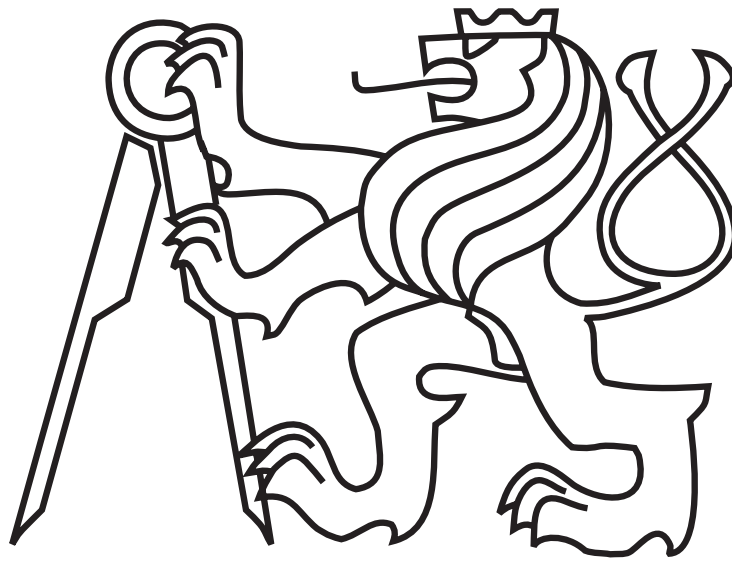


CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering

# BACHELOR THESIS



Michal Eliáš

**Mobile Robot Capable of Autonomous Navigation**

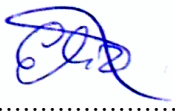
Department of Cybernetics

Thesis supervisor: Ing. Tomáš Krajník

## Declaration

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

In Prague on.....27.5.2011.....

  
.....

## **Acknowledgements**

I would like to thank my supervisor Ing. Tomáš Krajník, for showing me the world of mobile robotics, also to Ing. Martin Nečas, MSc., Ph.D. for great help with mechanical parts manufacturing. I would also like to thank my girlfriend, for helping me with testing, and to my family, that was supporting me.

### *Abstrakt*

Tato práce se zabývá návrhem a stavbou mobilní robotické platformy. Je zde popsán proces vývoje a výroby mechaniky, elektroniky, firmware a software. Vývoj byl směřován k outdoorové platformě vážící 5-10 kg, se zvýšenou světlostí podvozku a vyšším výkonem motorů. V práci jsou popsány základní typy senzorů použitelných pro autonomní navigaci. Podstatnou část práce tvoří vývoj komunikačního rozhraní propojujícího jednotlivé funkční bloky platformy a vývoj algoritmů pro zpracování a segmentaci obrazu z kamery v reálném čase. Důraz je také kladen na bezpečnost – v případě selhání systému dojde k automatickému zastavení. Provozní testy manuálního řízení a autonomní reaktivní navigace byly provedeny v městské zástavbě a během robotické soutěže Robotem Rovně 2011.

### *Abstract*

The goal of this thesis is development and manufacturing of robust mobile robotic platform, weighing 5 to 10 kg, that will be able to autonomously travel on unpaved roads. Mechanics, electronics, firmware and software development is discussed, as well as basic sensors types and image analysis and segmentation. For safety reasons, several protections were incorporated – namely automatic shutdown in case of system failure, as well as battery monitoring. Testing of autonomous navigation and manual wireless control was done in urban environment and during ‘Robotem Rovně’ competition.



## BACHELOR PROJECT ASSIGNMENT

**Student:** Michal Eliáš  
**Study programme:** Electrical Engineering and Information Technology  
**Specialisation:** Cybernetics and Measurement  
**Title of Bachelor Project:** Mobile Robot Capable of Autonomous Navigation

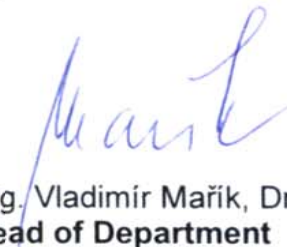
### Guidelines:

1. Design and assemble a mobile robotic platform, which would be able to carry the sensory and computational hardware necessary for navigation in outdoor environment.
2. Get to know the sensor data processing methods used in mobile robotics.
3. Select a suitable navigation method for an outdoor mobile robot.
4. Select suitable sensors and computational hardware for the selected method.
5. Construct a platform capable of carrying the selected hardware.


**Bibliography/Sources:** Will be provided by the supervisor.

**Bachelor Project Supervisor:** Ing. Tomáš Krajník

**Valid until:** the end of the winter semester of academic year 2011/2012

  
prof. Ing. Vladimír Mařík, DrSc.  
**Head of Department**



  
prof. Ing. Boris Šimák, CSc.  
**Dean**

Prague, February 18, 2011

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Sensors</b>	<b>2</b>
2.1	Sensors for Dead Reckoning . . . . .	2
2.2	Sensors for Localization . . . . .	3
2.3	Sensors for Close Range Perception . . . . .	3
2.4	Selected Sensors . . . . .	5
<b>3</b>	<b>Mechanics</b>	<b>7</b>
3.1	Construction . . . . .	7
3.2	Implementation . . . . .	7
3.3	Motor and Gearbox . . . . .	9
<b>4</b>	<b>Electronics</b>	<b>12</b>
4.1	Overall . . . . .	12
4.2	Battery Type Selection . . . . .	12
4.3	Modules Summary . . . . .	14
4.4	Top Module . . . . .	15
4.5	Control Board . . . . .	15
4.6	Peripherals . . . . .	16
4.7	Li-Po Monitor . . . . .	16
4.8	Motor Driver . . . . .	20
<b>5</b>	<b>Firmware</b>	<b>22</b>
5.1	Code Reusability . . . . .	22
5.2	Protocol Communication . . . . .	22
5.3	Control Board . . . . .	23
5.4	Motor Driver . . . . .	25
<b>6</b>	<b>Software</b>	<b>26</b>
6.1	RoboControl Software . . . . .	26
6.2	Image Processing . . . . .	26
6.2.1	Strategy Results . . . . .	27

<b>7 Experiments</b>	<b>29</b>
<b>8 Conclusion</b>	<b>31</b>

## List of Figures

1	Absolute rotary encoder with Gray code . . . . .	2
2	Signal from relative encoder, with direction information . . . . .	2
3	Example of touch sensor . . . . .	4
4	Phototransistor . . . . .	4
5	Sharp rangefinder . . . . .	5
6	Laser rangefinder SICK LMS 100 . . . . .	5
7	Sonar SRF05 . . . . .	5
8	Hokuyo Laser rangefinder . . . . .	5
9	Triangulation principle explanation . . . . .	6
10	Camera mounted on platform . . . . .	6
11	Ackermann steering . . . . .	7
12	Differential steering . . . . .	7
13	P3AT robotic platform . . . . .	8
14	QUIDO robotic platform . . . . .	8
15	Concept 1 . . . . .	8
16	Concept 2 . . . . .	8
17	Common synchro belt design . . . . .	9
18	One synchro belt for each wheel . . . . .	9
19	Two machined jackels . . . . .	9
20	Assembled jackels . . . . .	9
21	Reduction shaft . . . . .	10
22	RC wheel . . . . .	10
23	Bearing . . . . .	10
24	Bearing housing . . . . .	10
25	DC motor savaged from printer . . . . .	11
26	Brushless sensorless DC motor . . . . .	11
27	Overall design with LED neon . . . . .	12
28	Development version . . . . .	12
29	Lead battery discharge curve . . . . .	14
30	Li-ion battery discharge curve . . . . .	14

---

31	LiPol battery with balancer plug . . . . .	15
32	Simplified diagram of modules . . . . .	16
33	Control Board is based on LM3S6965. . . . .	17
34	BQ76PL536 integrated circuit block diagram . . . . .	18
35	Block diagram of LiPol monitor board . . . . .	19
36	Removing paper from PCB . . . . .	20
37	Etched and cleaned board . . . . .	20
38	Finished charger top . . . . .	20
39	Finished charger bottom . . . . .	20
40	Electronic speed controller . . . . .	20
41	Soldered Motor driver board . . . . .	20
42	Vanishing point strategy . . . . .	28
43	Watershed strategy . . . . .	28
44	Well detected road edges . . . . .	28
45	One edge almost out of image . . . . .	28
46	Both algorithms failing . . . . .	28
47	Correct classification . . . . .	28
48	Prague traveled path . . . . .	29
49	Official competition route . . . . .	29
50	Final preparations . . . . .	30
51	Autonomous drive test . . . . .	30

**List of Tables**

1	RoboControl classes hierarchy . . . . .	27
2	Prague Experiments . . . . .	29
3	‘Robotem Rovně’ results . . . . .	30
4	CD Content . . . . .	35

## List of Algorithms

1	Initialization of Control Board . . . . .	24
2	Router Mchanism . . . . .	24

# 1 Introduction

In the past, intelligent robotics was a science fiction, unavailable to anyone except the well financially supported research teams. It has always been a challenge, to improve past designs, introduce new principles, ideas, concepts and solutions. But due to limited amount of projects, limited cooperation, virtually no existing target market and expensive technologies, intelligent robotics was not for everyone.

Only recently there has been a rapid progress – with new and cheap sensors, high computational power of handheld computers and new means of communication, general public became interested in robotics. Now anyone who has some technical background can build simple line following or obstacle avoiding robots. To address this increasing popularity, several robotic competitions emerged, where enthusiasts and universities compete against each other, not for winning, but for sharing knowledge and experience. Most of the competitions were indoor - namely Robot-sumo, Robot Football and Eurobot [1].

Outdoor robotics has shorter history – probably first competition in Czech Republic was RoboTour [2] 2006, followed by ‘Robotem Rovně’ [3] and by RoboOrienteering [4]. Also outdoor robotic platforms are mostly unique – every team has a different means of navigation, sensors and strategy. Some platforms are made from scratch, others based on RC cars, or even on wheelchairs. Robots in hands of their makers evolve, just like a living organisms would. Every new iteration is more capable than the previous, and so on. Interesting is also development based not only on own mistakes, but also on mistakes of others. Getting better every year is the main motor of this so-called robotic evolution. We were fascinated by seeing how it works, and decided to join this compete-and-learn cycle. First attempt was during ‘Robotem Rovně 2010’ with a small outdoor robot named Brimstone [5]. After that, it was decided to enhance this concept – develop much larger, robust platform, with more advanced vision algorithms.

Result of this work is an affordable all terrain outdoor platform, with top speed around  $20 \text{ km}\cdot\text{h}^{-1}$ . Main constraint was the price of the solution, because this project is completely self-financed. Bill of materials was created for future reference. Price of the complete solution (including netbook and batteries) is around \$723, which is great when compared with price of outdoor platform Pioneer 3-AT [6] starting at \$12 500.

First, we discuss several sensor types and their use in robotics. The following chapters describe several areas – development of the mechanical construction, where several concepts are mentioned. Electronics chapter contains manufacturing and assembly of the printed circuit boards. Firmware followed by Software deals with protocol communication, navigation and image processing. Several experiments and results are presented in chapter Experiments.



## 2 Sensors

Sensors are the only means how the robotic platform could perceive the surrounding world. Information from sensors have a major impact on how well are decisions made. For navigation during competitions, most important is to stay on a given path, by continuously correcting the heading. Obstacle avoidance is also very important because every collision would end the trial. A brief description of various sensor types is given in this section

### 2.1 Sensors for Dead Reckoning

Dead reckoning is the process of estimating one's current position based upon a previously determined position, or fix, and advancing that position based upon known or estimated speeds over elapsed time [7]. It is one of the basic localization techniques. Dead reckoning suffers from accumulated error, as described in [8].

#### Encoders

Encoders are devices for rotation measurement, that can be mounted onto motor shaft, to determine number of revolutions per minute (robot speed), and possibly to estimate robot position, by a method called odometry.

Encoders from physical point of view can be optical, mechanical, magnetic and inductive. They can be absolute (angle measurement) on figure 1 or relative (speed measurement) on figure 2.

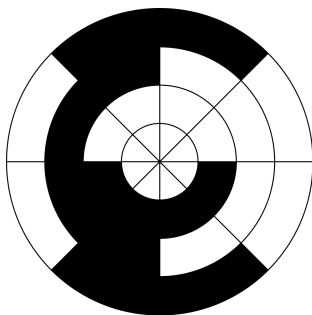


Figure 1: Absolute rotary encoder with Gray code

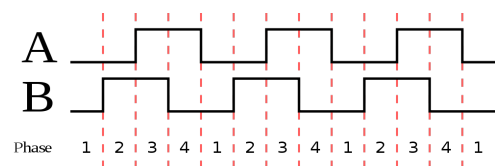


Figure 2: Signal from relative encoder, with direction information

#### Gyroscopes

Gyroscopes are devices used for orientation measurement. Thanks to the principle of angular momentum conservation the mechanical gyroscopes can also maintain orientation.

For low-cost robotic applications, electronic MEMS gyroscopes (MicroElectroMechanical systems) are good choice.

### **Accelerometers**

Accelerometers are devices measuring device acceleration in given direction. An accelerometer can measure in 1-axis, two-axis and even three-axis. Affordable accelerometers are based on MEMS technology.

### **Inertial Measurement Unit**

An inertial Measurement Unit (IMU) is essential for boats, UAV's (Unmanned Aerial Vehicles), for velocity, orientation and gravitational measurement. It consists of accelerometers and gyroscopes.

## **2.2 Sensors for Localization**

Process of determining the position of mobile robot is called localization. In outdoor robotics, task can for example be the localization in a park - given vector map of roads, robot must be able to determine its position.

### **GPS**

Global Positioning System is a space-based global navigation satellite system, that provides location and time information anywhere on (or near) the Earth [9]. The GPS is a very important device for map-based navigation, but with limited accuracy. It is therefore advantageous to fuse GPS data with data from IMU, as described in paper [10].

### **Magnetometer**

Magnetometer is a device for measurement of strength or direction of the magnetic field. In robotic, it is often used as a complement of gyros in dead reckoning systems.

## **2.3 Sensors for Close Range Perception**

### **Touch Sensors**

Simplest sensors possible can be represented for example by switches - mechanical, optical, capacitive and inductive. Imagine a blind man – he will be able to avoid obstacles, by 'touching' the ground and surrounding objects with his Blind Cane. Touch sensors in

outdoor robotics can be used to detect collision, as described in [11]. Switch positioned on front bumper will report any imminent contact with obstacle.

### Light Sensors

As small insects will react to light (fly attracted to light bulb – phenomenon called phototaxis), so robot can sense changes in light intensity. From absolute light intensity can be determined if other sensors are usable (camera in the dark won't see). Very simple photoresistor or phototransistor can be used for light source following or avoiding.

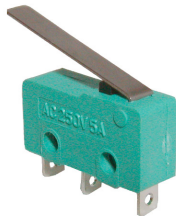


Figure 3: Example of touch sensor

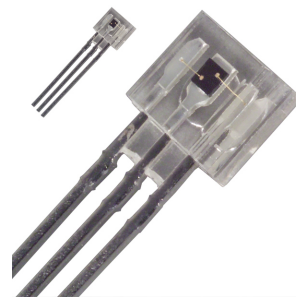


Figure 4: Phototransistor

### Distance Sensors

To know about location of obstacles without physically touching them, a distance sensor can be used. When distance to obstacles around a robot is measured continuously, it is possible to avoid them. There are many different principles of operation. Infra-Red Rangers on figure 5, manufactured by Sharp company, work on principle of triangulation – pulse of infrared light is emitted, if it hits obstacle, it is reflected and measured by the detector. Distance to obstacle than can be calculated from the position of the impact onto detector. This is illustrated on figure 9. Other principle is listening for echo of sent ultrasonic pulse (Sonar), shown on figure 7. Or Time of Flight to obstacle and back can be determined (sensors with lasers) on figure 6. Last type is phase shift measurement of returned modulated light on figure 8 that is less precise, lower cost alternative to the Time of Flight method.

### Sound Sensors

Sound sensors can be used for voice-control, and with two sensors placed on different sides of platform, source of sound can be tracked. For sound sensing, it is possible to use the electret microphone.



Figure 5: Sharp rangefinder



Figure 6: Laser rangefinder SICK LMS 100



Figure 7: Sonar SRF05



Figure 8: Hokuyo Laser rangefinder

## Image Sensors

Image sensors are devices able to convert optical image into electrical signals. They can have different resolution, depending on application and requirements. Examples are CCD image sensor and CMOS sensor in digital cameras. Images taken around robot can be interpreted to search for possible paths, to locate familiar objects, to determine speed and even to avoid collision.

### 2.4 Selected Sensors

As main sensor, camera was selected, due to its general use. Inspiration from our perception was taken – humans use mainly vision for navigation. Interpreting image into usable data for navigation is not a simple task, two approaches are evaluated in section Software 6. Low-cost camera salvaged from chassis of old Asus notebook, on figure 10 was used during all tests. Native resolution of this camera is 640 x 320 pixels, but much more suitable (less noisy) resolution is 320 x 240 pixels.

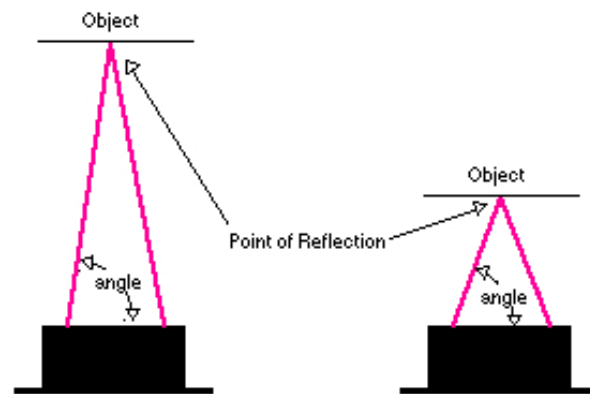


Figure 9: Triangulation principle explanation



Figure 10: Camera mounted on platform

## 3 Mechanics

### 3.1 Construction

Mobile robotic platforms can be divided into two main groups – indoor and outdoor. Although they are used in different environments, main concepts remains the same.

From the point of steering, we recognize **Ackermann Steering** – geometric arrangement of linkages in the steering of a vehicle designed to solve the problem of wheels needing to trace out circles of different radii [12]. Each turnable wheel has its own pivot, thus minimizing wheel slip. See figure 11. **Differential Drive** – if mobile robot has two separately driven wheels on both sides of body (figure 12), than turning is possible by changing the relative rate of wheels rotation. There are several advantages of the differential drive, namely easier movement calculations, simpler construction (compared to Ackermann model) and possibility of arbitrary rotation on a spot. There are also disadvantages - need for higher torque of motors due to the friction, created by rotation movement, loss of precision in odometry due to slipping sideways or lower achievable speeds.

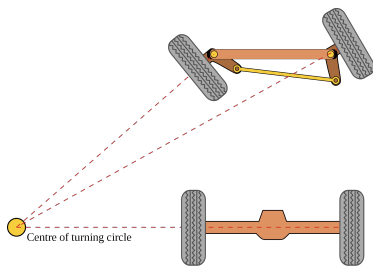


Figure 11: Ackermann steering

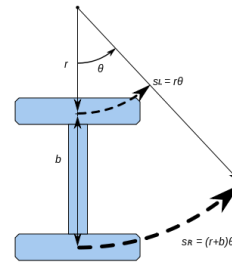


Figure 12: Differential steering

There were many outdoor platforms developed in the past, for example Pioneer 3-AT [6] which is differential type, with four wheels. This platform have been used by Czech Technical University for many years. Another example of a mobile robot is QUIDO [13], utilizing Ackermann steering.

### 3.2 Implementation

Development of the mechanics was the most challenging part. Several constructions were designed. Basic requirements are:

- Manufacturable mainly at home with basic tools,
- Big enough to carry notebook and sensors,
- Durability to withstand outdoor environment,



Figure 13: P3AT robotic platform



Figure 14: QUIDO robotic platform

- Made from widely available materials,
- Top speed at least  $3 \text{ m} \cdot \text{s}^{-1}$ ,
- Powerful engines,
- Ability to turn on spot,
- At least half hour of autonomous drive on single battery pack,
- Remote control,
- Transparent top chassis.

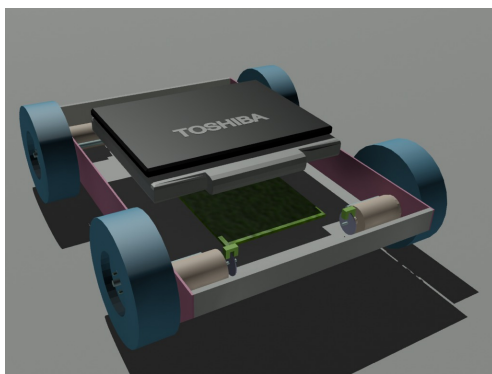


Figure 15: Concept 1

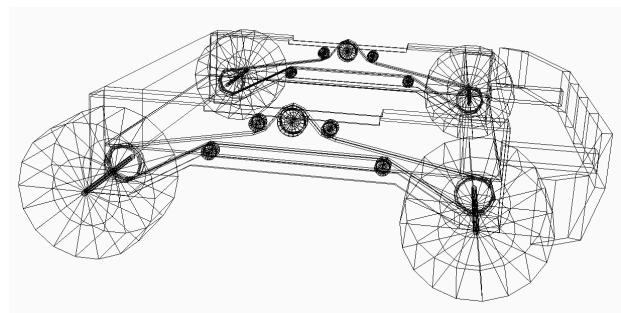


Figure 16: Concept 2

To fulfill the aforementioned requirements, main material was selected to be aluminum and Plexiglass. Main support chassis was made from two 2.5 mm thick aluminum profiles, 500 mm long, with 100x40 mm outer profile. Interconnection was made from four modular 20x20x300 mm universal construction profiles. Outer dimensions of this configuration are 500x380x100 mm. This is illustrated on figures 19 and 20



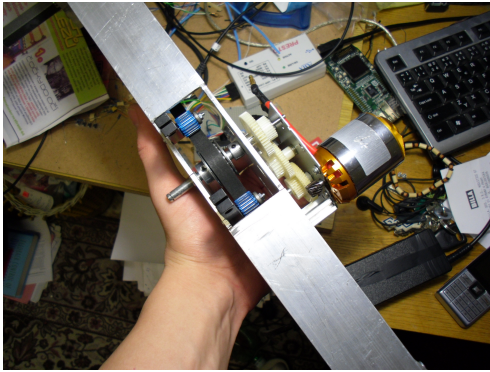


Figure 17: Common synchro belt design

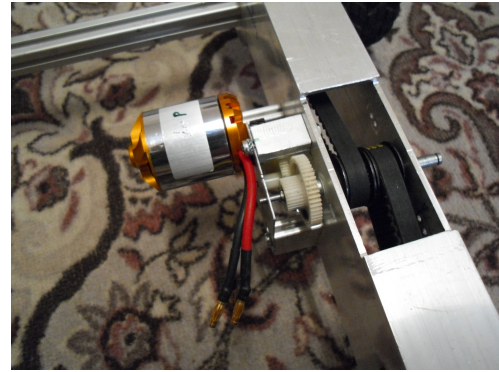


Figure 18: One synchro belt for each wheel

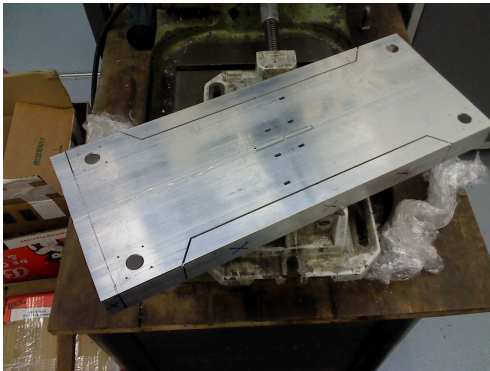


Figure 19: Two machined jackels

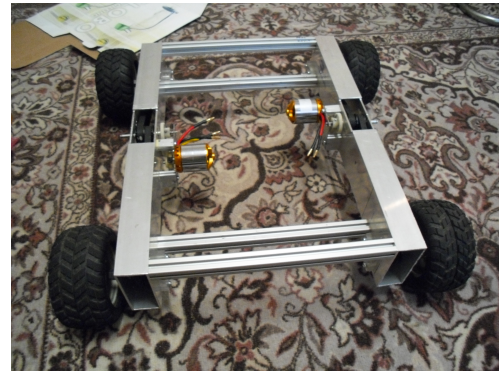


Figure 20: Assembled jackels

Wheels were originally made for 1:6 sized radio-controlled car 22. To be able to mount them, the reduction shaft 21 was created, from 14 mm hexagonal pole. To mount shaft into profiles, bearings 23 8x22x7 mm were used.

### 3.3 Motor and Gearbox

With simplicity in mind, differential steering was selected. Major disadvantage was the need the of at least two motors. First design incorporated four DC motors RS-455PA [14], with 65 W of output power, nominal voltage 42 V, max efficiency at 0.33 A, max load 1.85 A, weight 145 g, probably more suitable for indoor robotic platform up to 2-3 kg. Motors were a gift from Radio Club Pisek, salvaged from old laser printers.

Advantage of laser printer motors also was a good gearbox, originally used for drum rotation. Stripped down gears were reassembled into more suitable configuration, to create ratio of about 12:1. Unfortunately, after series of trials and development it was found, that higher torque is needed. It was decided to utilize sensorless brushless motors, due to their high efficiency, low cost and better weight-to-power ratio. Their disadvantage is the need



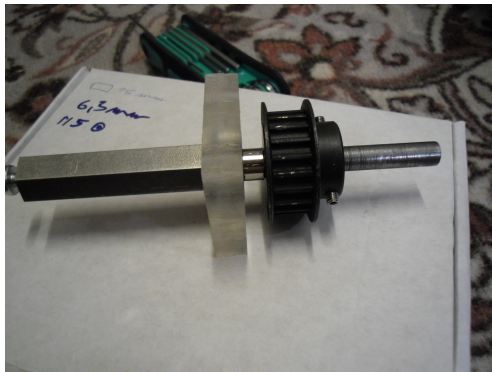


Figure 21: Reduction shaft



Figure 22: RC wheel



Figure 23: Bearing

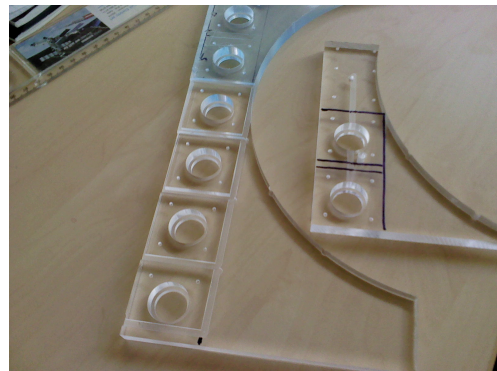


Figure 24: Bearing housing

of a sophisticated electronic speed controller. An important parameter also is the motor velocity constant, measured in RPM per Volt ( $K_V$  rating). It was desired to select as low  $K_V$  as possible, thus simplifying the gearbox, (lower speed - lower required gears ratio). Both DC and brushless motors have high no load revolutions per minute (from 4 000 to 10 000 RPM). Common robotic platform moves at speed varying from tenths, up to tens meters per second. It is therefore essential to decrease speed and increase torque. Two Brushless sensorless DC motors were selected. They are outrunner type (external rotor) TR 42-60C [15], 500  $K_V$ , output power 900 W, max efficiency at 38 A, max load 50 A, weight 250 g. These motors are usually used in the biggest RC airplane models. ‘No load’ current is about 4.3 A. Recommended electronic speed controller rating is 60 A.

Electronic speed controller (ESC) drives the motor – directs the rotor rotation, based on rotor orientation. Current rotor position can be estimated either from Hall effect sensors, rotary controller, or by measuring back EMF [16] (back electromotive force). Described motor is a sensorless type, so only method remaining, is a measurement of back EMF, which is done by sensing voltage on momentarily non-driven coils to infer rotor position. A challenging task is to initiate rotation, because without rotation, there is no back EMF. One solution is to incorporate a predefined starting sequence to give motor enough spin to

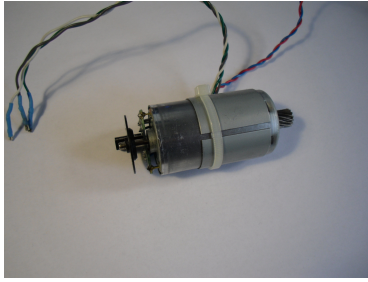


Figure 25: DC motor salvaged from printer



Figure 26: Brushless sensorless DC motor

initiate correct sensing. This creates a minor drawback, because lowest stable achievable speed is in orders of hundreds RPM. Even with a high ratio gearbox, the minimal speed of developed platform is around  $0.1 \text{ m}\cdot\text{s}^{-1}$ . This may be a too high in the indoor, but is sufficient in outdoor robotics.

## 4 Electronics

### 4.1 Overall

Electronics consists of several interconnected modules, a battery pack and mounted netbook. Completed robotic platform with some of the electronics is displayed on following figures:

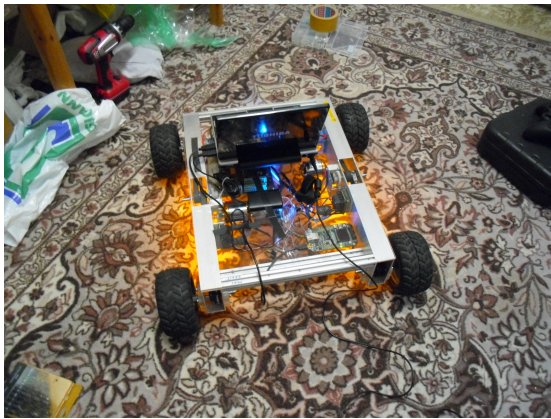


Figure 27: Overall design with LED neon

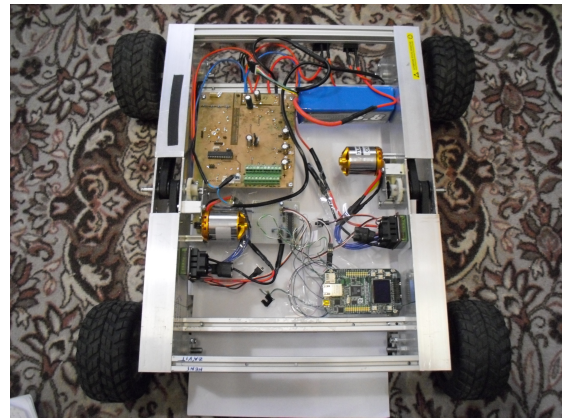


Figure 28: Development version

### 4.2 Battery Type Selection

There are many different widely available rechargeable battery types on the market, namely:

- Lead-acid (Pb)
- Nickel-cadmium (NiCd)
- Nickel-metal hydride (NiMH)
- Lithium-ion (Li-ion)
- Lithium-polymer (Li-Po)
- Lithium iron phosphate (LiFePo4)

#### Lead-acid

Lead battery is the oldest type of a rechargeable battery [17], known for its low energy-to-weight ratio, as well as high surge currents. Main advantage of this type is its low cost. Discharge characteristics are on figure 29, taken from [18].

### **Nickel-cadmium**

NiCd batteries are mainly used in RC cars and wireless telephones. Because of their low internal resistance, high current surges are possible. Their nominal voltage is 1.2 V [19]. It is common to wire more batteries in series to create higher voltages. Cadmium used inside is highly toxic and is hazardous for environment. This battery type suffers a memory effect problem (degradation over time when not used properly) [20].

### **Nickel-metal hydride**

NiMH is a similar technology to the NiCd, but NiMH battery uses a hydrogen-absorbing alloy for the negative electrode, instead of cadmium [21]. Advantage is a higher energy density, but for a price of lower surge currents. Common capacity is around 2500 mA·h. As well as Nickel-cadmium type, the NiMH suffers memory effect.

### **Lithium-ion**

Li-ion is a very power-efficient battery when compared to the previous battery types. Lithium ions move from the negative electrode to the positive electrode during discharge, and back when charging [22]. Li-ion battery is common in portable electronics and consumer electronics in general. Its main advantage is low self-discharge rate, but suffers low current rating. Can be charged from any state (no memory effect), but cannot be deeply discharged. When several cells are wired in series to increase voltage (each cell have 3.7 V when charged), care must be taken. Li-ion is very sensitive to overcharge, as well as undercharge, so cell balancing is essential to prolong the battery pack life [23]. Discharge characteristics are on figure 30.

### **Lithium-ion polymer**

Li-Po is a similar technology to the Li-ion. But Li-Po is logical successor, by having both lower weight as well as higher discharge current. When more cells are used in series, also balancing is recommended. This technology is preferred for hybrid electric vehicles [24]. Can supply very high currents but when not handled carefully, the battery might explode. Price of Li-Po battery is high, when compared to other types, but is dropping constantly.

### **Lithium iron phosphate**

Lithium iron phosphate battery have  $\text{LiFePO}_4$  as a cathode material. They have superior thermal and chemical stability over standard lithium-ion batteries, are less combusive, but have a lower power density [25].

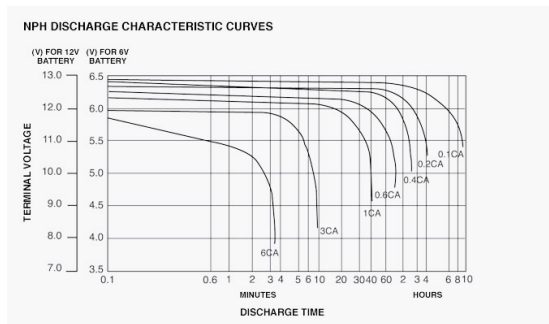


Figure 29: Lead battery discharge curve

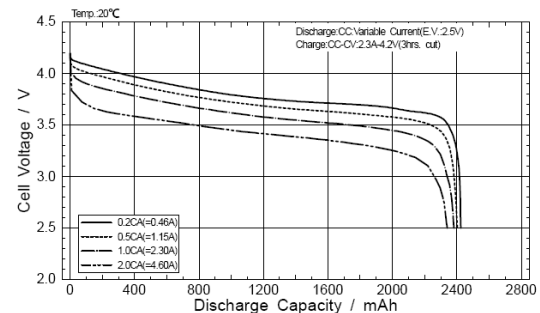


Figure 30: Li-ion battery discharge curve

## Selected Battery Technology

Robotic platform sets several limitation on what battery can be used, main criteria were voltage from 12 to 17 V, weight up to 3 kg, limited outer dimensions (defined in chapter 3.2), high capacity (at least 5 Ah) and low cost

Two battery types were selected as candidates – **Lead-acid** 12.0 V, 7.0 Ah, weight 2.25 kg, dimensions 151x65x95 mm, **Li-Po** 14.8 V, 5.8 Ah, weight 0.62 kg, dimensions 150x49x40 mm.

Both types are said to have almost same amount of stored energy, Lead type costs \$19 and Li-Pol \$51. Li-Po pack weights 3.6 times less and takes 3 times less space than Lead type, but costs 2.7 times more. But main difference is in amount of current drawn from battery – if Lead type is discharged at 4.2 A rate, it will have only 4.2 Ah of capacity [26], instead of declared 7 Ah! While Li-Po capacity is defined at 145 A constant discharge rate [27]. Thus knowing, that robotic platform can have constant current draw of 10 A, lead battery would be a very bad choice – it would act as if the capacity is somewhere around 2 Ah. Discharge curves for both battery types are on figures 30 and 29.

So Lithium polymer technology was selected. Two Turnigy 5800 mAh battery packs with 4 cells in series safe discharge current of 145 A were bought from China supplier, for \$60 including VAT. This battery back is on figure 31.

## 4.3 Modules Summary

The electronic system is decomposed into modules, with hierarchy shown on figure 32. Highest (TOP) modules are running on computer with OS Linux or OS Windows. Main purpose of the computer is strategy coordination and processing of video stream. Control Board utilizes SPI bus for subsystems and USB 2.0 for interconnection with computer. Peripheral modules are for example motor driver and LiPol monitor. Intelligent sensors are special group – they can break system hierarchy. Sensors can contain cameras, possibly rangefinders, touch sensors and even emergency button.





Figure 31: LiPol battery with balancer plug

## 4.4 Top Module

The top module performs all high level tasks as navigation, decision making, image processing or user interaction). Every Top module is truly platform independent, capable of running both on Linux and Windows operational system. Graphical interface, as well as decision making, wireless control and other less time critical modules are written in language Java. Computational intense modules like image acquisition and analysis, are written in C++. For example image acquisition as well as processing and analysis are such tasks. The USB and remote control drivers are incorporated into main Java application, called RoboControl. Purpose of the RoboControl also is to run and interpret hot-pluggable modules like image processing. This configuration ensures, that even in case of system failure (broken camera, corrupted images or total module crash) can RoboControl can simply unload any module and rerun it.

## 4.5 Control Board

The Control Board, which is based on evaluation kit LM3S6965, is on figure 33. It acts as a bridge between lower-level systems and Top modules. Consists of 32b Texas instruments microcontroller Cortex M3, an USB slave, a 100Mbps Ethernet and SPI Master bus. The purpose of this system is reporting status of robotic platform to modules, interpreting commands from Top modules, as well as forwarding them to appropriate peripheral module on SPI bus. Firmware is written in language C (event based, no RTOS capabilities), compiled with IAR Embedded Workbench for ARM.

Main parameters are: LM3S6965 microcontroller with fully integrated Ethernet controller, OLED graphics display with 128 x 64 pixel resolution, User LED, switches, Magnetic speaker, MicroSD card slot, JTAG debugger, SPI, I2C and USB connector.

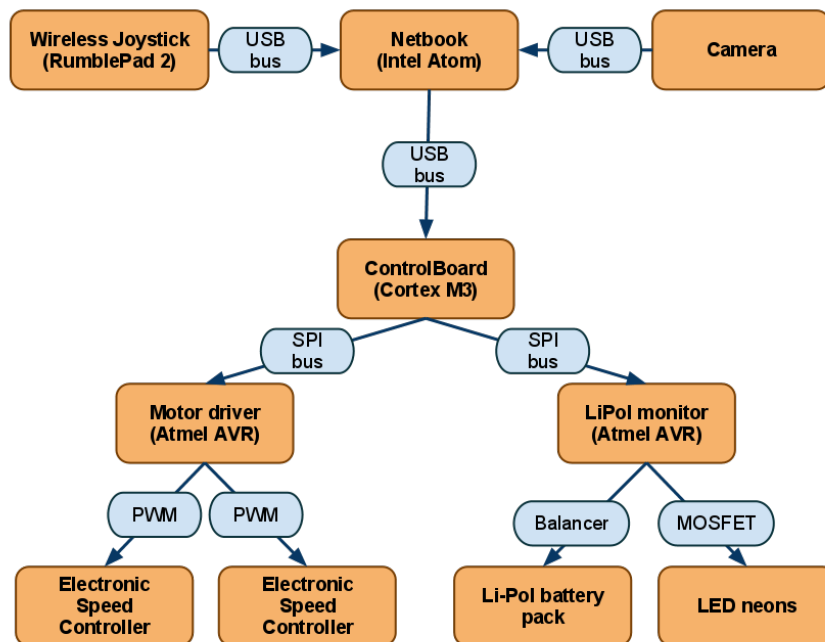


Figure 32: Simplified diagram of modules

## 4.6 Peripherals

Two peripherals were developed so far – Li-Po Monitor and Motor driver. Both contain 8b AVR ATmega8 microcontroller clocked at 8 MHz and SPI bus for interconnection.

## 4.7 Li-Po Monitor

The monitor is classified as a Peripheral. To enable monitoring the battery and balancing of the pack, the system must be able to measure voltages of all cells in battery. Since one 14.8 V pack contains four single cells (each at 3.7 V), four differential analog to digital converters are needed. Pack voltage monitoring is not necessary, because it can be easily calculated as sum of voltages of each cell, but adding this redundancy opens the new ways of error detecting (broken wires from balancer for example).

### Board Function

Required functionality:

- Measure voltage of each cell,



Figure 33: Control Board is based on LM3S6965.

- Measure current flowing from battery pack,
- Regulate unstable 14.8 V input into 12 V 1 A, 5 V 1 A and 3.3 V 1 A,
- Standard SPI (Serial Peripheral Interface) communication bus,
- Charger connector,
- Voltage terminals with stable voltage for powering external boards,
- Balancing circuit,
- High current solder pads for battery connectors.

## Monitor

Most important part of the Li-Po management is an integrated circuit bq76PL536 from Texas Instruments [28]. Several features of this solution are 3 to 6 series cell support, built-in comparators (secondary protection) programmable thresholds and delay times, over-voltage and under-voltage fault signals, cell balancing control outputs, temperature sensors, SPI slave interface (managed by on-board microprocessor, separated from global robotic SPI bus). This integrated circuit is a part of Texas Instruments new line of products targeted on hybrid cars, electronic bikes and scooters. It is not yet commercially available, samples were courtesy of Texas Instruments.



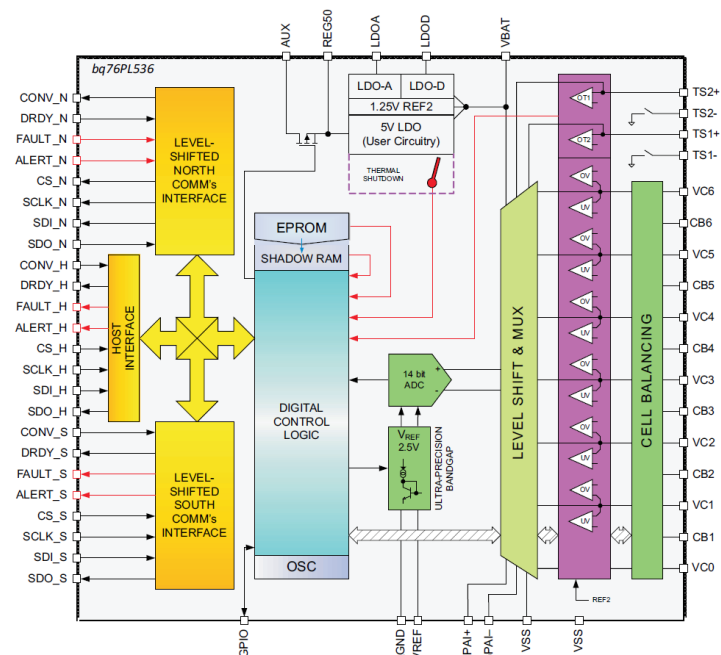


Figure 34: BQ76PL536 integrated circuit block diagram

## DC/DC Converters

To enable future development of peripherals and sensors, expansion power supply modules were created. They consist of one standard linear LM7812 12 V output regulator, three 1 A step-down voltage regulators LM2595 [29] – two with 5 V fixed output voltage, one with 3.3 V output. These regulators are power supply for the on-board systems as well as for the external systems connected to the expandable connector with screw terminals.

## Schematics and Layout

Schematic drawings as well as printed board layout were developed in OrCAD package and can be found on enclosed CD.

## Printed Board Manufacturing

Most of the today's boards are made using SMT (Surface Mount Technology). Because integrated circuit bq76PL536 is in PQFP-64 package (0.27 mm wide pins, 0.5 mm spacing) printed circuit board had to be very precise. It was decided to manufacture it at home, with only basic tools. Photo-resist is not available everywhere, and requires a lot of chemical components as well as special equipment like an UV lamp. Proposed manufacturing method utilizes black toner from laser printer, to act as a mask for etching. The idea is simple –

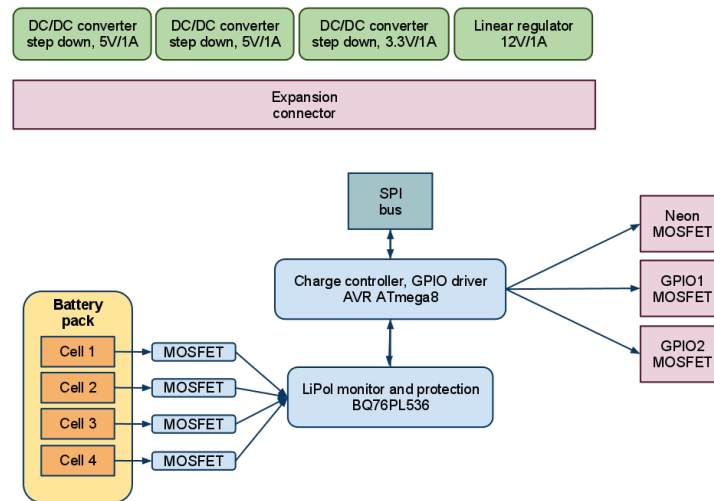


Figure 35: Block diagram of LiPoL monitor board

a toner is very resistant to copper etching solutions, so it has to be transferred onto the board to protect areas of future traces.

Simplified guide:

- Prepare new printed circuit board,
- Remove oxides from copper layer, using some cleaning chemical,
- Print mirrored board layout from CAD software onto papers with dry glue (colored papers for children),
- Put newly printed paper onto clean board – printed side to copper (must be printed on laser printer!),
- Use household clothes iron to heat PCB with paper on it, apply mild pressure,
- Put ironed board into water, let the glue dissolve, then remove paper 36,
- Etch board in ferric chloride solution,
- Remove residue toner with acetone 37,
- Drill board using small diameter drill,
- Create vias (in case of two side board) with thin wire (like sewing all together), then solder.

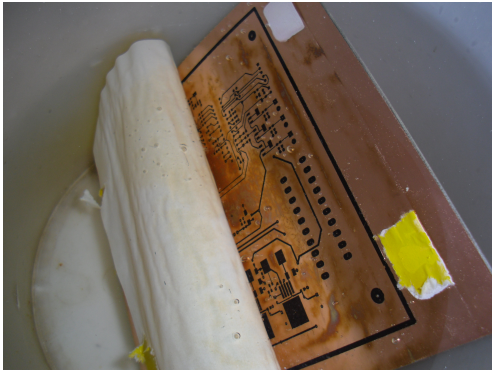


Figure 36: Removing paper from PCB

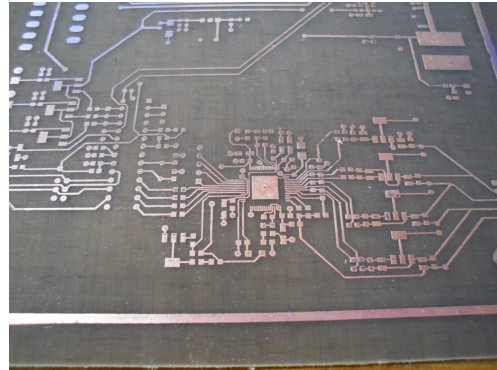


Figure 37: Etched and cleaned board

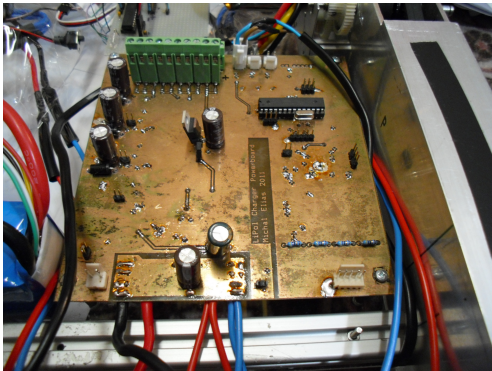


Figure 38: Finished charger top

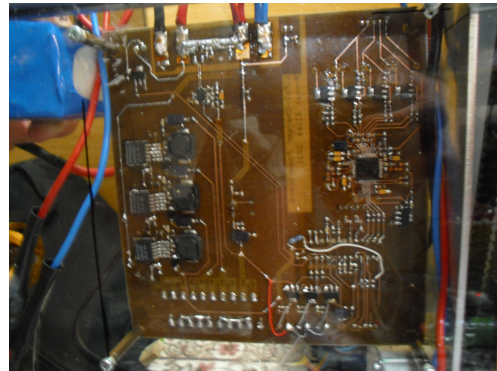


Figure 39: Finished charger bottom

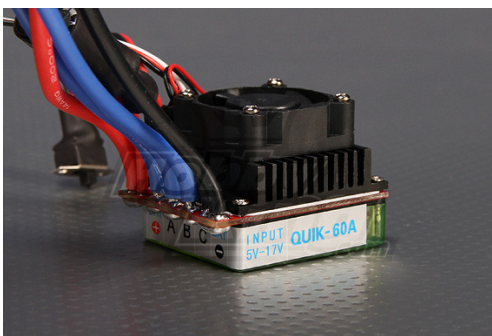


Figure 40: Electronic speed controller

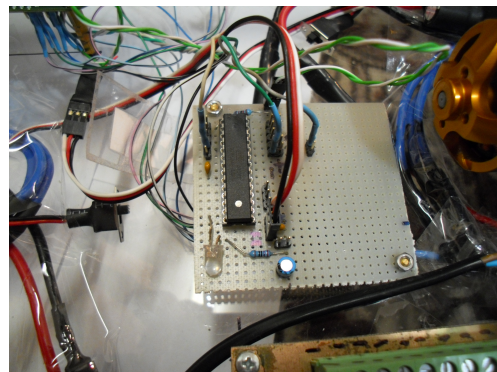


Figure 41: Soldered Motor driver board

## 4.8 Motor Driver

Brushless electronic speed controllers (ESC) are consumer-grade, made originally for motors in RC car models. They require standard input signals used for example to drive servos in hobby products. Because of their low price, there is no advantage in building them

(raw materials cost more than purchased controllers) [30]. The parameters of the purchased ESC on figure 40 are: input voltage 5 to 17 V, continuous current 60 A, low voltage battery protection, throttle signal lost detection and overheat protection. Continuous 50 Hz signal have to be provided, with 0.5 ms (maximum reverse speed) to 2.5 ms (max forward speed) wide pulse (1.5 ms is a neutral). This signal is synthesized by the Motor driver peripheral. The speed is determined from packet sent from Control board. Physical wiring is done on universal drilled solder board, as shown on figure 41.

## 5 Firmware

### 5.1 Code Reusability

Lots of people understand the need for the clean code and coding standards. But code reusability is often somehow underestimated. Example of reusable code in this project are protocol routines – the same packet style communication exists on every type of the Peripheral board, and even on the Control board. It is important to keep enumerations of the robot packet types and data sizes consistent through every platform and device. Also when user wants to incorporate new peripheral type into the platform, new packet type and data sections have to be defined – this simply cannot be done in every copy of source code. These rules come from many mistakes done in past projects by trial and error. Several things should be done to ensure that C/C++ code can be reused again, even on different architectures:

- Never use system specific code (for example registers in microcontroller),
- Avoid using uncertain data types – integer can be 32b wide on i386, 64b on x64 platform),
- Define types describing size in their names – uint8, uint16, uint32, int8,
- Don't use system calls, if needed, use pointers on functions,
- Never copy shared code – use references,
- Constants used in code should use #define macro, avoid use of 'magic constants' like 666 directly in functions.

### 5.2 Protocol Communication

Every streamed type of bus should have a well-defined data format. Common approach to sending commands over asynchronous data link is text oriented or byte oriented, that means, that command have either predefined length (byte oriented) or predefined array of characters (ASCII formatted, terminated by end of line). In theory, everything work as planned – when command is known to be six bytes long, then six bytes are parsed and interpreted. When text oriented command is 'speed=10', terminated by end of line, it is again parsed very well by receiving side. Both approaches are sensitive to byte corruption – speed=10 can easily end up as speed=90. This vulnerability is limited by use of cyclic redundancy check [31]. Idea is to send not only usable information/commands, but also CRC calculated over the entire message, almost eliminating the possibility of misinterpreted message.

But what if one character is lost or corrupted due to noise on signal path? Byte oriented approach will interpret first five bytes of command, but instead of sixth lost byte, interpreted is first byte of next command, and so on. Situation is improved a little in case of text oriented type – when one character is lost, than command is not recognized and safely discarded. But when termination character is lost, two messages are mixed together, possibly resulting into two commands lost or misinterpreted.

All previously mentioned problems can be solved by HDLC [32] (High-Level Data Link Control). Main idea behind HDLC is to bond data into frames with specific identification for beginning and end of each frame. This identification can never appear inside of any frame, because it will always mean beginning or end of frame. Any number from 0 to 255 (one byte) can represent this frame delimiter. In this example, it will be a number 0x7E. When 0x7E is what should be transmitted as data, another specific byte is sent – 0x7D, followed by originally desired byte (0x7E) xored by number 0x20. Also when data contains 0x7D, first is sent 0x7D, followed by 0x7D xored by 0x20. This may seem rather complex on first look, but is actually quite simple in the code.

Advantages of this approach are that start and end of frame are always recognizable even when several frames are corrupted. Furthermore the misalignment of data observed in byte oriented data link cannot happen. With header defined, frame will become a packet, which can have different recipients, variable data length. When CRC is computed over entire packet and sent with data, frame can always be interpreted correctly (or thrown when corrupted). An error in any byte inside of a frame will always cause only one frame to be lost (the defective one)

Only disadvantage over byte oriented data link is some data overhead – extra 2 bytes to delimiter frame, extra 1 byte for each control character appearing in data (bytes 0x7E and 0x7D) and 2 more bytes when 16 b checksum is used. So to transfer 20 bytes of data, about 25 bytes are used.

### 5.3 Control Board

As already mentioned in section 4.5, the Control Board is based on a 32 b microcontroller Cortex M3. There was a possibility to use RTOS (Real Time Operational System) like uClinux [33], and gain advantages like threading and already defined peripheral drivers. However there are several disadvantages of RTOS – steep learning curve that can affect global project time schedule and higher memory requirements. It was decided to write firmware for simplicity in language ANSI C. IAR Embedded Workbench is a professional compiler and debugger, supported by many major companies developing microcontrollers. It can be used in evaluation mode for free, if the resulting code size is smaller than 32 kB. Milestones in development of firmware were – installation of all tools (compiler, programmer drivers, support software), learning Cortex M3 architecture from datasheet [34], learning how different peripherals of the microcontroller should be configured, debugger

configuration, writing ‘hello world’ application, development of routines for peripheral configurations (SPI driver, USB driver, ..), implementation of packet generator, interpreter and packet router.

Code fragment of peripheral initialization:

```
int main() {
    // Initialize peripherals:
    // Prepare processor
    McuInit();
    // Configure OLED display
    OledInit();
    // Turn display off for now
    RIT128x96x4DisplayOff();
    // Init UART to USB bridge
    UartInit();
    // Configure debugging LED
    LedInit();
    // Timers to be used for task scheduling
    TimerInit();
    // Serial Peripheral Interface bus
    SpiInit();
    // Packet buffer for incoming data cache
    BufferInit();
    // Enable processor interrupts
    IntMasterEnable();
}
```

Listing 1: Initialization of Control Board

Main routing mechanism periodically checks if new packet is received. If new data are found, packet type is determined, and an appropriate reaction is performed. Simplified sorting can be observed on following fragment:

```
// Check data in incoming USB buffer
if (PacketAvaible(&inPacket)) {
    // New packet received, parse type
    switch ((enum packetType) inPacket.type) {
    case ping: {
        packet.counter++;
        // Ping type – respond with Pong
        packet.type = pong;
        packet.size = 0;
        // Calculate CRC for outgoing packet
        packet.crc = CrcCalc(&packet);
        // Mark packet as ready for sending
        SendPacket(UartSendChar, &packet);
        //LedToggle();
        break;
    }
    case setSpeed: {
        // Computer wants to set speed of motors
        packet.counter++;
    }
    }
}
```

```
    packet.type = setSpeed;
    packet.size = 2;
    // Create new packet for SPI devices, copy data bytes
    packet.dataSection[0] = inPacket.dataSection[0];
    packet.dataSection[1] = inPacket.dataSection[1];
    // Calculate CRC for outgoing packet
    packet.crc = CrcCalc(&packet);
    // Mark packet for SPI transfer
    SendPacket(SpiSendChar, &packet);
    // Signal by LED
    LedToggle();
    break;
}
}
```

Listing 2: Router Mechanism

Two packet types are shown – Ping and SetSpeed. Reaction to Ping is simple, only thing that needs to be done is to respond with Pong packet back to computer. Situation is more difficult in case of SetSpeed packet type – because speed of robotic platform is maintained by Motor driver peripheral, packet has to be forwarded by the SPI bus.

## 5.4 Motor Driver

Consists of one microcontroller ATmega8. Main features of this MCU, as described in [35] are: advanced RISC Architecture, 8 KB of flash memory, 512 B of EEPROM memory, 1 KB of internal SRAM memory, two 8 bit Timer/Counters, one 16 bit Timer/Counter, three PWM channels, 6 analog to digital converters, programmable USART, Master/Slave SPI interface.

Communication with the Control board is SPI-based. Originally developed discrete PID regulator is not used for now, will be part of second revision. Firmware for the Motor Driver is written in language ANSI C, and compiled by AVR-GCC compiler.



## 6 Software

### 6.1 RoboControl Software

As mentioned in chapter 4, RoboControl is an application written in language Java. Its purpose is to communicate with the robotic platform over the USB interface. Several platform-independent libraries are used, for example RXTX serial port (COMM) library and f joystick library (Logitech RumblePad 2 Java driver).

RoboControl software can be run on Microsoft Windows XP, Vista and 7, GNU Linux – virtually any distribution capable of running SDL (Simple DirectMedia Layer) and JRE (Java Runtime Environment), both 32 bit and 64 bit platforms. Functionality was tested on Linux Fedora 12 native 64 bit, Windows XP and 64 bit version of Windows 7 (without image processing module). On every tested operational system, it was possible to establish connection to robotic platform over the USB interface, run joystick driver and control mobile robot movement. Image acquisition and processing software was tested only on Linux, but should be working on Microsoft Windows as well, if recompiled.

### 6.2 Image Processing

Development of Image processing software, based on OpenCV [36] (Open Computer Vision) started with idea to create complete robotic control environment, but eventually was profiled towards implementation of experimental reactive navigation algorithms. It is written in C++ language and at this moment contains implementation of two algorithms – Vanishing point detection and Watershed image segmentation.

#### Vanishing Point Strategy

A vanishing point is a point in a image (captured by a perspective camera) to which parallel lines appear to converge [37]. Goal of this strategy is to find both sides of the road, approximate them with line and calculate their intersection, so called Vanishing point. If correctly found, following this point will help the mobile robot to stay in the middle of the road. This was proved in paper [38].

#### Watershed Strategy

Watershed is an algorithm transforming pixel intensity into height of the relief, which is then flooded by water from several preselected regions (defined by mask). The places where different water sources meet, constitute results of watershed. Watershed implementation in OpenCV, which is used in our robot, is based on [39]. How custom predefined mask looks like is visible on images in the next section.

Package	Class name	description
	Main	Runs Graphical User Interface
	Processor	Listening for events from modules, main decision class
	TimerFace	Provides timer routines for Processor class
.comm	CommIface	Interface for different ways of communication
	OnPacketReceive	Interface for listeners of arrived packets
	OnRawDataReceive	Interface for listeners of raw packet data
	SerialPortDriver	Encapsulates library for COMM port transmissions
	SerialReader	Listens for new byte event on COMM port
	SerialWriter	Packs raw packet data into HDLC protocol
	Udp	Unfinished class for Ethernet-based communication
.gui	Gui	User interface, robot status display
.joystick	RumblePadDriver	Wireless controller driver
	Axis	Joystick axis state class
	Buttons	Joystick buttons state class
	POV	Joystick POV (Hat switch) state class
	OnJoystickEvent	Interface for listeners of joystick event
.packets	Packet	Abstract class, set of methods for packet assembly
	InPacket	Received packet without specific type
	Ping	Service packet type
	Pong	Service packet type
	SetSpeed	Packet type, sets speed of robot
	Types	Packet types enumerator, defines data sections size
.external	Camera	Communication with external image processing software

Table 1: RoboControl classes hierarchy

### 6.2.1 Strategy Results

To test the developed algorithms on as many images as possible, a dataset was provided by thesis supervisor captured on Pioneer 3AT robotic platform, during tests of Monocular Navigation System [40]. Dataset is named robotour2008 and contains 661 images, from Stromovka park in Prague. For each image, Watershed and Vanishing point strategies were applied. All resulting images with additional imprinted classification information are stored on the attached CD.

For outdoor tests and a contest mentioned in next section Experiments 7, the Watershed strategy was selected. According to test results, it is more robust, doesn't depend on sharpness of the image and has better results in terms of road detection. Examples are on the following figures 44, 45, 46, and 47. Vanishing point detection is on the left side, Watershed on the right.

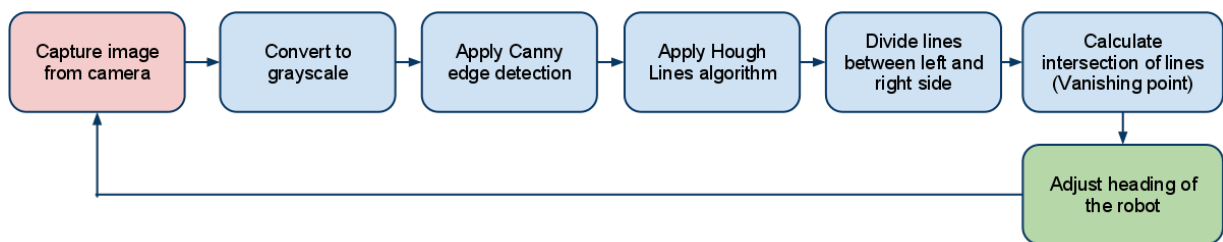


Figure 42: Vanishing point strategy

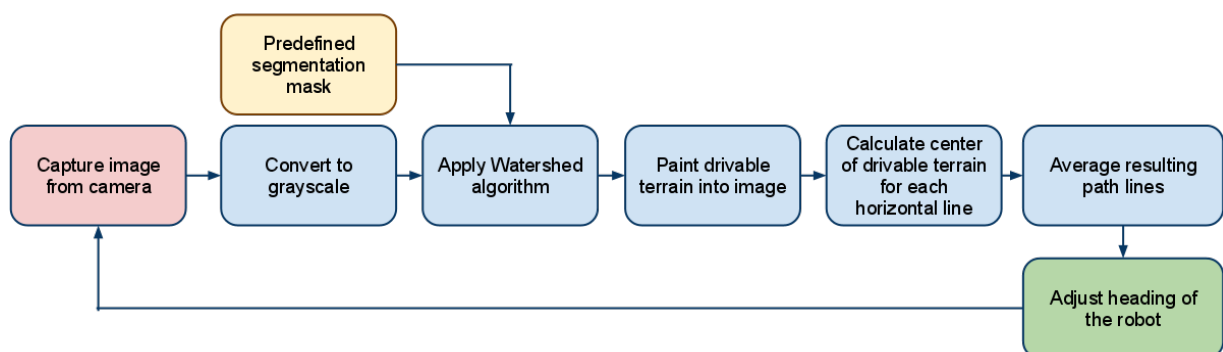


Figure 43: Watershed strategy



Figure 44: Well detected road edges

Figure 45: One edge almost out of image



Figure 46: Both algorithms failing

Figure 47: Correct classification

## 7 Experiments

### Prague Experiments

First autonomous experiments were made in Prague urban area, on an approximately 300 meters long sidewalk, shown on figure 48.

The goal was to stay on the sidewalk without manual interventions into steering. When collision with objects or grass was imminent, emergency button on wireless controller was pressed, thus ending the trial. Speed of the platform was set around  $5 \text{ km}\cdot\text{h}^{-1}$ . Length of the traveled path was estimated from location on the map.

Id	Traveled distance [m]
1	50
2	200
3	140
4	260

Table 2: Prague Experiments

### Robotem Rovně Contest

To test the developed robotic platform in challenging outdoor conditions, we participated on ‘Robotem Rovně’ contest (translated as Robot, go straight!), held every year in Písek, Czech Republic. Goal of this event is to autonomously pass 314 meters long path in local park ‘Palackého sady’, visible on map 49. Each robot had 3 attempts, each could be ended by collision, contact with grass, or wrong turning on crossroad. Totally about 20 teams had participated, not only from Czech republic but also from Slovakia and Germany.



Figure 48: Prague traveled path



Figure 49: Official competition route



Figure 50: Final preparations



Figure 51: Autonomous drive test

The traveled distance was measured by official judges. Results are in the following table 3. Results are worse than the ones of Prague tests, due to many visitors, blocking the path.

Id	Traveled distance [m]
1	29
2	114
3	115

Table 3: ‘Robotem Rovně’ results

## 8 Conclusion

The main goal, development of the outdoor robotic platform, wasn't an easy task. Many problems emerged during every stage of this work. Hardest part was probably mechanics, due to my limited knowledge of mechanical engineering. But with every resolved problem, I learned a lot of new things, and was able to solve later issues more efficiently. Working on complex task is a very good experience, because when something is not done properly, it will affect platform as whole. It is better to focus on things that can be done in time, instead of chasing after perfect solution and fail to finalize more important parts.

Although developed platform does not carry every possible sensor, it was observed that it can still perform better than some less robust, but more equipped solutions presented during outdoor competition. There are still a lot of things that could be improved, and future research will focus mainly on obstacle avoidance and transition from reactive navigation to advanced navigation strategies and Simultaneous Localization and mapping.

## References

- [1] MFF UK. Indoor robotic competition.  
<http://www.eurobot.cz/>.
- [2] Robotika. Outdoor robotic competition.  
<http://robotika.cz/competitions/robotour/cs>.
- [3] Radioklub Pisek. Outdoor robotic competition.  
<http://www.kufr.cz/index.php>.
- [4] Robotika.cz. Outdoor robotic competition.  
<http://robotika.cz/competitions/roboorienteeing/en>.
- [5] Monika Svědřihová. Mobile robot brimstone.  
<http://robotika.cz/articles/brimstone/en>.
- [6] Adept mobilerobots. Pioneer 3-at robotic platform.  
<http://www.mobilerobots.com/researchrobots/researchrobots/p3at.aspx>.
- [7] Wikipedia. Dead reckoning.  
[http://en.wikipedia.org/wiki/Dead\\_reckoning](http://en.wikipedia.org/wiki/Dead_reckoning).
- [8] Johann Borenstein and L. Feng. Umbmark – a method for measuring, comparing, and correcting dead-reckoning errors in mobile robots, 1994.
- [9] Wikipedia. Global positioning system.  
<http://en.wikipedia.org/wiki/GPS>.
- [10] S. Sukkarieh, E.M. Nebot, and H.F. Durrant-Whyte. A high integrity imu/gps navigation loop for autonomous land vehicle applications. *Robotics and Automation, IEEE Transactions on*, 15(3):572 –578, jun 1999.
- [11] V.J. Lumelsky and T. Skewis. Incorporating range sensing in the robot navigation function. *Systems, Man and Cybernetics, IEEE Transactions on*, 20(5):1058 –1069, sep/oct 1990.
- [12] Wikipedia. Ackermann steering geometry.  
[http://en.wikipedia.org/wiki/Ackermann\\_steering\\_geometry](http://en.wikipedia.org/wiki/Ackermann_steering_geometry).
- [13] Najvárek Jan et al. Roboauto, robot quido.  
<http://www.roboauto.cz/foswiki/bin/view/Projekt/0ProjektuRoboAuto>.
- [14] Mabuchi. Rs-455pa printer dc motor.  
[http://www.mabuchi-motor.co.jp/cgi-bin/catalog/e\\_catalog.cgi?CAT\\_ID=rs\\_455pa](http://www.mabuchi-motor.co.jp/cgi-bin/catalog/e_catalog.cgi?CAT_ID=rs_455pa).



- [15] HobbyKing.com. Brushless turnigy outrunner motor.  
[http://www.hobbyking.com/hobbyking/store/uh\\_viewitem.asp?idproduct=6231](http://www.hobbyking.com/hobbyking/store/uh_viewitem.asp?idproduct=6231).
- [16] Acroname. Back-emf sensing.  
<http://www.acroname.com/robotics/info/articles/back-emf/back-emf.html>.
- [17] Wikipedia. Lead-acid battery.  
[http://en.wikipedia.org/wiki/Lead-acid\\_battery](http://en.wikipedia.org/wiki/Lead-acid_battery).
- [18] YUSA. Lead battery.  
[http://www.yuasaeurope.com/eu/industrial/products/en\\_enl/](http://www.yuasaeurope.com/eu/industrial/products/en_enl/).
- [19] Wikipedia. Nickel-cadmium battery.  
[http://en.wikipedia.org/wiki/Nickel-cadmium\\_battery](http://en.wikipedia.org/wiki/Nickel-cadmium_battery).
- [20] Wikipedia. Memory battery effect.  
[http://en.wikipedia.org/wiki/Memory\\_effect](http://en.wikipedia.org/wiki/Memory_effect).
- [21] Wikipedia. Nickel-metal hydride battery.  
[http://en.wikipedia.org/wiki/Nickel\\_metal\\_hydride\\_battery](http://en.wikipedia.org/wiki/Nickel_metal_hydride_battery).
- [22] Wikipedia. Lithium-ion battery.  
[http://en.wikipedia.org/wiki/Lithium-ion\\_battery](http://en.wikipedia.org/wiki/Lithium-ion_battery).
- [23] Wikipedia. Cell balancing.  
[http://en.wikipedia.org/wiki/Battery\\_\(electricity\)#Prolonging\\_life\\_in\\_multiple\\_cells\\_through\\_cell\\_balancing](http://en.wikipedia.org/wiki/Battery_(electricity)#Prolonging_life_in_multiple_cells_through_cell_balancing).
- [24] Wikipedia. Lithium-ion polymer battery.  
[http://en.wikipedia.org/wiki/Lithium\\_ion\\_polymer\\_battery](http://en.wikipedia.org/wiki/Lithium_ion_polymer_battery).
- [25] Wikipedia. Lithium iron phosphate battery.  
[http://en.wikipedia.org/wiki/Lithium\\_iron\\_phosphate\\_battery](http://en.wikipedia.org/wiki/Lithium_iron_phosphate_battery).
- [26] Shimastu. Lead-acid battery, 12 v, 7 ah.  
[http://www.gme.cz/\\_dokumentace/dokumenty/540/540-304/dsh.540-304.1.pdf](http://www.gme.cz/_dokumentace/dokumenty/540/540-304/dsh.540-304.1.pdf).
- [27] HobbyKing Turnigy. Li-pol 14.8 v, 5.8 ah battery.  
[http://www.hobbyking.com/hobbycity/store/uh\\_viewItem.asp?idProduct=14482](http://www.hobbyking.com/hobbycity/store/uh_viewItem.asp?idProduct=14482).
- [28] Wikipedia. Lithium-ion battery monitor.  
<http://focus.ti.com/docs/prod/folders/print/bq76pl536.html>.
- [29] National Semiconductor. Switching power supply.  
<http://www.national.com/mpf/LM/LM2595.html#Overview>.



- [30] HobbyKing.com. Brushless car esc.  
[http://www.hobbyking.com/hobbyking/store/uh\\_viewItem.asp?idProduct=11743](http://www.hobbyking.com/hobbyking/store/uh_viewItem.asp?idProduct=11743).
- [31] Wikipedia. Cyclic redundancy check.  
[http://en.wikipedia.org/wiki/Cyclic\\_redundancy\\_check](http://en.wikipedia.org/wiki/Cyclic_redundancy_check).
- [32] Wikipedia. High-level data link control.  
[http://en.wikipedia.org/wiki/High-Level\\_Data\\_Link\\_Control](http://en.wikipedia.org/wiki/High-Level_Data_Link_Control).
- [33] Arcturus Networks Inc. uclinux.  
<http://www.uclinux.org/>.
- [34] Texas Instruments. microcontroller lm3s6965.  
<http://www.luminarymicro.com/products/lm3s6965.html>.
- [35] Atmel Corporation. microcontroller atmel atmega8.  
[http://www.atmel.com/dyn/resources/prod\\_documents/doc2486.pdf](http://www.atmel.com/dyn/resources/prod_documents/doc2486.pdf).
- [36] WillowGarage. OpenCV library.  
<http://opencv.willowgarage.com/wiki/>.
- [37] Wikipedia. Vanishing point algorithm.  
[http://en.wikipedia.org/wiki/Vanishing\\_point](http://en.wikipedia.org/wiki/Vanishing_point).
- [38] Hui Kong, J.-Y. Audibert, and J. Ponce. Vanishing point detection for road detection. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 96–103, june 2009.
- [39] Meyer F. The morphological approach to segmentation: the watershed transformation. 1992.
- [40] T. Krajník, J. Faigl, V. Vonásek, H. Szücsová, O. Fišer, and L. Přeučil. A Monocular Navigation System for RoboTour Competition. *AT&P journal PLUS 2*, 18(1):57–63, říjen 2010.

# Appendix

## CD Content

In table 4 are listed names of all root directories on CD

<b>Directory name</b>	<b>Description</b>
bp	bachelor thesis in pdf format.
sources	source codes
dataset	images for road recognition
electronics	schematics, boards
mechanics	3D concepts, photos

Table 4: CD Content