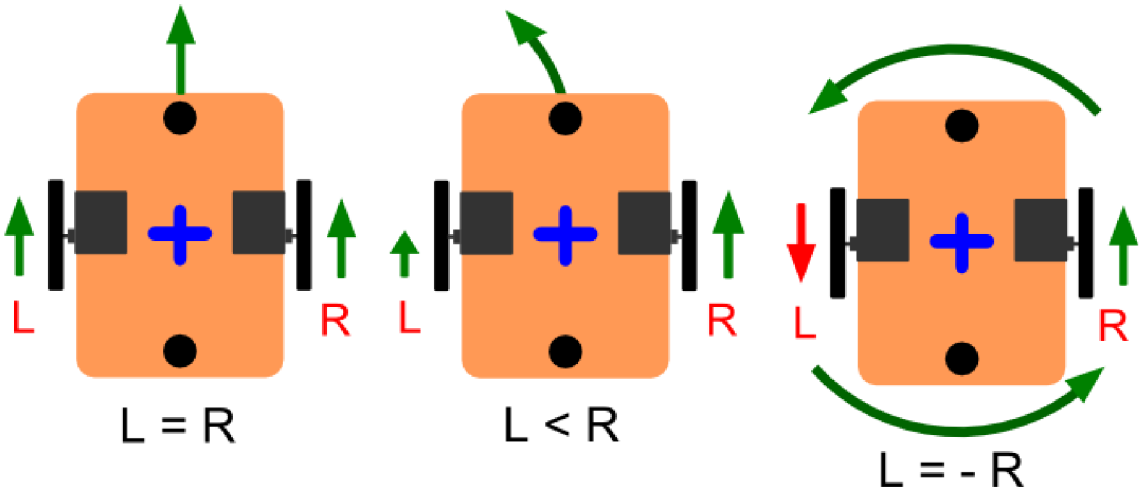


Fig. 2.1: Differential drive principle [1]

2.2 Trial-and-error method

This is the oldest method usable almost for everything. The reason, why is it so widespread is its simplicity. It is all about changing constants that determines relation between inputs and outputs, observing how system changed its behaviour and trying to find the best combination. But the simplest way do not have to be the shortest. It can take very long time to find solution for more difficult (more constants) systems and still, most likely, it will not be the most optimal one. So maybe it is good for everyday life but not for research. For my purposes I will choose one of following, more sophisticated methods.

2.3 Mathematical description - transfer function

One of the most powerful methods how to control anything is to find its mathematical description. Science discipline that deal with this is called theory of control and it is part of bigger unit called automation. The basic idea is, that each system of every kind can be described by differential equations where outputs are depended on inputs and initial conditions. These can be transformed to transfer function using Laplace transform. Transfer function is defined as the ratio of the output Y(s) of a system to the input U(s) of a system, in the Laplace domain considering its initial conditions and equilibrium point to be zero. (2.1)

$$G(s) = \frac{Y(s)}{U(s)} = \frac{b_m s_m + ... + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + ... + a_1 s + a_0} \tag{2.1}$$

Once we are in Laplace domain, in mathematical world, everything is possible. There are already quantity of verified methods designed for controller designing. Some of them are:

- Nyquist's stability criterion (Nyquist, 1932)
- H. S. Black's analysis of the feedback amplifier (Black, 1934)
- H. W. Bode's frequency domain analysis (Bode, 1940)
- W. R. Evans' root locus method (Evans, 1948)

[10]

2.4 Feedback

There is no sense of any controller if we did not define what is the correct state. In other words, what is the objective of robot's behaviour. In my case feedback would be provided by use of camera module and object of feedback will be caused deviation. As you can see in Figure 8.1 feedback feature is enclosing the control

loop. There is also a famous and little bit funny proverb beetwen automatics: there is not automation without measurement.

2.5 PID control applied on a line-follower AGV using a RGB camera

This article [2] is about using AGVs (Automated Guided Vehicles) in order to optimize material handling and similar systems. The reason is still increasing demand for efficiency in industry in order to increase productivity and reduce costs. On the other side, the use of inefficient methods and equipment may result in high costs due to the repetition of activities and time spent on these tasks. The other advantage of machines like ITSs (Intelligent Transportation Systems) is a potential to improve the safety and productivity on port operations.

There is one essential difference between AGVs and classical machines: AVGs are guided and positioned automatically, that means without a driver. The AGV control is generally based on laser, camera (optic) systems, magnetic or radio systems. There is possibility to use a combination of some or all systems, applying data fusion techniques.

To avoid collisions and deadlocks of the vehicle in its environment, there is need of consideration of kinematic and dynamic of the vehicle as well as an accurately execution of the trajectory provided by the route planner. The mathematical model of the vehicle can be designed as a SISO (Single Input Single Output), MISO (Multiple Inputs Single Output), SIMO (Single Input Multiple Output) or MIMO (Multiple Inputs Multiple Outputs) model, increasing or decreasing the complexity of the applied control.

My intention, as well as their, is to use PID control method after obtain data from image. The PID controller is a classic control method as it is one of the simplest methods capable of stabilizing a system without continuous oscillations commonly used, even though the non-linearities of the system are not considered. Its results are in their research compared with more complex control methods and their analysis demonstrated the non-necessity of control complexity which can save processing power and make control algorithm quicker.

As a robot model are they using the Pioneer 3DX because of their vastly applications, industrial simplicity and compatibility with systems like MORSE and ROS. They consider the kinematic model simplified as a punctual mass and the outputs of their model for actuators are linear (m/s) and angular (rad/s) velocity of vehicle.

2.5.1 Mathematics for camera used as a sensor

Interesting part of their research is image processing. They are using other technique than I have intended so it can be very enriching. They are using also another camera, but the idea could be useful for me as I know all essential constants of my camera that are used in the calculations. The basic difference in out ideas is that they are calculating the angle between robot axis and guide line Figure 2.2, what is smart, but it can be insufficient for track with sharp twists. My intention is to take first few lines of taken image can calculate the line offset from the image axle. Then I want to take few rows in two thirds of image and make some conclusions for future controlling and forward velocity.

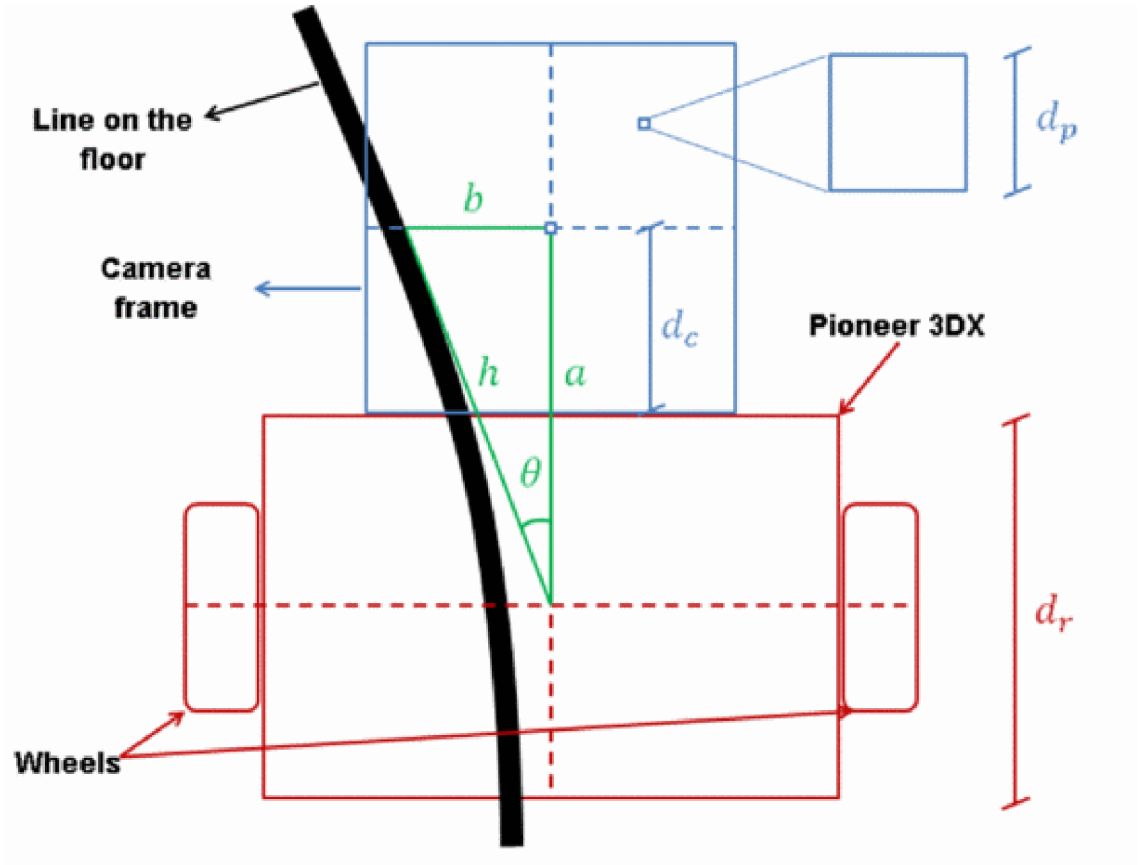


Fig. 2.2: Angular position θ, given with the camera frame [2]

The angle θ can be obtained from a and b, see (2.2)

$$\theta = \pm \arctan(\frac{a}{b}); \tag{2.2}$$

The b side is calculated at each sample as a count of white pixels from the centre of the camera frame to the reference line multiplied by the horizontal size of each pixel. From a known focal distance d_F (I need to calculate it or find in data-sheet)

and a known sensor width d_S is possible to obtain the image width (2.3). The sketch with their original values: Figure 2.3.

$$\tan \alpha = \frac{d_S}{2xd_F} = \frac{IM}{2xID} \tag{2.3}$$

, where IM is a distance between lens and an observed object (paper), in this case it is equal to $200mm$.

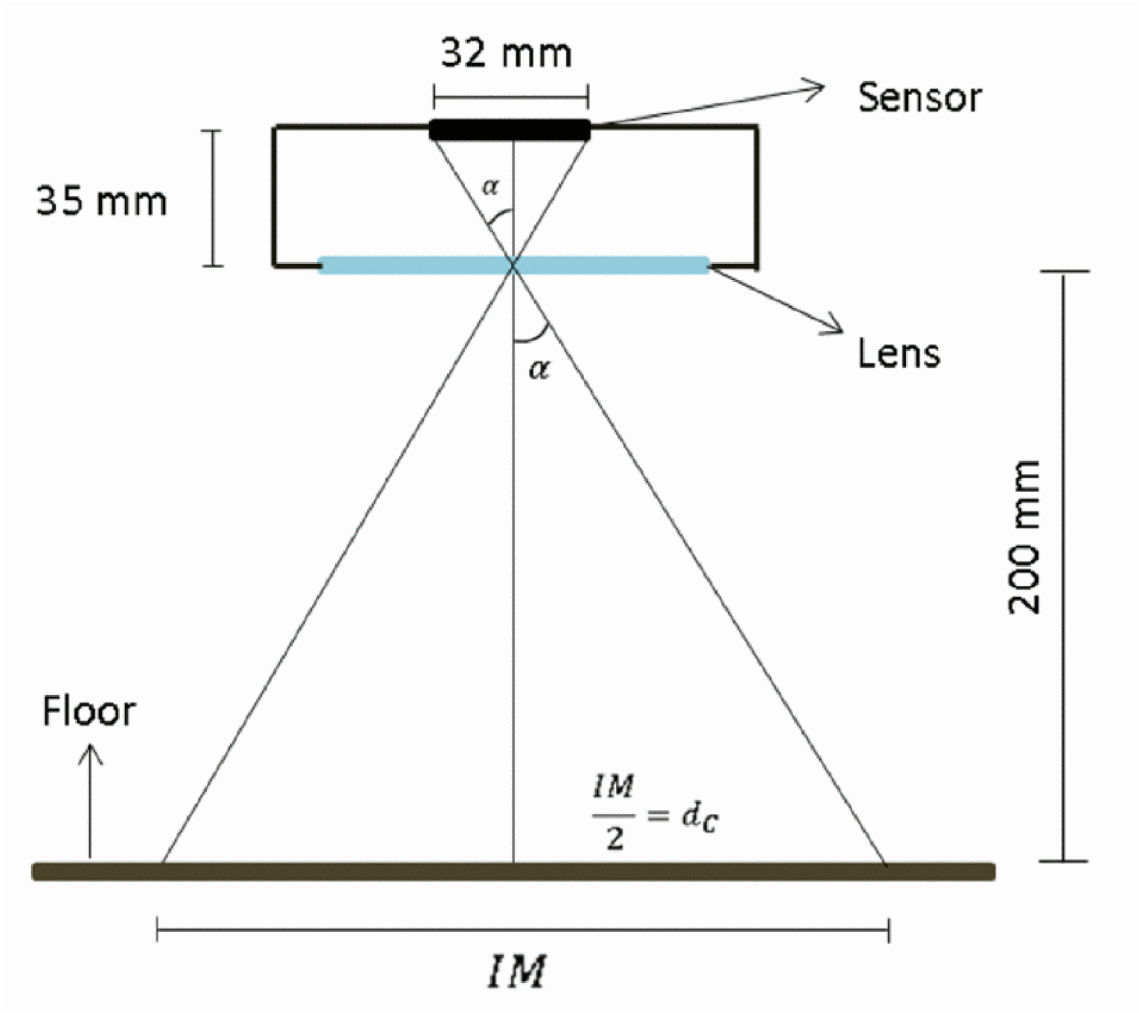


Fig. 2.3: Dimensions of the camera set [2]

Now we can calculate a horizontal width of one pixel d_P as $\frac{IM}{N_H}$; N_H is horizontal quantity of pixels. Therefore b is equal:

$$b(t) = N_p d_p \tag{2.4}$$

, and parameter a which is constant is equal:

$$a = d_C + \frac{d_p}{2} \tag{2.5}$$

, where d_C and d_p are constant, see Figure 2.2.

The final result, angular deviation, can be obtained from (2.2) and than it can be used as feedback for designed controller.

3 DIGITAL IMAGE PROCESSING

Digital image processing means the use of computer power and algorithms in order to perform image processing on digital images. From its essence it allows to use much wider range of algorithms to be applied to the input data. The other advantage is the absence of problems such as the build-up of noise and signal distortion during processing. As one of fields of digital signal processing, digital image processing has many advantages over analogue image processing. While images are usually defined over two and more dimensions, digital image processing should be modelled in the form of multidimensional systems. The effort is to reach the biggest versatility. [11]

3.1 History

Before talking about digital image processing, it is convenient to make a point on the history of digital images.

It could be said that the newspaper industry was a pioneer in one of the firsts applications of digital images. In the early 1920s, when newspaper images were sent between London and New York by submarine cable, the introduction of the Bartlane cable picture transmission system was a leap forward for this industry. In fact, it decreased the time to transport pictures across the Atlantic from more than a week to less than three hours. At that time, specialized printing equipment coded the images for cable transmission and then reconstructed them. While at the beginning the images were coded with 5 gray levels, in the next nine years this capability was already increased to 15 levels. The field of digital images was taking off.

Despite these cases are directly related with digital images, they are not considered as consequences of digital image processing, because of computers were not implicated in their creation. In this manner, the history of digital image processing is intimately fixed to the development of the digital computer.

After the development of the computer industry, we can approximate the date of the birth of digital image processing to the early 1960s. Exactly, in 1964 at the Jet Propulsion Laboratory (Pasadena, California), when the pictures of the moon needed to be processed to be watched in the television, it began to work on using computer techniques for improving images, but because of fairly high computational cost of processing its progress was not very fast.

In parallel with these space applications, digital image processing techniques began to be used in medical imaging in the late 1960s and early 1970s, highlighting

the invention of the computerized axial tomography (CAT), also called computerized tomography (CT), and the discovery of the X-rays in 1895. Over the years, the field of image processing has grown vigorously. Big number of the techniques of digital image (digital picture) processing have been developed with application to satellite imagery, wire-photo standards conversion, videophone, character recognition and photograph enhancement among others.

Now, in addition to applications in medicine and the space program, digital image processing techniques can be used in a wide variety of applications. Today could be images processed in real time, for some dedicated problems such as television standards conversion. As computers intended to general were been becoming faster, they started to take over the role of dedicated hardware for almost all operations. Digital image processing has become the most common form of image processing with the arrival of fast computers and signal processors in the 2000s. [4][12]

3.2 Tasks

Digital image processing opens up new possibilities. Among others it allows the use of much more complex algorithms, so it can offer both more sophisticated performance at simple tasks, and the implementation of methods which would be impossible by analog methods. There are several problems for which we do not know another solution than using digital image processing:

- Feature extraction and pattern recognition
- Multi-scale signal analysis
- Projection
- Classification

Some techniques which are used in digital image processing include:

- Pixelation
- Anisotropic diffusion
- Linear filtering
- Hidden Markov models
- Image editing and restoration
- Independent component analysis
- Neural networks
- Partial differential equations
- Principal components analysis
- Self-organizing maps
- Wavelets

3.3 Image Enhancement

The principal objective of enhancement is to process an image modifying its attributes with the objective of getting as result an image more appropriated than the original for a specific application. It's important to point out the concept of specific application, because depending on this fact, it is wise to use different methods or specific techniques. Unfortunately, there is not exits a general method that satisfy all our wants, so it is obvious that the methods for example used for detecting breast cancer are not the same than the methods used for enhancing pictures of Mars. It is difficult to give a generic method that works well on all pictures.

The field of image enhancement approaches fall into two categories: spatial domain methods and frequency domain methods. [4]

3.3.1 Image Enhancement in the spatial domain

The term spatial domain refers to the image plane itself. The techniques used in this domain are based on direct manipulation of the pixels of the image. The processes can be denoted by the expression:

$$g(x, y) = T[f(x, y)] \tag{3.1}$$

where f(x, y) is the input image, g(x, y) is the processed image, and T is an operator on f, defined over some neighborhood of (x, y). [4]

3.3.2 Image Enhancement in the frequency domain

Unlike Spatial domain, based on the direct manipulation of the pixels, Frequency domain processing techniques are based on modifying the Fourier transform of an image. All the operations are realized over the Fourier Transform of the original image and then the Inverse of this transformation is performed to get the result.

The Fourier transform, F(u), of a single variable, continuous function, f(x), is defined by the equation:

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux}\, dx \tag{3.2}$$

where j = $\sqrt{-1}$. Conversely, given F(u), we can obtain f(x) by means of the inverse Fourier transform:

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{-j2\pi ux}\, du \tag{3.3}$$

These two equations comprise the Fourier transform pair and indicate the important fact that a function can be recovered from its transform. Starting from them, it is easy to obtain the same equations for two variables.

The Fourier transform of two variables, continuous function:

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-j2\pi(ux+vy)}\,dxdy \tag{3.4}$$

and its inverse:

$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{-j2\pi(ux+vy)}\,dudv \tag{3.5}$$

Also, for discrete functions, we can obtain the Fourier transform of two variables:

$$F(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(x)e^{-j2\pi ux/M}\, foru = 0,1,2,...,M-1 \tag{3.6}$$

and its inverse:

$$f(x) = \frac{1}{M}\sum_{x=0}^{M-1} F(u)e^{-j2\pi ux/M}\, foru = 0,1,2,...,M-1 \tag{3.7}$$

Knowing these equations and their properties it is possible to work on the frequency domain. [4]

3.4 Main techniques/methods

Currently, there exists a big set of techniques or methods that can be used for image enhancement. Most of them are usually combined to obtain different algorithms for process the image.

3.4.1 Filtering

One of the biggest fields of digital image transformations is called filtering. That means the use of digital filters to sharpen and blur digital images. There are several ways to perform filtering. The most widespread are performed in the spatial domain by convolution with specifically designed kernels (filter array), or in the frequency (Fourier) domain by masking specific frequency regions [4]. The following examples shows some of them:

- Spatial Lowpass
- Spatial Highpass
- Fourier Representation

- Fourier Lowpass
- Fourier Highpass

The simple example of the convolution matrix use is on Figure 3.1. For more

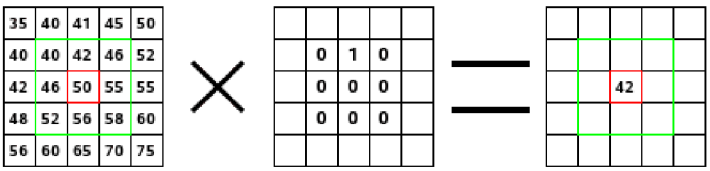


Fig. 3.1: Convolution matrix use example [3]

examples and further explanation see [3].

3.4.2 Gray-level Transformation

Continuing with the study of all of these techniques, we talk about gray-level transformation functions. It could be said that these are among the simplest of all image enhancement techniques.

In any of its varieties, a gray-level transformation can be denoted by the expression:

$$s = T(r) \tag{3.8}$$

where T is a transformation that maps a pixel value r into a pixel value s. So, the values of pixels before processing are denoted by r and the values of pixels after processing are denoted by s.

Depending on this operator T, it can be different type of gray-level transformations. The three basic types of functions used frequently are linear (negative and identity transformations), logarithmic (log and inverse-log transformations) and power-law (nth power and nth root transformations).

Apart from these typical functions, it must be empathized the technique known as contrast stretching. This technique is based on darkening the levels below some level m and heightening the levels above it. Firsts values of the input image are compressed by the transformation function into a narrow range of s, toward black, and the opposite effect takes place for values above m. The Figure 3.2 shows one possible form of a transformation function. The level m will depend on the circumstances of the particular image. [4]

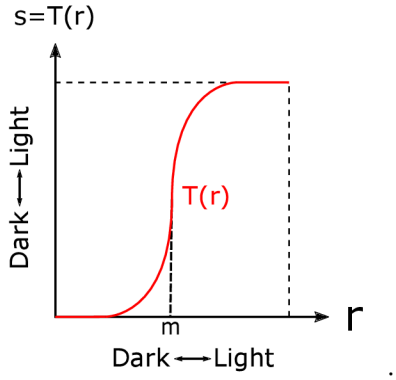


Fig. 3.2: Form of transformation function of contrast stretching [4]

The result of the contrast stretching will be an image with higher contrast than the original one, its histogram will be spread. The Figure 3.3 is an example of a real image that shows it. The histogram of the input image ranges from 79 to 136, however, the histogram of the output one is between 0 (black) and 255 (white).

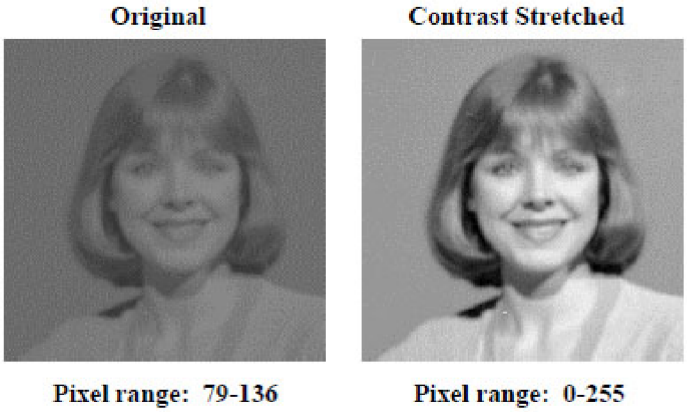


Fig. 3.3: Example of contrast stretching [5]

3.4.3 Gray-level discontinuities

In a digital image, we may find three types of gray-level discontinuities: points, lines, and edges.

In this case we only focus on edge detections. Although the objective in this project of processing of image is to find the line, edge detection is by far the most common approach for detecting meaningful discontinuities in gray level, so, despite changes in the lighting conditions or changes in the illumination angle, obtaining

the different discontinuities of the picture we can binarize the picture and detect the track.

There is a wide variety of mathematical methods that has the ability to identify brightness changes sharply. One of the most used is the Canny Edge Detector. It is an edge detection operator that using a multi-stage algorithm can detect an extensive range of edges in images. The operating process of this algorithm can be divided into 5 steps:

1. Filter the image with a Gaussian filter to smooth the image with the purpose to reduce the noise and reduce detail.
2. Find the intensity gradients of the image (directional change in the intensity).
3. Apply non-maximum suppression to dispose of spurious response (discarding the variables that are not causally related to each other).
4. Apply double threshold to determine potential edges.
5. Track edge by hysteresis: Suppressing all the other edges that are weak and not connected to strong edges.

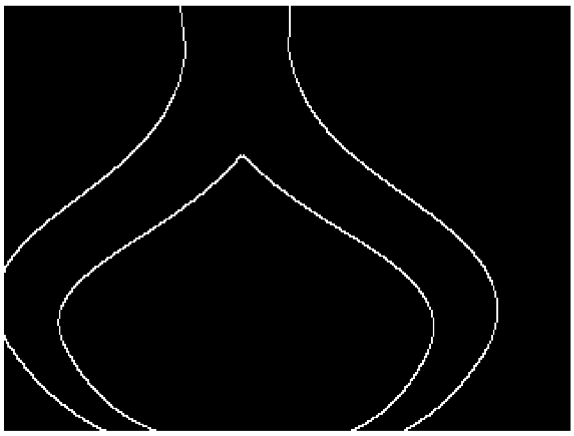
Besides, during the process there are a number of variable parameters which can be modified. These can affect the computation time and effectiveness of the algorithm, so, depending on the objective and their conditions shall be adjustable. These parameters are:

- The size of the Gaussian filter: The use of small filters produce less blurring on the result and permit the detection of small lines. A larger filter causes more blurring, spreading the value of a given pixel over a larger area of the picture.
- Thresholds: The application of a threshold too high can discard relevant information of the image. Conversely, a threshold set too low could wrongly associate irrelevant information as important.[4][13]

The Figure 3.4 is an example of the application of the Canny Edge Detector in our algorithm. On one side, the Figure 3.4a shows an original image of the track obtained directly from the camera, on the other side, the Figure 3.4b shows the result of the application of this method. Parameters used for this example:

- *Size of the kernel of the Gaussian filter: 3*
- *First threshold for the hysteresis procedure: 50*
- *Second threshold for the hysteresis procedure: 205*
- *Size of the Sobel kernel to be used internally: 3*

(a) Original image



(b) Result of the Canny Edge Detector

Fig. 3.4: Application of the Canny Edge Detector

3.4.4 Otsu's method

The Otsu's method is one of the most used methods in segmentation. This global algorithm chooses automatically the threshold level with the purpose of minimize the intraclass variance of black and with pixels or even to reduce a gray level to a binary image. Exactly, what it does is work with the histogram of the images assuming that the image contains two classes of pixels, thus selecting the optimum threshold level to separate them.

The most notable problem is that it is very sensitive to variations of brightness so depending on the situation it is not always advisable to use it. Besides, as the classes of the image increase, it takes more time.

The Figure 3.5 shows an example of the Application of the Otsu's algorithm. [6]

(a) Original image

(b) Image thresholded using Otsu's algorithm

Fig. 3.5: Application of the Otsu's algorithm [6]

4 USED SOFTWARE AND HARDWARE

4.1 Chassis

The Texas Instruments Robotics Systems Learning Kit (TI-RSLK) is a low-cost robotics kit which provide students with a deeper understanding of how electronic system designs work. Developed in collaboration with Dr. Jon Valvano, professor, electrical and computer engineering at The University of Texas at Austin.

The first in its series is the Maze Edition which comes with more than twenty learning modules covering basic to lot of advanced topics. Texas instruments created this robotics learning kit to fill a gap in existing engineering curriculum by introducing comprehensive course-ware that addresses the fundamentals of embedded systems and progresses to more advanced applications learning. [7]

The full planted chassis with all electronic from TI-RSLK advanced kit is in Figure 4.1.

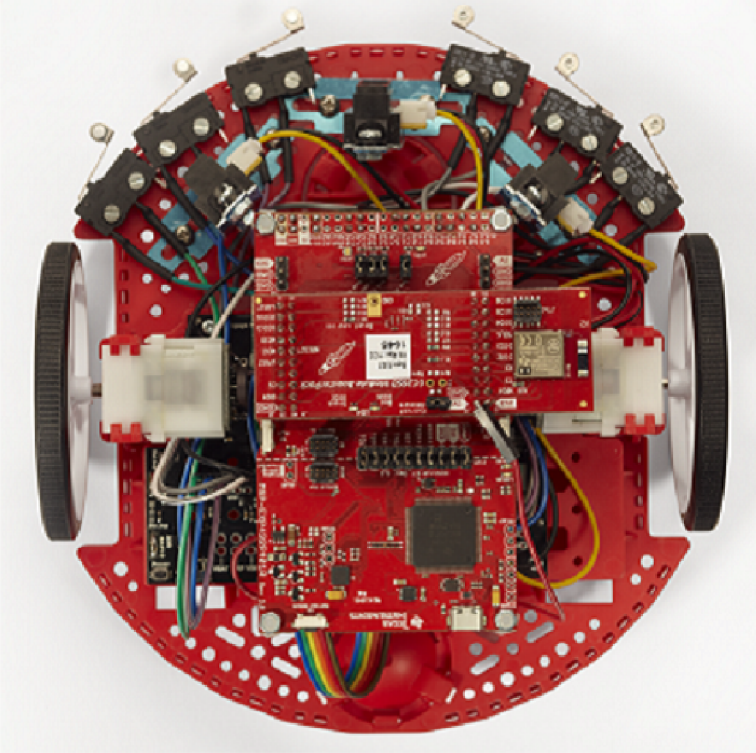


Fig. 4.1: TI RSLK advanced KIT [7]

For my purpose I will use only chassis with motors and theirs driver-board rom02a from Pololu.

4.2 DriverBoard

Rom02a board is motor driver and power distribution board for romi chassis Figure 4.2 designed by Pololu especially for chassis I am using. Some of features it offers are battery contact slots, several power switching options, reverse voltage protection and easy access to the various power buses. There is also two-channel motor driver and powerful switching step-down regulator (MP4423H) that can supply a continuous 2.5 A at 5 V or 3.3 V. I am using default pre-set 5 V regulator to power Raspberry pi. This kit is also designed to allow use Romi Encoder Pair Kit which can be used to track the rotational speed and direction of the robot's drive wheels, but I have decided not tu use them, because camera should be enough feedback for me.

Exact type of motor driver is DRV8838 from Texas Instruments and it offers a simple two-pin PHASE/ENABLE control interface which makes available DIR and PWM control for each motor. The DIR pin controls the motor direction (low = forward) and the PWM pin controls motor speed through PWM signal. The DIR and PWM control pins are pulled low through internal pull-down resistors (app. 100 kohm). Dynamic braking while PWM pin is low is secured with shorting motor outputs to ground. There are also one SLEEP pin for each motor (by default connected together) which can be driven to low to get drivers into a low-power sleep mode and turn off the motor outputs. [8]

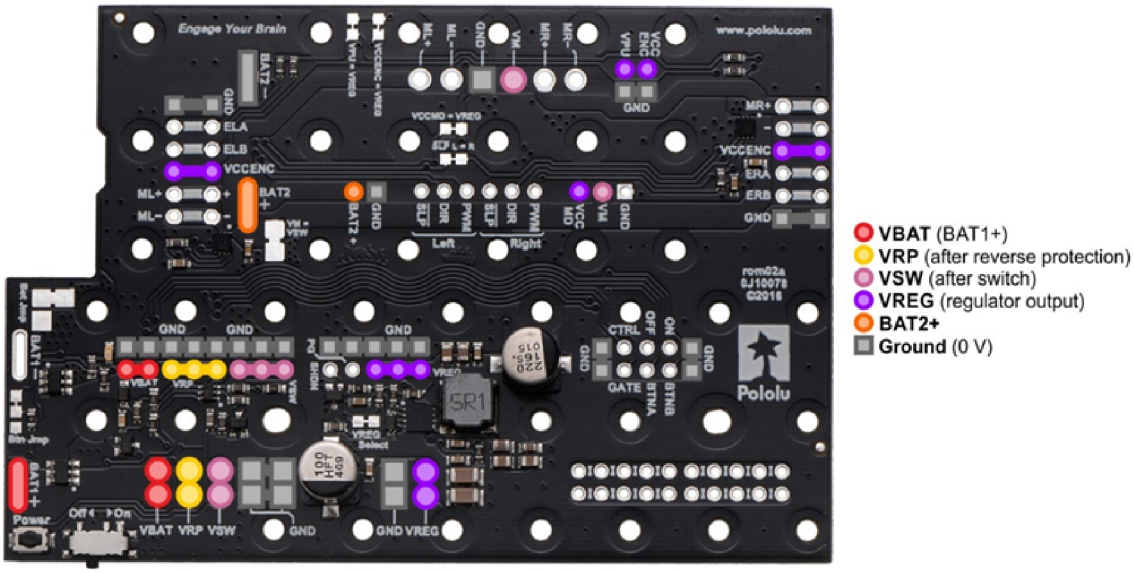


Fig. 4.2: Rom02a layout of the power distribution buses and access points on the board [8]

4.3 Raspberry pi 3

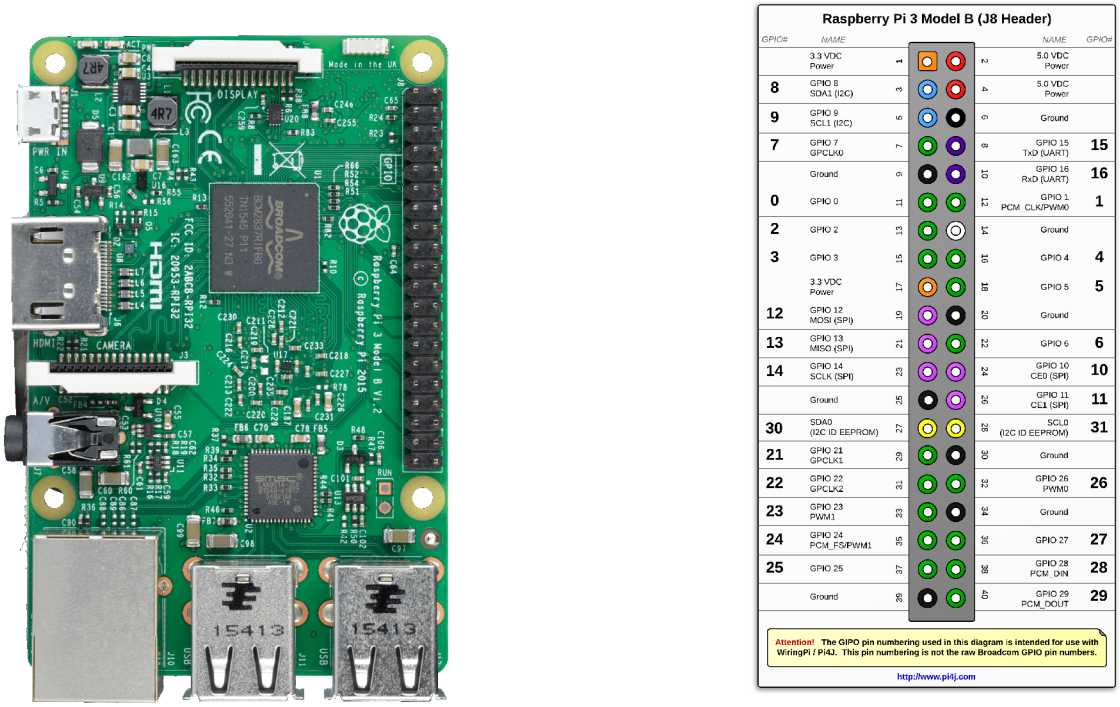
It is a small (85mm x 56mm) single-board computer (Figure 4.3a) developed to promote the teaching of basic computer science. Exact type of raspberry I am using is Raspberry pi 3 model B and it is the newest generation in time of my study. Most relevant specifications are:

- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU - it is almost twice more powerful than the second generation, what is really important for me: image is processing is computationally demanding and for purpose of real time controlling it should be very fast
- 1GB RAM
- BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board - I have created a hotspot on my raspberry so I can be directly connected to it through wi-fi without another equipment. It is practical for sending code to it and also for logging real-time data while improving controllers
- 40-pin extended GPIO Figure 4.3b - two of them are able provide physical PWM (BCM 13 and BCM 18) so I have decided to use raspberry to directly control motor drivers and not to use real-time micro-controller between computing and power part as it is usual. On the other hand, raspberry pi is not a real-time system. If it will not fast enough I will consider real-time microcontoller use (e.g arduino or atmega)
- 4 USB 2 ports - I used them at the beginning to connect mouse and keyboard in order to setup and enable SSH
- 4 Pole stereo output and composite video port
- Full size HDMI - As USB ports, I used it first time
- CSI camera port for connecting a Raspberry Pi camera - connecting camera directly to processor is the fastest way for getting images so it is better solution than using web-camera and USB port
- DSI display port for connecting a Raspberry Pi touch-screen display
- Micro SD port for loading your operating system and storing data - all data including operating system are on 8 gb micro SD card
- Upgraded switched Micro USB power source up to 2.5A

[9]

First I wanted to use BeagleBone board developed by Texas Instruments which is based on very similar idea as raspberry, but then I changed my mind for several reasons. The less essential are my previous experience with raspberry compared to beaglebone about that I heard first time now and that there is wider support for raspberry as it is more widespread. The main reason is, that raspberry is more

powerful, it has direct connection for camera and is more up-to-date. My raspberry pi 3 model B version is from 2017 but newest beaglebone Black is from 2013.



(a) Looks (b) Pinout

Fig. 4.3: Raspberry pi 3 model B [9]

4.3.1 Operating system

I am using Raspbian stretch lite as operating system, with newest kernel version 4.9. Raspbian is debian-based computer OS ans it is the officially supported by Rpi Foundation's. There is possibility to install it with NOOBS or directly download the image from their WEB-site. I have used the second one. Raspbian comes pre-installed with plenty of software for education, programming and general use, but I needed to install some other. There is their own subsections below for some of them. Important note is that Raspbian is not a real-time operating system, what can cause problems in my case.

4.3.2 Process priority

As mentioned above, I am working with no real-time operating system. sys/resource.h library - setPriority();

4.3.3 Hotspot setting

Because I want to work with raspberry using SSH and also because I am going to develop code remotely I need to create LAN net between my computer an raspberry. Doing that with cable is, as I see it, restrictive so I have decided to create wireless LAN. Using wi-fi router would not be the best choice also, because I would need to have router everywhere I want to work with raspberry. The best choice I have decided for is to create access point directly on raspberry, so I can connect to it any time. I had to install DNS mask, what is something between DNS and DHCP server, on it to work properly and I have created bridge between Ethernet port and wireless module on raspberry for the purpose of forwarding the internet.

Firstly I have used RaspAP application to configure access point, but there were som problems with adjusting settings so I have decided to set it in my own changing configuration files. The necessary changes are:

- **IP adress configuration:** in /etc/dhcpcd.conf I set a new static profile and its use in case of fall-back in the end of file. That happens when DHCP server does not give me an IP address, because that is, how it works here in ESIEE school. I need to set a special static IP address in order to connect to internet.
- **Access point configuration:** HostAPD (Host access point daemon) software allows me to connect to Raspberry from other devices like my laptop. Raspberry works like router with DHCP server and it is also forwarding internet. All settings are set in /etc/hostapd/hostapd.conf. There is also possibility to change ssid or password.
- **Select default daemon configuration:** in /etc/default/hostapd I had set Packet forwarding etc/sysctl.d/30-ipforward.conf. There was also need to set firewall in /etc/iptables.ipv4.nat because internet network on univesity requires it. Firewall settings are saved in /esiee_proxy.sh.

4.3.4 Samba share

Samba is a free software, originally developed by Andrew Tridgell, of the SMB/CIFS networking protocol which provides file and print services for various Microsoft Windows clients and can integrate with a Microsoft Windows Server domain. Samba runs on most Unix, OpenVMS and Unix-like systems, such as Linux, Solaris, AIX and the BSD variants and it is standard on nearly all distributions of Linux and is commonly included as a basic system service on other Unix-based operating systems as well. Samba is released under the terms of the GNU General Public License.

After I made required configuration in samba.conf file I can map memory of raspberry to my laptop as external disc through wi-fi net. All source codes with

netbeans project and other data are stored in raspberry's memory (and also in git) and I am working with them remotely. In this way I can see saved images without need of copying them. [14]

4.3.5 X Window System

Also X11 or only X is name of software which is used for creating of graphical user interface (GUI). Most often is it used in unix systems where it is almost standard. The base model it is using is client-server and it consists of several independent components like X server, X protocol, Xlib library etc. This X protocol, transferred via IPC or TCP/IP, provides feature that running application can be displayed on another computer than on which it was launched. I am using this opportunity for seeing (almost immediately) images taken and processed by robot on my laptop using software called Xming section 4.7. It is very useful for checking correctness of my algorithms by sight. [15]

4.4 OpenCV

OpenCV (Open Source Computer Vision Library) is an open source software released under a BSD license and hence it is free for both academic and commercial use. All around the world there are more than forty-seven thousand people of user community and estimated number of downloads exceeding fourteen million. Usage ranges from interactive art, to mines inspection, stitching maps on the web or through advanced robotics. It has C++, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android, hence there is wide spectrum where it can be used. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. While raspberry pi 3 has quad-core processor I will try to utilize this opportunity. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform. Current newest version I am using is 3.4.1.

Officially is openCV released compiled only with Visual Studio compiler for windows. While I am using MSYS2 MinGW-w64 compilers on my laptop and standard gcc on raspberry I had to compile it on my own. I have used CMake [16] software in order to ease the compilation. After successful installation have I needed to configure NetBeans to work properly with this library. Important information is that openCV is now divided into more smaller libraries, not only one like before, hence I need to include all parts I am using.

In the same way, there is not official compiled distribution for raspberry platform, so I have downloaded openCV from [17] and than, using cmake and standard commands 'make -j 4' and 'install', compiled and installed it. To compile code using openCV correctly there has to be additional parameter ''pkg-config opencv –cflags –libs'' while calling compiler. [18]

4.4.1 WiringPi

Open source library wiringPi released under the GNU LGPLv3 license is intended to PIN based GPIO access. It is written in C for the BCM2835, BCM2836 and BCM2837 SoC devices that are used in all Raspberry Pi versions and it is usable from C, C++ and RTB (BASIC) as well as many other languages with suitable wrappers. Its design is familiar Arduino 'wiring' system1, but it is also intended for use by experienced C/C++ programmers.

WiringPi includes a command-line utility gpio which can be used to control the GPIO pins. This feature is suitable for reading and writing the pins from shell scripts. While there is no native analogue hardware on a Pi by default, modules are provided to support the Gertboards analogue chips and other A/D and D/A devices can be implemented relatively easily.

Firstly I had problem with manipulating pins because of rights, which are in Linux policy very important. While I was logged in as normal user 'pi' I did not have rights to control physical interface. Gpio utility worked good, but when I tried to run my own script raspberry became frozen and I needed to do hard restart. Than I find out that it is working while running as root using sudo command. Although in documentation is that newest WiringPi should not have this problem I did not find other way than running as root. That is the reason why I have set Netbeans to log in raspberry as root, however I am realising that it is not very good solution. [19]

4.5 Netbeans

Netbeans is an open source project developed under CDDL licence. Netbeans IDE is an integrated development environment for JAVA, there are extensions for other programming languages like PHP, C, C++ and HTML5. It allows applications to be developed from a set of modular software components called modules what could be useful for my intentions. While it is written in Java it can be transferred to run on Windows, Linux, Mac OS X, Solaris, OS/2 and more. An IDE is more than only a text editor: it can indents lines, matches words and brackets, and highlights source

code syntactically and semantically. It lets you easily refactor code, with a range of handy and powerful tools(I appreciate rename tool - to rewrite ale istancies of some variable), while it also provides code templates, coding tips, and code generators.

The main reason why I decided to use this one is one more time my previous experience, open source character and possibility to develop C/C++ project remotely. Writing my code directly on raspberry would be restrictive because I would need external display, mouse and keyboard and what is worse, there are no so handy text editors. This way I can test my application on raspbian system without even leaving the IDE. Once I have set up the remote build host I even do not see any difference in work-flow between doing local and remote development. Another advantage is, that I can develop algorithm for image processing locally, what is faster than on raspberry, and then just simply copy it. [20]

4.5.1 MSYS2

MSYS2 is a software distribution and building platform for Windows which provides a bash shell, auto-tools, revision control systems and the like for building native Windows applications using MinGW-w64 tool-chains. Its core is an independent rewrite of original MSYS, based on modern Cygwin (POSIX compatibility layer) and MinGW-w64 with the aim of better interoperability with native Windows software. It features a Pacman package management system to provide easy installation of packages. It brings many powerful features such as dependency resolution and simple complete system upgrades, as well as straight-forward package building. The MinGW-w64 gcc and gpp compilers are exactly the reason why I have used it. With older MinGW compilers I had problem while compiling openCV. [21]

4.6 Matlab

This software combines a interactive desktop environment tuned for iterative analysis and design processes with a programming language that expresses matrix and array mathematics directly. Its biggest advantage is great number of external tools and supplements which can be used to solve as very easy tasks in easy way so also very complex and difficult problems. The tool I am using is called Simulink. It is a block diagram environment for multi-domain simulation and Model-Based Design. This is exactly what I need, because I want to create mathematical model of my robot and then create controller using model-based design technique. Simulink provides a graphical editor, wide customizable block libraries, and solvers for modelling and simulating dynamic systems. It also supports many more functions, like system-level design, simulation, automatic code generation, and continuous test and

verification of embedded systems, which will not be very useful for me. What is very useful is that it is integrated with matlab, enabling to incorporate matlab algorithms into models and export simulation results to MATLAB for further analysis. That means that I can provide parameters to simulation from matlab file and, in similar way, save results from simulation to workspace, process them or just plot them. I will use this feature while designing controller using ITAE criteria.

There are many third part tools like *judp*, which is free in contrast with original matlab supplements. It is a small piece of code, which allows communication trough UDP connection. I am using it for sending some data, such deviations and taken time, to matlab and the process and plot them.

4.7 Xming

Xming is an open-source(GNU-GPL) port of X server for Microsoft Windows OS. The main difference with Cygwin is, that Xming does not provide full-fledged Linux environment. It allows to access to XDMCP (X display manager is system process in graphical system X window which allows user to log in from local computer or trough computer network) session running on remote computer with another X server and start, see and manipulate with remote X application using tunnelling of SSH protocol. This is possible with PuTTy, but there is need to allow X11 forwarding. In very similar way, this feature is provided by NetBeans when developed code is running on remote machine. I am using it for purpose of controlling my image processing and deviation obtaining algorithms. After I process image I use an openCV function $cv :: imshow()$ which opens my image in, only connected, remote display provided by Xming. In this way I can see what my robot see and it is very fast, what is important for me, in order to keep the sample period as short as possible.

5 MATHEMATICS OF DIFFERENTIAL DRIVE

5.1 Linear model of my robot

Firstly I want to try to create liner model of my robot, because in that case I could use many methods which were developed for designing controllers for linear systems. Because differential drive is non-linear system also its mathematical description is non-linear. There is way how to linearise non-linear system, but such simplified model is correct only around working point which could be insufficient for my purpose. While I can use powerful computers and professional software, firstly I will try to design controller on non-linear system using model-based design.

5.2 Differential drive kinematics

Differential drive is one of the most used drive for mobile robots. The basic idea consists of two independently driven wheels on one common axis. While we can adjust velocity of each wheel in both directions (forward and backward), the robot has to rotate about the ICC (Instantaneous Centre of Curvature) point, which has to lie along centre common axis of both wheels. Than I can write following equations (5.1), which determine velocities of individual wheels from known angular speed ω and wheelbase l, which is distance between centres of wheels. R is distance from middle-point between the wheels to ICC point and V_r and V_l are velocities of right and left wheel. [22]

$$\begin{aligned} V_r &= \omega \left(R + \frac{l}{2} \right) \\ V_l &= \omega \left(R - \frac{l}{2} \right) \end{aligned} \tag{5.1}$$

From these velocities is possible to determine R (5.2) and ω (5.3) in any time:

$$R = \frac{l}{2} \frac{V_r + V_l}{V_r - V_l}; \tag{5.2}$$

$$\omega = \frac{V_r - V_l}{l} \tag{5.3}$$

For better understanding of these equations see the Figure 5.1.

With this kind of robot drive there are three interesting cases:

- $V_l = V_r$ -> straight forward motion; R = ∞ and $\omega = 0$
- $V_l = -V_r$ -> rotation about the midpoint of the wheel axis; R = 0 => rotating in place
- $V_l = 0$ or $V_r = 0$ -> rotation about one of the wheels; R = $\frac{l}{2}$ for this case

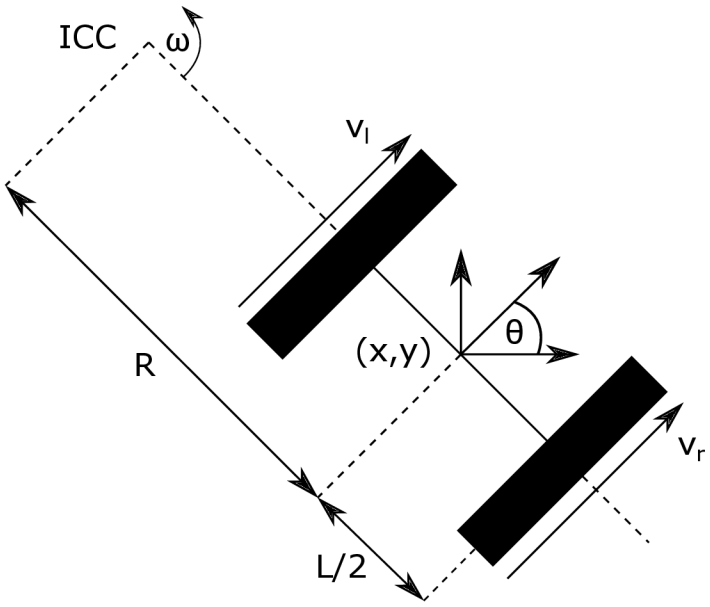


Fig. 5.1: Kinematic of differential drive [1]

5.3 Forward kinematics for differential drive robot

Now we can assume, that robot is at position (x, y) making angle θ with X axis. Now we can get robot to move to new position and orientation by manipulating the control parameters V_l, V_r.

Using equation (5.2) we can from known velocities find current ICC position (5.4):

$$ICC = [x - R\sin(\theta), y + R\cos(\theta)] \tag{5.4}$$

and determine the robots position in time $t + \delta_t$ (5.5):

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos(\omega\delta_t) & -\sin(\omega\delta_t) & 0 \\ \sin(\omega\delta_t) & \cos(\omega\delta_t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - ICC_x \\ y - ICC_y \\ \theta \end{bmatrix} + \begin{bmatrix} ICC_x \\ ICC_y \\ \omega\delta_t \end{bmatrix} \tag{5.5}$$

This equation is simply describing the motion of a robot rotating with an angular velocity of ω in a distance R about its ICC.

There are two special cases, where previous equations could be unnecessary or even not applicable. First of them is case, when is truth, that $v_l = v_r = v$, both wheels have the same velocity what cause, that robot is moving in a straight line. In a such case it is obvious from (5.2), that R value become equal to infinity(there is division by zero), which means, that ICC point is also in infinity distance. That is

real problem, because it cause my solution get into singularity point. The adjusted motion equations are:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x + v\cos(\theta)\delta_t) \\ y + v\sin(\theta)\delta_t) \\ \theta \end{bmatrix} \tag{5.6}$$

The second special become, when $v_r = -v_l$, the absolute values of both velocities are equal, but they have different sign. In this case, there is a denominator equal to zero in (5.2), the ICC point is exactly in the middle of wheel axis. In the result robot rotates in place and the equations become:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta + 2v\delta_t/l \end{bmatrix} \tag{5.7}$$

[23]

5.4 Inverse kinematic of differential drive robot

As I explained, the forward kinematics objective is to provide new position from given wheel speeds (or encoder counts). Obviously, then we can also think to an inverse problem: Robot is on position (x, y, θ) at time t. Determine control parameters v_l and v_r such that the new position at time $t' = (t + \delta_t)$ is (x', y', θ').

There is one unpleasant fact in way of finding solution. Differential drive belongs to the group of non-holomic systems(as the most of vehicles and robots). That means certain limitation in its movement. For example, robot cannot move directly along axis of its wheels. Car is also a non-holonomic and that is why pocket parking is so hard. The result is, that I cannot simply specify an arbitrary robot pose (x, y, θ) and find the velocities that will get us there.

5.5 Continuous model

Although previous model is correct, after I modelled it in Matlab I realised, that it is discrete. Second problem is, that I need equation in ODE (Ordinary differential equation) shape, what means, that on left side are derivations of state variables and on right side their function. The equations I found (5.8) are much more simple than previous description. [24]

$$\begin{aligned} \dot{x}(t) &= v(t)cos(\theta(t)) \\ \dot{y}(t) &= v(t)sin(\theta(t)) \\ \dot{\theta}(t) &= w(t) \end{aligned} \tag{5.8}$$

5.6 Constants of my robot

Specific values for my robot are:

- Wheelbase, distance between centres of wheels $= 0,14m$
- Diameter of wheel $= 0,07m$
- Maximal PWM value is equal to 1023, because there is ten bit resolution. The insensitivity starts at PWM $= 70$.
- Maximal speed of wheel $= 0,725m/s$
- Maximal angular speed $= 9,0625rad/s$

5.7 Motors characteristic

I am using linear motors to drive my robot. Linear motors have very unstable characteristic: it is always more-less linear, but it can move up and down depending on its load. How great is load affecting motors can depend on wheel size, robot weight, acceleration and friction. Also state of accumulators can influence characteristic of motors which powers. This is a big contrast with stepper motors where one step corresponds to a specific angle and in reasonable range of load it does not change.

In order to provide the best conditions for controllers I measured characteristic of used motors. I was doing this with fully assembled robot in order to obtain useful information. Changing PWM values for both motors in step of size 100 I measured the distance driven in particular time. From this values is very easy to calculate speed and plot Figure 5.2 the PWM dependence on required speed. I watched that when I set same PWM for both motors it is not going straight, but it is turning, so I decided to measure one characteristic for each motor. Although, after several measurements it starts to turn to opposite side. That means, that motor's inaccuracy is so big, that there is no sense to make two characteristics. I used least square method (in excel) and find the polynomial function of second degree that describes my characteristic.

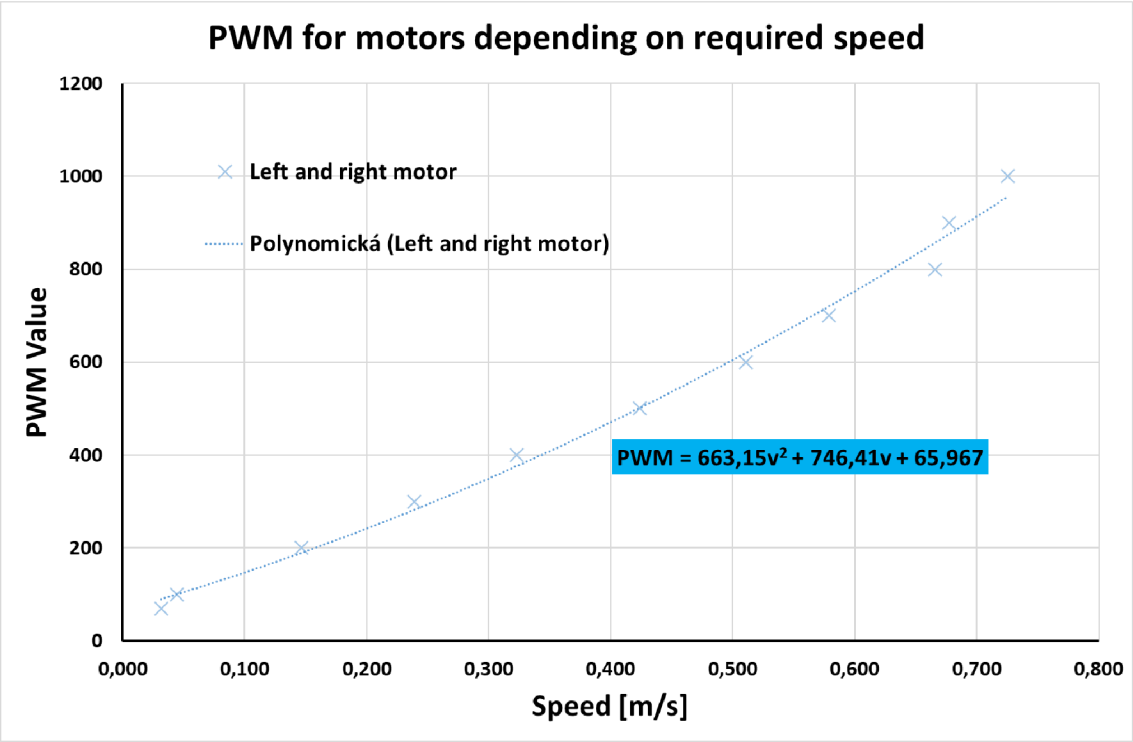


Fig. 5.2: PWM on forward velocity dependence for used linear motors

6 TRACK DESIGNING

One of the most important part while designing something is testing. If tests are not detailed and strict enough, there is no certainty that machine or whatever will works right. Time invested into making tests is not wasted. In my case I have to design a testing track. It will consists from white ground (paper) and black line made from black electrical tape wide 17 mm. There could be also some obstacles, what would be very advanced task.

Testing track has to contain:

- **Straight route** - velocity controller testing. When there is a relatively straight line before the robot, it should accelerate in order to reach the best time.
- **Curve** - this track feature will tests the angular velocity controller. Robot has to be able to compensate any deviation between its centre and line centre - that is its basic purpose.
- **Sharp curve** - also for velocity controller testing. In similar way, when the track before the robot is winding, robot has to slow down to reach necessary reaction time.
- **Crossing** - line crossing is very often seen feature of testing tracks and it is not so difficult to treat it, because we know, that there are no right angles on track. Robot should continue following its own line.
- **Split** - also very frequent feature. Usually, there are given rules in advance, what robot should does in a such case. If it should follow left or right line, or there are more difficult conditions.
- **Interruption** - when there is an interruption in line, robot should be able to find line one more time. Elementary, this could be implemented just as it will continue driving directly in its current direction. However, with camera, there can be implemented more sophisticated algorithms for line re-find
- **Interruption in curve** - very similar to previous feature. The only difference is, that after line lost robot will not continue in direct motion, but really in its current motion. That means, there has to be buffer that will contains several previous values of its speed, so it can truly continue in its previous motion.
- **Two lines near each other** - more complex test. When there are two lines very near each other, it is easy to mistaken the actual followed by the second one.
- **Light changing** - while using camera, lighting conditions are very important. There is needed sufficient external lighting, because camera does not have its own source of light, but it can usually vary. That is the reason, why there has to be implemented quality and robust feature, that will provide optimal

threshold level for each image. Second huge problem are shadows, but that is out of scope of my thesis.

For the track background I am using nine white hard papers conglomerate together by duct tape because I did not obtain large enough paper in one piece without pattern. Only problem which can rise is, that paper joins are imperfect what can cause difficulties in image processing. The route itself I painted with black colour and its borders are made with fix. The specific track I designed with all its features and subsequently tested my robot on it is in Figure 6.1.

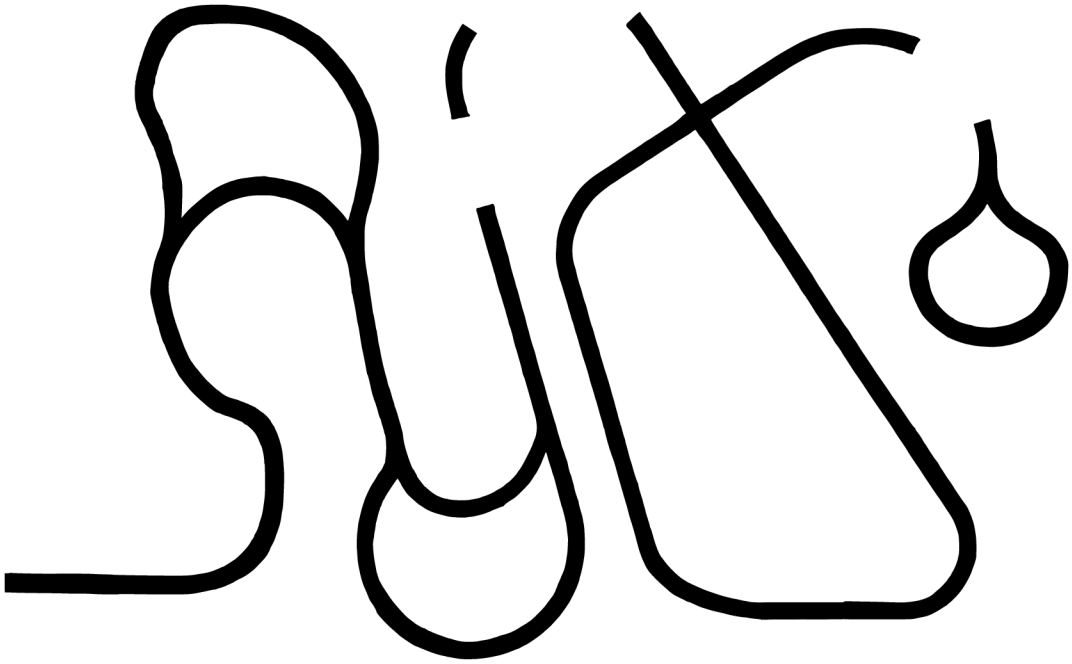


Fig. 6.1: Designed robot testing track

7 MODEL

Considering the relatively very powerful computers and software we have available nowadays, one of the best ways how to design controller is to create mathematical model of given system. Then we can can design arbitrary number of controller and test them virtually, in theoretical sphere. There are two biggest advantages in this approaches are: speed and safety. In speed I mean, that simulated time can run much more fast then real time, it depends only on computer performance, so it can be theoretically infinitely fast. The second advantage, safety, is coming from the nature of this approach: we are not using real system, so we cannot damage it while using incorrectly set controller.

There are also several disadvantages. Most common between them is, that usually it is not easy to find exact mathematical model of our system so in most of cases are we using simplified model where we neglect some aspect which, as we hope, have small enough impact on mode behaviour. Eventually, this can have considerable big influence while using very sensitive system, non-robust controller or long time of simulation. In my work, e.g. I am neglecting wheel steering, level of battery recharge, non-linearity of motors, noise etc.

My model is divided into three main parts: controlling(two controllers), model of robot and model of camera.

7.1 Model of robot

Mathematical models of differential drive are described in chapter chapter 5. As Matlab allows continuous simulation I am using continuous model. In real robot I am calculate real speed for left and right motor from normalized values for forward and angular speed. That would not be need in simulink model, because input values are forward and angular speed. Problem is, that in real code I am implemented also ramp in order to avoid very big change in required speed. These ramps are applied on left and right wheel separately, so also in model I need to first calculate right and left speed (7.1), apply ramp and then calculate back forward and angular speed for model. Following equations show, how to obtain velocity for both wheels from robot velocities, all in normalized values.

$$\begin{aligned} L_n &= v_n + \omega_n \\ R_n &= v_n - \omega_n \end{aligned} \tag{7.1}$$

Obviously, the sum of forward and angular speed has to be lower than one in order to keep motor speed normalize in range <-1, 1>. For this purpose I am using anti wind-up method described in subsection 7.3.6

Next step is to determine real speed from normalized values. For this purpose we need maximal values which corresponds with normalized values equal to one. I measured that maximal forward speed is $v_{max} = 0,725m/s$. From this I can calculate maximal angular speed as following:

$$\omega_{max} = v_{max}\frac{WHEELBASE}{2} = 9,063rad/s \tag{7.2}$$

Because dependence of motor speed on written PWM is not really linear, in real robot I am using measured polynomial characteristic from section 5.7.

7.2 Model of camera

In order to design usable controller I need to have real feedback from camera also in simulation. That means what value would camera return in specific position of robot and guide line in specific time. For this purpose I need to know what camera really see. Because camera and track are not making a 0° but 45° angle, also real array which camera sees would not be regular square but trapezoid with bases b_1 and b_2 and sides both s_1 and s_2. Centre of shortest base is $0,10m$ from robot's centre, as it is shown in Figure 7.1 , where

$$\begin{aligned} s_1 = s_2 &= 0,20m \\ b_1 &= 0,10m \\ b_2 &= 0,22m \end{aligned} \tag{7.3}$$

In my C++ code I am looking in one row in order to find deviation. I need to do the same in my mathematical model. To find required equations I need to make sketch first: Figure 7.2.

Now I can express coordinates for both P points, which are border points on my 'what camera see' line as following:

$$\begin{aligned} P_{1x} &= x_0 + s * cos(\theta + \alpha) \\ P_{1y} &= y_0 + s * sin(\theta + \alpha) \\ P_{2x} &= x_0 + s * cos(\theta - \alpha) \\ P_{2y} &= y_0 + s * sin(\theta - \alpha) \end{aligned} \tag{7.4}$$

Next step is to find P_m point (7.5), which is middle point between P_1 and P_2. It is useful, because I want to parametric description of line, so parameter t will be

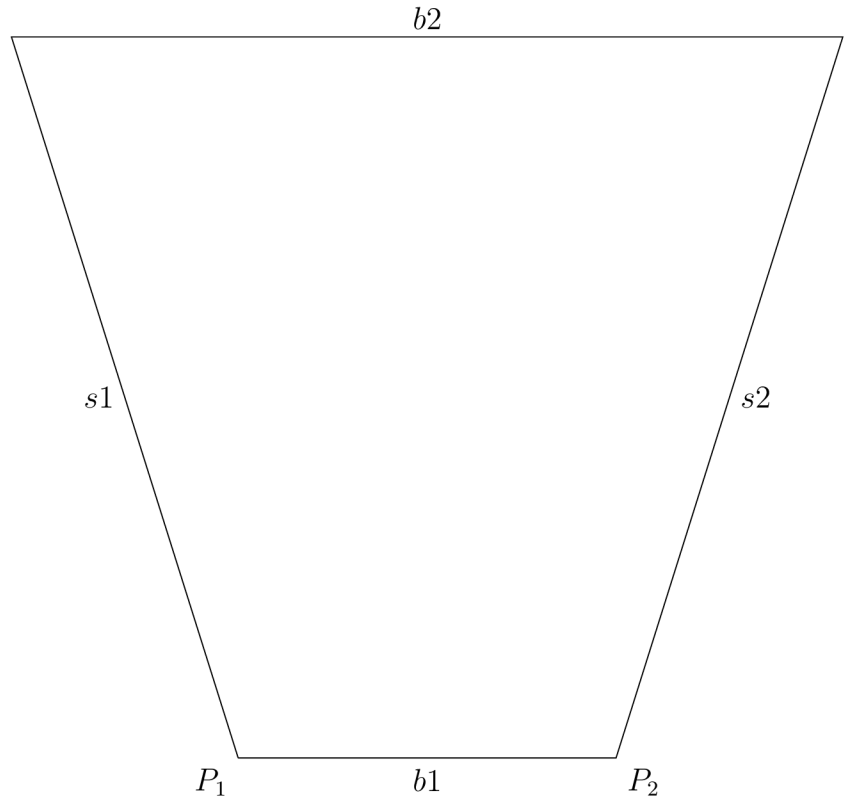


Fig. 7.1: Camera view geometry

equal to deviation I want to find. Because of the same reason I need to find also vector u (7.6) in direction from P_m to P_1. P_1 is on left side of my robot, so also the sign will be the same as for real deviation (minus on left).

$$P_m = [\frac{P_{1x} + P_{2x}}{2}; \frac{P_{1y} + P_{2y}}{2}] \tag{7.5}$$

$$u = P_1 - P_m \tag{7.6}$$

Finally, the parametric description of my 'what robot see' line is following:

$$\begin{aligned} x &= P_{mx} + t * u_x \\ y &= P_{my} + t * u_y \end{aligned} \tag{7.7}$$

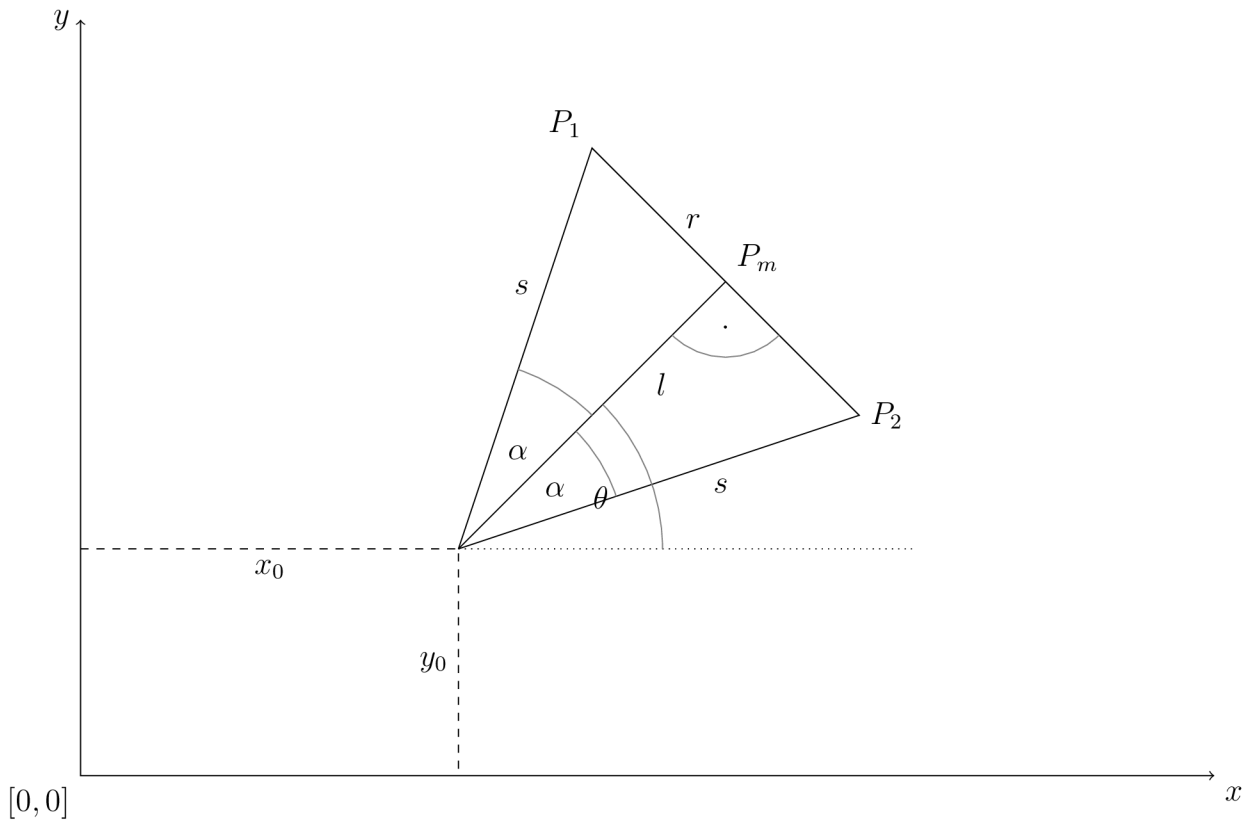


Fig. 7.2: Camera math

, where parameter t is in range $< -1, 1 >$ for each point which lies in visible zone of my robot.

In order to find out actual deviation have to I found the intersection between guide line and 'what robot see' line. I am using in my simulation only linear guide line described as following:

$$\begin{aligned} x &= A_x + k * v_x \\ y &= A_y + k * v_y \end{aligned} \tag{7.8}$$

, therefore it is easy to express value of parameter t in point of intersection:

$$t = \frac{A_x - \frac{v_x}{v_y} A_y - P_{mx} + \frac{v_x}{v_y} P_{my}}{u_x - \frac{v_x}{v_y} u_y} \tag{7.9}$$

In special case, when guide line is parallel with x axis, previous equation would not work because there will be division by zero. Hence the equation will be:

$$t = \frac{A_y}{u_y} + \frac{P_{my}}{u_y} \tag{7.10}$$

The parameter t is then directly the deviation. If its value is not in range $< -1, 1 >$, that means that robot cannot see line and I will end my simulation.

7.3 Controllers

This is the most important part of my work. Good controller is core of every automotive system. Theory of control offers wide scale of type of controllers on techniques for their tuning, but most of them are developed for linear systems. Its objective is to keep controlled system in specific state, so it needs some feedback from system, how far our it is from required state. Controller than determine how big should be action taken in the system. In this way there rise a closed loop which we call control loop. The basic control loop, containing controller and controlled system, looks like Figure 7.3. , where

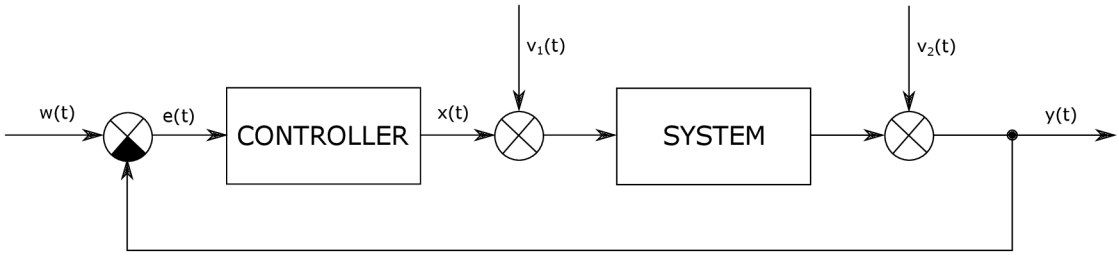


Fig. 7.3: Simple control loop example

- w(t) - wanted or required value - required deviation of robot and guide line (because my objective is to follow line, it is always equal to zero (t) = 0).
- y(t) - controlled value - actual deviation of robot and guide line. This will be measured by camera.
- e(t) - controlling error - difference between required and actual deviation e(t) = w(t) - y(t). This is an input for my controller.
- x(t) - action - action(output) of controller, in my case it is speed of both motors.
- v(t) - fault quantities - they appears in different places and controller should compensate them.

This is loop only with one controller and all values are continuous. In my case there will be two controllers and only robot system will be continuous, all other like controller (implemented digitally) and feedback provided by camera (with specific frame-rate) will be discrete. My control loop then looks more like Figure 7.4.

7.3.1 PID controller

A proportional–integral–derivative controller (PID controller or three term controller) is a control loop feedback mechanism widely used in industrial control systems and a variety of other applications requiring continuously modulated control.

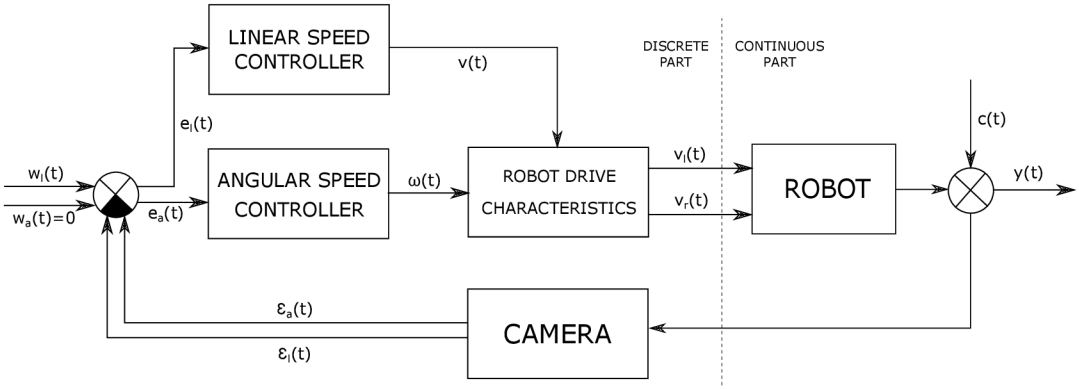


Fig. 7.4: Control loop with two controllers and continuous and discrete part

A PID controller continuously takes an error value $e(t)$ and applies a correction based on proportional, integral, and derivative terms (denoted P, I, and D respectively) Figure 7.5 which give the controller its name. By nature of my robot and

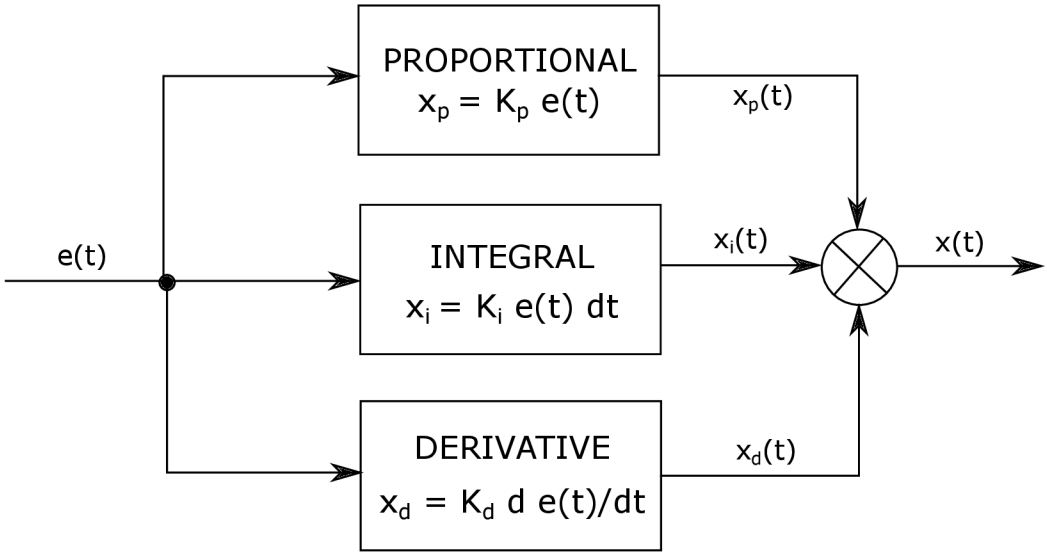


Fig. 7.5: PID controller diagram

way of its control there is an integral element already in system. The reason is that I am controlling the error - deviation(distance) by speed of rotation. My active value is derivation of controlled value so there is an integral between them in direction of control loop.

7.3.2 PSD controller

My controller will be implemented in raspberry pi, so I will use its discrete variant called PSD (proportional–sum–difference) controller. The only difference is, that instead of integrator there is an accumulator (sum) a instead of differentiator there is difference of actual and previous value. These elements are doing their function always once in sample period.

7.3.3 Forward speed controller

After several attempts I decided that controller for forward speed will contain only P term and constant value. The resulting forward speed can be calculated as following:

$$v_L = v_{LC} + v_{LV} * (1 - e_R) \tag{7.11}$$

, where v_L is resulting forward speed, v_{LC} is constant term of resulting forward speed and v_{LV} is its variable term. e_R is roughness error value determined by computer vision module in range $< 0, 1 >$.

7.3.4 ITAE Criterion

Integral of **T**ime-multiplied **A**bsolute value of **E**rror criterion is very simple and very powerful method of tuning PID controllers. Its objection is to minimize value given by (7.12). We start at certain initial set of constants and then use iterative minimization algorithm like simplex method to find the minimum. For this purpose am I using $fminsearch()$ function implemented in matlab standard library. By nature, obviously, it is not an algebraic method nor it is brute-force method, but it is almost impossible to use it without computer. Also with computer it takes about several tens of seconds to find the result. Advantage of this method is, that it can work with slightly non-linear system, but it do not have to always find real minimum and it can be very unstable. For instance, when I am running the same simulation with different simulation time it always find very different set of constants.

$$I = \int_0^\infty |e| \cdot t \cdot dt \tag{7.12}$$

7.3.5 Fuzzy control system

Another, in machine control widely used, control system based on fuzzy logic mathematical system. It analyses analogue input values in terms of logical variables that take on continuous values in range $< 0, 1 >$ which are equal to probability for given variable. In contrast, classical or digital logic operates on discrete values of either 1 (true) or 0 (false). The term *'fuzzy'* express the fact that the logic involved can

deal with concepts that cannot be expressed only as the *'true'* or *'false'* but rather as *'partially true'*. Although alternative approaches such as genetic algorithms and neural networks can perform just as well as fuzzy logic in many cases, fuzzy logic has the advantage that the solution to the problem can be cast in terms that human operators can understand, so that their experience can be used in the design of the controller. This makes it easier to mechanize tasks that are already successfully performed by humans. Obviously, it has a sense to use it when there are more input and output variables which are depended on each other. In my case, when there are only one input and one output variable, there is no great sense to use it in comparison with PID controller, which is very well developed nowadays. [25]

7.3.6 Anti wind-up

Wind-up is a very common phenomenon when output of physical systems is limited (and it always is) and in certain point in saturate, but integrator element in controller, in order to compensate error as soon as possible, is still integrating. In this case, the output of controller is still raising without any effect on controlled system. Problem happened when error is finally compensated, but integral has already integrated very big number and there is need to un-integrate it. This takes such a significant amount of time to recover within the operating range of the actuator and so causes a lag in response. This process can repeat itself as a limit cycle, or eventually converge towards the commanded value depending on the set gain and system response. There are many ways how to prevent this effect, which are very easy to implement in digital controller. The way I chose is based on creating new closed loop from point after saturation of actuator to point before integrator. In this way, the value which is saturated multiplied by constant is deducted from error value which is integrated. [26]

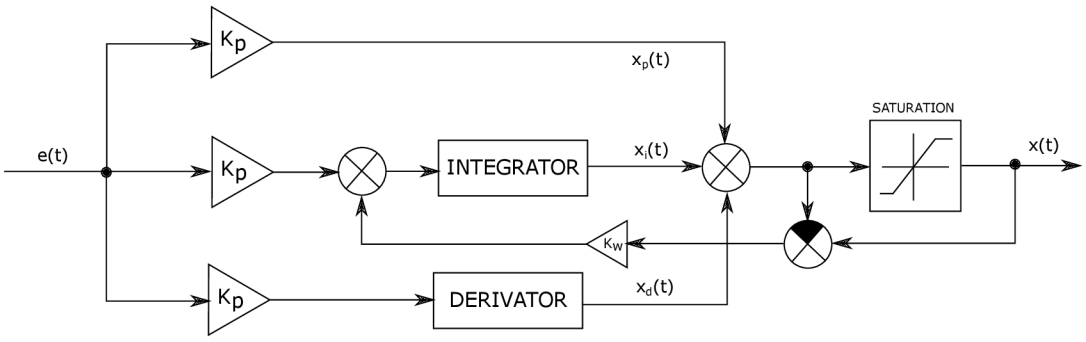


Fig. 7.6: Scheme of anti wind-up method

7.4 The resulting model

The resulting model, created in simulink, is outcome of many previous sub-models and improvements and it is part of more complex script which provides variables for model, data logging, their processing and plotting and some more complex tasks like application of ITAE criteria. How it looks is shown in Figure 7.7.

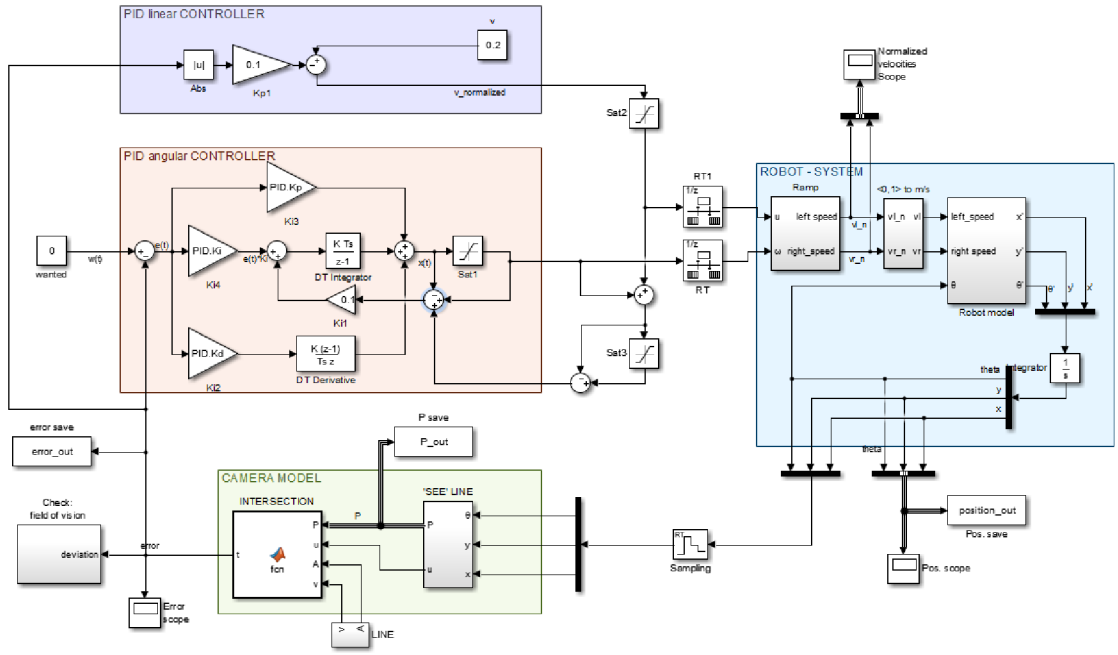


Fig. 7.7: System model consisting of models of robot, its controller and camera in simulink

7.4.1 Simulation results

Below is a table Equation 7.4.1 with several combinations of initial position

$$[x\_init,\ y\_init,\ theta\_init] \tag{7.13}$$

and guide line defined by point

$$A = [a_x,\ a_y] \tag{7.14}$$

and vector

$$\vec{v} = [v_x,\ v_y] \tag{7.15}$$

and determined constants

$$[K_p,\ K_i,\ K_d] \tag{7.16}$$

for PID controller for angular speed. Shift value means how far is robot centre from guide line and angle value is angle between robot main axis and guide line.

Initial position	Guide line definition		Shift	Angle	PID constants
$[m,\ m,\ °]$	$[a_x,\ a_y]$	$[v_x,\ v_y]$	$[m]$	$[°]$	$[K_p,\ K_i,\ K_d]$
$[0.00,\ 0, 0]$	$[0,\ 0]$	$[1,\ 1/3]$	0.00	19.47°	$[0.48,\ -0.01,\ 0.01]$
$[0.00,\ 0, 0]$	$[0,\ 0]$	$[1,\ 2/9]$	0.00	12.83°	$[0.53,\ -0.01,\ 0.01]$
$[0.00,\ 0, 0]$	$[0,\ 0]$	$[1,\ 4/9]$	0.00	26.39°	$[0.47,\ -0.01,\ 0.01]$
$[0.00,\ 0, 0]$	$[0,\ 0]$	$[1,\ 1/2]$	0.00	30.00°	$[0.47,\ -0.01,\ 0.01]$
$[0.01,\ 0, 0]$	$[0,\ 0]$	$[1,\ 1/3]$	0.01	19.47°	$[0.48,\ -0.04,\ 0.01]$
$[0.02,\ 0, 0]$	$[0,\ 0]$	$[1,\ 1/3]$	0.02	19.47°	$[0.48,\ -0.06,\ 0.01]$
$[0.03,\ 0, 0]$	$[0,\ 0]$	$[1,\ 1/3]$	0.03	19.47°	$[0.47,\ -0.06,\ 0.01]$
$[0.04,\ 0, 0]$	$[0,\ 0]$	$[1,\ 1/3]$	0.04	19.47°	$[0.47,\ -0.10,\ 0.01]$

Average values for resulting constants are $[0.48, 0.05, 0.01]$ and dispersion $[0.02, 0.03, 0.00]$; While I and D terms are negligibly small (the order of the hundredths) I will implement them into robot as zero. This can be caused by I term that is already in my controlled system-robot: I am controlling deviation s by velocity $\frac{ds}{dt}$.

Graphic view of calculated result (first row of table) can be seen in Figure 7.8. This graph shows 2D space where blue line is a guide line, green one is centre of robot and red one is centre of line where robot measures its deviation from guide line.

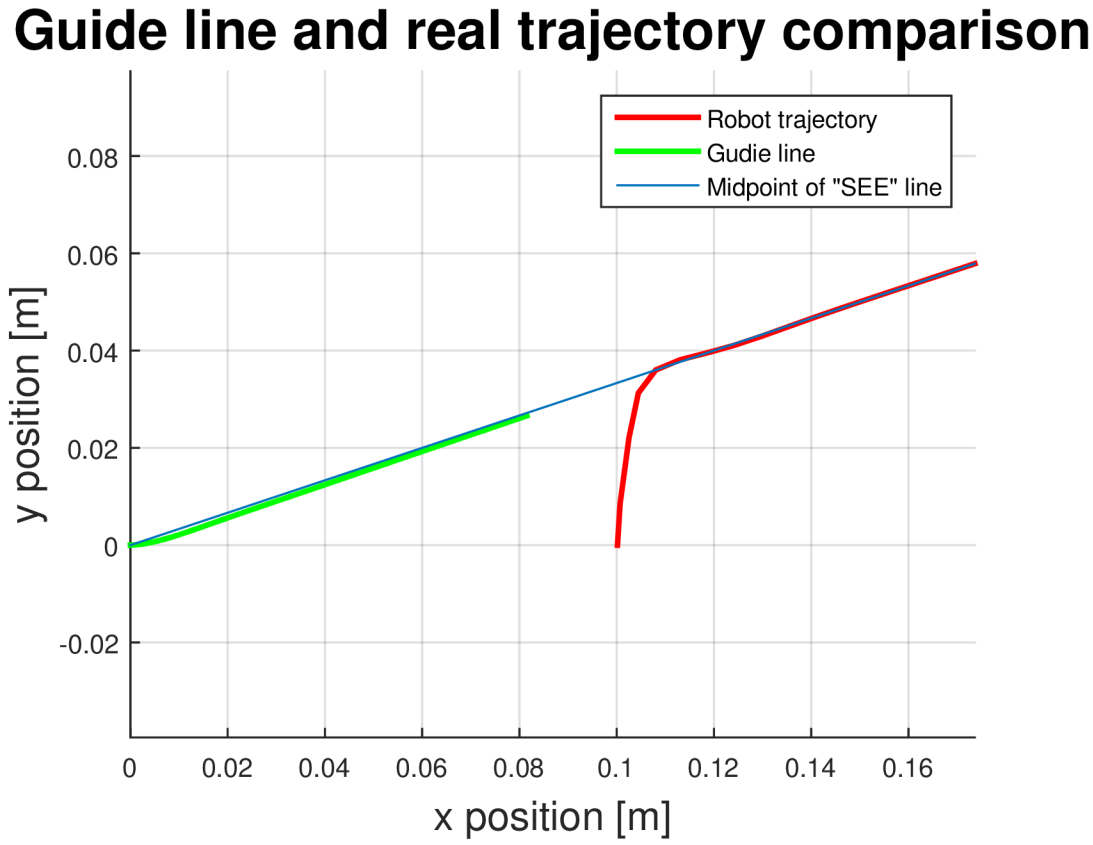


Fig. 7.8: Simulated robot control with calculated optimal controller constants

8 IMPLEMENTATION

When working on some project that is no strictly specialized, good idea is to make it modular. That means every module lives its own life independent on others and they communicate only by theirs interfaces. Biggest advantage is that when one change part of real system (camera for IR sensors or LIDAR or drive for another type) there is need to write again (or change) only that one specific part of code. That caused development much more faster and code reusable.

Basic idea of my control loop, how it is implemented in code, can be seen in Figure 8.1

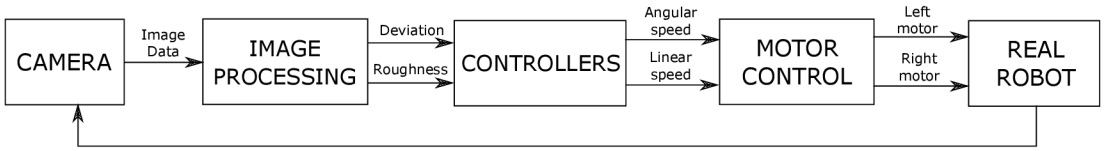


Fig. 8.1: Control diagram of modular code

I have split my code into five main blocks/modules:

- Main
- Computer vision
- Controllers
- Motor control
- Other support

8.1 Main code

In this file, main.cpp, is only one function main(). It should be as simple and summary as possible so there are only function calls from other files. In the beginning of main() function, there are setup function calls which should be done only once. Follows infinite while loop which contains calls of functions required for image taking, its processing, controlling and data logging. This cycle is executed every sample period. In the beginning of every cycle it checks if certain pin is shorten to ground and in a such case is the while cycle finished using break(). Subsequently, function for safe finish like motors cut-of and file closing are called and whole process is finished.

8.2 Computer vision code

This module contains everything which is related with camera, image processing ant its saving. There is defined that all images in my code resolution will be $320x240$ pixels. I need to have image as small as possible in order to keep process as fast as possible but detailed enough for data obtaining purposes.

8.2.1 Camera set-up

Called in the beginning of main loop. First it tries to open camera with $ID = 0$ (there is only one camera connected) and in case of failure it writes error message and exit application. Next step is to set above mentioned resolution.

8.2.2 Image saving

Save image from argument to file */home/pi/ROBOT/CAM/* as *im* with suffix *i* which is a number also passed by parameter in bitmap (.bmp) format. These images can be seen by any remote FTS client or by mapping raspberry memory to working laptop.

8.2.3 Image showing

For showing images real-time. As my virtual display created using Xming section 4.7 is only connected display they will be shown there on my laptop. It is very good way for controlling what really going on inside.

8.2.4 Find threshold level

Very important part of image processing. When good threshold is done and there stays only guide line in image, obtaining data is much more easy. In other case it is very difficult to decide what is really line and what is not. In some cases e.g. when one side of image is much more lighter the other mere threshold is insufficient and there are more complex method but I will not use them because of their computational difficulty. I have tried several methods of obtaining threshold level, among them: average, median, histogram or only constant. It was really surprising that poor constant 100 works very well and it takes almost no time to find it. I have tried also some very complex methods (contrast stretching, canny for edges obtaining and Otsu's method) using openCV core libraries which works also very well, but it takes $15ms$ that is unaffordable for my purposes.

Example what 'robot sees' after it takes and threshold image is in Figure 8.2.

Fig. 8.2: What robot sees after it takes and threshold image example

8.2.5 Buffering

In computer vision header file there is defined a buffer class which is storing last n (count depends on actual linear velocity) values of previous deviation and in case of line lost or other complication it predicts new deviation as average of stored values. This allows to continue in right way when intersection is located in arc.

When linear speed is at 10% of maximal speed ($7,25cm/s$), I am storing 20 previous values to buffer and sample period is $40ms$ am I considering values stored in previous $5,8cm$ while predicting new value.

8.2.6 Find errors function

Mother function for obtaining errors. It calls other created functions designed for obtaining line error in one specific line and use them to determine actual deviation and roughness of guide line. In some cases when these functions throws exceptions above mentioned buffer class is used to find deviation.

Roughness value is calculated as a sum of absolute values of deviations in several (I am using five) rows across whole height of image. If a line is not find in specific row value 0.5 is used. This sum is subsequently normalized: it is divided by 3 although maximal value of this sum is theoretically equal to 9. I do not assume that line can be so winding in reality. After all this value is cut to one because it could be potentially greater than one because of my assumption.

8.2.7 Find errors in line functions

I have created two of this functions. First starts from centre and the second starts from right edge. In first case I start with supposition that from nature of this task (line following) the biggest probability for guide line position is centre of image. I start with centre of considered image line and check if it lies on guide line. If it is so I will find its left and right edges. In other case I will start to inquire alternately on one side and the other in an effort to find line and, eventually, its edges. After that there several tests on found result. First I check if I found both borders or at least one of them. If no border was found I will throw an exception (probably it can be guide line intersection); if one was found I set the second one to its maximal value. In next step the width of potential line is controlled; there are defined assumed maximal and minimal values. Last test is about centre shift: if it is too long probably it is a blunder because there is no way for so fast change of robot position. If found result is evaluated as valid centre is calculated as (8.1) and normalized deviation as (8.2)

$$centre = \frac{left + right}{2}; \tag{8.1}$$

$$deviation = \frac{2 * centre}{im\_width} - 1; \tag{8.2}$$

Second function is very similar, bit easier but no so resistant to errors in image processing. Main difference is that I am starting from right edge because one of requirements is that robot should follow right branch when guide line is divided into two. All test as in first function are executed and if found line is satisfactory (centre shift is most important here) the deviation is calculated an returned. If found line is not satisfactory algorithm try to continue to left and find real guide line.

8.3 Controllers code

Discrete forms of designed controllers and theirs constants are implemented here. In its header file is declared the *error* structure which contains two used errors for

two controllers: deviation from line centre and roughness of guide line. Part of controllers is also anti wind-up as described in subsection 7.3.6.

8.4 Motors control code

Objective of this module is to manage motors. First of all there are defined pin numbers for communication with motor driver as can be seen in Figure 4.3b, chassis mechanical constants and maximal values of velocity and PWM, as written in section 5.6.

8.4.1 Motors set-up

This function has to be called before motors usage. Sets modes for all pins used for motor control and proceed initial writing to them. Stop pin and its pull-up are also initialized here.

8.4.2 Write to motors

Most important part of this part of code because it provides writing to motors itself. It is called automatically by interruption every $5ms$. The reason why is it so is ramp implementation. First I control if received signal number is really that one which should call this function.

Second step is to obtain normalized velocities for left and right motors from normalized linear and angular speed using equations from section 7.1. We assume that anti wind-up was implemented in control module so absolute value of results should be less than 1.

Above mentioned ramp is a way how to avoid huge changes of motor rotation which could potentially damage them or their control electronic because of induced currents which occurs as a response to sudden variation. The values written to motors are changed every $5ms$ by 10% of its maximal value so their progress looks like stairs.

Subsequently function for obtaining PWM from speed, subsection 8.4.3. Last step is to write actual values of PWM to motors. This has to be done in two steps: absolute value of PWM into PWM pin and its signum to direction pin.

8.4.3 Normalized speed to PWM

Here are implemented measured characteristics from section 5.7. Obtained polynomial equation are used for obtaining PWM from given required normalized speed.

8.4.4 Motor stop

Has to be called before and of whole process. It stops (writes PWM = 0) and breaks both motors. Only then can be execution finished securely.

8.5 Support supplements

Some support functions as logger and UDP communication with matlab are implemented here. Support header contains two inline functions which simplify measuring and working with time, new type for time points and enums definitions for logger.

8.5.1 Logger

Easy function which objective is to logging messages into file and possibly write them into stream. There are several levels of logging: ERROR, WARNING, INFO, DEBUG. These can be logged as several types of message: NON, MAIN, CAMERA, MOTORS, CONTROL. I am using integrated C libraries in order to obtain actual time and then format it by *strftime()* function.

There are also set-up and close functions for logger which opens and closes log file *'/home/pi/ROBOT/files/logger.txt'* as *ofstream*. *Stdout* buffer is set to zero to insure the best feedback while debugging.

8.5.2 UDP communication

For sending data to matlab am I using UDP (User Datagram Protocol). First step is to set-up UDP communication. I need to create socket, and set structure of type *sockaddr_in* with specific IP address, port number and other configurations. Second function is intended for data sending. As argument it takes errors_t structure and optional time information of integer type in ms. Subsequently it convert it all to strings and join with semicolons among them. Final string is send by *sendto()* function to my laptop where these data are processed.

Received errors data are plotted as actual value and its history as can bee seen in Figure 8.3.

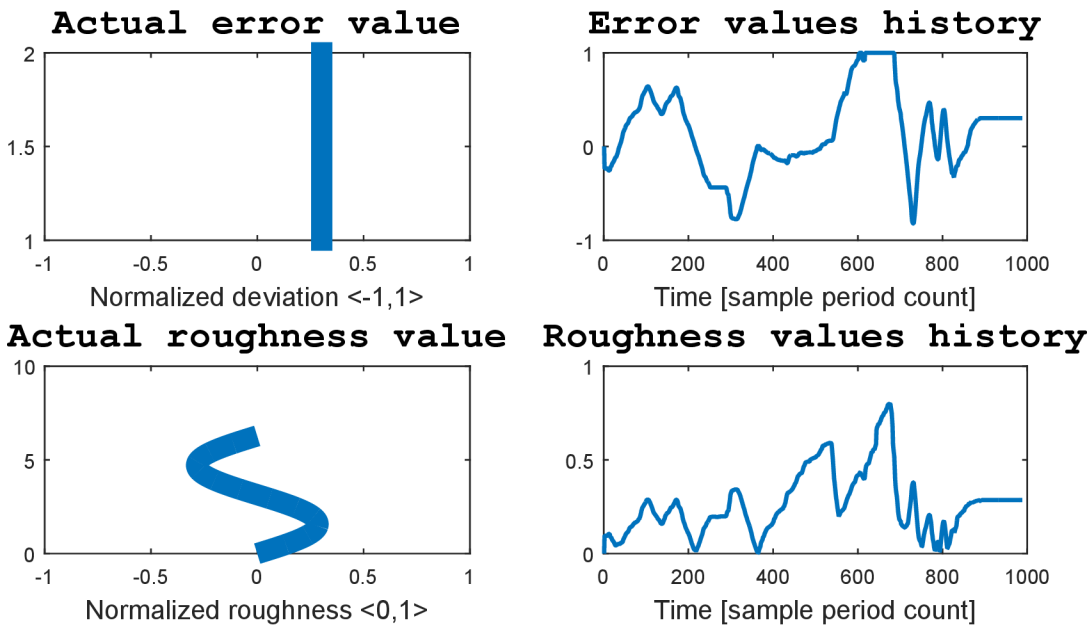


Fig. 8.3: Measured deviation and roughness and their history progresses

Optional time values are stored into array and then, after communication is completed, there are calculated some statistics values as maximum, minimum, average (mean) and standard deviation. These are:

$$\begin{aligned} Minimum\ sample\ period &= 27\ ms \\ Maximum\ sample\ period &= 38\ ms \\ Average\ sample\ period &= 33\ ms \\ Statistical\ dispersion &= 2\ ms \end{aligned} \tag{8.3}$$

Histogram is showed in Figure 8.4.

In implemented code I am compensating all sample periods to 40 ms.

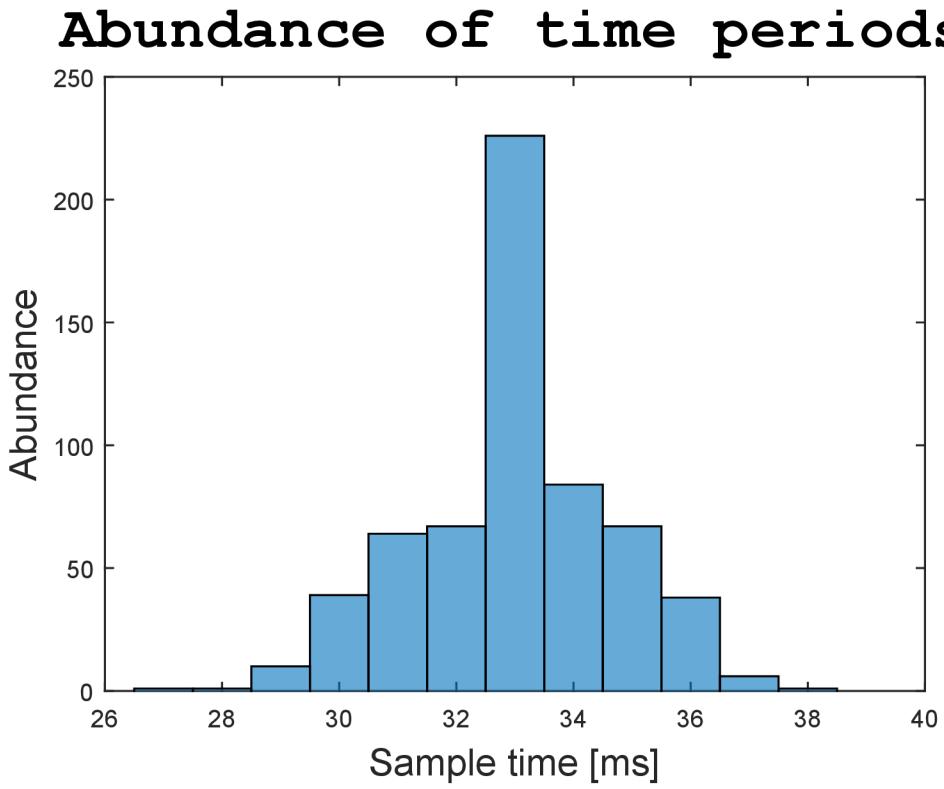


Fig. 8.4: Histogram (abundance) of measured sample time periods of main loop

8.6 Makefile

Make program is there to simplify the compilation process, speed it up and automate. It reads file called Makefile which contains instruction how my program should be compiled. In variables there are defined final program name and its cpp source files. Follows flags (write extra errors, all openCV required flags etc.) and external libraries (wiringPi, openCV libraries). In next step object files with same base-names as source files are created. The last ones are several rules for compilation and cleaning using above defined variables and implicit variables.

9 CONCLUSION

Objective of my work was to design and create small wheeled robot which should follow the marked line using computer vision. I can say I have done this but I have to admit I am still in the beginning concerning this field of technology. The biggest problem I have encountered is speed of calculations although raspberry Pi 3 is relatively powerful computer. To take image small like 320x240 pixels takes approximately $28ms$. With other tasks I round the sample period up to $40ms$, that is 25 times in one second. To be able to control it I can use maximal forward velocity of robot only equal to 10% of maximal power of used motors and still it oscillates sometimes.

First important part was to process captured image, that means to obtain image where guide line is black and all other is white. I have tried many methods, among others also constant threshold value which works very well and takes almost no time. Other extreme was complex function which consists of several ideas like edge detector, contrast stretching and Otsu's method. Its result are very good but it takes unaffordable long time.

The other part was to find needed information in these image data. While there is huge amount of information in image there is also big variance and obtaining only what is important for me is much more complicated. The algorithms I designed are relatively simple which is caused partially by limited frame of my project and partially by limited power of used hardware. They works good enough but it is obvious that it is far away from perfection.

Last step was to use above mentioned data and drive a real robot. I needed to know (and be able to describe) it from mechanical point of view so I measured all required constants of used drive and camera and found out all necessary mathematics. There was also need to install all used software and libraries like openCV or wiringPi. While debugging I used one way communication with matlab to visualize and analyse measured data and I created a mathematical model also in matlab to design the best PID controller using ITAE method.

BIBLIOGRAPHY

[1] 42 Bots. Differential drive princip. URL: http://42bots.com/tutorials/differential-steering-with-continuous-rotation-servos-and-arduino/.

[2] O Morandin M.Y. Gomes, L.A Bassora. Pid control applied on a line-follower agv using a rgb camera. *International Conference on Intelligent Transportation Systems*, 2016. URL: http://ieeexplore.ieee.org.ezproxy.lib.vutbr.cz/xpls/icp.jsp?arnumber=7795553.

[3] Gimp. Convolution matrix - generic filter. URL: https://docs.gimp.org/en/plug-in-convmatrix.html.

[4] Rafael Gonzalez. *Digital Image Processing*. Pearson Hall, 2008.

[5] Jeff Schuler. Biomedical image processing. 2009. URL: http://engineersphere.com/biomedical-image-processing-v.

[6] Otsu's method. *Wikipedia - the free encyclopedia*, 2018. URL: https://en.wikipedia.org/wiki/Otsu%27s_method.

[7] Texas Intruments University. Ti robotics systems learning kit. URL: https://university.ti.com/en/faculty/ti-robotics-system-learning-kit/ti-robotics-system-learning-kit.

[8] Pololu. *Pololu rom02a board*. Pololu Corporation. URL: https://www.pololu.com/product/3543/resources.

[9] Raspberry pi Foundation. *Raspberry pi 3 documentation*. Raspberry pi Foundation. URL: https://github.com/raspberrypi/documentation.

[10] Gjerrit Meinsma Okko H. Bosgra, Huibert Kwakernaak. *Design Methods for Control Systems*. DISC, 2007-2008.

[11] Digital image processing. *Wikipedia - the free encyclopedia*, 2018. URL: https://en.wikipedia.org/wiki/Digital_image_processing.

[12] Azriel Rosenfeld. *Picture Processing by Computer*. Academic pr, 1969.

[13] Canny edge detector. *Wikipedia - the free encyclopedia*, 2018. URL: https://en.wikipedia.org/wiki/Canny_edge_detector.

[14] Andrew Tridgell. Samba documentation. *Samba*. URL: https://www.samba.org/samba/docs/.

[15] Michael Larabel. The high-profile x.org. *phonorix*, 2010.

[16] *CMake*. URL: https://cmake.org/download/.

[17] *OpenCV GitHub download*. URL: https://github.com/Itseez/opencv.git.

[18] Robert Laganiere. *OpenCV 2 Computer Vision Application Programming Cookbook*. PACKT BUBLISHING, Birmingham, May 2011.

[19] Gordon. Wiring pi - gpio interface library for the raspberry pi. *WiringPi Website*, 2018. URL: http://wiringpi.com/.

[20] Alyona Stashkova Catherine Pickersgill. *NetBeans Developing Applications with NetBeans IDE*. ORACLE, 8.2 edition, September 2016. URL: https://docs.oracle.com/netbeans/nb82/netbeans/NBDAG/toc.htm.

[21] Alexpux Martell Malone Ray Donnelly David Macek Renato Silva niXman. Msys2. 2017. URL: https://github.com/msys2/msys2/wiki.

[22] Michael Jenkin Gregory Dudek. *Computational principles of mobile robotics*. 2000.

[23] Thomas Hellström. Kinematics equations for differential drive and articulated steering. *Department of Computing Science*, 2011.

[24] Karl Worthmann Mohamed W. Mehrez Mario Zanon George K. I. Mann Raymond G. Gosine Moritz. *Regulation of Differential Drive Robots using Continuous Time MPC without Stabilizing Constraints or Costs*. IFAC (International Federation of Automatic Control), 2015.

[25] Witold Pedrycz. *Fuzzy control and fuzzy systems (2 ed.)*. Research Studies Press Ltd, 1993.

[26] Prof. Alberto Bemporad. Anti-windup techniques. University of Trento, 2011.