Appendix 1
Pielikums 1

National Research Programme

„Cyber-physical systems, ontologies and biophotonics for safe&smart city and society”

(SOPHIS)

Project No.1

Development of technologies for cyber physical systems with applications in medicine and smart transport”

(KiFiS)

Scientific report

Period 3

2016

Contents

Glossary and abbreviations

ADAS	- Advanced Driver Assistance Systems;
CPS	- Cyber-physical systems;
ITS	- Intelligent transport systems;
ECG	- Electrocardiogram;
EDI	- Institute of Electronics and Computer Science, Riga, Latvia;
EMG	- Electromyogram;
KiFiS	- Cyber-physical system technology development and their applications in medicine and intelligent transport systems (Project No. 1);
SoC	- System on Chip;
SOPHIS	- Cyber-physical systems, ontologies, and bio-photonics for safe&smart city and society;
VPP	- State Research Programme;

Chapter 1.1

Introduction

State research program "Cyber-physical systems, ontologies, and bio-photonics for safe&smart city and society" (VPP SOPHIS) and the included project No. 1 "Cyber-physical system technology development and their applications in medicine and intelligent transport systems" (KiFiS) include tasks for the development of new generation of embedded systems – cyber-physical systems (CPS).

Cyber-physical systems include communication, data processing, and control elements, as well as interfaces to the physical world. These systems monitor the processes in real world, process the data, decides on the actions of controlling and improving the situation and enacts these decisions in the physical environment. Such cycles happen endlessly, and both on low level (such as a single room) and high level (such as a smart city). CPS provide a way for solving the economic problems, by providing us with "smarter", more intelligent, more energy efficient, more comfortable vehicles and transport systems, medical services, places of employment, communication systems, houses, cities and personal devices.

To make this vision a reality, there is a range of serious scientific and technological problems, that still need to be solved, connected to data gathering, electrical and optical signal processing, monitoring, control functions, while at the same time providing high enough level of security, stability and privacy. In addition, the system must be low energy, small, mobile and adaptable to new circumstances, as well as oriented to development of user friendly software and its usability. Scientific problems are connected to defining of new paradigms, concepts, platforms (hardware and software) and tool sets for the future development of CPS.

Because of limited resources available in this research project a subset of three specific CPS-related research areas were selected, matching the overall goals of the state research programme, and assigned to three groups of researchers:

- Group *TestBed*: To facilitate the production, programming and usage of CPS and by doing so, also to facilitate development of economically competitive innovative CPS based products while also facilitating their everyday use and reducing the digital divide;
- Group *MedWear*: To improve the quality and convenience of medical

services, while facilitating more efficient prophylaxis, more timely diagnostics and more successful treatment and rehabilitation based on innovative solutions both face to face or remotely in telemedicine;

- Group *SmartCar*: To improve the road safety and convenience of using road vehicles, by the use of smart transport system technologies.

In each of these three groups new concepts and platforms are developed based on results of previous state research projects, and improving on existing state of the art. These concepts and platforms are evaluated by comprehensive modeling and simulation research, thus selecting the perspective solutions, which are researched empirically, by creating experimental mock-ups, conceptual demonstrators, software libraries. The technologies which are economically competitive, will be approbated ir real or close to real conditions, in cooperation with partners from the economy.

The following document contains detailed description of these research activities in each of the three groups -

- *TestBed* in section 1.2 describes a testing/prototyping environment for wireless sensor system development and testing;
- *MedWear* in section 1.3 describes smart wearable sensor network infrastructure for energy efficient data gathering for measuring human biomechanics, and other health parameters (such as ECG), with applications in medicine and rehabilitation;
- *SmartCar* in section 1.4 describes development of Advanced Driver Assistance Systems based on smart image and sensor signal processing, and communication with other vehicles as well as development of a car based test platform for these systems and related algorithms.

Chapter 1.2

TestBed - Smart sensor and their network innovative hardware and software platform

1.2.1 Introduction

To ease the production, programming and usage of CPS, thus promoting competitive production of innovative CPS based products in economy, as well as facilitating their everyday usage and bridging the digital divide a set of research was done in the field of sensors, sensor networks and their hardware and software platforms.

Designing Wireless Sensor Networks (WSN) is time consuming process that involves many subsequent steps:

1. definition of WSN use case,
2. design and debugging of hardware,
3. development and debugging of software,
4. evaluation of designed WSN performance,
5. adoption of WSN for real-world operation.

Design of WSN begins with definition of use case – number of sensor nodes, operating environment and desired up-time. There are two possibilities for hardware design (step 2): first – to use commercially available sensor nodes TelosB[1], MicaZ[2] EPIC Mote[3], XM1000[4], second – to build hardware from ground. The first option requires only adaptation of existing hardware to the desired operation and therefore requires less effort compared to the second option. However, by using the second option it is possible to design hardware that is optimized for specific task and has no redundant components. Sometimes software development (step 3) may reveal that some changes to the hardware or even whole architecture of developed node are necessary or beneficial (to increase performance or reduce software complexity). Therefore, steps 2 and 3 must be iteratively repeated. In step 4 designed WSN is tested in controlled environment to evaluate power consumption, radio communication performance, and other parameters. In step 5 designed WSN is scaled (for operation using 25+ nodes) and/or adapted to real-world operation. Failure to accomplish step 5 may require repetition of previous steps and redesign of WSN.

Without specialized tools efficient execution of mentioned steps can be very challenging. For instance, there are a lot of routine manipulations that slow down the design process, like mounting, reprogramming of sensor nodes and connection of measurement equipment. To decrease development time of WSN, mentioned steps must be simplified.

This project works on solving these problems from two directions - (1) providing tools and know-how for faster development of WSN hardware and software, (2) reducing the time and effort required for testing and validating each of the mentioned steps.

The first direction is concerned with both low level tools such as Operating System for WSN (MansOS)[5], and high level programming tools (Seal/Blockly)[6]. Most research in these tasks is left for periods 3 and 4 of this project, while concentrating the limited resources of periods 1 and 2 (described in this report) on the second direction - testing and validation.

In this direction several new hardware and software solutions were developed, and used in development of a large (25+) WSN TestBed where users can perform different tests at different levels of abstraction - from low to high.

The WSN TestBed design challenges can be divided into three main sub-categories - architectural, hardware and software:

1. Architectural problems - Scaling, upgrading, and adding a new custom hardware;

2. Hardware problems - Selected hardware define overall WSN TestBed performance;

3. Software problems - The most efficient way to use available hardware resources for desired functionality. User-friendly front-end implementation for intuitive TestBed usage, without compromising data acquisition, processing, structuring from TestBed user point of view

Because of the limited resources of the project, the scope of the research has been narrowed to work on specific unsolved parts of these research problems, while reusing the existing state-of-the-art knowledge and results from previous projects for other parts of the testing environment surrounding them.

In this section our specific approach to solving these problems is described, as well as the vision for the system as a whole, defining the direction of the future research in the project.

1.2.2 Background

Wireless Sensor Networks (WSN) are broadly used in different types of applications, from agriculture to medicine and on body sensor networks. Essence of WSN is to observe the surrounding environment parameters at macroscopic level. For example WSN can be used to monitor temperature distribution in building or vibration levels at the bridge. The quality of designed WSN is defined by individual autonomous devices, called sensor nodes or simply motes, performance, WSN covered area and count of placed nodes.

Typically WSN consists of sensor nodes, that communicates between each other using radio link. Each mote has specific sensor set that is necessary to measure desired environmental parameters. Acquired data are gathered from sensor nodes to super node or sink. Typically super node is connected to the local or global network and also is used as bridge to provide easy data access for end-users.

1.2.3 Related work

There are many designed TestBeds for WSN. We will review most popular WSN TestBeds.

1.2.3.1 The TKN Wireless Indoor Sensor network TestBed (TWIST)

This TestBed is developed by the Telecommunication Networks Group (TKN) at the Technische Universität Berlin. It is one of the first largest academic WSN TestBed[7] for indoor deployment scenarios, deployed in 2005 year. It is located across 3 floors, resulting in more than 1500 m2 of instrumented office space. Currently they are using two types of sensor nodes - 102 Tmote Sky[8], 102 eyesIFX[9].

TWIST TestBed architecture is hierarchical in nature, consisting of three different levels of deployment: sensor nodes, micro-servers, central server. A high level view of this architecture can be seen in Fig. 1.2.1 below.

1.2.3.2 MoteLab

MoteLab[10] has been deployed on a network of 30 Ethernet-connected MicaZ [2] sensor nodes distributed over three floors of Maxwell Dworkin, the Electrical Engineering and Computer Science building at Harvard University. Also this WSN TestBed is freely available as open source, and several universities and research labs have chosen to use it for their projects.

MoteLab was the first designed WSN TestBed with reduced usage complexity. This was achieved due to the fact that there was implemented a rich-set of features: user-friendly web interface, remote access, automatic data logging for offline data processing, job scheduling, quota system for fairly TestBed usage.

MoteLab consists of several different software components. The main pieces are:

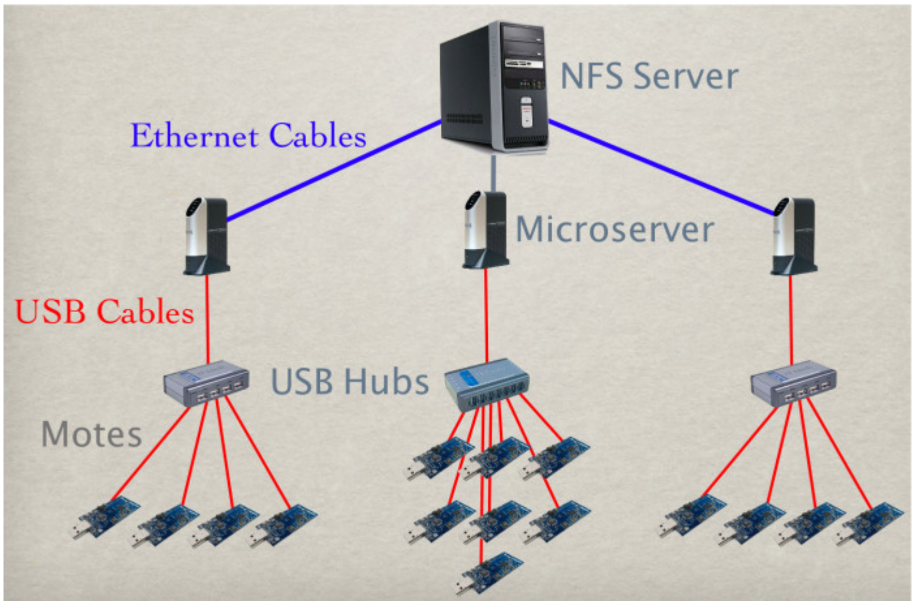


Figure 1.2.1: TWIST TestBed architecture

- **MySQL Database Backend** : Stores data collected during experiments, information used to generate web content, and state driven TestBed operation description,
- **Web Interface** : PHP-generated pages present a user interface for job creation, scheduling, and data collection, as well as an administrative interface to certain TestBed control functionality,
- **DBLogger** : Java data logger to collect and parse data generated by jobs running on the lab,
- **Job Daemon** : Perl script run as a cron job to setup and tear down jobs.

1.2.3.3 Indriya: A Low-Cost, 3D Wireless Sensor Network TestBed

INDRIYA is a three-dimensional wireless sensor network deployed across three floors of the School of Computing, at the National University of Singapore[11]. 100 TelosB[1] nodes and 25 Arduino[12] devices are used in INDRIYA TestBed. The INDRIYA WSN TestBed is build on TWIST architecture[7] with modifications regarding cost reduction. To reduce system cost INDRIYA uses MAC Mini devices that is capable of controlling 127 USB like sensor nodes. In this way micro-server count is reduced, thus cost are reduced.

1.2.4 Our approach

After analyzing the state-of-the-art WSN TestBeds it was decided that there are still many unsolved problems in this field and to properly develop and test potential solutions to these problems it was decided to develop a TestBed of our own, capable of implementing our research results. TWIST[7] was chosen as the ground truth architecture for the TestBed, but two major modifications were implemented:

1. Ethernet switches were replaced with PoE switches. PoE switch supports data transfers and power delivery. There are two PoE IEEE standards: 802.3af, max power rating is 15.4W, second - 802.3at, max power rating 25.5W. This modification allows us to decrease set-up costs and place micro-servers more freely in desired places, but we can't exceed power limitations, thus power efficient micro servers must be used.

2. Additional module, EDI TestBed adapter (described in subsection 1.2.4.5), is introduced. It is placed between micro-server and sensor node. Our developed module allows users to accurately evaluate designed WSN performance. It provides additional information like: power consumption measurement, battery discharging emulation, real-world sensor data emulation, analog/digital signal debugging.

In the following sections a more detailed view of the developed TestBed is provided, including description of architecture and solutions to specific researched problems, including work on the main challenges:

1. Distribution of the computational resources: The challenge is to split computational power to use full hardware potential, thus reducing computational load from main server;

2. Control all of the TestBed devices: The challenge is to reprogram all sensor nodes (DUT) as well as TestBed adapters remotely and at the same time. This also includes network health monitoring;

3. Efficient data acquisition and structuring: The challenge is to effectively acquire data and structure it in user-friendly manner.

1.2.4.1 TestBed architecture

The architecture of the TestBed (Figure 1.2.2) consists of the server (section 1.2.4.3), routers (section 1.2.4.4), TestBed adapters (section 1.2.4.5) and devices under test (DUT) (section 1.2.4.2).

Routers are connected to the server via Ethernet with PoE. To each of the routers a TestBed adapter is connected via USB connection and to finally the DUT is connected to the adapter either via USB connection (if it supports it) or some other configuration specific wiring.

Physically the TestBed will be installed in EDI building, across five floors - first, second, third, fourth, seventh. About 20 TestBed workstations will be placed in each floor. The placement grid was defined as irregular. Such sensor node placement assures different environment for testing radio communications and a more realistic close-to-real-world testing environment with different types of rooms, people density and obstacles. It is intended that 100 TestBed

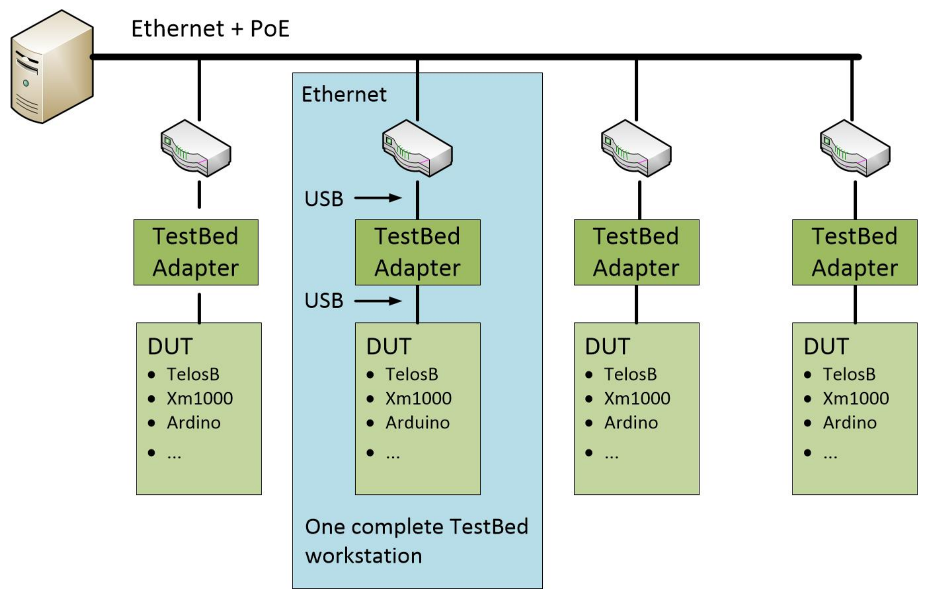


Figure 1.2.2: EDI TestBed Architecture

workstations will be installed, 90 of them across five floors, 10 outside of The EDI building utilizing also the nearby forest and exterior structures for even more varied test environment. In figure below you can see TestBed workstation placement in the third floor, Fig. 1.2.3.

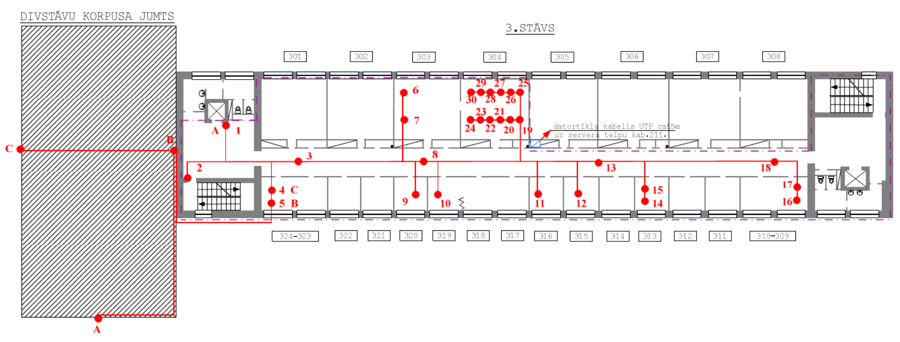


Figure 1.2.3: Sensor node placement in the 3rd floor

At the end of the second period of the project most of the TestBed workstations are already complete and are in the process of being deployed to their planned locations.

1.2.4.2 TestBed Devices Under Test (DUT)

We have designed our WSN TestBed in a way that it puts as few restrictions as possible on devices which can be tested. Devices Under Test can be operated under any WSN operating system that the device is capable of running. As for the device itself, we have 100 XM1000[4] sensor nodes which we will use as default sensor nodes, but theoretically any device can be used as long as it can operate from USB power source and use Serial communication through USB, the only restriction being that device must be compatible with reprogramming scripts used in routers, but this can be quite flexible since routers are running Linux operating system and we can add new reprogramming scripts quite easy.

1.2.4.3 TestBed server

The physical parameters of the server are:

- Manufacturer: ATEA
- Model: sVectron TS26
- CPU: Intel(R) Xeon(R) CPU E3-1220 V2
- RAM: DDR3-1333 4Gb,
- HDD: 1TB, 7200RPM , 64MB cache, SATA3,
- Case: Rackmount type (height 1U).

The server acts as a link between the TestBed and its users as well as a centralized data storage and configuration platform. It provides several critical services, such as:

- Centralized configuration services;
- Web interface for the TestBed;
- Centralized reprogramming control for the connected DUT;
- Online code editor for DUT;
- Data gathering, visualization and analysis utilities.

1.2.4.4 TestBed routers

Currently the routers used in the TestBed setup are Alix 2d2. The main parameters of these routers are:

- CPU: AMD Geode LX800, 500 MHz;
- DRAM: 256 MB DDR DRAM;
- Expansion: 2 miniPCI slots, LPC bus;
- Storage: CompactFlash socket, 44 pin IDE header;
- Operating system: Linux.

The routers are set up with 'Ubuntu 12.04 LTS' Linux operating system with custom modifications in kernel. Not all of the Linux kernel modules are needed in the current system and by customizing the kernel, we can achieve better performance and get support for additional hardware. Specifically FTDI drivers were added to the kernel, and redundant modules removed.

The routers provide the base functionality of data transport within the TestBed.

1.2.4.5 TestBed adapter

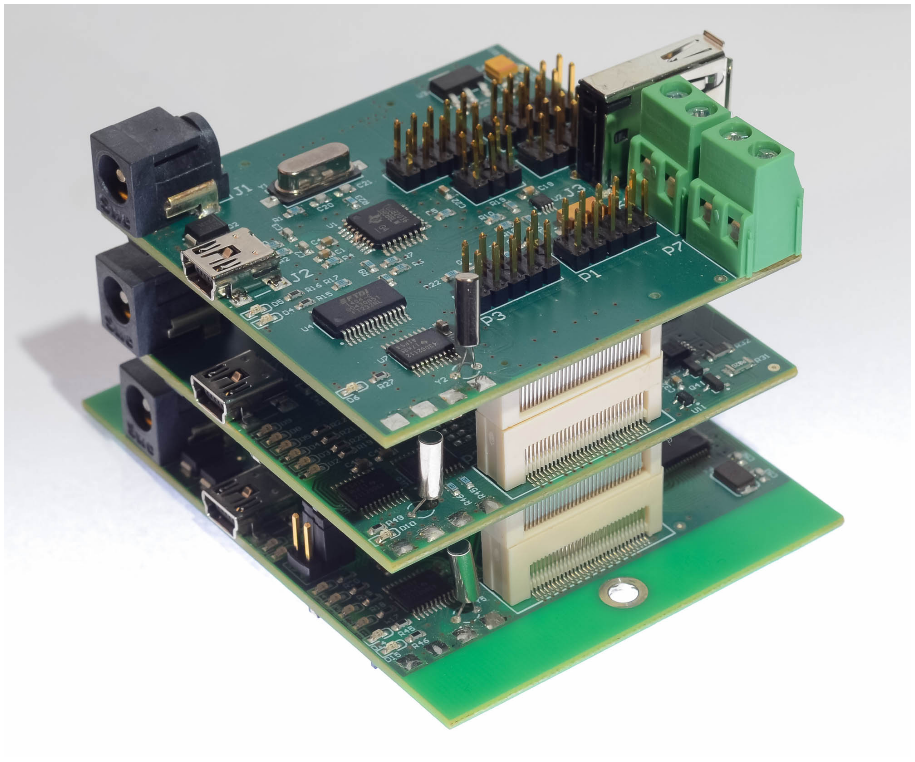


Figure 1.2.4: EDI TestBed adapter

EDI TestBed adapter (Figure 1.2.4) is a result of previous research work in the institute, related to prototyping and profiling low power embedded systems, such as EDIMote[13]. It is intended to test and debug low speed embedded devices, especially wireless sensor nodes.

EDI TestBed adapter has many features to extend testability for embedded devices:

- Emulate battery discharging,
- Measure consumed current of sensor node,
- Generate - digital and analog signals,
- Measure - digital and analog signals,

- Store measured data on local SD cards.

The adapter is designed with modularity in mind, so that new functionality may be added later as required. Three specially synchronized modules are processing and debugging data. Modules are – communication, power meter and signal conversion. Each module has it own task respectively.

- Communication module: Controls data flow between router and sensor node and other adapter modules. With additional software, stored on router, it is possible to access every stacked module, through communication module. Only one connection at the time, between router and modules can be established;
- Power metering module: This module evaluates connected sensor node power consumption and supply voltage stability;
- Signal conversion module stores and processes data to avoid data loss or modifies the current data as needed.

1.2.5 Achieved in 3rd period

The following sections describe the main advancements in the Testbed during the 9 months of the shortened 3rd period.

1.2.5.1 Implementation

We have divided the data management task into five logical steps which directly correspond to our TestBed data flow architecture:

1. TestBed adapter and device under test data generation
2. Data gathering on router
3. Data processing on router
4. Data insertion in database
5. Data representation on web interface

1.2.5.1.1 TestBed adapter and device under test data generation

All of the interesting data is generated or acquired at device under test(DUT) or TestBed adapter, both of them are connected to router via USB cable. As you can see in figure 1.2.5, technically DUT is connected to TestBed adapter, but since TestBed adapter includes a USB hub, we can access DUT as a different USB port from router, so we can gather data from DUT and TestBed adapter simultaneously.

To access TestBed adapter modules we need to switch between them, since we have only one USB connection to TestBed adapter itself, but multiple TestBed

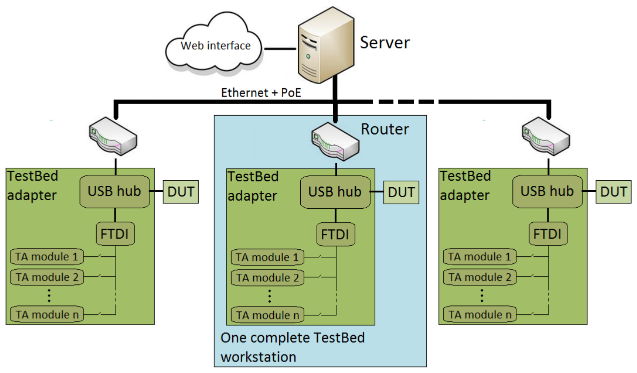


Figure 1.2.5: TestBed data flow

adapter modules we need to communicate to. To solve this we have software controllable switches on UART lines after FTDI chip, so we can enable or disable communication to each TestBed adapter module. At the moment we have 3 TestBed adapter modules:

- Main control module
- Power module
- ADC & DAC module

Data transfer To make this communication protocol robust and scalable we decided to use High-Level Data Link Control(ISO 13239) style packets, the packet format for both directions is shown in Figure 1.2.6.

1 byte	1 byte	1 byte	n bytes	2 bytes	1 byte
Start flag	Length	Command	Payload	CRC	End flag

Figure 1.2.6: TestBed data packet

Total length of packet minus flags is length + 4 bytes. 16 bit Cyclic Redundancy Check(**CRC**) is used to ensure that received data is valid. Field **command** defines what kind of command/data is transmitted and field **payload** contains data transmitted or command parameters depending on command.

Data flow path Not all modules have the same requirements regarding data flow, understanding where the most of our data comes from helps us to understand how to build an optimal TestBed adapter data management life-cycle.

Main control module contains only few switches controllable by software for DUT control and serves as controller for UART line switching e.g. enabling or disabling communication for other TestBed adapter modules and thus it needs negligible bandwidth.

Power module contains ADC for DUT power consumption estimation and digital potentiometer for battery discharge simulation. Battery discharge simulation is controlled by simply setting discharge rate throughout the experiment or providing a constant battery voltage at certain experiment points, it requires negligible bandwidth to operate. The ADC on power module has 1 channel with 16 bit resolution and it can operate with frequencies up to 500 kHz, which means that the maximum bandwidth it can produce is about 1 MB/s.

ADC & DAC module responds for external signal generation and capturing regarding DUT operation. ADC installed on this module have 8 channels and each channel has 12 bit resolution and it can operate with frequencies up to 1 MHz, which means that the maximum bandwidth each channel can produce is 1.5 MB/s, leading to total maximum bandwidth of 12 MB/s. DAC installed on this module have 8 channels and each channel has 12 bit resolution, it can be controlled by simply setting output value pattern for each channel or controlling each channel at the course of experiment, either way the bandwidth necessary is negligible. ADC & DAC module differs from other modules since it has two FTDI chips, so it basically consists of two logical modules on a single physical module.

Device under test We are also developing communication library for MansOS [5] operating system which uses the same principles as communication with TestBed adapter, so users could easily implement the same communication standard for DUT to router communication with ease.

1.2.5.1.2 Data gathering on router

Since we have separate USB connections to TestBed adapter and DUT we can read data from both of them simultaneously. This leaves us with only one problem - handling data acquiring from TestBed adapter modules, since we have only one USB connection, but 4 logical modules we need to communicate to. This can be achieved by using the UART line switch and communicating with each TestBed adapter module in turns. This means, that we can't allow TestBed adapter modules to send data whenever they have anything to send, because we can't assure that the UART lines are connected, so we have to develop a communication protocol where router asks for data to TestBed adapter modules and TestBed adapter modules respond only when asked.

To ensure that no data is lost we implemented the communication protocol in such a way that after the request for data has been sent to TestBed adapter, we wait for response, and if we get no response in some defined time, we retry the request up to 3 times. If there is no response after 3 retries we have to assume that the current TestBed adapter module has failed, there are two possible scenarios after such event. If experiment supervisor has allowed to restart failed TestBed adapter modules, we restart the failed module and continue the experiment as planned. But sometimes there might be some very important data saved on TestBed adapter and maybe it is continuing to work and only the communication part has failed, so experiment supervisor has forbidden to restart TestBed adapter modules and after the end of experiment TestBed adapter SD card can be manually examined to understand what exactly caused the fail and check for any usable data the failed module may have saved.

1.2.5.1.3 Data processing on router

Every message received from DUT and TestBed devices must be parsed in order to structure and save necessary data. Parsing is not very heavy work but it can be problem if carried out in great scale, especially if parsable data is made up from relatively big argument list, for example large vector or matrix. To tackle this problem and introduce more scalability to this TestBed solution a decision was made to do all necessary data parsing on router.

In addition to this basic functionality new firmware version also introduces possibility to do effective data preprocessing on router, distributing computational load and reducing data flow in the network. It is useful in situations when relatively large data sets are being collected, but useful information are obtained only by running algorithm with collected data sets(histogram for example). Custom preprocessing algorithms can be developed and carried out on routers by extending according Python classes. For trivial preprocessing algorithm implementation all that must be done is defining input data length - data that must be collected before running algorithm, output data length - data size that will be passed to database after algorithm execution and algorithm function itself. This way only meaningful data is sent to database and computing is done as close as possible to leafs of the network, increasing network scalability.

1.2.5.1.4 Data insertion in database

In previous solution query for writing new data in database was called from main server, but for more scalable system design a slightly different approach was chosen. Instead of sending data to daemon running on server for later saving, router itself connects to database and calls save query on parsed data every time new data is available. This way not only some load is taken off from main server, but independent connectivity provides easier network expanding and better device health monitoring, because router automatically adds itself to database as connected and state of device can be monitored.

1.2.5.1.5 Data representation on web interface

Every data set saved in database can be visualized in web interface. When call for data visualization is made, query is called to database and received data drawn in necessary format. Data is also refreshed dynamically. Gathered data is streamed to user(web interface session) near real-time. It is done by asking server(sending AJAX request) for changes in data set every 300 ms, this sort of solution does generate overhead which can be avoided by using web socket technology, but it would put more computational pressure on server and heavily increase system complexity.

1.2.5.2 Evaluation

1.2.5.2.1 TestBed adapter hardware estimation

To improve the accuracy and quality of the measured data and find the relevant values, it is essential to apply post-processing techniques. When a large collection of numbers is assembled, usually the actual value is not in the individual numbers, but rather in certain descriptive quantities such as the average or the median. In our case, the actual value is obtained using averaging and the Gaussian distribution functions as explained in more details in the material

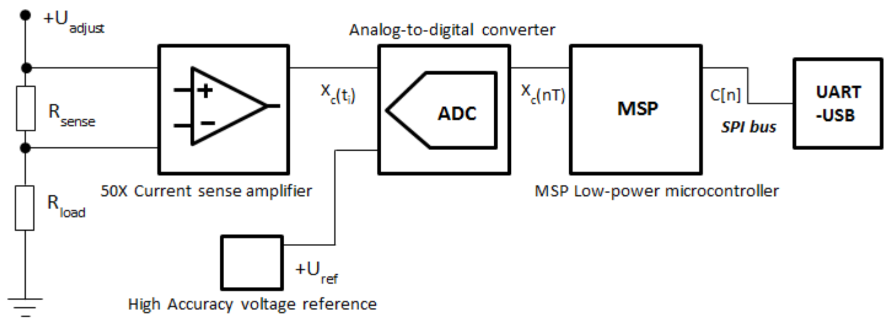

+U adjust
R sense
R load
50X Current sense amplifier
X_c(t_i)
Analog-to-digital converter
ADC
X_c(nT)
MSP
MSP Low-power microcontroller
C[n]
SPI bus
UART -USB
+U ref
High Accuracy voltage reference

Figure 1.2.7: Basic block diagram of current measure circuit

Current measurement circuit is shown in Fig.1.2.7. It consists of three parts: Analog value acquisition circuit, with low-noise-current-sense amplifier, digital pre-processing system and finally, digital value processing and preparation for serial data translation.

All collected data, from the ADC is received in the MSP microcontroller and transferred to the PC, using UART-USB data bus. This information contains true value and noise. Necessary evaluation must be done by using computer software or different high quality system to preserve the original quality of the data.

Calculation steps, of actual value of measured data is shown in equation 1.2.1, where Q is the number of quantization bits, $C[n]$ that came from some continuous-time signal $X_c(t_i)$ (Fig.1.2.7), is a binary formed data, but during measuring process this value has been changed, $X_c(nT) = X_c[n]$ as data samples with a specific frequency. $X_c(t)$ is an analog value of the line received by the ADC.

$$I_{sense} = (X_c[n] \cdot Q) \cdot \frac{1}{50} \cdot \frac{1}{R_{sense}} \tag{1.2.1}$$

$$Q = \frac{U_{ref}}{2^{bit}} \tag{1.2.2}$$

Quantization bit value can be estimated by equation 1.2.2. U_{ref} is reference voltage provided to ADC and powered by the value of ADC bit resolution number

This value after formula 1.2.2 consists of several components noise and true value, but this data is scattered. To get measured value of data stream, we must understand the quality of signal and noise. Because, sometimes there are lot of distractor components which affect each other. E.g. digital clock signal is generating high frequency noise, but this noise can be found using Fourier transformation. At first, it is necessary to analyze data. To make sure, the mean value conforms to the measured, calculations are made in equation 1.2.3.

$$X_c(t_i) = mean(X_i) = \frac{1}{N}\sum_{i=1}^{n} X_i \tag{1.2.3}$$

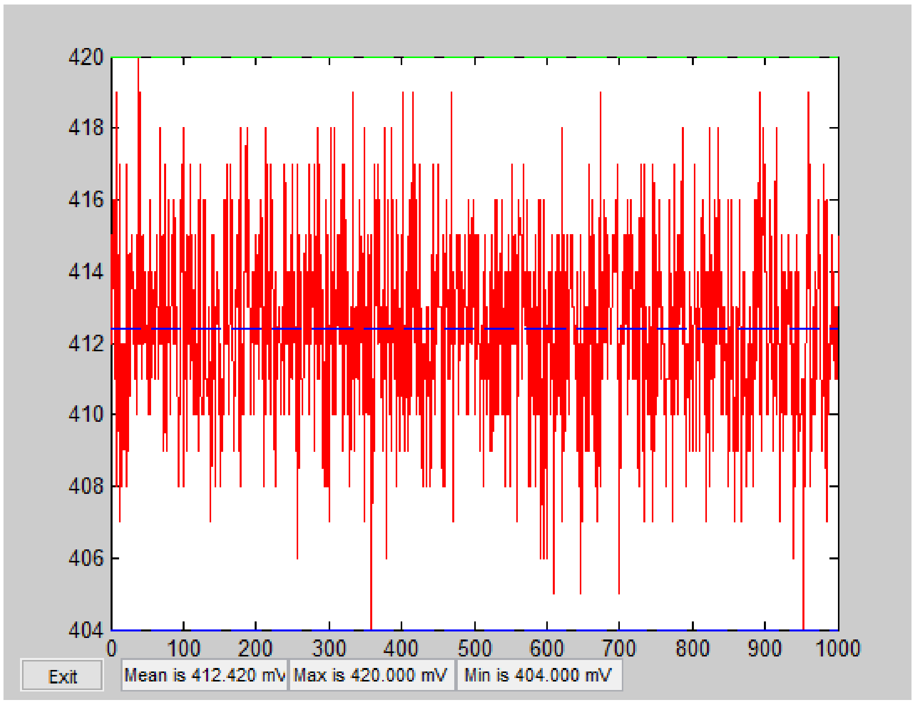


Figure 1.2.8: Data stream, measuring passive load in current sense circuit

In figure 1.2.8 a data stream stream is shown, measuring passive load (resistor) using 4.096 reference voltage and 3.00 supply voltage in current sense circuit U_adj.

$$X_i = mean(X_i) - X_{dir} \tag{1.2.4}$$

Equation 1.2.4 returns mean value of data streaming and this value conforms to real, expected unit by comparing with value from another measuring device. Calculating the systematic error of measured signal, attention must be paid to direct component. In this circuit there is direct component caused by current sense circuit in analog part of circuit. To get this value measurements with open circuit in current-sense part must be made (without any load), after calculating, must subtract this direct component from $mean(X_i)$ value.

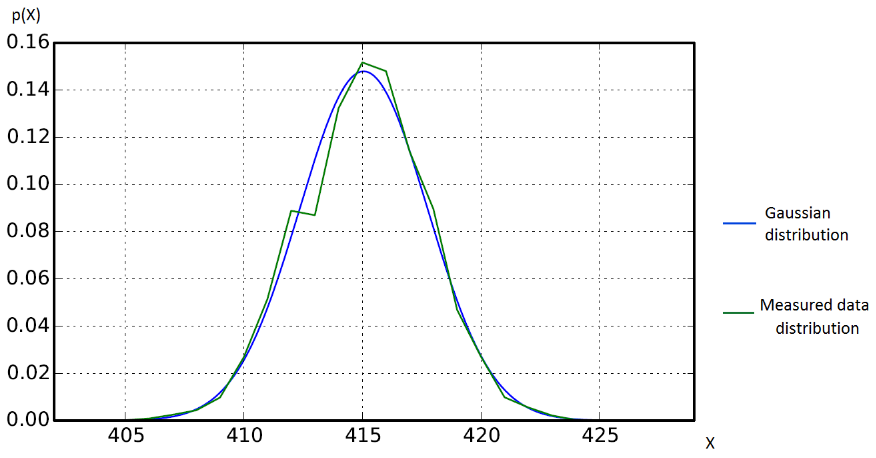


Figure 1.2.9: Measured probability density function - green line. Gaussian distribution calculated for measured dispersion σ and mean value $\bar{X}$ - blue line

Even after this calculation eliminates the systematic error, this value still contains noise and electrical disturbance. It is easy to calculate and compare the data after using formula 1.2.3, but to determine the nature of the error, most common solution is processing data with a Gaussian distribution and standard deviation functions. In Figure 1.2.9 two lines are shown, blue line is calculated using Gaussian distribution function, but green line presents a measured data probability density function.

The results of processed data are as shown in equation 1.2.5 below. For this the Gaussian distribution was calculated by equation 1.2.6, Sigma was calculated by equation 1.2.7 and ΔX was calculated by equation 1.2.8 meaning standard deviation.

$$\begin{aligned} X \pm \Delta X &= 415.056 \pm 0.054 (For 10000 samples) \\ X \pm \Delta X &= 415.87 \pm 0.17 (For 1000 samples) \end{aligned} \tag{1.2.5}$$

$$f(x) = \frac{1}{\sqrt{2 \cdot \pi \cdot \sigma}} \cdot e^{-\frac{(X - X_{mean})^2}{2 \cdot \sigma^2}} \tag{1.2.6}$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (X - X_{mean})^2}{n - 1}} \tag{1.2.7}$$

$$\Delta X = 2.0 \cdot \frac{\sigma}{\sqrt{n}} \tag{1.2.8}$$

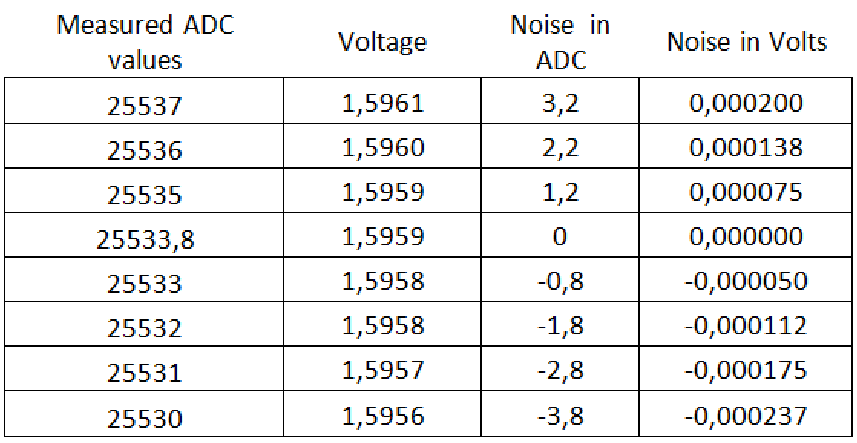
In these equations, the number X is combination of ADC value and error. Error depends on the number of samples; in the case of 10000 samples error is rather small. From the graph it can be seen, that the error has the nature of a Gaussian distribution. It means the error is random caused by some physical parameter like heat, low level electron flow etc.

1.2.5.2.2 Power module current measuring and error

Evaluate each TB-adapter, must be calibrated accordingly, to acquire real (expected) value coincides with measured values. Before this process, has to evaluate error and noise distraction.

Measured data from ADC consist of real value and error, in this case must do the calculation to get real value out of the data value stream, as mentioned before in the material.

Analyzing Testbed-adapter power measurement module functioning and try to find the cause of data scattering, lot of experiments was made. To find their exact cause of the data scattering, in this measuring system, the easiest way is to use a low noise power source, what is a noise-free element, several tests was done using this method. The obtained results are showed in 1.2.10) and it present, that measurement results contain variance, it means that the noise is created inside the measuring system.



Measured ADC values	Voltage	Noise in ADC	Noise in Volts
25537	1,5961	3,2	0,000200
25536	1,5960	2,2	0,000138
25535	1,5959	1,2	0,000075
25533,8	1,5959	0	0,000000
25533	1,5958	-0,8	-0,000050
25532	1,5958	-1,8	-0,000112
25531	1,5957	-2,8	-0,000175
25530	1,5956	-3,8	-0,000237

Figure 1.2.10: EDI TestBed adapter, power module measured data table,with constant power source measurement

Dispersion of the measured data are in microvolts. Comparing by constant flow current, the noise level is obtained low, but a significant problem is DUT - (device under test) and it low-state power consumption as "Sleep mode" or "Idle mode".

Working around this problem, several methods were used to avoid data scattering. Main part takes the reference voltage source and test results sowed, that small data dispersion is created in this section, for about 200-300 micro volts. This noise is determined with reference measuring method. This is one of the most important points, because reference voltage source determines the fair value of accuracy and, if the source is noisy, then all the results will be inaccurate. Simply to solve the dispersion by adding a filter after reference power source is the easier solution, otherwise, all system have to be modify or even rebuild, and it would exceed unambiguous high costs. Results after small modification in the measuring system is shown below in 1.2.11).

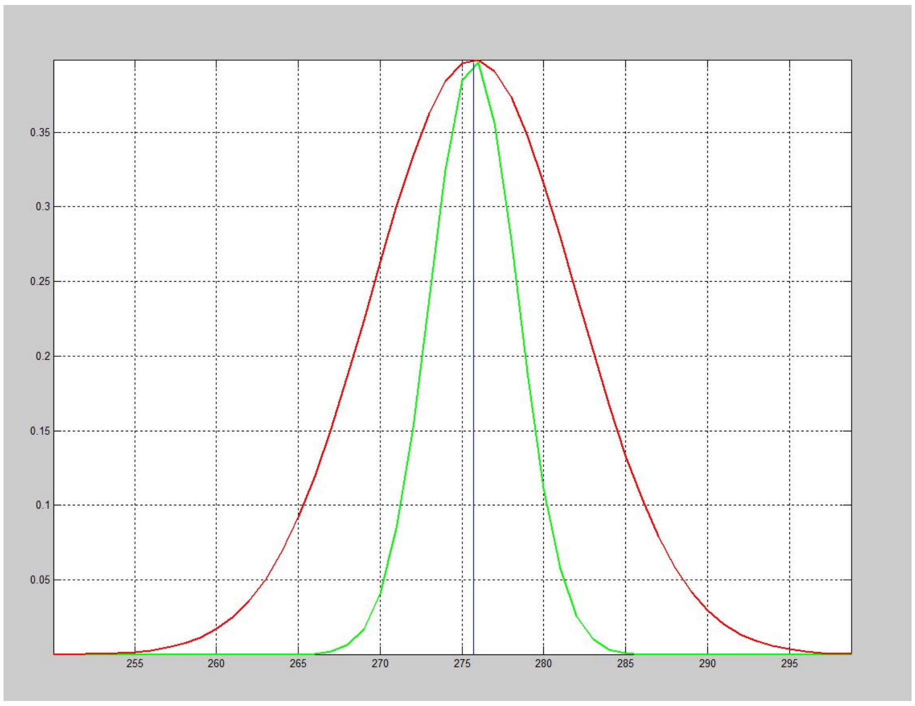


Figure 1.2.11: EDI TestBed adapter, current measurement data calculation of Gaussian function and compare with improved results

In 1.2.11) red line presents data dispersion using original (unmodified) measuring system with ADC value

$$X \pm \Delta X = 275.7 \pm 0.394$$

. Green line presents the same current ADC value, but with system Modification, with ADC value

$$X \pm \Delta X = 275.7 \pm 0.016$$

After the system modification, improvement is about 55% of data dispersion. This is a huge progress, but the downside of this improvement is a systematic error enlarging. In ADC values it is about, 62.6 units in this particular case.

1.2.5.2.3 TestBed adapter and device under test data generation

We measured how much data can we read on TestBed adapter ADC & DAC module from ADC. Test was performed reading all 8 channels sequentially, this way we can read data from ADC at about 187.8 KB/s, which is a lot less than theoretical 12 MB/s, this is due to MCU not being able to read ADC fast enough.

We measured how much data can we read on TestBed adapter Power module from ADC. Test was performed reading 16 bit value repeatedly, this way we can read data from ADC at about 600.2 KB/s, which is less than theoretical 1 MB/s, this is due to MCU not being able to read ADC fast enough.

1.2.5.2.4 Data gathering on router

We performed multiple bandwidth tests using different payload sizes, from 5 to 255 bytes with step of 5 bytes, and different test scenarios, to evaluate the bandwidth we can achieve with current hardware and software, the results are shown in figure 1.2.12. Testing was done using constant data stored in TestBed adapter module MCU memory, before every message sent by router it switched to target TestBed adapter module to ensure that the correct module is selected, this is also applied to tests where only one TestBed adapter module was used. Each test with different payload size was continued until at least 10 KB of data was transferred. We performed such test for each TestBed adapter module except main module, because main module does not have an MCU with full UART connection capabilities. We also performed a mixed test, where random TestBed adapter module was chosen for each message, for example, when using payload size 100 bytes, we need to send 100 messages to send total of 10 KB of data, so for each of those 100 messages a random target TestBed adapter module was used.

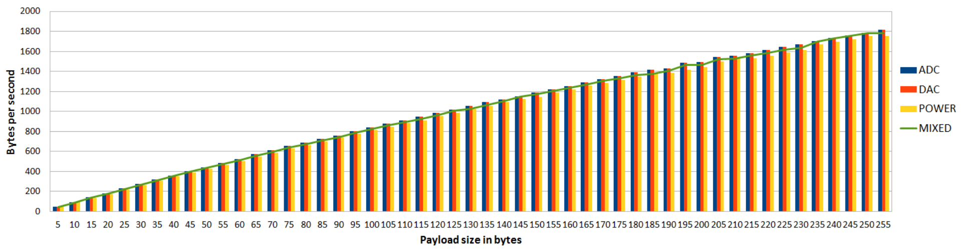


Figure 1.2.12: TestBed module data bandwidth

The results show that there is no noticeable bandwidth difference between TestBed adapter modules. Also it is clear that when using mixed data sending, it does not affect bandwidth, so we don't need to worry about very careful scheduling of module switching. At larger payload sizes the bandwidth increases because the ratio between payload and overhead increases. When using payload of 255 bytes, we can achieve the bandwidth of approximately 1.8 KB/s, which is way less then the bandwidth theoretically produced by ADC on TestBed adapter modules, which means that we will have to lose or pre-process the data before sending, for example we could send some average values over time or send only the relative differences when they occur.

1.2.5.2.5 Data processing on router

As mentioned before, in current software version, all data received from adapter and DUT is parsed by router. Data parsing speed on router is so effective that it is hard to generate data sets that would create noticeable lag. Roughly one millisecond for 1000 elements with 3 arguments each. In real life scenario DUT can't generate data sets of that size in such a short time. The only time when parsable data can grow larger than 1000 elements is when adapter returns gathered data from long period of time - energy consumption etc. But because it happens over large time span it is not even close enough for system slowdown and thus growth of parsable queue is not likely to happen. Pre-processing functionality performance is hard to measure, because it depends on algorithm implementation, but it is faster than processing data on main server or client machine for several reasons. There is no need to call database queries for required data because all data that is needed for algorithm is saved in router RAM or disk memory based on required data size when it is received from DUT or adapter. Data is stored until necessary set of data is collected for algorithm to perform and only then result is saved to database. In comparison, if target algorithm would be carried out on main server or client, it would need an extra database query for data. It would put noticeable pressure on database and server if large sets of data are necessary for algorithm and also generate huge amounts of network traffic if carried out on clients machine. For some algorithms it is also possible to generate intermediate results from incomplete data sets making calculations while the data is still being collected. This sort of solution can only be effective if all data is stored in memory. It greatly increases parallelism and network scalability.

1.2.5.2.6 Data insertion in database

In previous software version, as mentioned before, data was written in database by daemon running on same hardware as database engine. This functionality was moved to the router because tests prove that under pressure(when more than 20 write operations are executed at once) writing speed is almost identical even with data traveling over local network. Write speed on current hardware is about 8ms per data vector which consists of long integer, string with length of 255 symbols, integer and timestamp. Writing to database directly from router was slightly slower(9ms-10ms), as expected, but by structuring data path in this manner router becomes independent from main server. This property can be of great use in the future, for example, if data needs to be streamed to different database servers or connected with other TestBed solutions with different architecture.

1.2.5.2.7 Data representation on web interface

Data representation can be divided in two time consuming actions. First one is loading and visualizing already existing data from data base and second is refreshing data set while it is being viewed by user. For both actions elapsed time consists of initial AJAX call, reading from database, data parsing and callback. Tests were carried out on local network with initial data set of 1000 elements and 3 new elements added every second. AJAX call and callback tends to run in time frame between 30ms - 50ms, but, of course, it is heavily dependent on network speed and data set size. Database read speed is about one millisecond for 10 elements and is also almost identical in both actions. Data parsing happens in an instant - one millisecond for 1000 elements. As mentioned before, data is only asked for server once every 300ms which is main slowdown in data refreshing. Web sockets can be used to eliminate this overhead, but, depending on implementation, it would disturb router and its network child nodes ”independence” and put more pressure on database, router or client machine.

1.2.6 Results

During the first three periods of the project a TestBed system is continually developed, including development of a new modular TestBed adapter which allows complex control and analysis of DUT. The adapter hardware has been tested and some problems have been found with communications through TinyMCU, but they have been resolved.

The TestBed server has been upgraded and moved to a dedicated server room in EDI.

The TestBed user interface has been developed to provide a more convenient way of using TestBed and programming the DUT (including start of development of online code editing tools).

TestBed communications architecture and protocol has been reworked by shifting some work load from the server to the other TestBed components thus resulting in a more decentralized, fast and effective design.

In the third period of the project a lot of tests have been performed to make sure that TestBed is working as expected. We have developed more reliable overall TestBed architecture to distribute load evenly among different TestBed components and thus achieve even better overall performance; Also we have improved all of our communication protocols to allow faster and more reliable data transmissions and more precise experiment control as well as support scalability for future TestBed improvements; one publication (indexed in SCOPUS) has been accepted for publication describing part of the new results.

1.2.7 Discussion and future work

In the final period of the programme we are planning to achieve multiple long-term and short-term goals to make EDI TestBed a convenient and powerful tool helping people around the world to develop their wireless sensor networks or test their prototypes faster. This includes:

- Improving our user interface to make it more convenient as well as more functional;
- Improving data visualization by introducing a lot of different ways of graphical data representation to suit every need;
- Adding support for cloud program compiling and online code editor linked together with MansOS and SEAL for faster and easier code development;
- Calibrating and improving TestBed adapter to provide more meaningful and precise data about Device Under Test;
- Validating the TestBed system by approbating it for use in other projects and by third parties (such as businesses working with WSNs).

Chapter 1.3

MedWear - Medicine and telemedicine uses of CPS

1.3.1 Introduction

In this section healthcare applications of cyber-physical systems are researched to reach the goal of improving the quality and convenience of face to face medical services as well as remote telemedicine services, while facilitating more efficient prophylaxis, more timely diagnostics and more successful treatment and rehabilitation based on innovative solutions.

The evidence for beneficial applications of wearable computing can be found in many different fields, such as medicine [14] from early detection [15][16][17][18] to treatment [19][20], care for the elderly [21][22][23] and frail[24], exercise [25][26][27], mental health [28], entertainment [29][30] and potentially many others.

Also it is not hard to imagine, that both patients and doctors could benefit from low-cost unobtrusive smart clothing, which could be used for patient compliance monitoring, long term medical data analysis in patients everyday environment as opposed to special testing facilities, telemedicine, unobtrusive vital sign and traumatic event monitoring etc.

Unfortunately each of these examples of previous work in the field of wearable sensor applications have been working on their own custom wearable sensor platform. Because of this a lot of time and energy goes into reinventing the wearable data gathering infrastructure, which could be better spent in researching beneficial applications to wearable technologies.

A standardized architectural, hardware and software process for development of smart clothing could provide tools for development of customizable wearable sensor and actuator networks with little effort and low budget. In addition such wearable sensor ecosystem could provide similar benefits to the growth of the field of wearables as personal computers did for the field of computing - competing companies could work on specific parts, such as specific sensors and not be concerned about the complexity of developing and marketing a full wearable system.

Because of this our team researched the state-of-the-art in these fields and defined a vision of smart clothing architecture capable of providing these benefits. Because of the limited resources available in this research project, priority directions were selected and specific key parts of this vision were researched (such as key architectural elements or use of specific wearable sensors) thus ensuring, that the overall vision of smart wearable systems is moved forward, and an effective foundation is laid for future applications of this research.

A variety of different illnesses or injuries, for example, cerebral palsy, can often lead to posture and different body part control problems in a number of daily situations. Correct posture, body part alignment, etc. is essential to ensure accurate operation of respiratory system, cardio-vascular system and other body functions. In addition, this information can provide context of human activity that allows to better interpret data obtain with other sensors. Currently posture and body alignment monitoring and training in a variety of different rehabilitation programs normally is done in close supervision of medical specialist. This approach limits availability of rehabilitation and maintains high work load of medical staff. In addition, the monitoring duration is limited to special dedicated sessions, making the daily monitoring of patient outside medical facilities practically impossible. To overcome previously stated limitations a variety of aiding technical apparatus are being used, however, the availability of these are limited in terms of functionality and also ease of use. In this project, a particular subtopic is devoted for development of methods that allow monitoring of human posture, different body part alignment and movements in real time with wearable devices.

1.3.2 Architecture for multi-sensor smart wearable systems

While working on wearable computing and smart clothing and reviewing the state-of-the-art, common use case scenarios have been defined and common requirements have been based on those, providing basis of a universal smart textile as described in the introduction[31].

The defined common use scenario of a smart textile, facilitating rapid, low-cost development and deployment of wearable computing, is as follows (Fig. 1.3.1):

1. The cloth with wiring integrated within in a universal grid is mass produced resulting in rolls of smart textile, similar to the rolls of ordinary textile;

2. Clothing is tailored from the smart textile in the same way as ordinary clothing;

3. Wiring from different physical parts of the clothing, such as sleeves and torso, is connected together in the same network;

4. At some point in the clothing a special connector and pocket is added for central data gathering endpoint containing a battery and wireless

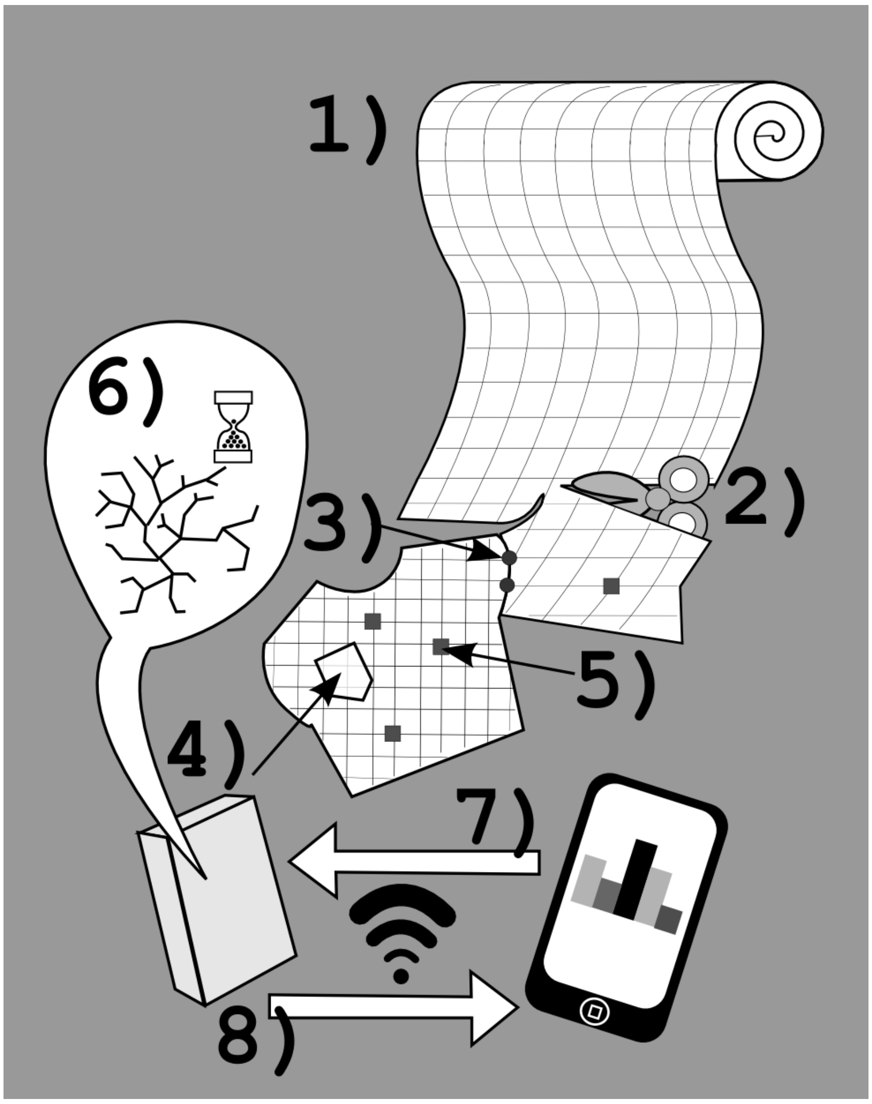


Figure 1.3.1: Envisioned use scenario of a universal smart textile for medical applications

communication module;

5. Depending on the specific task for which the smart clothing is being created sensors are selected and placed in the relevant spots of the ready clothing turning it into a wearable sensor network;

6. The system is turned on and it runs self-diagnosis mapping the network and routing the power, data and clock lines through existing wire connections for optimal data gathering;

7. If needed, specific sensor or actuator drivers are loaded to the whole sensor network, and these drivers are seamlessly distributed throughout the piece of smart clothing;

8. The data is gathered by the central data gathering endpoint and transmitted over wireless connection to a computer or a mobile device and there software can be developed for the specific purpose envisioned.

To facilitate this scenario a series of basic requirements must be met, which the authors have defined as follows:

- Wires - Wiring system for sensor node connections providing power and signal lines to each of the sensor nodes must be realized with the minimum number of wires. This is because wires are hard to implement in stretchable textiles and they are the most fragile and expensive part of such a smart textile requiring that their number should be kept to a minimum;
- Physical layout - A regular pattern of densely placed sensor node connection points is preferable allowing more universal solutions instead of sensor positioning for a specific problem. Each square meter of textile should include more than 100 sensor nodes - the more potential connections for sensor nodes, the better the chance that a specific node will be close enough to a specific point on the body required for a specific use case. Also a better resolution can provide a more complete picture of the data;
- Connections - Wiring topology connecting the sensor node connection points should provide redundancy for rerouting in cases when some wires are cut while creating clothing from the smart textile - as custom electronics manufacturing is very expensive, it would be more beneficial to develop a universal smart textile. This means, that while tailoring the smart clothing cuts could be made almost anywhere and this potential damage to the network must be taken into account;
- Electronics - The hardware should be small and unobtrusive enough to integrate in the textile and elastic and robust enough to survive everyday use, moisture and other typical stresses. The technology will not gain mass market appeal if it is not easy to wear and will not be robust enough to pay back for its value. Electronic chips and sensors usually are quite resilient. The most fragile part of such smart clothing is usually the wiring and connectors. It must be applied with specific technology to provide elasticity and at the same time maintain the specific electric properties, such as resistance and at the same time protect the electronics from outside elements, such as moisture (The requirement for constant resistance is one of the reasons common methods of developing elastic conductive threads[32] are not a good solution);
- Data gathering endpoint - At any point at the discretion of the clothing designer there should be a possibility to add an endpoint capable of gathering the data from the smart device, providing power to the wearable sensor network and transmitting the gathered data further for processing and displaying. The battery should be small enough to be unobtrusive and capable enough to power the whole device for a full day of use. This means

that the whole system must be built on energy efficient components and principles of energy saving;

- Software - The software accompanying the smart textile should be able to gather the data from all sensor nodes irrespective of the final configuration of the smart clothing and the place where the endpoint is connected. This requires smart mapping of the final garment;
- Updates - Irrespective of the number of sensor nodes and topology of the network it should be easy to reprogram the system with new software or drivers for sensors or actuators without disassembling the network. This means, that sensor nodes should be re-programmable through the network, without direct hardware access to each node. System updates should come from the wireless connection of the endpoint and propagate throughout the network, while it is still powered on;
- Speed - The system should provide data rates of at least several full frames per second for the system to be usable in real time - this includes gathering data of every sensor in the system in each frame and transmitting the gathered data for processing;

In addition - for the system to become widely used and accepted it has to have a potential to be economically viable through mass production, there should be standardized software tools for using the gathered sensor data and the hardware and software standards should be open and accessible for mass acceptance.

This architecture concept has been partially based on the line-topology smart clothing architecture described by the institute in previous state research project. In this project specific parts of this architecture have been developed further, as well as some specific sensors and applications for this architecture have been researched.

1.3.3 Our approach

Our approach to this problem is to identify specific parts of the overall solution and concentrate project resources on those parts e.g. specific problems in data gathering architecture or specific sensors and their synergy with other sensor types, as well as energy efficiency and data gathering/transfer.

In below sections these specific tasks, on which our team has worked during the project are described in detail, as well as the resulting innovation from this research.

1.3.3.1 Overview of technologies developed in periods 1 and 2

Even though these technologies have already been described in detail in the project reports of previous periods, short overview is added here as well for context of achievements in this period.

1.3.3.1.1 Multi-branch architecture

During State research program "IMIS" project no. 2 „Innovative signal processing technologies for smart and effective electronic system development" a method was developed to form a network with large number of low-power sensors and acquire their data in real time. This method was based on the daisy-chain principle. It provides a number of benefits when sensors can be connected in series.[33] The proposed enhanced daisy-chained SPI network greatly reduces the amount of wires in the system. New sensors can be easily connected at the end of the chain, with little software adjustments. View of this architecture can be seen in Fig. 1.3.2 below.

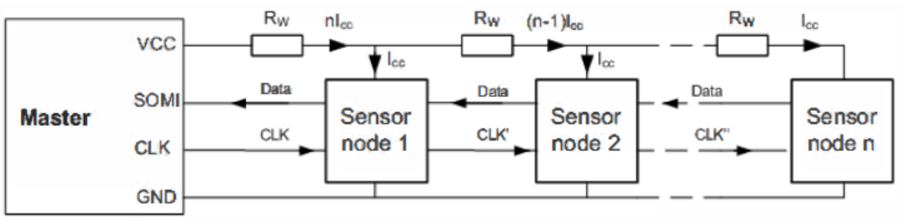


Figure 1.3.2: Enhanced daisy-chained SPI [33]

Several potential improvements to this architecture were considered in this project to limit its drawbacks and improve its properties when gathering data from sensors on the whole human body not just specific parts. One of such improvements was the multiple branch network architecture. Main goal for using this architecture is to increase the rate at which whole system measurements are received as well as to simplify access to remote parts of human body (e.g. feet, palms). This architecture also has to provide the option to connect non-homogeneous sensors and sample them at different frequencies. This architecture consists of one master controller with multiple branches of series connected smart sensors.

Structure of proposed multiple branch architecture can be seen in Fig. 1.3.3 below. For this application, previously described enhanced daisy-chain connection is used for every branch.

A master controller module is responsible for gathering the data from the network and forwarding it for processing via Bluetooth or Bluetooth LE (low energy) connection. It provides clock signals and control for the whole network. A simplified flowchart of master controller module program is displayed in Fig.1.3.4.

This architecture provides the possibility to connect different kinds of sensors with different sampling and data rates to the same wearable sensor network.

1.3.3.1.2 Gathering wearable sensor data on a remote server

In various projects we propose systems, that are using mobile platforms in order to visualize, analyze and store data. Mobile platform was chosen taking its availability, ease of use and high computing power into account. Collected data

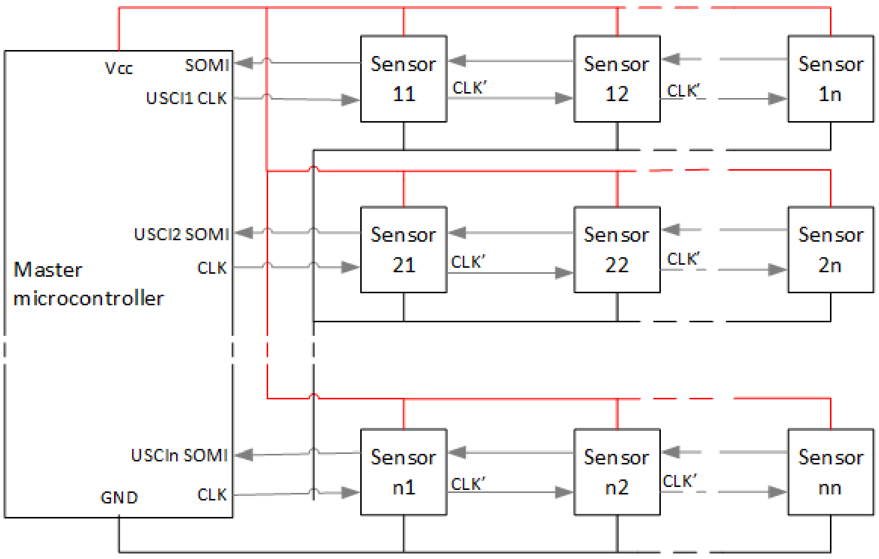
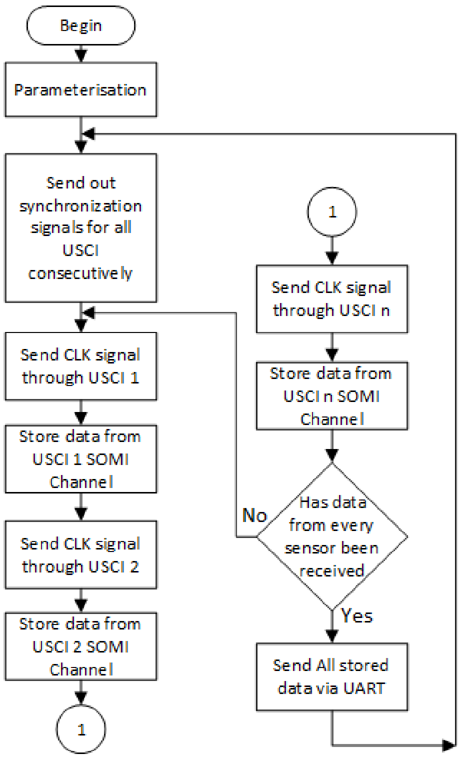


Figure 1.3.3: Multiple branch BSN SPI structure



Begin
Parameterisation
Send out synchronization signals for all USCI consecutively
Send CLK signal through USCI 1
Store data from USCI 1 SOMI Channel
Send CLK signal through USCI 2
Store data from USCI 2 SOMI Channel
1
1
Send CLK signal through USCI n
Store data from USCI n SOMI Channel
Has data from every sensor been received
No
Yes
Send All stored data via UART

Figure 1.3.4: Master controller code flowchart

on the smartphone can be stored in cloud infrastructure in order to provide data accessibility and security for both patient and health specialist. By using cloud services it is possible to provide communication in the scope of application between two parties: patient, doing rehabilitation and sharing his statistics with rehabilitation specialist, who can manage prescriptions and view data from patient.

1.3.3.1.3 Sensors for biomechanic

In this project, methods are being developed, that allow monitoring of human posture, different body part alignment and movements with wearable sensors in real time. In this reporting period a number of new approaches and methods for wearable sensor node signal processing were studied and implemented. Particular focus was aimed towards sensor fusion of inertial and magnetic sensors, which allows to obtain more accurate orientation leading to better measurements of human body biomechanics.

Acceleration and magnetic sensor network for shape sensing During State research program "IMIS" project no. 2 „Innovative signal processing technologies for smart and effective electronic system development" , an acceleration sensor network [34] was developed that allows to monitor shape of the fabric and can be used in smart wearable garments for posture monitoring [35].

A significant drawback arise from the fact that accelerometers alone can only measure direction of single reference vector, providing incomplete sensor orientation estimate. In order to obtain full 3D surface model another reference direction have to be measured such as Earth magnetic field.

Together acceleration (in static conditions - dynamic acceleration $<< g$) and magnetic sensors provide two vector observations, which is enough for full orientation determination, therefore, to obtain orientation data some deterministic algorithm such as TRIAD [36] can be used. Triad is one of the fastest, singularity free and computationally simple algorithms for orientation calculation from vector observations. This results in acquiring sensor orientation relative to Earth reference frame.

This allows the reconstruction of the shape of the fabric by approximating it to many rigid segments for which the orientation is known, as seen in the surface model in Fig. 1.3.5 and resulting in a 3D point cloud defining the shape of the body similar as in a 3D scanner.

Proof-of-concept prototype was designed to demonstrate feasibility of our system and validate performance. An architecture described in [33] was used to test surface with 63 sensor nodes. Data processing was implemented on PC in Matlab environment and as Java application. In addition, Android application was developed demonstrating ability to implement real-time system on portable devices with limited processing power (Fig. 1.3.6).

More detailed information about tests with experimental setup as well as computer simulations and evaluation can be found in [37].

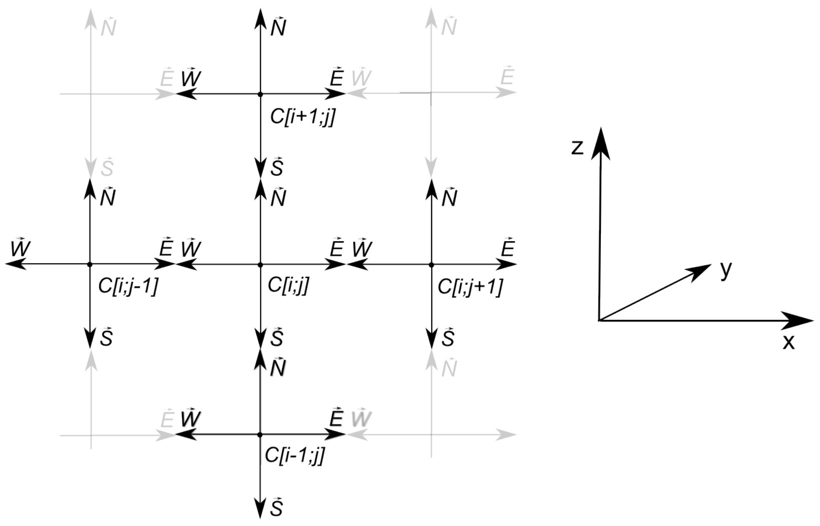


Figure 1.3.5: Surface segment structure. Each segment consists of center C and four direction vectors $\vec{N}$, $\vec{E}$, $\vec{S}$ and $\vec{W}$.

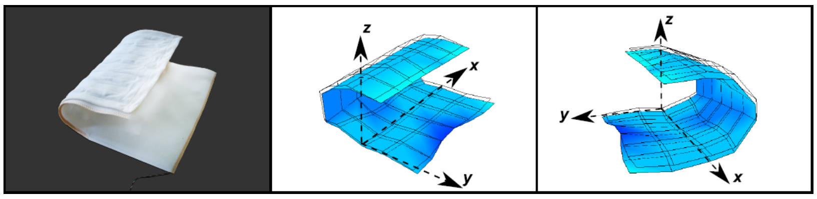


Figure 1.3.6: Sensor sheet and reconstructed shape rendered in Android application with 10 Hz frame rate.

Inertial/magnetic sensor module for motion tracking Fast dynamic accelerations as well as electromagnetic noise introduce high frequency noise in the reference vector measurements that are used for orientation estimates. This is a particular problem when fast human movements have to be measured which requires bio-mechanics model estimation in dynamic conditions. To eliminate this problem acceleration and magnetic sensor data are often fused with gyroscopes. For this reason an additional sub-task is dedicated for design of small/low-cost/low-power sensor node that can accurately estimate orientations in human bio-mechanics model during dynamic movements.

Several methods for acceleration, magnetic and gyroscope sensor fusion have been explored and a prototype was built using complementary filters [38] originally designed for applications in Micro Areal Vehicles. This algorithm was adapted for application in wearable sensors for human bio-mechanics measurement. A specific sensor node was designed consisting of 9-axis sensor (accelerometer/magnetometer/gyroscope) and microcontroller (Fig. 1.3.7, that can be connected to our previously used sensor architecture [33].

In Fig. 1.3.8 a demonstration of an early human hand bio-mechanics monitoring prototype can be seen.

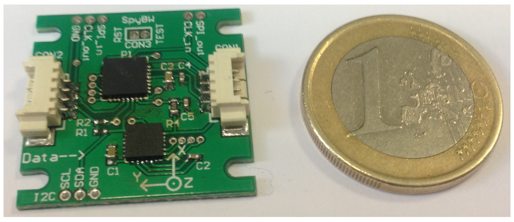


Figure 1.3.7: Prototype of wearable sensor node with accelerometer, magnetometer, gyroscope sensor module.

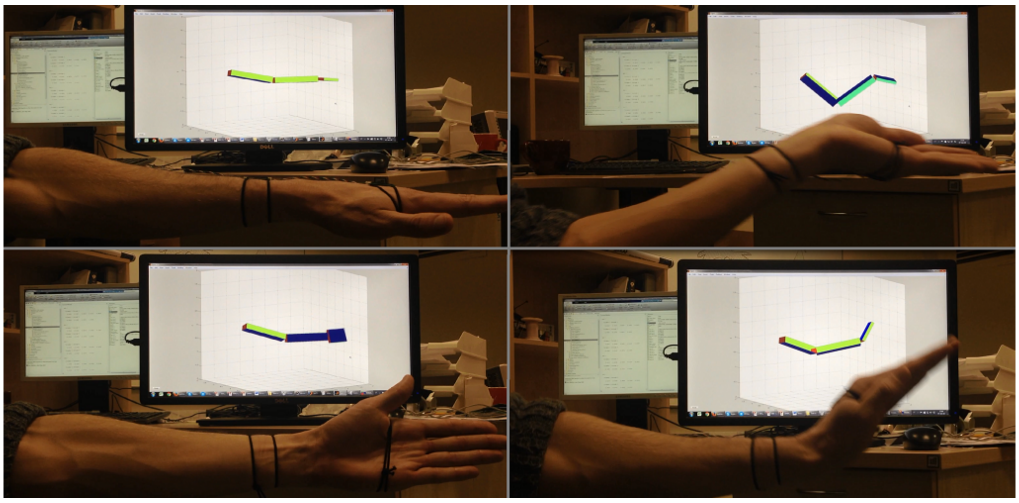


Figure 1.3.8: Human hand movement bio-mechanics reconstruction from wearable sensor node data.

Calibration of inertial/magnetic sensor modules A major drawback for low-cost MEMS inertial and magnetic sensor application for orientation estimation arise from imperfect sensor parameters such as scale errors, bias errors and axis cross sensitivity and misalignment. Work has been done to develop automatic calibration without external equipment, that would allow on-the-fly calibration during the use of the device. This approach can provide essential contribution to inertial/magnetic sensor systems applications in general by removing the necessity of cumbersome calibration routines, that are usually required before each use of the system. Results of these studies are expected during the next periods of the project.

1.3.3.2 Progress in 3rd period

1.3.3.2.1 Motion tracking visualization on Android with Unity game engine

For motion tracking system to be usable for physiotherapists or trainers the software for visualization was developed. Initially motion tracking data acquisition, processing and visualization was implemented in MATLAB as it allows rapid development and testing of algorithms. Physiotherapy specialists and trainers are usually not familiar with MATLAB computation environment, and also MATLAB framework is commercial software and in order for them to use software developed for this framework, they would need to purchase MATLAB license which is costly. To avoid that the software for motion tracking sensor data acquisition, processing and visualization was developed on Android OS. To run this software only device with Android OS is required, which allow system to be easily deployed for specialists that could be interested in motion tracking system.

Sensor data acquisition through Bluetooth and some parts of sensor data processing algorithms where already implemented using Android framework in Java previously for different applications which used sensor network architecture. To reuse existing code, architecture shown in Fig. 1.3.9 was implemented. It can be seen that Android framework is used for sensor data acquisition and processing, then the computed body segment orientations are transferred to UNITY engine, where 3D scene is rendered, and transferred back to android application, where scene is displayed on screen. The android application is illustrated in Fig1.3.10

1.3.3.2.2 Mobile device for ECG monitoring

Abstract Judging from Disease prevention and control center data the main cause of death in Latvia still is the heart and vascular diseases. The most important heart and vascular disease test is Electrocardiogram (ECG). There are three different ways to do this test – stationary, load, and long term. The most common is stationary, which is also the most accurate, but the length of the test is less than a minute. Load test is for patients who complain about problems when doing some physical activities. Long term test is for patients who need to

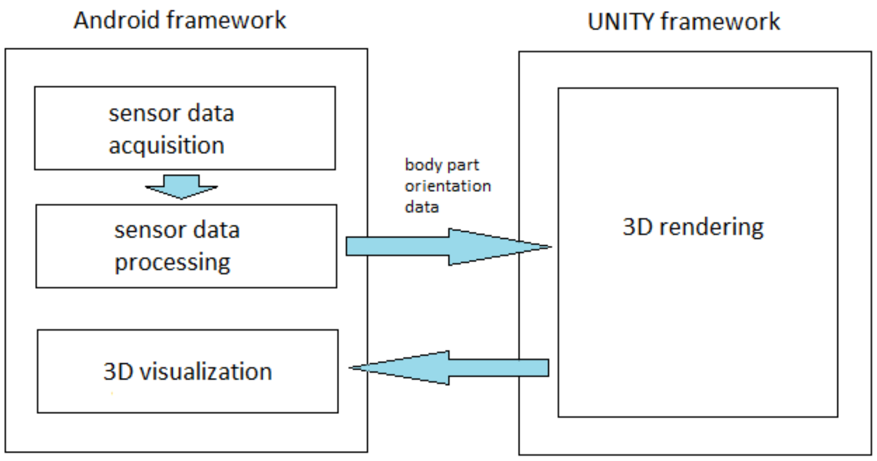


Figure 1.3.9: Motion tracking visualization software diagram on Android using Unity

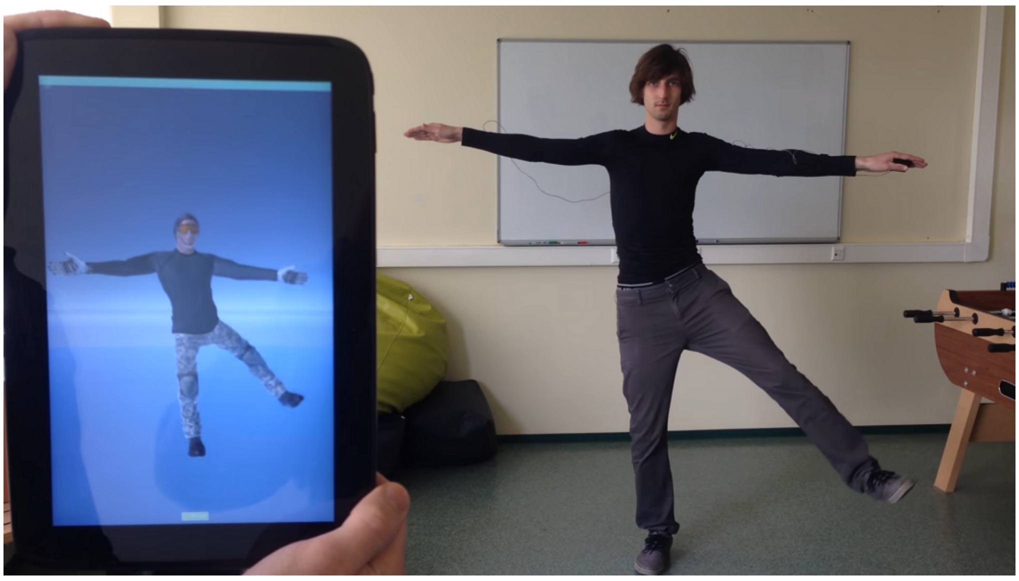


Figure 1.3.10: Motion tracking Android application

monitor heart for longer periods, for example 24 hours. Using Holter system for these long tests is uncomfortable for patient because it is necessary to deliver this system back to doctor for analysis. As a part of our MedWear sub-project we are developing a sensor for patients heart monitoring with potential to integrate it in the overall smart clothing architecture and gain the added benefits of activity monitoring together with ECG monitoring. We are making an embedded system that is easy to use for the patients, doctors and supervising staff. The collected data is sent to the database over the air using the advantages of wireless sensor networks. Doctor can monitor patient's health from distance and decide what to do before the system is even returned. Human movement and environmental factors are interfering with data that is why there is a need for data analysis to filter noise and get more believable results. Using other MedWear sensors we can monitor patients movements and predict noise that needs filtering.

Introduction Electrocardiogram (ECG) is the most important test that is used for patients with potential heart diseases. Wilhelm Einthoven was the first who invented this system in 1903, later in 1924 he got the Nobel Prize in medicine for his invention.

The same method is still used nowadays. Different electrical signals from heartbeats are monitored using electrodes that are connected to patients flesh. ECG test provides important information about inner workings of the heart and helps diagnose serious heart rhythm disturbances. To use this test firstly patient has to have some complaints about dizziness, blackouts, palpitations etc. ECG is important for patients of all ages, because it provides important information about patient's health.

Most common is the stationary test that is concluded in doctor's cabinet and is no more than a minute long. That means that there is a small chance that in this moment doctor can see the real problem with a patient's heart for example arrhythmia, if it appears only once in a week. To get more accurate readings doctor sends patient to get Holter monitoring system which monitors patient's heart for at least 24 hours. This system lets patient go on with his everyday life while it collects the data. In addition patient has to fill a diary about his feelings, like dizziness, coming blackout, or even an argument with a neighbor. Using this system patient has some discomfort, because, patient can't get wet not to break the system. But the biggest discomfort of all is that patient has to deliver the system to doctor on his own, so that the data can be read and analyzed.

Previously we reviewd various producers of ECG and holter devices and concluded that all producers of ECG and holters at the same time are the direct sellers of their own production first and only then their production has been delivered by other dealers.

Our group made our first prototype Mobile ECG monitoring device based on CC3200 ARM Cortex M4 Microcontroller, with embedded WiFi chip, for data streaming via network channels. At the heart of our device data acquisition is ADS1292R analog to digital converter, that is build for the specific purpose of gathering various heart signals, and amplifieing them with programmable gain starting from 1 going up to 12.

As the first prototype board was finished it was subjected to various tests starting from basic Ground and Power supply inputs, ADC's internal square wave tests and heart rate signal simulator circuit inputs. The tests and the results are explained below.

First prototype mechanical debugging As the first prototype was completed we encountered couple of problems associated with its hardware design:

- **Flash memory chip not working** Used to store program code. Solution was to tie the HOLD pin to power supply line;
- **SPI signals not reliable.** Used in communication with CC3200 and ADS1292R. Solution was to add couple of pull-up resistors to each of SPI data pins and pull-down resistor to clock pin;

First prototype tests For the first test, all input channels were down, ADS1292R generated test signal by it self and sent it over SPI. As was expected, the CC3200 could gather data from ADS1292R over SPI and send them to the PC. At this moment we was not able to monitor the form of the signal. Using processing language and Processing IDE was written UART plotter which could draw the receiving data and save then in .txt file.

Next task was, to apply some output signal to the ADS1292R inputs and draw corresponding graph on the PC. Sending to the ADS1292R square wave with 1kHZ frequency and 2.5V amplitude, we were able to observe the save square wave on the UART plotter on the PC.

But, according to that the heart generates about 5mV, which is several orders of magnitude lower than the previous test signal amplitude, next test signal have to have amplitude about 5mV. The minimal amplitude of frequency generator, which is at our disposal, is 50mV. The parameters of the following test signal are 1kHZ frequency and 50mV amplitude. The waveform of the plotted signal reminds nothing but noise, as shown on the figure 1.3.12.

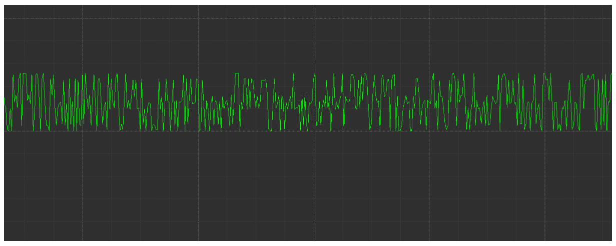


Figure 1.3.11: 1kHZ, 50mV signal after digitization

Unfortunately, the noise level of the surrounding space is commensurable with the heart generated signal. The solution was to reconfigure ADS1292R, put higher gain on the first channel, enable reference buffer and change data rate to 125SPS. After reconfiguration, we were able to observe the signal without noise.

For the following tests were used the ECG Simulator which could generate sine wave with 10/60/100HZ frequency, square signal with 0.125hz and 2 hz, 2hz saw tooth signal, and hear beat signal with 30/60/120/240 heart beats per second with 2mV amplitude.

As shown on the figure 1.3.13, the green graph represents the signal from first channel and the yellow one from the second. The same tests were made for the following signals: square signal 2HZ, 2mV; saw tooth 2HZ, 2mV; heart signal with 30/60/120/240 bpm. If the output square signal was more or less commensurable with the input, the saw tooth brought more dissonant results.

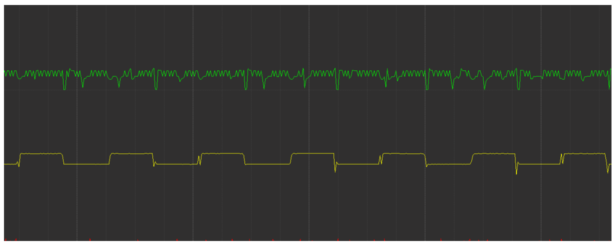


Figure 1.3.12: 1kHZ, 50mV square signal after digitization, ADS is reconfigured. Green - CH1. Yellow - CH2

As shown on the figure 1.3.14, the yellow signal is clear, but the form of the signal doesn't reminds the saw tooth. However, it looks like the signal is separated in three pars, that could be because of the overflow.

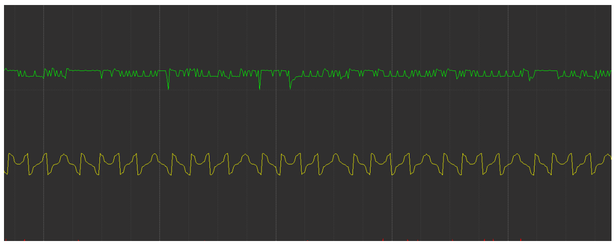


Figure 1.3.13: 2HZ, 2mV saw tooth signal after digitization, Green - CH1. Yellow - CH2

The same problem with overflow, we could see on the following figure 1.3.15, where is displayed the digitized heart beat signal at 60bpm.

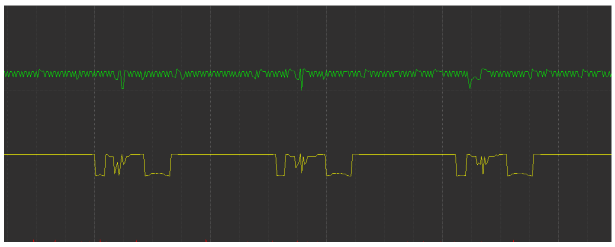


Figure 1.3.14: 60bpm heart beat signal after digitization, Green - CH1. Yellow - CH2

Besides problems related with the digitization, data storage problems have been found. The CC3200 cant store this amount of data that we gonna receive from the ADS1292r, so that means that we have to use external storage. All SDcards, which could be used as the external storage are using SPI interface to write and read data. But, CC3200 have only one SPI communication interface, and it is used by the ADS1292r already. There are two solutions - the first one is to share SPI interface between SDcard and ADS1292, but that means that we could not be able to read data from ADS and write them to SDcard as the same time, as well as we could not be able to send data over WiFi to the database and read data from the ADS. The second solution is to change CC3200 to another microprocessor with same parameters and 2SPI interfaces.

1.3.3.2.3 EMG device in wearable network system

In previous period of this project we adopted commonly used EMG amplifier scheme with instrumental amplifier and actively driven ground and successfully tested it with dry contact electrodes to obtain EMG signal from eye squeezing muscle during subsequent eye squeezes.

In this period of the project we examined and tested three signal processing approaches which have been proposed for detecting the on and off timing of the muscle:

- single-threshold method with raw and filtered amplitude,
- double-threshold method described in [39],
- detector matched to MUAP (muscle activation potential) shape described in [40].

We tested all three methods with EMG signal acquired from eye squeezing muscle. The results are depicted in fig. 1.3.15.

All examined methods were able to successfully detect all 4 eye squeezing events, but computation time for double threshold method was significantly longer because of methods computational complexity. Measured computational timing in Matlab for all examined methods with our test signal (5000 samples) is listed bellow:

- single-threshold method (raw amplitude): 0.219298 s,
- single-threshold method (filtered amplitude): 0.203529 s,
- double-threshold method: 43.578387 s,
- detector matched to MUAP shape: 0.242128 s.

Detector matched to MUAP shape was slightly slower than single-threshold detector, but still fast enough for real time applications and it also yielded better sensitivity to EMG because of filter matching to the MUAP shapes.

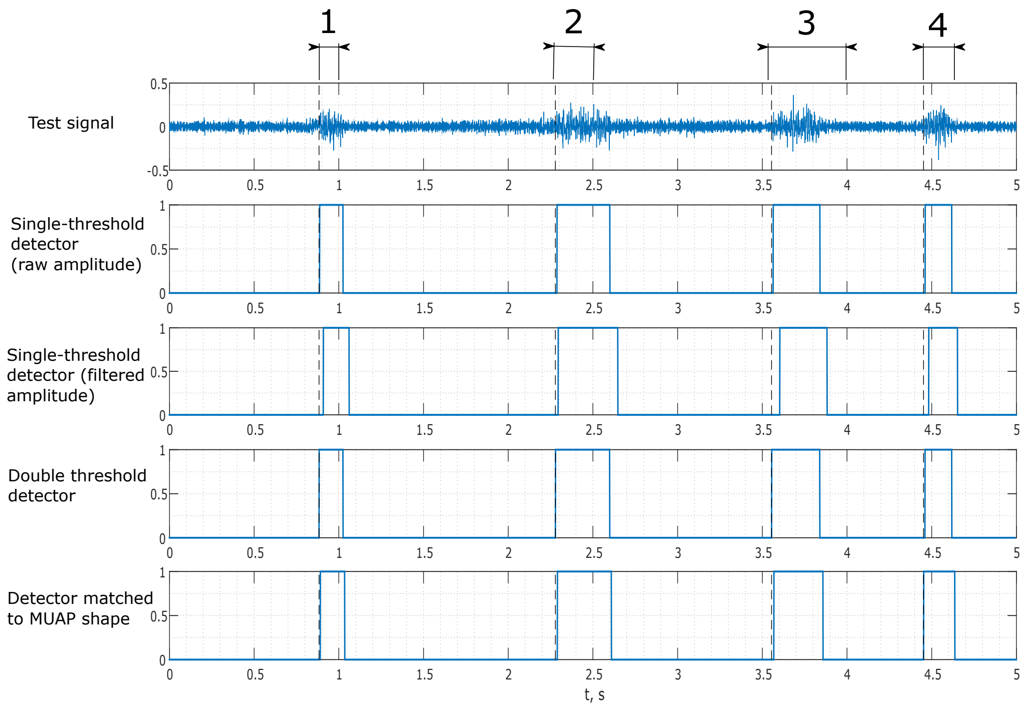


Figure 1.3.15: EMG detection approach comparison

We combined head device for alternative communication developed in previous period of this project with EMG detector matched to MUAP shape to supplement it with acceptance switch. A wearable prototype headband which contained EMG electrodes, accelerometer sensor and and data acquisition PCB with Bluetooth module (fig.1.3.16) was designed to transfer EMG samples and accelerometer data in real time to a remote device for data processing. We created a test application in Matlab (fig. 1.3.17) which used accelerometer data to dynamically change object position in two dimensional coordinate (algorithm) plane and displayed the number of EMG signal detector recognized eye squeezing events. Matlab algorithm was tested with different detector thresholds during 100 subsequent eye squeezes. Results are depicted in fig. 1.3.18. They show that minimum number of errors occurred when threshold is set to 0.2 V, but optimal threshold depends on application device is used.

We adopted Matlab application to run on STM32F401VC ARM©Cortex©-M4 32-bit RISC microcontroller and combined it with nRF51882 Bluetooth Low Energy (BLE) chip. Then the BLE HID-over-GATT profile (HOGP) was used to emulate HID (human interface device) pointer using accelerometer data to calculate absolute cursor position and EMG signal from eye squeezing muscle to generate ”click” event. As a result our developed prototype is recognized as HID pointer device in operating systems supporting Bluetooth v4.0 or later without additional software or drivers.

Further work will be concentrated on minimizing system dimensions to implement it into easy to use wearable sweatband and textile electrode

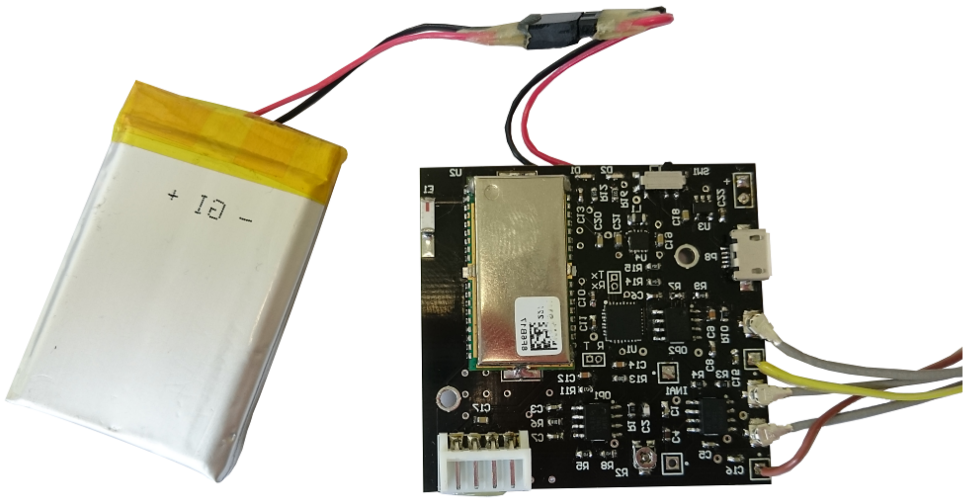


Figure 1.3.16: PCB for EMG and accelerometer data acquisition

implementation and testing. Also, complementing accelerometer data with data from magnetometer and gyroscope, more precise and convenient cursor control algorithm may be achievable.

1.3.4 Applications of technologies developed in the project

The technologies developed in this project has already been applied in several applications, initiated both by our research institute and approbated with external institutions, and also developed as contract research using knowledge and results reached in this research.

In the next subsections main examples of these applications are described.

1.3.4.1 Head position monitoring in therapy

There are patients with cerebral palsy or people after serious injuries who have difficulty maintaining vertical head position. In rehabilitation there are therapies to improve patient ability to maintain normal body position. Therapy with patients who suffer cerebral palsy can be problematic, because communication with some patients is troublesome or almost impossible. In collaboration with rehabilitation center ”MEL” idea emerged for system that

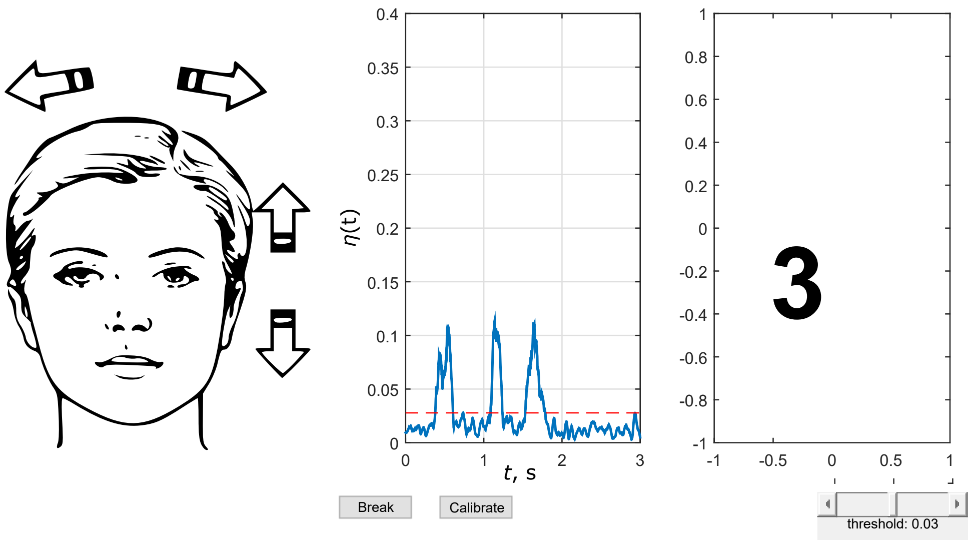


Figure 1.3.17: Matlab application controlled with head movements and eye squeezing

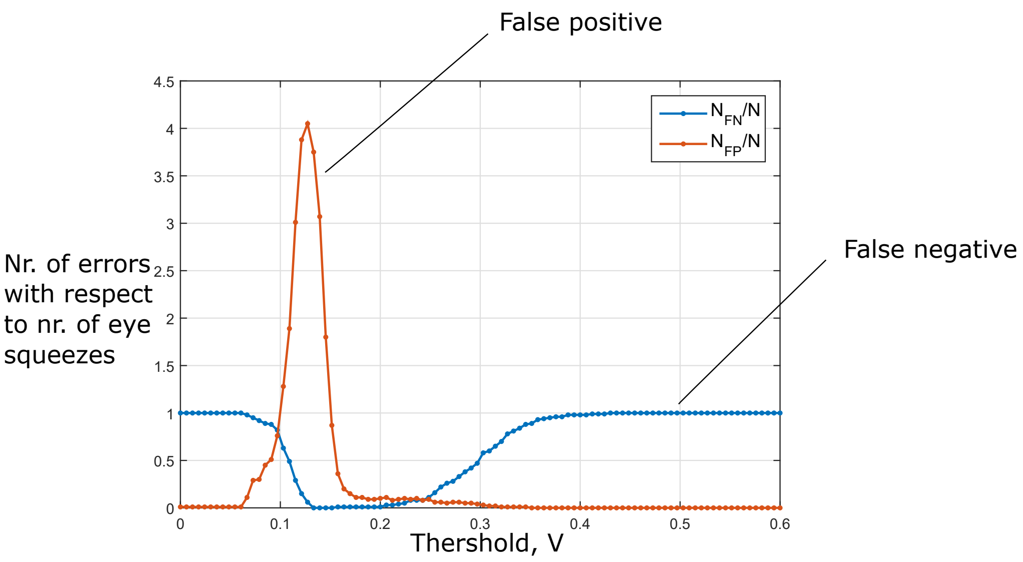


Figure 1.3.18: Detector errors with different detector threshold

could improve rehabilitation process for head position therapy. The system consists of previously developed accelerometer/magnetometer sensor device node using the same architecture for data transmission to smart phone or tablet. Sensor module consists of accelerometer and magnetometer sensors, bluetooth communication module for data transmission to processing device and battery. Module is attached to wearers head with elastic head band as seen in Figure 1.3.19. By using sensor integrated in module it is possible to estimate tilt of the

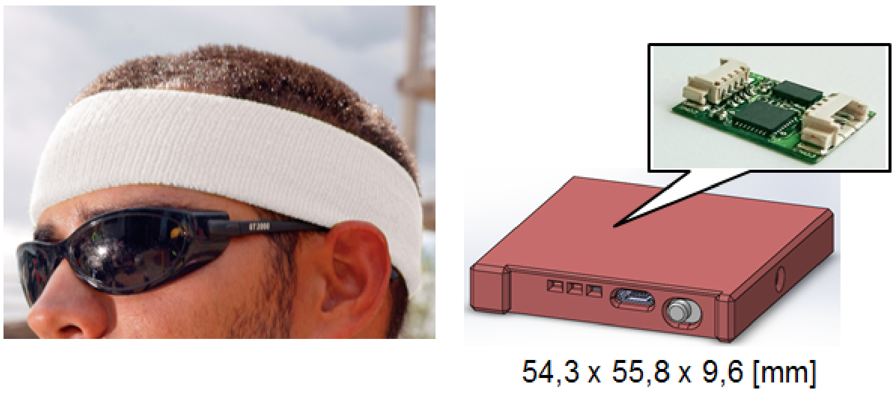


Figure 1.3.19: Sensor module for head position monitoring.

head relative to gravitation vector of earth.

Also specialized software was developed with simplified graphical user interface that provides feedback about head position to the wearer(Figure 1.3.20). The object, seen in application screenshot depicted in Figure 1.3.20, is controlled by wearers head movements. System is calibrated for preferred position of the head before the session. During session the task for the wearer of the sensor is to maintain the object in the marked area in the center of the screen. If object leaves marked area, feedback is triggered changing background of the screen to red color and producing sound alert. This simplified approach allows to perform therapy on patients who are hard to communicate with.

1.3.4.2 Head device for alternative communication

As mentioned in previous section, there are patients that are problematic to communicate with. For this reason the idea for system providing means of alternative communications emerged. Previously described system can be used as computer mouse like human-computer interface device, that is controlled by head motion.

In Figure 1.3.21. screen shot of smart phone/tablet application is shown used as tool for alternative communication. The application consists of moving object (yellow face) and two coloured regions (red and green). Regions represent two options: yes/no type answers. The medical staff can ask yes/no type questions, and respondent can move the object to region representing corresponding answer by moving his head. In current implementation answer is accepted if yellow face marker has been in answer region for specified amount of time. The accept of

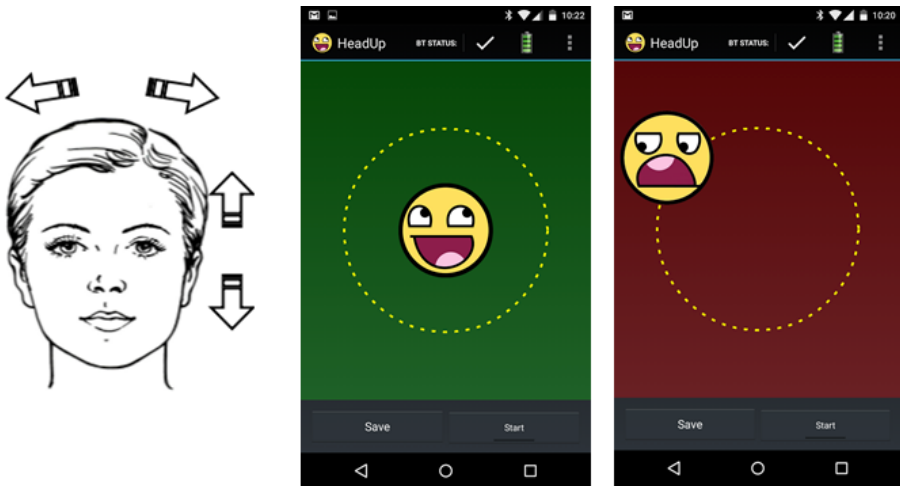


Figure 1.3.20: Head position monitoring application screenshot.

answer is represented with sound alert. The future implementation will have answer acceptance by using eyelid blinking using electromiography.

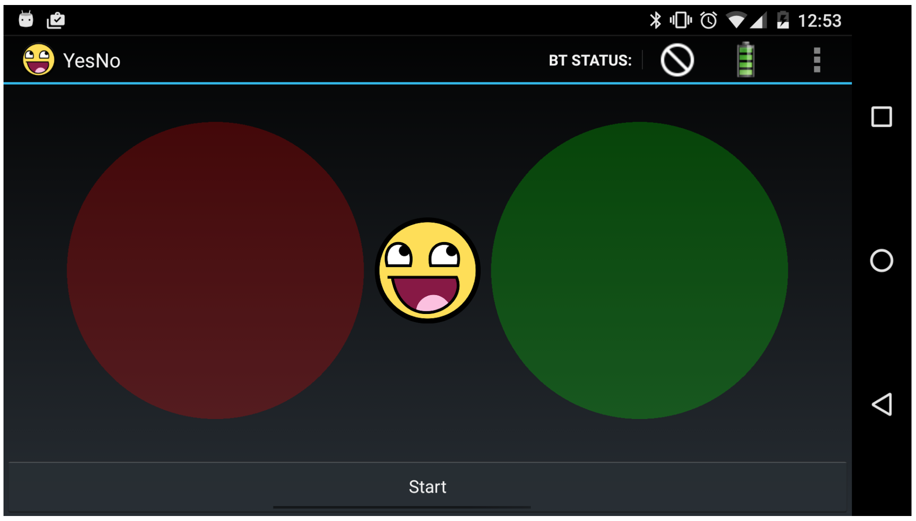


Figure 1.3.21: Application for alternative communication.

1.3.4.3 Posture and head position monitoring in therapy

To obtain combined information about patients head position and posture, prototype was developed in collaboration with rehabilitation center ”MEL”, that combines head position and posture monitoring functions. Specialized Vest was developed embedding 20 sensor nodes for posture monitoring and additional sensor for head position monitoring. Specialized application was developed for posture and head tilt monitoring providing feedback for the wearer of the system. Application also provides with data logging feature, that allows the data later to be visualized to provide better insight of the session as seen in Figure 1.3.22.

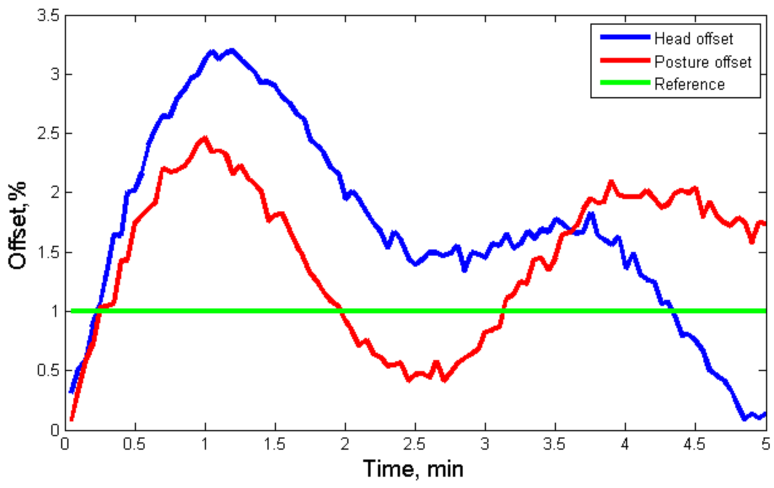


Figure 1.3.22: Head position and posture data visualization.

1.3.4.4 Knee device

Knee joint is the largest joint in the body, and one of the most easily injured. Knee injury is one of the most common reasons people see their doctors. In 2010, there were roughly 10.4 million patient visits to doctors' offices because of common knee injuries such as fractures, dislocations, sprains, and ligament tears. We propose a combination of a wearable sensor system and a mobile application in order to help patients successfully complete rehabilitation procedure. Wearable system consists of 4 sensor nodes used for knee joint flexion/extension angle calculation. After successful vital signs data acquisition from sensor nodes, data is being transmitted to mobile application. As part of the project a mobile application was developed to make calculations, analyze collected data from sensor nodes, store and visualize it 1.3.23. Another important functionality of application - communication with a patient using developed notification system. Notification system was implemented based on health specialist – patient communication style aiming to create a similar feeling of safety when a patient is near physiotherapist. Patient is notified when exceeding the flexion limit threshold and also when reaching the end of rehabilitation session. Developed solution was tested in dynamic conditions – patients used the system during their rehabilitation session in real life. System received positive feedback from participants and rehabilitation specialists 1.3.24. During rehabilitation session developed prototype data was compared to industrial digital Precision of the implemented solution was calculated: 0.79 degrees, fulfilling the requirements received from doctors on current condition.

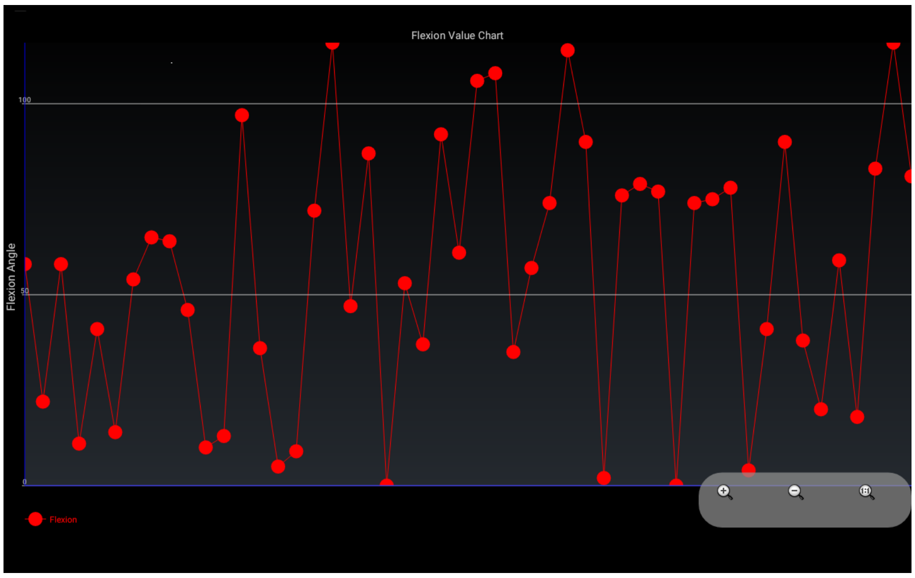


Figure 1.3.23: Knee flexion data visualization.

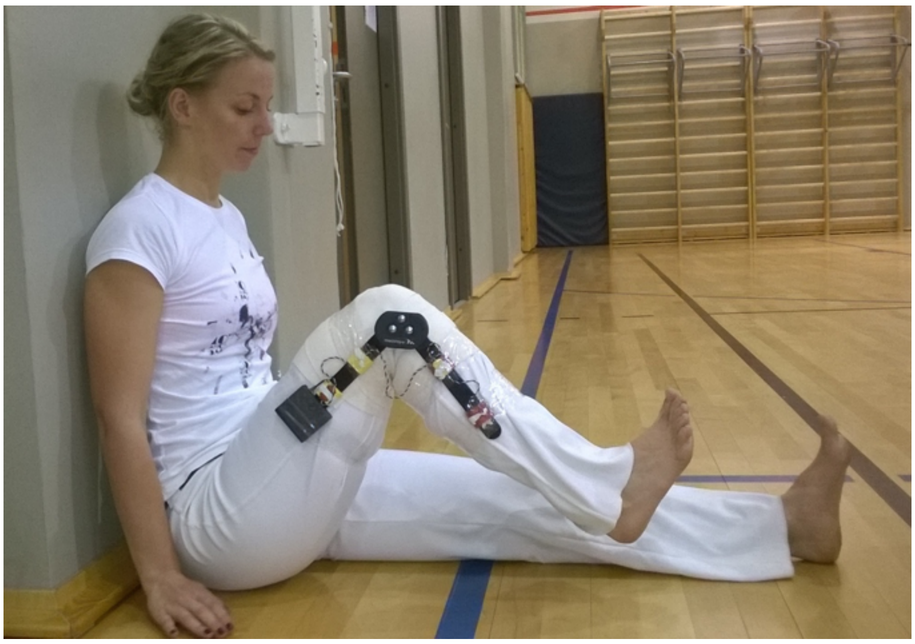


Figure 1.3.24: Patient participates in system prototype testing.

1.3.4.5 BT Smart Inhaler

Asthma patients use inhalers to deliver medication into the body via lungs. To help them to comply with their medication schedule and improve treatment, we proposed to use BT Smart technology to send notifications from inhaler to a smartphone or a tablet.

We developed and tested a proof-of-concept prototype using TiWi-Ub1

BT Smart module with minimum circuitry required for device programming, registering push events and sending notifications to BT Smart enabled device (fig. 1.3.25). We also made an Android application, which logs received notifications and assigns corresponding time stamps (1.3.26). In current consumption test, logging averagely 4 push events every day, device could continuously operate for two weeks powered by two AA batteries in series.

In future we are planning to improve device and application power efficiency to extend battery life to more than a month and implement time tracking into the inhaler device to eliminate necessity of connection to master device during push event to assign valid time stamp.

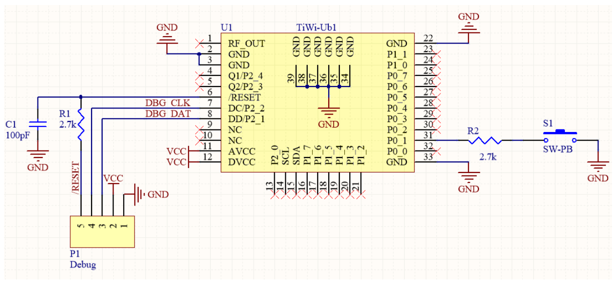
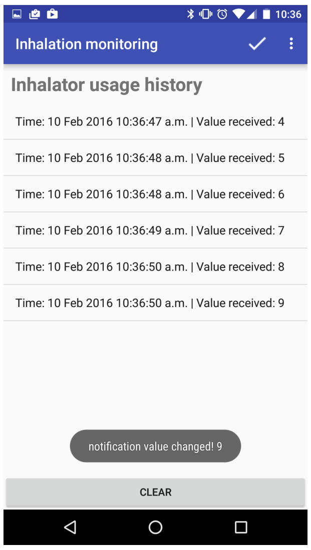


Figure 1.3.25: Proof-of-concept prototype schematic of inhalator monitoring device using TiWi-Ub1 module



Inhalator usage history

Time: 10 Feb 2016 10:36:47 a.m. | Value received: 4

Time: 10 Feb 2016 10:36:48 a.m. | Value received: 5

Time: 10 Feb 2016 10:36:48 a.m. | Value received: 6

Time: 10 Feb 2016 10:36:49 a.m. | Value received: 7

Time: 10 Feb 2016 10:36:50 a.m. | Value received: 8

Time: 10 Feb 2016 10:36:50 a.m. | Value received: 9

notification value changed! 9

Figure 1.3.26: Inhalator monitoring device Android application screenshot.

1.3.4.6 Palm prosthesis dynamics monitoring in therapy (in cooperation with Wide.Tech)

During this project a wearable sensor system for prosthesis motion tracking in order to improve rehabilitation process was proposed. We developed a sensor system for prosthetics that will be able to track angle of a joint flexion/extension in real time. Acquired data will be used to monitor rehabilitation process and patient progress. For palm prosthesis flexion/extension angle calculation was used a network that consists of two 3-axial accelerometers and magnetometers on-board. Special cases were designed and printed to ensure sensors, master board and battery protection (mechanical influence etc.) and calculation precision 1.3.27. Position of sensors can influence on received data precision – sensor nodes should be located on one straight line (one plane). Sensors should be attached to the prosthetic in a way to minimize free movement inside the printed cases. As part of the project a mobile application was developed to make calculations, analyze collected data from sensor nodes, store and visualize it. Cloud services were used to provide data storage and communication between patient and health specialists, so that patient will be using up-to-date rehabilitation recommendations and health specialist will be informed on actual status of patient's rehabilitation progress. In the scope of this project, prototype version uses Microsoft Azure cloud services 1.3.28. Proposed prototype version was created in collaboration with start-up company "Wide.Tech", participating in "Fabulous" EU acceleration program.

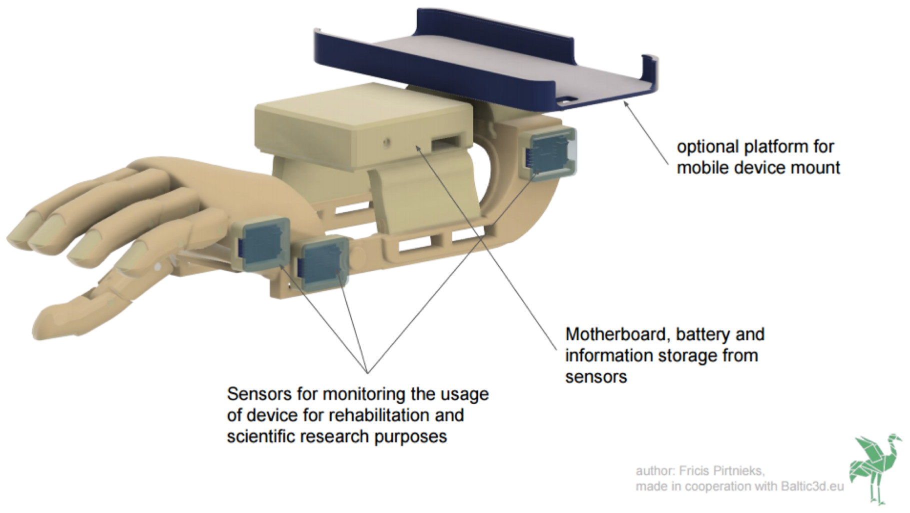
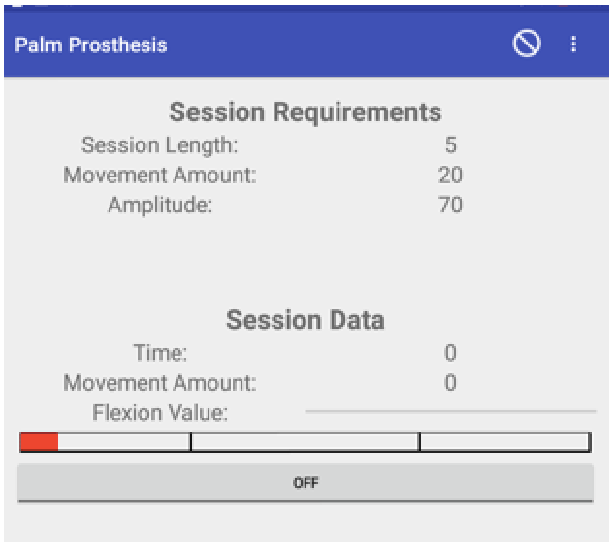

optional platform for mobile device mount
Motherboard, battery and information storage from sensors
Sensors for monitoring the usage of device for rehabilitation and scientific research purposes
author: Fricis Pirtnieks, made in cooperation with Baltic3d.eu

Figure 1.3.27: Palm prosthesis model with sensors attached.

1.3.5 Results

During the first three stages of the project a wearable sensor system is being developed, by both improving the architecture and developing new sensors for use in such system. In addition several of these results have been promoted to general public through dissemination efforts, and some have already attracted interested companies, thus resulting in contract research producing prototypes for specific applications.

Results of this research has also been presented in several conferences and


Palm Prosthesis

Session Requirements

Session Length:	5
Movement Amount:	20
Amplitude:	70

Session Data

Time:	0
Movement Amount:	0
Flexion Value:	

Figure 1.3.28: Prototype version of mobile application for palm prosthesis dynamics monitoring.

research papers, as described in the main report, to which this report is appended to.

1.3.6 Discussion and future work

In the final stage of the MedWear project the main goal is to further improve the data gathering architecture and research the potential uses of added sensor data for patient monitoring, such potential for motion/activity data to improve usefulness of gathered ECG results, or EMG data to improve the overall activity monitoring results in rehabilitation, and demonstrate these technological improvements in prototypes for actual applications.

Authors see system future research as a part of global human vital signs monitoring solution – developed device can be used in synergy with other systems to get more data and provide them to health specialist to better diagnostics. We are planning to create cloud based solution to analyze data from various patients with sensor nodes attached, in order to define risk factors of rehabilitation process.

Chapter 1.4

SmartCar - Intelligent transport systems

1.4.1 introduction

Every year, many car accidents due to driver fatigue and distraction occur around the world and cause many casualties and injuries. Various studies have suggested that around 20% of all road accidents are fatigue-related, up to 50% on certain roads[41][42].

To reduce the number of these incidents, generally improve road safety and improve driving convenience Advanced Driver Assistance Systems (ADAS) can be used.

In this project existing ADAS systems were reviewed and analyzed, and several research directions were selected in which potential improvements can be achieved through research in the confines of this project. Also to test and validate the results of this research a test platform (self-driving car) with an innovative control platform was developed which will be used not only for running real-life tests, but also for validating the devices in actual competition environment in Grand Cooperative Driving Challenge (GCDC).

Below sections describe the current state-of-the art, our vehicle control/test platform architecture and specific research in ADAS systems.

1.4.2 Background and state of the art

Because many road incidents are a cause of driver error or drowsiness a significant part of ADAS systems is dedicated to driver monitoring and providing additional information to the driver, that normally is not accessible.

Some of the current systems learn driver patterns and can detect when a driver is becoming drowsy. Various technologies can be used to try to detect driver drowsiness[43] and generally assist driver in driving more safely and comfortably, thus improving road traffic safety through intelligent transport system technologies:

- **Steering Pattern Monitoring.** Primarily uses steering input from electric power steering system;
- **Vehicle Position in Lane Monitoring.** Uses lane monitoring camera;
- **Driver Eye/Face Monitoring.** Requires a camera watching the driver's face[44];
- **Physiological Measurement.** Requires body sensors for measure parameters like brain activity, heart rate, skin conductance, muscle activity;
- **Thermal imaging.** Requires thermographic camera to increase a driver's perception and seeing distance in darkness or poor weather.

1.4.2.1 Steering behavior

Features of steering wheel movement can be divided into time, frequency and state space domains. This assignment follows the first processing step of computing frame level descriptors, independent of feature characteristics of the second, contour describing, functional based processing step.

Time domain features: Within the time domain the following features can be extracted: regression descriptors (e.g. regression slope, intercept, maximum of regression error), class distribution measures (e.g. number of values within steering angle bin 0.0-0.1), peak amplitudes and distances (e.g. mean distance of peaks; maximum of peak amplitude), entropy, zero crossing distances and slope (e.g. maximum of distance between consecutive zero crossings; mean velocity of steering angle in zero crossings)

Frequency domain features: It is possible to obtain a set of frequency-related parameters. Usually a signal processing is applied to obtain signal spectrum which is evaluated in a frequency regions from which the descriptors are extracted. Frequency area is generally defined depending on the application. It is important how many samples will be used to calculate the spectrum of the signal and necessary descriptors will be extracted. Spectral features include: DCT coefficients, statistical moments, the standard deviation, the zero crossing frequency, etc.

State space: To extract the nonlinear properties of the steering angle signal, a three-dimensional state space (steering wheel position, steering wheel velocity, steering wheel acceleration) is computed, and a phase space is reconstructed. The geometrical properties of the resulting attractor figures can be described by trajectory based descriptor contours (angle between consecutive trajectory parts, distance to centroid of attractor, length of trajectory leg). The temporal information of the contours can be captured by computing functionals.

1.4.2.2 Vehicle Position in Lane Monitoring

The goal of the image processing is to extract information about the position of the vehicle with respect to the road from the video image. Two major processes are usually implemented: the pre-processing process and then the lane detection process. The goal of pre-processing is to remove image noise and make the images sharper. The goal of the lane detection is to detect the desired lane of the vehicle in order to obtain the look-ahead distance and the lane angle. This process is based on the real-time data of video sequences taken from a vehicle driving on the road. The processing steps of the example lane detection algorithm could be: image segmentation, edge detection, Hough Transform, and lane tracking.

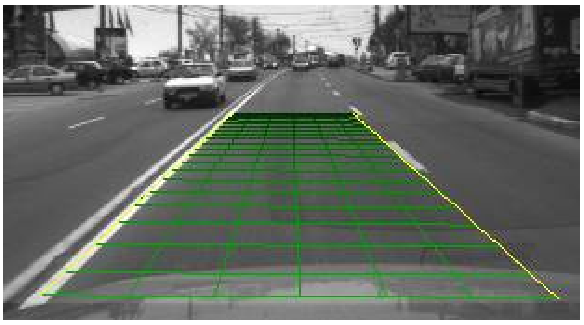


Figure 1.4.1: Lane tracking

1.4.2.3 Driver eye/face monitoring

Driver face monitoring: The driver face monitoring system is a real-time system that investigates driver physical and mental condition based on processing of driver face images. Driver status can be detected from eyelids closure, blinking, gaze direction, yawning and head movement. This system will alarm in hypo-vigilance states such as drowsiness, fatigue and distraction. The systems based on the driver face monitoring can be divided into two general categories. In first category, driver fatigue and distraction is detected only by processing of eye region. There are many researches based on this approach. The main reason of this large amount of researches is that main symptoms of fatigue and distraction appear in driver eyes. Moreover, processing of eye region instead of total face region has less computational complexity.

In the other category, symptoms of fatigue and distraction are detected not only from eyes, but also from other regions of face and head. In these approaches, not only activities of eyes are considered, but also other symptoms such as yawning and head orientation is extracted.

Software is the most important part of driver face monitoring system and is divided into two main parts: image processing algorithms and decision-making algorithms.

The main goals of image processing algorithms include preprocessing, detection and tracking of face, eyes and other facial components, and extraction of appropriate symptom from facial images.

After extraction of appropriate symptom from images, decision-making algorithms determine the level of driver alertness based on extracted symptoms. Finally, an appropriate output is generated for the system. In most of the driver face monitoring systems, face detection is the first part of image processing operations. In systems that fatigue and distraction is detected based on processing of facial region, face detection is considered as very important part of the system. Also in most of methods based on processing of eye region, due to difficulty of eye detection directly, face is detected at first and then eyes are detected. The most important problems of face detection are[45]:

- In-plane face rotation,
- Out-of-plane face rotation,
- The presence or the absence of makeup, beard and glasses,
- Mental conditions (happiness, crying, and etc.),
- Illumination conditions,
- Covering part of the face with an object,
- Real-time processing.

Face detection methods can be divided into two general categories[45]: (1) feature-based and (2) learning-based methods. Learning-based methods usually more robust than feature-based methods, but they often take more computational resources. However, these methods can achieve a detection rate about 80-90% or higher in laboratory conditions, but both of them usually fail in real conditions, especially in night light.

In feature-based methods, the main assumption is that face in the image can be detected based on some simple features, independent of ambient light, face rotation and pose. These methods are usually used for detection of one face in image. In noisy image or the environment with low illuminations, these algorithms have low accuracy[45].

Learning-based methods deal with face detection using a number of training samples. These methods benefit from statistical models and machine learning algorithms. Generally, learning-based methods have less error rates in face detection, but these methods usually have more computational complexity. Viola et al.[46] presented an algorithm for object detection, which uses very simple features named Haar-like features. In this algorithm, many Haar-like features are extracted from the image, and a number of effective features are selected using AdaBoost algorithm, and then these features are processed in a hierarchical structure similar to the decision tree. Due to the simple extracted features and selection of the best features, this algorithm is relatively fast and robust.

Driver eye monitoring: In all driver face monitoring systems, eye region is always processed for symptom extraction, because the most important symptoms are related to the eyes activity. Therefore, eye detection is required before processing of eye region. Eye detection methods can be divided into three general categories: (1) methods based on imaging in IR spectrum, (2) feature-based methods and (3) other methods.

Seeing Machines builds image-processing technology that tracks the movement of a person's eyes, face, head, and facial expressions. They've developed an ADAS technology which monitors driver fatigue and distraction events in real time, enacting an intervention strategy that improves driver and environmental safety.

Figure 1.4.2: Eye tracking device developed by Seeing Machines

1.4.2.4 Relationship between Physiological Signals and Drowsiness

Bioelectricity is generated on the cell level and acts as the charge flow on human surface. The electrical charges on the skin off the chest are mainly caused by the depolarization of heart muscles during each heartbeat cycle. In each cycle, nerve excitability is triggered by sinoatrial node, and then spreads through atrium, intrinsic conduction pathways and ventricles. As a result, it causes the change of action potential in cells manifested as the form of tiny rises and falls of potential on body surface. The electrical activity of heartbeat cycle is adjusted rhythmically by central and peripheral nervous system. Fatigue causes changes in spontaneous rhythmic activity, breathing, cardiovascular reflex activity, blinking, nodding, etc. The comprehensive regulation of these changes by the central nerve system will finally cause changes in the physiological signals.

A few physiological signals of drivers have been found to be good drowsiness indicators. It is generally believed that fatigue is the behavior of the central nervous system. When stress response of organs occurs during fatigue, cardiovascular nervous system will adjust accordingly. Therefore onset of fatigue causes changes in the bioelectrical signals, such as the electrocardiogram (ECG), a recording of electrical signals produced by the electro-dynamic functioning of the heart.

The methods for drowsiness identification based on ECG signal include Heart Rate (HR) analysis, Heart Rate Variability (HRV) analysis and amplitude analysis of T wave.

In period 1 and 2 of our research project we chose to use drivers face monitoring, because of the ease of use, non-intrusiveness and high reliability, based on the previous research results and commercial success.

1.4.2.5 Thermal imaging

An automotive night vision system uses a thermographic camera to increase a driver's perception and seeing distance in darkness or poor weather beyond the reach of the vehicle's headlights. Such systems are offered as optional equipment on certain premium vehicles. It is crucial to detect persons or animals on or on the side of the road before they appear in the drivers viewing range. This gives additional time for a driver to react and eliminate the accident. Many research groups have been working on this problem using methods like basic background subtraction, sliding window, keypoint detection and other approaches. In [47] researchers propose to use Maximally Stable Extremal Regions (MSER) to detect hot spots, verify detected hot spots using a Discrete Cosine Transform (DCT) based descriptor and a modified Random Naïve Bayes (RNB). Keypoint detection is prone to detect only few or no keypoints for low resolution objects. This leads to partial or missed detections. The sliding window approach is time consuming especially when many different object scale levels are considered. Background subtraction cannot be used with moving camera. Because of the low computational demand MSER method has been chosen for further research in the project. MSERs are the result of a blob detection method based on thresholding and connected component labeling[48].

1.4.3 Our solution

1.4.3.1 Introduction

To test and validate these solutions a decision was made to develop a testing platform - self driving car capable of housing these technologies and reliably validating them.

In the previous period a car was chosen and general scope of self driving car control system was outlined and some parts of the system were prepared. A list of used sensors was determined. A 3D point cloud extraction algorithm and object detection algorithm in 3D point cloud that were developed earlier ([49]) was adapted to PointGrey Bumblebee stereo camera for using it with the testing platform. New code was developed to extract 3D point cloud from Velodyne LiDAR and Oxts RT3003 RTK-GPS devices.

Car testing platform was developed with a goal of approbation of these technologies in i-GAME *(interoperable GCDC*[1] *AutoMation Experience)* - a cooperative autonomous driving competition that took place in Helmond, Netherlands on May 23-31 of 2016.

Car testing platform consists of the physical car and mechanical control mechanisms, electronic actuators, feedback systems, data gathering, analysis and decision making framework, and sensing systems for both the external and internal environment, as described in more detail in the sections below.

1.4.3.2 Background - GCDC competition

The GCDC competition differs from other autonomous car projects in the sense that the cars participating in a maneuver communicate precise information about their state (ie. coordinates, speed, direction) to each other via wireless communication network. For this purpose cars participating in the competition are equipped with communication infrastructure of unified standard (*ETSI*[2] *ITS G5* network stack). As a result every car receives real-time information about the others and can plan cooperative maneuvers in advance without unnecessary stress and without getting into situations where extreme braking or another extreme action is necessary as long as this communication standardization requirement is held. Another consequence from communication standardization: competition between different technical solutions, different vendors and different levels of complexity becomes possible.

In the GCDC-2016 competition the following scenarios were evaluated: *Cooperation on highway*, *Cooperative intersection* and *Emergency vehicle*[3].

In the GCDC competition scenarios emphasis is on cooperative rather than individual performance, by the group as a whole, consisting of all the cars participating in the maneuver. This second GCDC competition differs from the previous by the fact that automated lateral control of cars is also required in addition to longitudinal control introduced in 2011.

1.4.3.3 Control system description

1.4.3.3.1 Approach overview

Design of our cooperative car control system starts from lowest-level actuator control to high-level control such as motion planning, maneuvers and scenarios.

As a base for our experimental cooperative car we are using a 2004 Mazda 6 (see Figure 1.4.3). Some modifications were made to the Mazda 6 base car itself for steering, braking and throttle control to comply to GCDC requirements. These hardware changes are described in the further sections. The car control module including custom hardware and supporting software were developed.

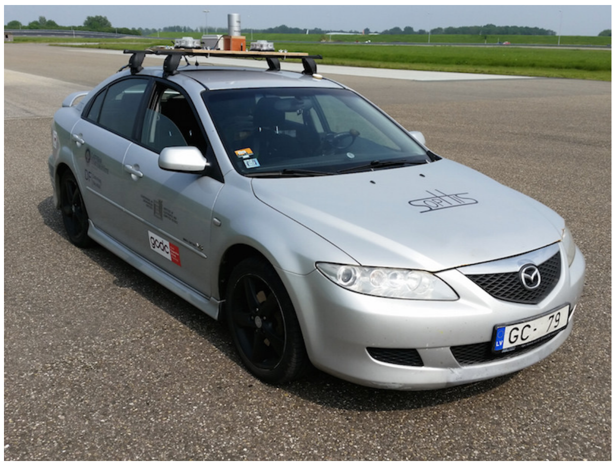


Figure 1.4.3: Photo of GCDC competition car (Mazda 6)

1.4.3.3.2 System architecture

The developed car control module (see Figure 1.4.4) is a hardware and software solution that continiously obtains a car data (speed, RPM, GPS position, acceleration data, position of the throttle and brake pedals, position of the steer) and its surrounding object data (car positions, dimensions) from sensor devices. Additional commands are accepted from human-machine interface (HMI) as well, for example, *operating mode* change from *software dashboard* or emergency interrupt signal (*STOP* and *Resume* buttons on *control panel*, a signal from car pedals or steering column that indicates an explicit driver action and the need to interrupt the automated operating mode).

The data obtained from sensors and HMI are stored into database as *system state*. The system state data is used to plan or replan motion of the car by *motion planning process* for some short future time period (1 second) at 5Hz frequency. Motion planning process describes the car's projected path in the future by the means of list of micro-goals for speed, steering angle or position relative to the other cars.

Motion plan execution process performs step-by step execution of current plan at 25Hz. The list of micro-goals describing the plan is converted to actuator control signals for throttle, steering and braking motors and red/green *beacon lights*.

Data from other cars (positions, directions, speeds) are received continuously at 25Hz via *ETSI ITS G5 network*. The position of our car is sent to other cars at the same frequency. Data from the road infrastructure stations are received via the same network.

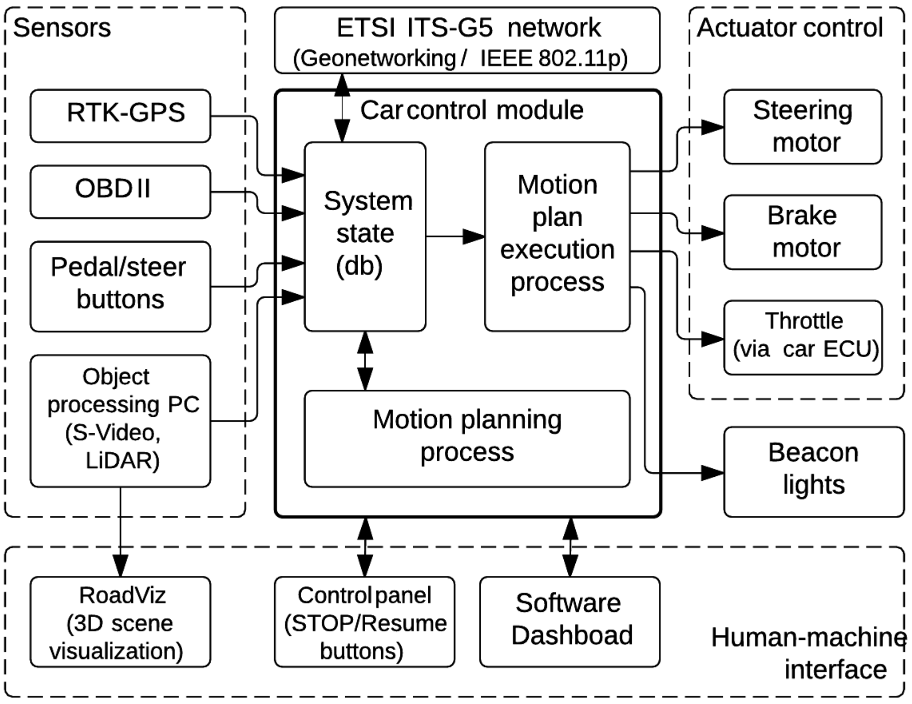


Figure 1.4.4: Car control system architecture

1.4.3.3.3 Software architecture

Software part of the car control system is developed in *Erlang* programming language and runs on *BEAM* virtual machine. Small sub-functions for low-level device control are written in C and accessed via Erlang linked-in port driver.

The car control system has an architecture of several independent software processes (Erlang light-weight threads) that are executed concurrently each with its configurable wake-up period (see Figure 1.4.5). The only way processes interact with each other is the system state that is stored in the in-memory *database server*.

Some processes just receive data periodically from sensor devices and store them into the database (ODB II data, GPS data, joystick position, surrounding objects, dashboard events and incoming network packets). Some other processes are reading system state data from the database and sending them to devices periodically (outgoing network packets, dashboard updates and car's surrounding objects). Separate processes are allocated for motion planning and motion plan execution.

[1]Grand Cooperative Driving Challenge
[2]European Telecommunications Standards Institute
[3]http://gcdc.net/en/

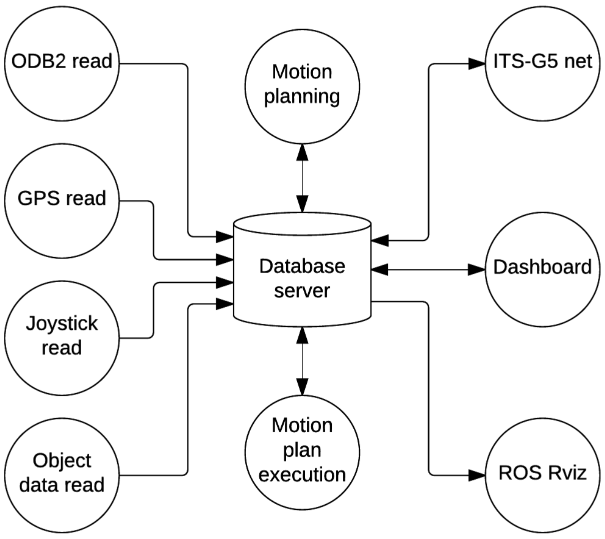


Figure 1.4.5: Software architecture of the car control system

1.4.3.3.4 Car modifications for actuator control and feedback

As our base car did not originally have x-by-wire ability, we have added controllable actuator motors for steering column and brake pedal and made some changes to the car that are described here.

- **Steer control:** Our base car has a hydraulic power assisted steering system. To add steer-by-wire functionality to it we use Electric Power Assisted Steering (EPAS) motor from *Renault Clio* that was integrated into our car's steering column (see Figure 1.4.6). This motor could be driven by PWM signal of 12V amplitude.

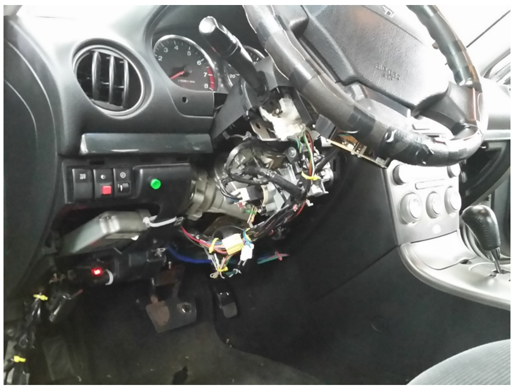


Figure 1.4.6: Electric Power Assisted Steering motor assembled in Mazda 6 steering column

For steering angle feedback we use data from steering wheel angle sensor from *Renault* (see Figure 1.4.7) that is also assembled in steering column. Steering angle data from sensor are received via CAN bus interface.

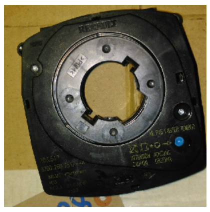


Figure 1.4.7: Steering wheel angle sensor

- **Brake control:** Brakes of our base car can not be controlled by wire. Therefore we added a motor from Electric Parking Brake Module from *Renault Laguna* that is used to pull brake pedal using wire rope (see Figure 1.4.8). Parking brake motor is also driven by PWM signal of 12V voltage amplitude.

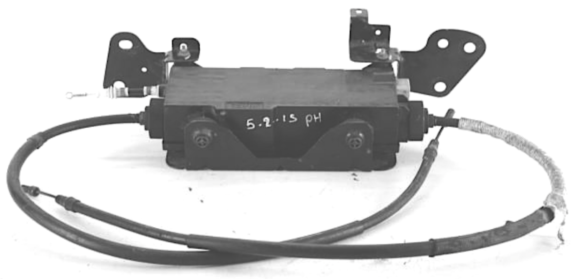


Figure 1.4.8: Electric Parking Brake Mechanism

For brake position feedback we use voltage formed by rotary potentiometer connected to brake pedal and read by appropriate ADC device.

- **Throttle control:** We have added input selection ability to the wired connection between the throttle pedal and car's ECU to enable feeding ECU throttle control input signal from either pedal or the DAC of our control system as necessary.

  Our base car throttle signal from pedal to ECU is controlled by two voltages: V_1, V_2. V_1 must be in range from $1.57V$ to $3.3V$ and V_2 must be from $1.02V$ to $2.75V$. The difference between V_1 and V_2 must be exactly $0.55V$.

  These voltages are linear to throttle pedal position and motor RPM value.

1.4.3.3.5 System state and its transitions

By *state of the car control system* we mean current operating mode of the system (ie. MANUAL, CC etc.). Each state could have one or more attributes.

Car control system may operate in different operation modes like MANUAL, AUTO and its sub-modes — JOYSTICK, CC, etc. Behavior of the whole system in each individual moment depends not only on car data that are stored in the database (ie. speed, position, other car positions etc.) but also on a set of micro-goals that the system currently has to achieve. The set of micro-goals always depend on current mode the system is operating in. For example, in CC mode the goal is to achieve and hold given speed value, but in JOYSTICK mode car's RPM value must strictly follow a y-axis value obtained from the joystick device but angle of steering wheel at the same time must follow joystick's x-axis value.

A transition structure of system operating modes could be described in the form of state transition diagram (see example state diagram in Figure 1.4.9).

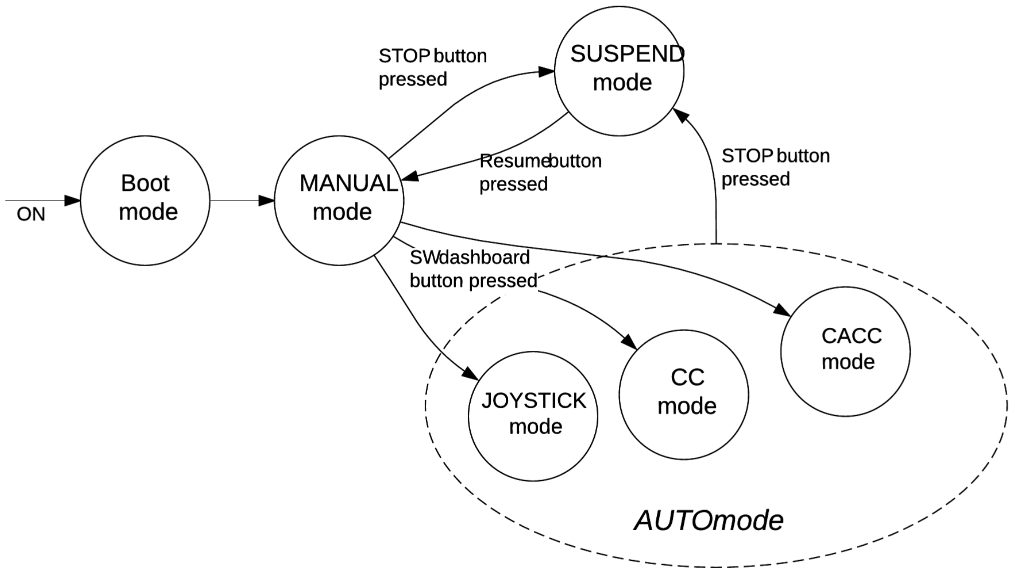


Figure 1.4.9: State transition diagram for car mode control

State transition rules are controlled by state machine. Some state transfers (ie. transfers to and from SUSPEND mode) are triggered by hardware circuit and are related to emergency takeover mechanism that is described in next subsection. Others could be triggered by a software dashboard (an Android application, see Figure 1.4.12).

When system is turned ON, after successful boot sequence a control system is left into MANUAL mode. In this mode car's throttle, steering and brakes are fully controllable manually by the driver while system is started and current operating mode and car attributes such as RPM and speed are already reflected on the dashboard. If operating mode transfer rules allow, it is also possible to change operating mode or its attributes using buttons on the dashboard. There are three first-level states that correspond to the operating modes of the system: MANUAL, SUSPENDED and AUTO.

A state could have zero or more sub-states. For example, in our case the state AUTO has three sub-states: JOYSTICK (car driven by joystick), CC (given fixed speed must be hold) and CACC (platooning, where car must follow front car's trajectory while holding fixed (x, y) position relative to the front car). Each sub-state may have no more than one direct parent.

Each state has an *entry condition* that is a logical expression that must

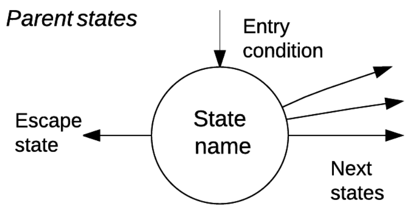


Figure 1.4.10: State model

evaluate to *true* to allow a transfer to this state (see Figure 1.4.10). Successful evaluation of the entry condition of a state with parent implies that entry conditions of all levels of its parents must also evaluate to *true*.

For example, to make transfer to the MANUAL state it is not necessary to satisfy any specific entry conditions. But transfer from MANUAL to AUTO state is possible only when the sensor and actuator systems are ready to work. Next, for instance, to be able to enter CACC mode, in addition to AUTO entry condition there particular other conditions must be held, such as existence of a candidate car in the front of our car with specific parameters in a specific range (ie. relative position, relative speed).

While the system is in a state with specific entry condition, it must hold *true* while the system remains in this state. Also the entry conditions of all its parent levels (if any) must hold *true*. If any of these entry conditions fail to satisfy, the system is transferred to *escape state* of an outermost of all states that failed to satisfy its entry condition.

A state has a *boolean* property called *interactive*. If a state is not interactive and its entry condition evaluates to *true* then transfer to the state is performed by the state machine immediately. But if the entry condition of the state evaluates to *true* but the state is interactive then transfer to this state just becomes enabled for the driver on the software dashboard (see Figure 1.4.12), otherwise disabled. Then the driver could make the decision to transfer system to this state by pressing a button on the dashboard.

In the case when the entry conditions of several non-interactive states simultaneously evaluate to *true* then all of these options are displayed in dashboard allowing driver to take decision manually.

1.4.3.3.6 Emergency takeover mechanism

GCDC rules required that the control system should provide manual emergency takeover feature for driver to switch a car to manual mode in case if something goes wrong while the car is in AUTO mode or its sub-modes (for instance in case of a software error, a system error or power outage). After emergency takeover all automatic control should be removed and car must switch to the manual control.

System must provide basic indication of driving mode using beacon lights (see Figure 1.4.3). Green light indicates that car is in AUTO mode or its sub-modes. Red light indicates that MANUAL mode is active.

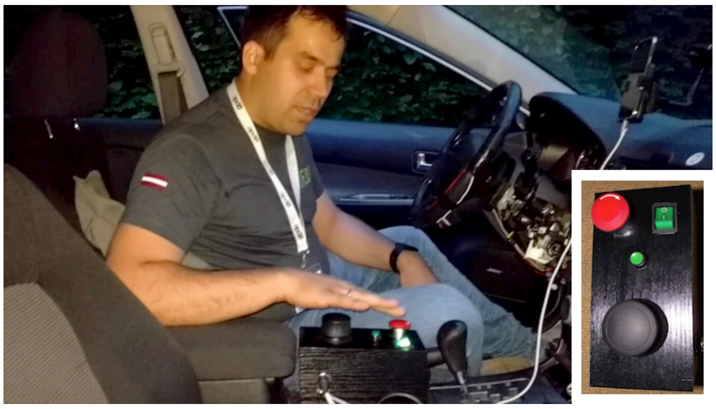
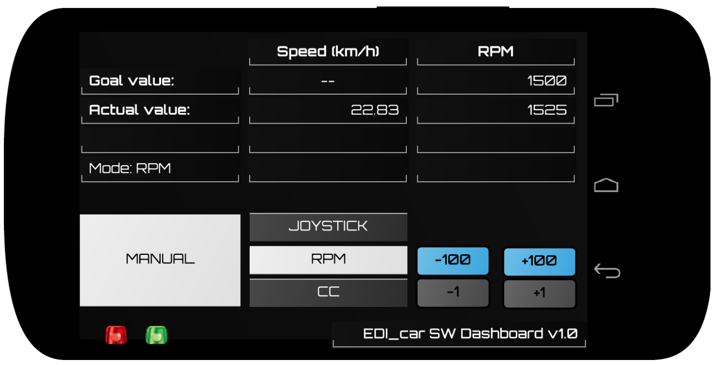


Figure 1.4.11: Control panel

Emergency manual takeover must be triggered by any of the following events:

- **By STOP button:** System must switch to manual mode if a special red button on control panel near driver's right hand (see Figure 1.4.11) is pressed. By pressing it the car is transferred to manual mode but control system is set to a special SUSPEND mode (see Figure 1.4.9) and all running software programs are interrupted. There is also a green button on the control panel that transfers system from SUSPEND mode back to WARM MANUAL.
- **By pedal:** System must switch to manual mode if throttle or brake pedal is pressed.
- **By steering wheel:** System must switch to manual mode if steering wheel is touched.

1.4.3.3.7 Human-machine interface



	Speed (km/h)	RPM
Goal value:	--	1500
Actual value:	22.83	1525
Mode: RPM		

Figure 1.4.12: Software dashboard

In addition to control panel mentioned above the car control system's HMI includes a very simple software dashboard — an Android application that shows current values of important car parameters such as operating mode, speed etc. The dashboard also provides buttons for driver to change system operating mode or to adjust mode parameters (see Figure 1.4.12).

1.4.3.3.8 Car position and speed

Common approach for determination of car position is usage of GNSS signals supplemented with additional data sources as inertial measurement units (IMU). The fusion of data from multiple sources allows to compensate their disadvantages and therefore improve the position accuracy and stability. Different smart car categories require different car position accuracy. For example, Google's autonomous cars mainly rely on very detailed maps of the roads and terrain as well as data from LiDAR and RADAR sensors and wheel encoders, therefore temporal outage of GNSS signals or inaccuracy by several meters do not immediately suspend the operation of the whole system [50]. Cooperative driving approach as in GCDC is based on common world model based on each participant's supplied own position data therefore the accuracy of each car position, in the common case 1m 2DRMS or better, is utmost important. This accuracy is provided by usage of high class GNSS systems based on Real Time Kinematic (RTK-GPS) technique [51].

We use a combined inertial and satellite-based navigation system *Oxts RT-3003* to obtain position, direction, speed and other parameters of our car. This device can be augmented optionally by ground reference station that allows to use RTK correction and provide position accuracy of 1cm at 100Hz. Data from *RT-3003* is received via Ethernet cable.

As a cheaper alternative to obtain speed from car ECU itself we used *ELM327* OBD II device for some experiments.

1.4.3.3.9 Communication with other cars

To get data via network about other cars position, speed etc. and to send our position to others we use *Cooperative awareness message (CAM)* and i-GAME *Cooperative lane change message (ICLCM)* formats. To receive messages from road infrastructure we use *Decentralized environmental notification message (DENM)* format. CAM and DENM packet formats are described in *ETSI ITS-G5* standard but ICLCM format was developed by organizers specially for GCDC competition.

A C function library was developed to encode/decode CAM, DENM an ICLCM payload using code generated by *ASN.1* compiler *asn1c*. Payload packet is sent and received using *IEEE 802.11p* at data-link level and is wrapped into simplified versions of *GeoNetworking* and *BTP* packets at network and transport levels respectively. A PC Engines APU platform board with Voyage Linux as network communication device is used.

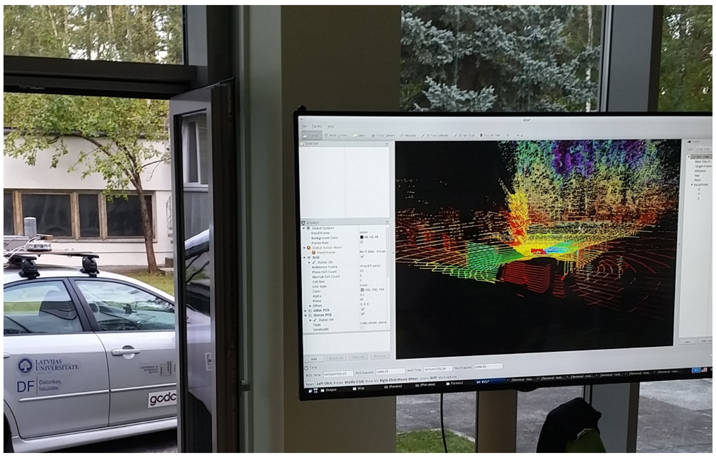


Figure 1.4.13: 3D point-cloud visualization on a common *ROS Rviz* scene from different data sources (LiDAR and stereo video)

1.4.3.3.10 Surrounding object detection

Although it is not required by GCDC rules we added some sensors for detection of 3D point-cloud around our car and detecting objects such as pedestrians or road boundaries in them. Detecting other cars around our car might also be useful for additional correctness checks of data received from the network. 3D point-cloud is also used as stand by data source in moments of short interruption in network communication.

We use two independent sources for 3D point-cloud — LiDAR and stereo video camera (see Figure 1.4.13). Data from both data sources is sent to object processing PC where objects in point-cloud are detected and "distilled" list of detected objects is sent to motion planning module of the car control system. Optionally a scene of car and its surroundings (car, 3D point-cloud, detected objects, other cars, planned path) could be visualized using *ROS Rviz* program or logged to file.

- 3D point-cloud from LiDAR: We are using a *Velodyne HDL-32E* device to get cloud of distances to objects around the car. This device has 32 infrared 905nm lasers that measure a distance to objects from 1m to 70m with precision 2.5cm at 10Hz speed. Data from *HDL-32E* is sent to object processing computer via Ethernet cable where point-cloud data is extracted and processed.

- 3D point-cloud from stereo video camera: For detection of objects in front of the car we use stereo video camera *PointGrey Bumblebee XB3 1.3 MP*. Video shots are captured simultaneously by two cameras located in parallel at a fixed distance from each other ([49]). Pair of frames are sent via *Firewire* to object processing PC where 3D point-cloud is computed from them.

- Object detection in 3D point-cloud: After 3D point-clouds from LiDAR

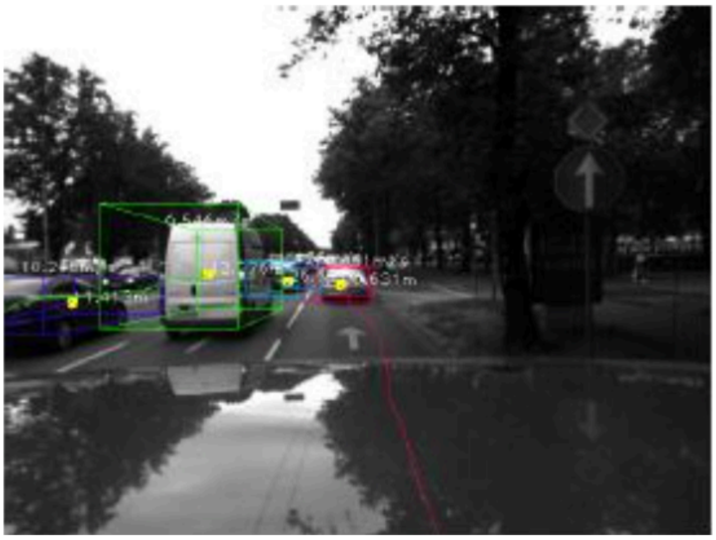


Figure 1.4.14: Object detection example from road stereo-video [49]

and Stereo Video devices is obtained/computed, road objects are detected (see Figure 1.4.14) in those point-clouds [49].

A dedicated PC with *Intel i7* processor is used for processing stereo video and 3D point-cloud data from video camera and LiDAR. It takes significant computing power to process point-clouds and video so it is put on a separate PC and in separate Ethernet segment. Only the distilled list of detected objects is sent to the car control system for further motion planning. Number of surrounding objects usually will be less than 20.

1.4.3.3.11 Motion plan execution process

An outline of the plan execution algorithm is shown in Figure 1.4.15.

Motion plan execution process starts on timer events at strict frequency 25Hz and performs step-by-step execution of motion plan prepared by motion planning process. First, the existing timer that started current code is canceled and rescheduled after 40ms. Then current system state and its ordered set of micro-goals Gs for the current state is read from database. State specific handler code is executed. In handler code current actuator values for steering and braking motors, throttle and beacon lights are read. New control values are computed for every actuator. Finally, a state context is stored back to database.

When actuator control value $u(t)$ [1.4.1] that yields to satisfied micro-goal cannot be computed directly (for example, when throttle voltage must be determined to achieve given speed in CC mode) then just approximated step of control value to the right direction is made. The step value is computed using PID controller based specific *control type* for each of micro-goals.

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \tag{1.4.1}$$

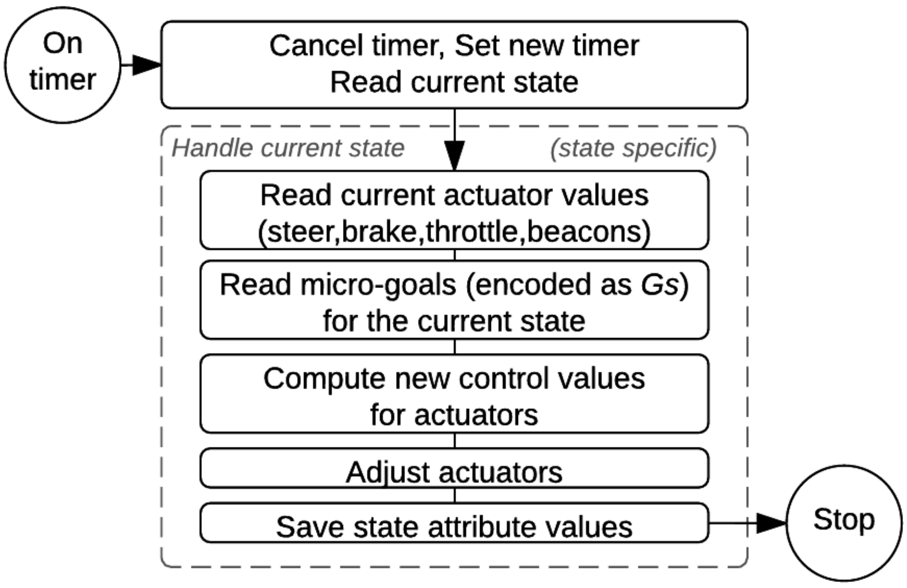

On timer
Cancel timer, Set new timer
Read current state
Handle current state
(state specific)
Read current actuator values (steer,brake,throttle,beacons)
Read micro-goals (encoded as Gs) for the current state
Compute new control values for actuators
Adjust actuators
Save state attribute values
Stop

Figure 1.4.15: Motion plan execution algorithm

Proportional-Integral-Derivative (PID) controller is simple and robust feedback control algorithm commonly used to control 95% of real-world systems and processes [52]. PID controller also is well suited for automated car actuator control applications (see Section 1.4.4).

There are certain standard control types to be used in the motion plan execution to describe state behavior.

Only *id* field of the record is always mandatory. Others are mandatory only if required by the control they're used with.

In this section by G the first element of Gs is denoted. Following standard control types are used in the system:

- S (speed) control uses throttle and brake motor to reach and hold the car's speed to a given value $G.v_g$. S control is used in CC mode;
- SP (speed path) control is similar to S, except after the goal is reached, next G is chosen and S is repeated. When no elements are left in Gs, actual throttle, brake and steering motor control signals are set to idle values;
- SD (speed-direction) control uses throttle, brake and steering motors to reach and hold the car's speed to a given value $G.v_g$ and direction to a given $G.theta_g$;
- SDP (speed-direction-position) control is similar to SD, except after the point $(G.long_g, G.lat_g)$ crossing line perpendicular to $G.theta_g$ is reached, actual throttle, brake and steering motor control signals are set to idle values;
- $SDPP$ (speed-direction-position path) control is similar to SD, except after the goal is reached, next G is chosen and SD is repeated. When no

elements are left in Gs, actual throttle, brake and steering motor control signals are set to idle values;

- RP (relative position) control uses throttle, brake and steering motors to reach and hold its relative position $(G.xr_g, G.yr_g)$ to other car $G.car_g$. RP control is used in CACC mode;

1.4.3.3.12 Motion planning process

An outline of the planning algorithm is shown in Figure 1.4.16.

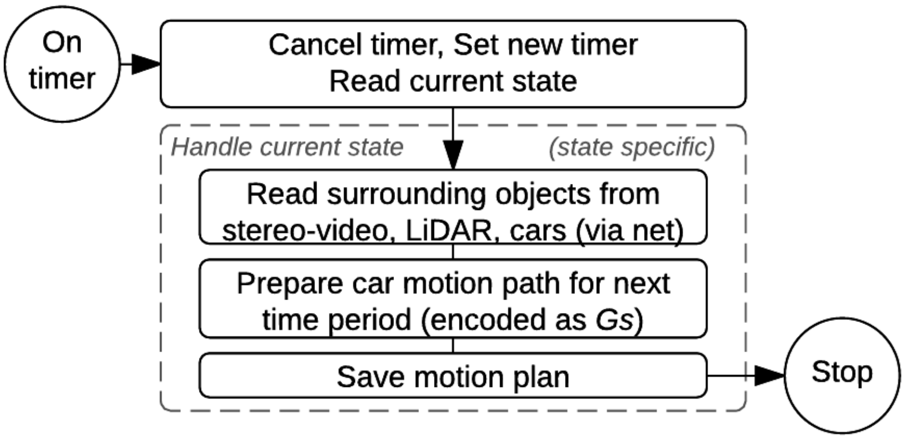

On timer
Cancel timer, Set new timer
Read current state
Handle current state
(state specific)
Read surrounding objects from stereo-video, LiDAR, cars (via net)
Prepare car motion path for next time period (encoded as Gs)
Save motion plan
Stop

Figure 1.4.16: Motion planning algorithm

Motion planning process starts on timer events at frequency 5Hz and creates motion plan for next 1 second.

First, existing timer that started current code is canceled and rescheduled after 200ms. Then current system state is read from the database. Then car's surrounding objects that may impact motion plan are read (objects detected in 3D point-clouds from LiDAR and stereo-video point-clouds, other car data from network). Then a motion plan for next 1 second is computed, encoded as ordered set of micro-goals Gs and stored to database.

1.4.3.4 mCDC - mini Cooperative Driving Challenge

To create more affordable infrastructure for development of cooperative driving algorithms and to have an ability to test them with many cars simultaneously a mini cooperative driving infrastructure development was started (abstract cooperative driving algorithms are the same for real cars and for model cars). The infrastructure includes a track for specially equipped model cars, positioning system infrastructure and control software (see Figure 1.4.17).

We are using base chassis from standard RC model car (scale 1/18) that is equipped additionally with MCU, wireless network device and controllers for main motor and steering servo (see car prototype in Figure 1.4.18).

A positioning system prototype is developed (see Figure 1.4.19). It works like GPS system for real cars and provides positions and direction angles at

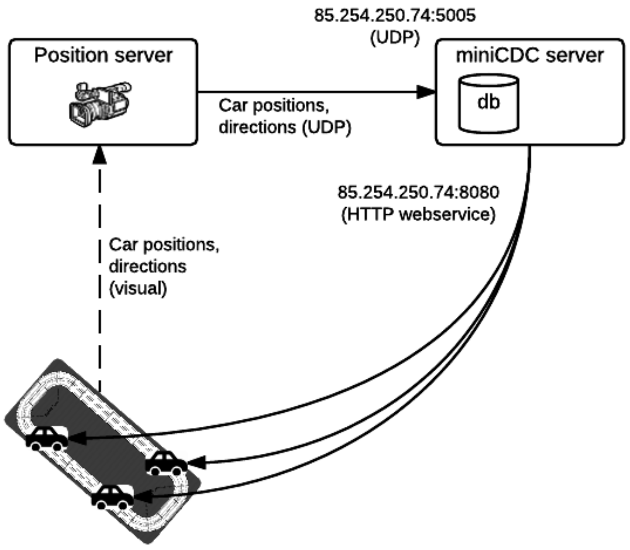


Figure 1.4.17: mCDC system architecture

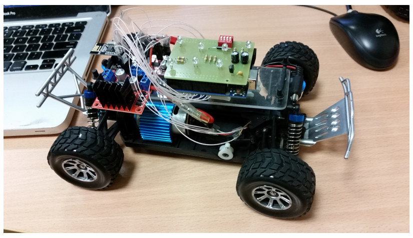


Figure 1.4.18: Mini car prototype

25Hz frequency for all cars on the track at the moment. Current precision of the system is around 1-2mm for position x, y coordinates and around 1 degree for direction angle.

Current version of the positioning system prototype uses one vertical video camera so the covered track area is currently limited to 3m to 2m (at maximum camera height of 5m above ground).

Currently work on positioning system is continued to support several cameras to increase covered track area without accuracy loss, as well as on the other parts of the mini cooperative driving infrastructure.

1.4.4 Results

In this period a test and validation platform for ADAS/ITS solutions was validated at GCDC event. A publication about the results of the development of this platform, developed technologies and the validation itself was submitted for publication.

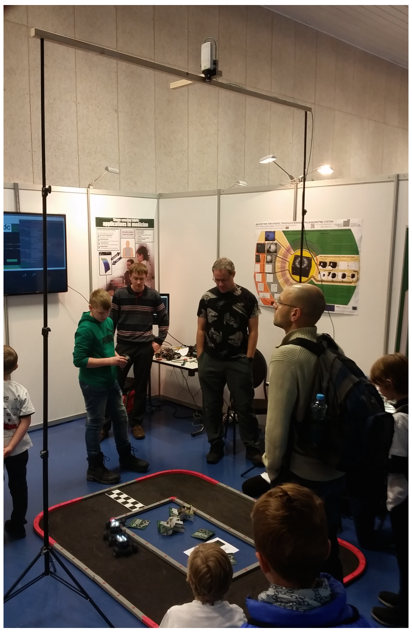


Figure 1.4.19: Mini-cooperative driving track and positioning system prototype

1.4.4.1 Prototype

A prototype of car control system is currently implemented (see Figure 1.4.20). At a hardware level our system's software processes (see Figure 1.4.5) are distributed among three *Raspberry Pi* v2 computers connected via 1 Gbps ethernet hub with *Raspbian* operating system. *Raspberry Pi* computers act as an connected Erlang nodes.

The RasPi 3 node (see Figure 1.4.21) is used to do motion planning an control beacon lights. RasPi 2 node is used to read steering angle sensor values using *MCP2515* CAN bus interface chip. The rest of functionality is done by RasPi 1 node. An *Atmel328P* MCU (*Arduino Uno v3* board) is used to control steering and braking PWM signals. A *MCP4822* DAC is used to send throttle control signals to car ECU. A *MCP3202* ADC is used to read brake position and throttle pedal voltages.

Two levels of relays are implemented for safety. Relay signal R_0 (see Figure 1.4.21) handling is implemented in hardware circuit. It will switch all car actuators to MANUAL mode on pressing STOP button on control panel (see Figure 1.4.11) even when system software is in non-working state. Signals from relays R_1,R_2,R_3 control only their respective actuators - steering motor, braking motor and throttle. These three relays are handled by software. In general, power from system to i-th actuator is passed only if R_0 is ON and R_i is ON. Also default state of relays is OFF when system is switched off or power outage happens.

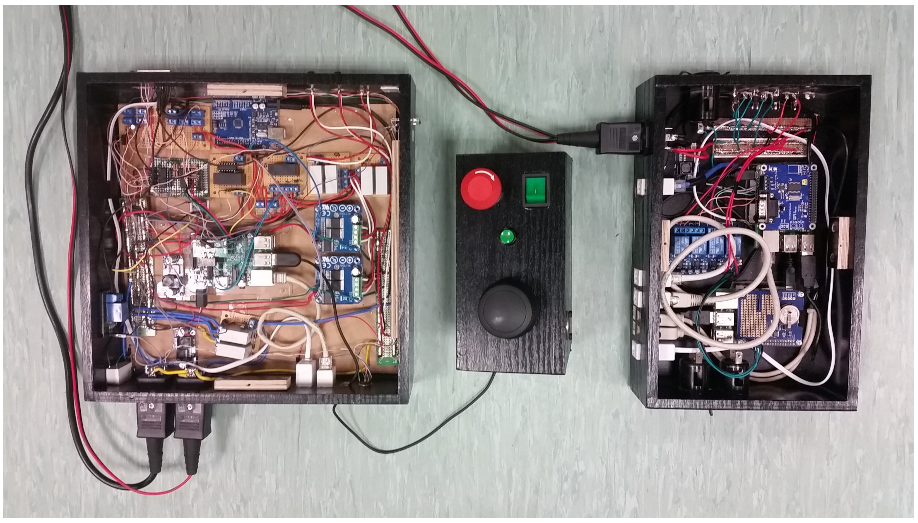


Figure 1.4.20: Car control module prototype

1.4.4.2 Prototype test

As demonstrative example of tests with prototype we present the results of test where control of engine RPM was performed.

Test scenario is as follows. When system is started, its operating mode is set to MANUAL and actual RPM value obtained from OBD II device is around 750 (idle, when throttle pedal is in neutral position). Goal RPM value is already set to 1000 but it has no impact to plan execution process while the system is in MANUAL mode. Throttle control signal is passed from pedal to car ECU as it is.

On the 3rd second of the graph the system is transferred (by dashboard or by command line) to RPM mode. Car ECU throttle signal source is changed from pedal to our DAC (see Figure 1.4.21). Now goal RPM value is started to be regarded as micro-goal and plan execution process is starting to adjust actual value to goal value. Plan execution process is starting to use a PID controller ($K_p = 0.05, K_i = 0.001, K_d = 0.05$) to compute throttle signal change steps and adjust signal until actual RPM value becomes near to goal value ($abs(rpm_{act} - rpm_g) < 1$). In graph could be seen that actual value converges to goal and remains around there until goal or operating mode changes.

At 32th second from the start the goal value of RPM is changed to 2000. Actual RPM value is automatically adjusted again to achieve goal. Then at 62th second the goal value of RPM is changed back to 1000 and actual value again could be seen converging to the new micro-goal value. There could be seen certain instability observed in the resulting actual RPM value. A possible reason of it could be objective interference of our PID controller with similar functionality of the car ECU itself. Also it could be seen that sharp changes of goal value (like at 32th and 62th seconds) raise significant instability of actual RPM value. Firmness of resulting RPM signal could be improved by further testing and adjusting PID coefficients K_p, K_i, K_d and adding additional

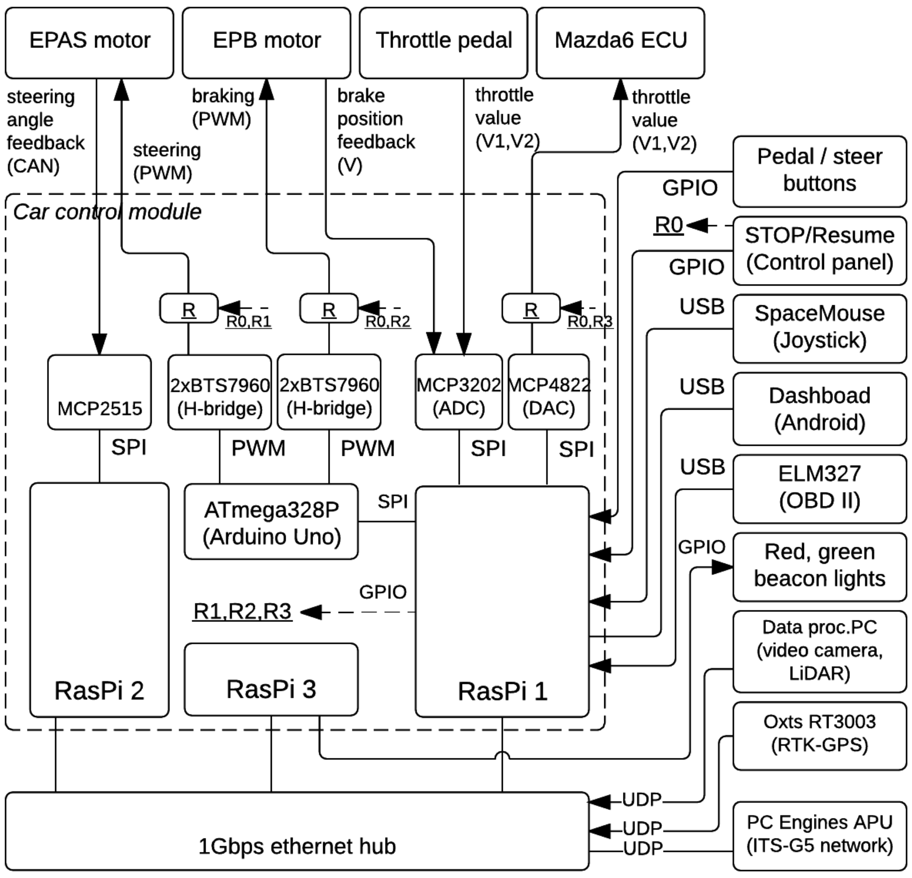


Figure 1.4.21: Car control module block diagram

constraints to control signal change computed with PID control algorithm.

1.4.5 Discussion and future work

In the last period of the project we plan to work towards several goals:

- To implement the stereo-vision algorithms on SoC;
- To augment/extend the senses of the driver, e.g. make infrared vision spectrum visible;
- Develop innovative systems for driver monitoring and feedback;
- To develop and use the mCDC track for gathering data and implementing collaborative driving algorithms and testing them on the same road as algorithms from different providers.

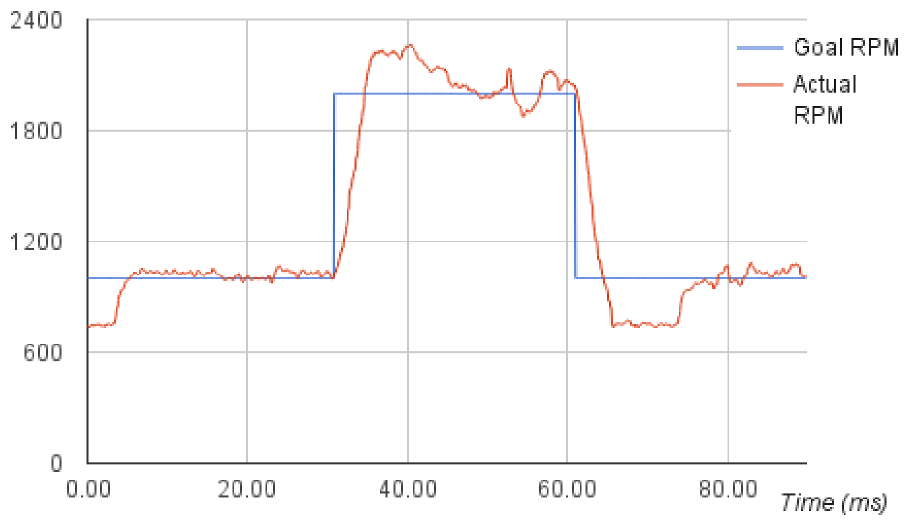


Figure 1.4.22: Car goal RPM vs actual RPM values

Bibliography

[1] Joseph Polastre, Robert Szewczyk, and David Culler. Telos: enabling ultra-low power wireless research. In *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pages 364–369. IEEE, 2005.

[2] *MICAz.*

[3] Prabal Dutta and David Culler. Epic: An open mote platform for application-driven design. In *Proceedings of the 7th international conference on Information processing in sensor networks*, pages 547–548. IEEE Computer Society, 2008.

[4] Advanticsys. *XM1000 datasheet.*

[5] Girts Strazdins, Atis Elsts, and Leo Selavo. Mansos: easy to use, portable and resource efficient operating system for networked embedded devices. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 427–428. ACM, 2010.

[6] Atis Elsts, Janis Judvaitis, and Leo Selavo. Seal: A domain-specific language for novice wireless sensor network programmers. In *Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on*, pages 220–227. IEEE, 2013.

[7] Vlado Handziski, Andreas Köpke, Andreas Willig, and Adam Wolisz. Twist: A scalable and reconfigurable wireless sensor network testbed for indoor deployments. In *REALMAN '06 Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality*, pages 63–70. ACM, 2006.

[8] Harward. *Tmote SKY datasheet.*

[9] Vlado Handziski, Joseph Polastre, Jan-Hinrich Hauer, and Cory Sharp. Flexible hardware abstraction of the ti msp430 microcontroller in tinyos. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 277–278. ACM, 2004.

[10] Geoffrey Werner-Allen, Patrick Swieskowski, and Matt Welsh. Motelab: a wireless sensor network testbed. In *REALMAN '06 Proceedings of the*

2nd international workshop on Multi-hop ad hoc networks: from theory to reality, pages 483 – 488. IEEE, 2005.

[11] Manjunath Doddavenkatappa, Mun Choon Chan, and Ananda A.L. Indriya: A low-cost, 3d wireless sensor network testbed. In *Testbeds and Research Infrastructure. Development of Networks and Communities*, pages pp 302–316. Springer Berlin Heidelberg, 2012.

[12] Arduino. *Arduino homepage.*

[13] Rinalds Ruskuls and Leo Selavo. Edimote: a flexible sensor node prototyping and profiling tool. In *Real-World Wireless Sensor Networks*, pages 194–197. Springer, 2010.

[14] Paolo Bonato. Wearable sensors and systems. *Engineering in Medicine and Biology Magazine, IEEE*, 29(3):25–36, 2010.

[15] Rune Fensli, Einar Gunnarson, and Torstein Gundersen. A wearable ecg-recording system for continuous arrhythmia monitoring in a wireless tele-home-care situation. In *Computer-Based Medical Systems, 2005. Proceedings. 18th IEEE Symposium on*, pages 407–412. IEEE, 2005.

[16] Winston H Wu, Alex AT Bui, Maxim A Batalin, Duo Liu, and William J Kaiser. Incremental diagnosis method for intelligent wearable sensor systems. *Information Technology in Biomedicine, IEEE Transactions on*, 11(5):553–562, 2007.

[17] Peter Leijdekkers and Valérie Gay. A self-test to detect a heart attack using a mobile phone and wearable sensors. In *Computer-Based Medical Systems, 2008. CBMS'08. 21st IEEE International Symposium on*, pages 93–98. IEEE, 2008.

[18] Alexandros Pantelopoulos and Nikolaos G Bourbakis. A survey on wearable sensor-based systems for health monitoring and prognosis. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 40(1):1–12, 2010.

[19] Shyamal Patel, Hyung Park, Paolo Bonato, Leighton Chan, and Mary Rodgers. A review of wearable sensors and systems with application in rehabilitation. *Journal of neuroengineering and rehabilitation*, 9(1):21, 2012.

[20] A Hermanis, K Nesenbergs, R Cacurs, and M Greitans. Wearable posture monitoring system with biofeedback via smartphone. *Journal of Medical and Bioengineering Vol*, 2(1), 2013.

[21] Yu-Jin Hong, Ig-Jae Kim, Sang Chul Ahn, and Hyoung-Gon Kim. Activity recognition using wearable sensors for elder care. In *Future Generation Communication and Networking, 2008. FGCN'08. Second International Conference on*, volume 2, pages 302–305. IEEE, 2008.

[22] Louis Atallah, Benny Lo, Guang-Zhong Yang, and Frank Siegemund. Wirelessly accessible sensor populations (wasp) for elderly care monitoring. In *Pervasive Computing Technologies for Healthcare, 2008. PervasiveHealth 2008. Second International Conference on*, pages 2–7. IEEE, 2008.

[23] Jay Chen, Karric Kwong, Dennis Chang, Jerry Luk, and Ruzena Bajcsy. Wearable sensors for reliable fall detection. In *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, pages 3551–3554. IEEE, 2006.

[24] R Ganea, Anisoara Paraschiv-Ionescu, Arash Salarian, Christophe Bula, Estelle Martin, Stephane Rochat, Constanze Hoskovec, and Chantal Piot-Ziegler. Kinematics and dynamic complexity of postural transitions in frail elderly subjects. In *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, pages 6117–6120. IEEE, 2007.

[25] Miikka Ermes, Juha Parkka, Jani Mantyjarvi, and Ilkka Korhonen. Detection of daily activities and sports with wearable sensors in controlled and uncontrolled conditions. *Information Technology in Biomedicine, IEEE Transactions on*, 12(1):20–26, 2008.

[26] Florian Michahelles and Bernt Schiele. Sensing and monitoring professional skiers. *Pervasive Computing, IEEE*, 4(3):40–45, 2005.

[27] Weijun Tao, Tao Liu, Rencheng Zheng, and Hutian Feng. Gait analysis using wearable sensors. *Sensors*, 12(2):2255–2283, 2012.

[28] Christian Peter, Eric Ebert, and Helmut Beikirch. A wearable multi-sensor system for mobile acquisition of emotion-related physiological data. In *Affective Computing and Intelligent Interaction*, pages 691–698. Springer, 2005.

[29] Flavia Sparacino. The museum wearable: Real-time sensor-driven understanding of visitors' interests for personalized visually-augmented museum experiences. *Museums and the Web 2002*, 2002.

[30] Konrad Lorincz, Bor-rong Chen, Geoffrey Werner Challen, Atanu Roy Chowdhury, Shyamal Patel, Paolo Bonato, Matt Welsh, et al. Mercury: a wearable sensor network platform for high-fidelity motion analysis. In *SenSys*, volume 9, pages 183–196, 2009.

[31] Krisjanis Nesenbergs and Leo Selavo. Smart textiles for wearable sensor networks: Review and early lessons. In *Medical Measurements and Applications (MeMeA), 2015 IEEE International Symposium on*, pages 402–406. IEEE, 2015.

[32] Robert Broadbent, Sidney Melamed, and Robert G Minton. Conductive nylon substrates and method of producing them, April 15 1975. US Patent 3,877,965.

[33] A. Hermanis, R. Cacurs, K. Nesenbergs, and M. Greitans. Efficient real-time data acquisition of wired sensor network with line topology. In *Open Systems (ICOS), 2013 IEEE Conference on*, pages 133–138, Dec 2013.

[34] A. Hermanis and K. Nesenbergs. Grid shaped accelerometer network for surface shape recognition. In *Electronics Conference (BEC), 2012 13th Biennial Baltic*, pages 203–206, Oct 2012.

[35] A. Hermanis, K. Nesenbergs, R. Cacurs, and M. Greitans. Wearable Posture Monitoring System with Biofeedback via Smartphone. *Journal of Medical and Bioengineering*, 2(1):40–44, 2013.

[36] M. D. Shuster and S. D. Oh. Three-axis attitude determination from vector observations. *Journal of Guidance and Control*, 4:70–77, 1981.

[37] Atis Hermanis, Ricards Cacurs, and Modris Greitans. Acceleration and magnetic sensor network for shape sensing. *IEEE Sensors Journal*, 16(5):1271–1280, 2016.

[38] Roberto Valenti, Ivan Dryanovski, and Jizhong Xiao. Keeping a Good Attitude: A Quaternion-Based Orientation Filter for IMUs and MARGs. *Sensors*, 15:19302–19330, 2015.

[39] P. Bonato, T. D'Alessio, and M. Knaflitz. A statistical method for the measurement of muscle activation intervals from surface myoelectric signal during gait. *IEEE Transactions on Biomedical Engineering*, 45(3):287–299, March 1998.

[40] A. Merlo, D. Farina, and R. Merletti. A fast and reliable technique for muscle activity detection from surface emg signals. *IEEE Transactions on Biomedical Engineering*, 50(3):316–323, March 2003.

[41] Royal Society for the Prevention of Accidents. (2001, February) DRIVER FATIGUE AND ROAD ACCIDENTS A LITERATURE REVIEW and POSITION PAPER. [Online]. Available: `http://www.rospa.com/rospaweb/docs/advice-services/road-safety/drivers/fatigue-litreview.pdf`.

[42] COMPASS. (2015, November) 4.1.03. Driver Drowsiness Detection System for Cars. [Online]. Available: `http://81.47.175.201/compass/index.php?option=com_content&view=article&id=506:413-driver-drowsiness-detection-system-for-cars&catid=22:smart-cars`.

[43] Robert Bosch LLC Frank Sgambati. (2012, January) Driver Drowsiness Detection. [Online]. Available: `http://www.sae.org/events/gim/presentations/2012/sgambati.pdf`.

[44] Dimitri J Walger, Toby P Breckon, Anna Gaszczak, and Thomas Popham. A comparison of features for regression-based driver head pose estimation

under varying illumination conditions. In *Computational Intelligence for Multimedia Understanding (IWCIM), 2014 International Workshop on*, pages 1–5. IEEE, 2014.

[45] Ming-Hsuan Yang, David J Kriegman, and Narendra Ahuja. Detecting faces in images: A survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(1):34–58, 2002.

[46] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

[47] Michael Teutsch, Thomas Muller, Marco Huber, and Jurgen Beyerer. Low resolution person detection with a moving thermal infrared camera by hot spot classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 209–216, 2014.

[48] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767, 2004.

[49] Harijs Grinbergs, Artis Mednis, and Modris Greitans. Real-time object tracking in 3d space using mobile platform with passive stereo vision system. In *International Conference on Signal Processing and Imaging Engineering (ICSPIE 2013), Hammamet, Tunisia*, 2013.

[50] Erico Guizzo. How google's self-driving car works. *IEEE Spectrum Online, October*, 18, 2011.

[51] Christoph Stiller, Sören Kammel, Benjamin Pitzer, Julius Ziegler, Moritz Werling, Tobias Gindele, and Daniel Jagszent. Team annieway's autonomous system. In *International Workshop on Robot Vision*, pages 248–259. Springer, 2008.

[52] Karl Johan Aström and Richard M Murray. *Feedback systems: an introduction for scientists and engineers*. Princeton university press, 2010.