BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF RADIOELECTRONICS

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV RADIOELEKTRONIKY

NEURAL NETWORKS IN INERTIAL NAVIGATION SYSTEMS

NEURONOVÉ SÍTĚ V INERCIÁLNÍCH NAVIGAČNÍCH SYSTÉMECH

DOCTORAL THESIS
DOKTORSKÁ PRÁCE

AUTHOR
AUTOR PRÁCE

ING. LENKA TEJMLOVÁ

SUPERVISOR
VEDOUCÍ PRÁCE

DOC. ING. JIŘÍ ŠEBESTA, PH.D.

BRNO 2017

KEYWORDS

IMU, INS, DR, inertial positioning, dead reckoning, artificial neural network, Arduino UNO, X-NUCLEO-IKS01A1, MATLAB™, Qt.

KLÍČOVÁ SLOVA

IMU, INS, DR, inerciální polohovací systém, umělá neuronová síť, Arduino UNO, X-NUCLEO-IKS01A1, MATLAB™, Qt.

DISERTAČNÍ PRÁCE JE ULOŽENA:

Ústav radioelektroniky
Fakulta elektrotechniky a komunikačních technologií
Vysoké učení technické v Brně
Technická 3082/12
616 00 Brno

CONTENTS

1 INTRODUCTION

An implementation of artificial intelligence into an automatic navigation systems is the one of opportunities how to improve performances of autonomous positioning systems. Well known positioning method which is used in many modern systems, such in cars, is the dead reckoning. This method is defined as the process of calculating current position by using a previous determined position and actual data from inertial sensors in combination with vehicle odometers. The implementation of this method defines actual position of moving object regarding to the initial position. It also defines the trajectory during the movement.

This topic is often discussed nowadays and the research in this field can be divided to many way. To providing of more effective solutions than independent processing of inertial sensor data offers, additional methods, systems and devices are required.

Research teams work on acquisition with intention to obtain more precise results provided by sensor data fusion, by increasing the number of sensors that are used to measure the same physical quantities, by adding various specific devices, such as Wi-Fi or other wireless equipment and its signal strength, by limitation of results determination, by monitoring of regularities in motions and finally by fusion with available GNSS/GPS, pedestrian navigation constrains, visual-aided constrains, map matching etc.

There are three main issues arising from the fundamentals of inertial navigation. The first of all is the Earth’s gravity. We can measure the acceleration. It contains both, a linear acceleration (that is needed to determine the position) and Earth’s gravity acceleration. This is good when the accelerometers are placed horizontally (flat). The precise strength of Earth's gravity varies depending on location, nevertheless, at the Earth's surface the nominal average value (standard acceleration of free fall) should be in our case subtracted, because we are located on the Earth’s surface. The accelerometers are generally never horizontally placed though the position of inertial measurement unit is often approaching this state. For that reason it is very hard to separate Earth’s gravity and linear acceleration, both measured together by accelerometer. We highly focus on this issue in this document. The second difficulty is Earth’s rotation around its axis by 15 degrees per hour and around the sun by 0.041 degrees per hour. This should be solved by using the gyrocompass and by implementation of proper compensations in computations. The third issue is a significant inaccuracy caused by sensitivity and typical characteristics of inertial sensors. Due to low signal to noise rate when the linear acceleration of the IMU or its orientation vary is the only inertial sensor navigation fundamentally inapplicable for precise localization. Thus nowadays, many localization methods are combined.

When we are talking about inertial sensor data fusion we are always confronted by real world challenges. It is thought that nothing is exactly accurate and therefore we have to consider deviations and errors as an inseparable part of technique. The task is to

use knowledge and enrich it by our own thoughts that complexly lead to invention of better solution, innovation. The application of inertial sensor data fusion brings thorough considerations of error models and their implementation in calculations. In combination with artificial neural network, Kalman filtering and with the support of GNSS/GPS the dead reckoning system may achieve a sufficient accuracy to determine the orientation and the position where the inertial navigation system (INS) is located.

An inertial navigation system (INS) is a navigation aid that uses a computer, motion sensors (accelerometers) and rotation sensors (gyroscopes, gyros) and maybe others to continuously calculate via dead reckoning (DR) actual position, actual orientation, and actual velocity (direction and speed of movement) of a moving object in time without any external references [1]. It has been called "Newtonian navigation" because its theoretical foundations have been known since time of Newton:

Given the position $x(t_0)$ and velocity $v(t_0)$ of a vehicle at time t_0, and its acceleration a(s) for times $s>t_0$, then its velocity, $v(t)$, and position, $x(t)$, for all time $t>0$ can be defined as (1.1), (1.2).

$$v(t) = v(t_0) + \int_{t_0}^{t} a(s)ds \tag{1.1}$$

$$x(t) = x(t_0) + \int_{t_0}^{t} v(s)ds \tag{1.2}$$

1. Sensors for measuring acceleration with sufficient accuracy:
    a. 3-axis acceleration sensor (accelerometer)
    b. 3-axis rotation sensor (gyroscope)
2. Compatible methods based on integration of the sensor outputs to obtain position
    a. Methods integrating the gyro outputs to determine the orientation of the accelerometer
    b. Methods integrating the accelerations to obtain the velocities and integrating the velocities to obtain the position
3. Hardware and software implementing these methods and for interpretation of the results
4. Applications that could justify the investments in technology required for developing the solutions to the capabilities listed above

Figure 1.1 Recorded trajectory.

This dissertation thesis heads with the state of the art, where the short history and current research state is described. The next Chapter outlines the objectives of the thesis and the rest of document deals with those objectives. At the end of the main document the results are discussed and the proposed method is evaluated. Annexes complement the described methods.

2 STATE OF THE ART

Inertial navigation is a self-contained navigation technique in which measurements provided by accelerometers and gyroscopes are used to track the position and orientation of an object relative to a known starting point, orientation and velocity. Inertial measurement units (IMUs) typically contain three orthogonal rate-gyroscopes and three orthogonal accelerometers, measuring angular velocity and linear acceleration respectively, [2].

By processing signals from these devices, it is possible to track the position and orientation of the device. This aim is often discussed nowadays and research is divided into many directions. To ensure better solution than which is offered by independent processing of sensor data, additional methods and equipment are required. Proposed inertial guidance system is based on dead reckoning method supplemented by artificial neural network (ANN) and Kalman filters (KF).

2.1 AN EXAMPLE OF IMPLEMENTATION

A very nice example of ANNs implementation for navigation system shows the paper from 12th International Conference on Control, Automation and Systems, Korea, 2012 [3]. Researchers developed the indoor navigation system based on pedestrian dead reckoning (PDR) that uses various sensors in a smartphone. MEMS IMU was mounted on the waist, using sensors and ANN status; they estimated the step length adaptively. They used a map-matching method in addition. If the estimated trajectory was tracked wrong way or the estimated position in unavailable place to go, map matching arranged the estimated position to the coordinate defined in a map. So the computed position was "snapped" to link in the map or to the corner when rotation rate measured by a gyroscope increased in the moment. A barometer was used for to distinguish the floor where the IMU belongs.

A major disadvantage of this method is that we need a map of the area where such a system is used. Without a map, the performance of positioning is not sufficiently accurate. Other examples can be found in [4] – [10].

2.2 A DIFFERENT APPROACH

The presented method approaches to the issue from another point of view than previous solutions of PDR inertial units. It is based on the fact that we need to apply DR (INS) while the terrain is unknown; that means wireless connections are not available, terrain map is not defined, and GNSS signal is not available. It was investigated that sensor errors, deviations and drifts achieve significant values, thus, the error in positioning is large. The fusion of sensor data, Kalman filtering and artificial neural network offer a solution for the purpose.

2.3 COORDINATE SYSTEMS

The coordinates for inertial systems are given to be natural to the problem at hand. We use LTP (local tangent plane) coordinates; first-order model of the earth as being flat, where they serve as local reference directions for representing vehicle attitude and velocity for operation – on the surface of the earth (or very close to). A common orientation for LPT coordinates has one horizontal axis (the north axis) in the direction of increasing latitude and the other horizontal axis (the east axis) in the direction of increasing longitude.

Furthermore, we have to specify the ECI (earth cantered inertial) coordinates that are the favoured inertial coordinates in the near-earth environment. The origin of ECI coordinates is at the centre of gravity of the earth, with axis directions:

x – the direction of the vernal equinox;

z – parallel to the rotation axis of the earth (north polar axis);

y – an additional axis to make a right handed orthogonal coordinate system.

The equatorial plane of the earth is also the equatorial plane of ECI coordinates, nevertheless the earth itself is rotating relative to the vernal equinox by about 15.04109 deg per hour[1]. ECEF (Earth Centred, Earth Fixed) coordinates have the same origin and third, polar axis as ECI coordinates, but rotate with the earth. Consequently, ECI and ECEF longitudes differ only by a function of time. ECI (indexed by "*I*"), ECEF (indexed by "*e*") coordinates and LTP are shown in Figure 2.1.

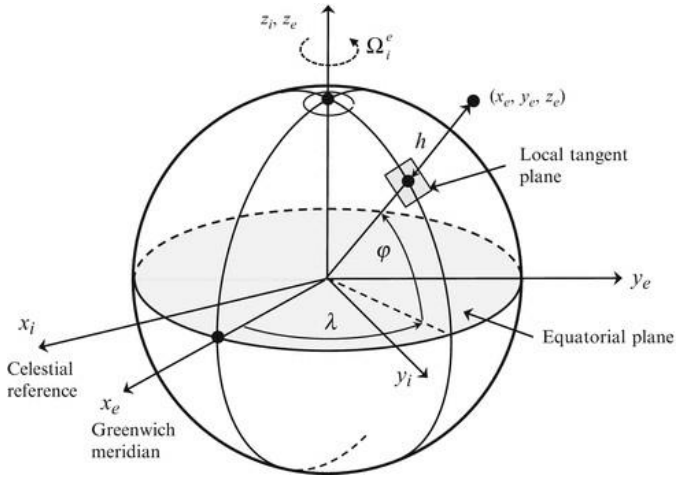

Figure 2.1 ECI, ECEF, and geodetic coordinate frame, [11].

In coordinate system NED (earth-fixed, north east down) right handed LTP system is preferred because the direction of a right (clockwise) turn is in the positive direction with respect to a downward axis and NED coordinate axes coincide with vehicle-fixed RPY (body fixed, roll pitch yaw) coordinates when the vehicle is in the flat position and headed to north. The other, commonly used right handed LPT system is ENU (east-

[1] *World Book Encyclopaedia* Vol 6. Illinois: World Book Inc.: 1984: 12. "It takes 23 hours 56 minutes 4.09 seconds for the Earth to spin around once 2π radians/86164.09 seconds"

north-up) and the transformation matrix between ENU and NED is described in my thesis. The ENU coordinate system is preferred in this thesis. The relation between ECEF coordinate frame and ENU coordinates can be found in APPENDIX A, part A.5.

RPY coordinates are vehicle fixed, as noted above, with the roll axis in the nominal direction of motion of the vehicle, the pitch axis out the right hand side, and the yaw axis such that tight turning is positive. This is used also for surface ships and ground vehicles, called SAE coordinates.

2.4 SENSOR MODELS

Inertial navigation performance is hardly limited by the performance of used inertial sensors. Basic formula, Newton's model, gives us an overview of the inertial navigation system's error evolution over time. The performance significantly decreases with the time and the system based only on integrated data from sensor is inapplicable.

The errors in measurement arise from many various reasons. Inertial navigation has been called "black box navigation" because it is entirely self-contained. It interferes what is going on outside by what it can sense inside. In addition, inertial sensors are called black boxes for the same reason. There are more events outside the sensor than just accelerations or rotations, see Figure 2.2.

The important fact to realize is that accelerometers do not measure gravitational acceleration, but inertial acceleration. That means, they measure "specific force" a=F/m, where F is the physically applied force and m is the mass it is applied to [12].

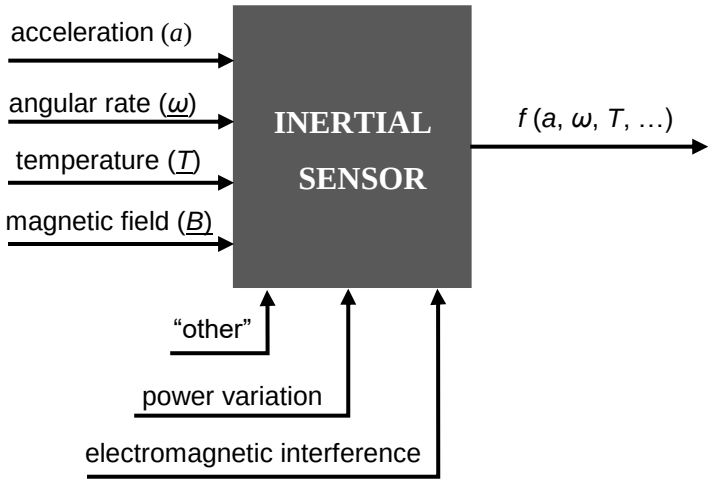

Figure 2.2 Sensor black-box model.

The dissertation includes definitions and appropriate suggestions for compensation of various sensor errors.

2.5 SENSOR CALIBRATION

To calibrate and compensate offsets, biases, scale factors and misalignments, affine (linear plus offset) model is used. Biases are included in offsets and the rest is linear. When we define output as shown in relation (2.1), where $\mathbf{z}_{input}$ is the vector representing the inputs (accelerations or rotation rates), $\mathbf{z}_{output}$ is the vector representing the

corresponding outputs, $\mathbf{b}_z$ is the vector of sensor output biases and **M** represents the linear input-output model.

$$\mathbf{z}_{output} = \mathbf{M} \cdot \left(\mathbf{z}_{input} + \mathbf{b}_z \right) \tag{2.1}$$

$$\mathbf{z}_{input} = \mathbf{M}^{-1} \cdot \mathbf{z}_{output} - \mathbf{b}_z \tag{2.2}$$

To estimate the values of **M** and $\mathbf{b}_z$, several pairs of given input-output vectors [$\mathbf{z}_{input,\ k}$, $\mathbf{z}_{output,\ k}$] have to be defined, . These outputs are measured while controlled calibration conditions, thus we get a pair of input-output recorded under these conditions and applicable for sensor compensation. This result can be generalized for a cluster of N ≥ 3 gyroscopes or accelerometers. For more information, see [12] and [13].

2.6 ORIENTATION DETERMINATION

The absolute orientation of the inertial measurement unit or its tilt is unknown in a real terrain and it is perhaps the most important step to estimate this state as accurate as possible. Any inaccuracy leads to wrong derotation from RPY coordinates to other coordinates, e.g. ENU, ECEF [12].

2.6.1 Euler angles

This way, the orientation might be defined as rotation angles about each of axes (vehicle roll, pitch and yaw axis), called Euler angles, named for the Swiss mathematician Leonard Euler (1707-1783). With this approach, it is always necessary to specify the order of rotations when specifying Euler angles. The rotation from RPY coordinates to NED coordinates can be composed from three Euler rotation matrices, consecutively yaw ψ, pitch θ and roll φ, as is shown in (2.3), respectively (2.4).

$$C_{NED}^{RPY} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\varphi) & -\sin(\varphi) \\ 0 & \sin(\varphi) & \cos(\varphi) \end{bmatrix} \tag{2.3}$$

$$C_{NED}^{RPY} = \begin{bmatrix} \cos(\psi)\cdot\cos(\theta) & -\sin(\psi)\cdot\cos(\varphi) + \cos(\psi)\cdot\sin(\theta)\cdot\sin(\varphi) & \sin(\psi)\cdot\sin(\varphi) + \cos(\psi)\cdot\sin(\theta)\cdot\cos(\varphi) \\ \sin(\psi)\cdot\cos(\theta) & \cos(\psi)\cdot\cos(\varphi) + \sin(\psi)\cdot\sin(\theta)\cdot\sin(\varphi) & -\cos(\psi)\cdot\sin(\varphi) + \sin(\psi)\cdot\sin(\theta)\cdot\cos(\varphi) \\ -\sin(\theta) & \cos(\theta)\cdot\sin(\varphi) & \cos(\theta)\cdot\cos(\varphi) \end{bmatrix} \tag{2.4}$$

This approach leads to problem with discontinuity when the pitch angle equals 90 degrees. Roll axis is then pointed upwards and any change in pitch or yaw causes ±180 degrees changes in heading angle. This is called “gimbal lock” and it is the reason why we do not use Euler angles for the orientation determination of IMUs.

In addition, it depends on the sample rate of angular rate sensing and how precise the sensor is, in the other words, computations of φ, θ and ψ during the time from gyroscope outputs, body angular rates, are mathematically very complicated.

Other methods that can be used for 3D position determination, such DCM (Direction cosine matrix) or Rotation vectors are described in my thesis. In this project, quaternions were chosen as the suitable tool.

2.6.2 Quaternions

Quaternions are members of a noncommutative division algebra first invented by William Rowan Hamilton. They are a single example of a more general class of hyper-complex numbers discovered by Hamilton, (2.5), [14]. While the quaternions are not commutative, they are associative, and they form a group known as the quaternion group, [15].

$$i^2 = j^2 = k^2 = i \cdot j \cdot k = -1 \tag{2.5}$$

Quaternion multiplication is noncommutative, the result depends on the order of multiplication. A single quaternion product, the final rotation, is determined by the quaternion product $\mathbf{q}_n$ x $\mathbf{q}_{n-1}$ … $\mathbf{q}_3$ x $\mathbf{q}_2$ x $\mathbf{q}_1$., can implement each successive rotation. The quaternion equivalent of the rotation vector $\vec{\rho}$ with $|\vec{\rho}| = \theta$, and where **u** is a unit vector, is (2.6).

$$\mathbf{q}\left(\vec{\boldsymbol{\rho}}\right) \overset{def}{=} \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) \\ \frac{\rho_1}{\theta} \cdot \sin\left(\frac{\theta}{2}\right) \\ \frac{\rho_2}{\theta} \cdot \sin\left(\frac{\theta}{2}\right) \\ \frac{\rho_3}{\theta} \cdot \sin\left(\frac{\theta}{2}\right) \end{bmatrix} = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) \\ u_1 \cdot \sin\left(\frac{\theta}{2}\right) \\ u_2 \cdot \sin\left(\frac{\theta}{2}\right) \\ u_3 \cdot \sin\left(\frac{\theta}{2}\right) \end{bmatrix} \tag{2.6}$$

When the two coordinate systems are aligned, the initial value of $\mathbf{q}_{[0]}$ equals $[1\ 0\ 0\ 0]^T$. In inertial measuring systems the initial $\mathbf{q}_{[0]}$ is determined during INS alignment procedure. We can then define the calibrated value of the orientation, quaternion $\mathbf{q}_k$, as a quaternion product as shows (2.7), where $\mathbf{q}_{k-1}$ is a prior value of attitude (a quaternion that is determined from the vector as [0; v_1; v_2; v_3]) and $\mathbf{\Delta q}$ is the change in orientation, all represented in quaternion form.

$$q_k = \Delta q_k \times q_{k-1} \times \Delta q_k^{\ *} \tag{2.7}$$

The attitude representations and rotation sequences for quaternion expressions are available in [16] and [17] for example.

2.7 ARTIFICIAL NEURAL NETWORK

An artificial neural network (ANN) enables to decide how the result of the issue should be, without any equations, relations between physical quantities, and probabilistic filters. It is based on an artificial intelligence (AI), which is the intelligence exhibited by machines or software and such problematics including learning, reasoning, knowledge, planning, communication, perception and the ability to move and manipulate objects, please see [18], [19] and [20].

Each ANN has its input values, variables on which the ANN output depends. As an output, in our case, we have one value that determines the state of the system. In other cases, more outputs may be present, see [21]. The number of hidden layers and the

number of neurons in particular layers depends on the complexity of the problem we solve, [22], lecture 2.

ANNs were used in systems for tracking, positioning or navigation as it is presented for example in [23], [24] or [25]. These applications nevertheless do not use the ANN to find out the state of the system and also these developments combine the IMU with an additional data sources. The ANNs are used to solve many specific types of issues. Always the appropriate type of ANN and the method of training and other parameters have to be chosen.

Our task is to correctly define the state in time. It offers the classification ANN as an appropriate network for the data processing. However, the task is time dependent and thus it is necessary to analyse the data during the time. The dynamic time series ANN was chosen, [26]. There are lot of kinds and types of ANNs that solve completely different issues, detailed information are provided in Chapter 5 of the thesis.

2.8 KALMAN FILTERING

Kalman filter (KF) (see literature [27] and [28]), also known as linear quadratic estimation (LQE), had become the important instrument of systems that integrate more data sources to give the solution. You can imagine this filter as an algorithm that uses sets of measurements observed over time (containing random variations of noise) and produces the estimates of unknown variables in order to obtain more precise result. An introduction to concepts gives P. S. Maybeck in [29].

Correction step:

a) Calculate the Kalman Gain

$$S = HP_k^- H^T + R$$

$$K_k = \frac{P_k^- H^T}{S}$$

b) Correct the a priori state estimate

$$x_k = x_k^- + K_k\left(z_k - h\left(x_k^-, 0\right)\right)$$

c) Correct the a posteriori error covariance matrix estimate

$$P_k = P_k^- - K_k H P_k^-$$

Prediction step:

a) Predict the state

$$x_k^- = Ax_{k-1}$$

b) Predict the error covariance matrix

$$P_k^- = A_k P_{k-1} A_k^T + Q_{k-1}$$

x_k^-	the predicted or a priori value of the estimated state vector
x_k	the corrected or a posteriori value of the estimated state vector
z	the measurement or observation vector
R	the sensor noise covariance or measurement uncertainty
Q_k	the dynamic disturbance noise covariance
P_k^-	the predicted or a priori value of estimation covariance
P_k	the corrected or a posteriori value of estimation covariance
K	Kalman gain
A	Jacobian of the system model with respect to state
H	the measurement sensitivity matrix or observation matrix
$h(x_k^-, 0)$	the predicted measurement
$z_k - h(x_k^-, 0)$	innovations vector

Figure 2.3 Kalman filter process, [33].

With respect to data from sensors and other available information, the KF estimates the result by using a form of a feedback control loop. The filter estimates the result and obtains feedback in the form of (noisy) measurement. After that, the process repeats

(see Figure 2.3). In our case, the KF gets the data from the IMU, please have a look at principles in [30]. The adaptive Kalman filtering for low-cost INS/GPS is shown in [31] and [32].The theory of the optimal state estimation is also described.

2.9 TRAJECTORY RECONSTRUCTION

Trajectory reconstruction is difficult process when the high precision is supposed to be reached and when there is not any support of additional external information system or auxiliary system implemented, [34], [35]. The successive computation of position is called strapdown navigation. In addition, Heading from the magnetometer should be taken into account. Nevertheless, surrounding environment may differ with the time and place where the measurement is performed. Thus the data from magnetometer is not always included into the strapdown IMU system. This issue is discussed in, [33].

The essential processing function includes double integration of acceleration to obtain the position, [36]. The measured angular rates are also integrated to maintain the knowledge of the IMU orientation. The initial position, velocity and orientation must be known before the initialization of integration [12]. Figure 2.4 shows the simple strapdown INS and its outputs.

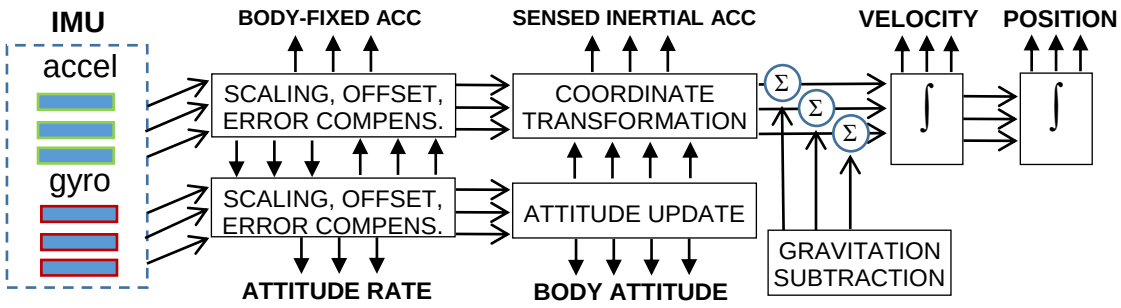

Figure 2.4 Simple strapdown INS and its outputs [12].

2.10 ISSUES AND DISSERTATION OBJECTIVES

– During the measurement, drifts and offsets arise on the output of the calibrated gyroscopes. The IMU orientation is defined (RPY to ENU). With time, the inaccuracy of orientation determination rises and thus the velocity and position is computed with enormous errors (in direction and in size).

– The acceleration also drifts during the time. Then the measured acceleration is not exactly 1 g while the IMU stays still. When the IMU stays still, the acceleration may be averaged and normalised. Nevertheless, during the walk or any other motion, the acceleration drift is not fully compensated and thus the estimated velocity (and position) may differ from the true values due to the integration of the acceleration.

– Metal objects placed close to the IMU affects the magnetometer output. It is called as soft iron offset. We assume that when the metal objects are present at a distance of at least fifteen centimetres, they do not affect measured values with the impact on the result (the magnetic north determination). The hard iron offset (the effect of the PCB, electronical components, etc.) has to be suppressed.

- SYSTEM FOR TERRESTRIAL EVALUATION OF THE CURRENT STATE

This may furnish for example GNSS navigation when the signal is lost, but, also and above all, this may be used for terrestrial indoor navigation or position determination for short distances, up to several meters, while walking, jogging, driving, etc.

The task is to develop a system that works without any step detection algorithms and map assigning, purely based on sensor outputs processing, with sufficient accuracy. It is also desirable to get a system that can be hold in the hand during its operation.

Therefore, we can define the following objectives of the dissertation:

- To develop the method for determination of the sensors orientation with respect to the navigation coordinates using only the sensor outputs while the system is essentially stationary.
- To develop the method for determination of the sensors orientation with respect to the navigation coordinates using only the sensor outputs while the system is not stationary and while it moves.

Those tasks lead to coordinate alignment ability and thus to ability of subtraction of split g-force (measured gravitational acceleration) from particular axes with eminent focus on accuracy.

- To create an artificial neural network (ANN) that recognizes and defines "what is going on" with the system and to implement it.

It leads to reduction of the positioning errors due to parasitic sensed rotation rates and accelerations. It also ensures that the integration errors will not be cumulated during whole measurement.

- To create a Kalman filter; that is a necessary element where data from the inertial sensors are used for the position determination.

After performing all these tasks, integrations may follow and the velocity and position in time may be computed as well as the trajectory of the moving object can be reconstructed. In addition, the determination of the unit orientation and heading as a function of time is available.

- To develop an IMU with application that evaluates all previous tasks and presents results.

Developed IMU will be based on proper modules with 3-D accelerometer, 3-D rotation rate sensor and 3-D magnetometer connected to appropriate MCU. Proposed application running on PC will process data from the IMU and MCU including graphical representation of results and verification of the system in which a new method of fully inertial positioning is implemented.

- To perform and evaluate series of experiments.

The complex system will be experimentally tested in different scenarios to verify improvements in positioning.

3 INERTIAL MEASUREMENT SYSTEM

These steps lead to obtaining of the basic orientation and linear acceleration used for IMU positioning from inertial sensors.

3.1 HARDWARE

The X-NUCLEO-IKS01A1 board that consists of motion MEMS and environmental sensors was chosen. It is compatible with Arduino Uno. Measured data are sent by BT or by USB cable to a PC and may be processed in real time or saved for further processing. This sensor board is designed around STMicroelectronics LSM6DS0 3-axis accelerometer and gyroscope, the LIS3MDL 3-axis magnetometer and the HTS221 humidity and temperature sensor and the LPS25HB* pressure sensor is available.

3.2 FIRMWARE

The function of Arduino UNO is to read data from sensors and send them directly to the PC. I^2C protocol is used for data retrieval from the sensors. Arduino Uno was programmed in Arduino IDE application using a programming language similar to C++.

The firmware code is divided into two sections, the first part (setup function) is performed on start-up of Arduino and applies the settings. Arduino is set as a master and sensors are set as slaves. The second part (loop function) reads the sensor data and sends them to the PC. Arduino Uno was programmed in Arduino IDE application using a programming language C++.

```
void loop()
{
 String output = "";
 readFrom(30, B00101000, 6);                          // Magnetometer reading
 while (Wire.available())
 {
   short c = Wire.read();
   c += (Wire.read() << 8);
   output += String(c, DEC) + " ";
 }
 readFrom(107, B00011000, 6);                          // Gyroscope reading
 while (Wire.available())
 {
   short c = Wire.read();
   c += (Wire.read() << 8);
   output += String(c, DEC) + " ";
 }
 readFrom(107, B00101000, 6);                          // Accelerometer reading
 while (Wire.available())
 {
   short c = Wire.read();
   c += (Wire.read() << 8);
   output += String(c, DEC) + " ";
 }
 output += String(millis(), DEC) + " ";                 // timestamp
    if(digitalRead(buttonPin) == HIGH)                  // button state
         output += "0";
    else output += "1";
    Serial.println(output);                             // send to PC
    delay(7);
}
```

3.3 DATA ACQUISITION

The firmware in Arduino defines the format in which the data are sent. In our source code, the package contains measured accelerations in *x*, *y* and *z*-axis from accelerometer (Figure 3.2), angular rate in *x*, *y* and *z*-axis from gyroscope (Figure 3.1) and magnetic field in *x*, *y* and *z*-axis from magnetometer (Figure 3.3). There is the possibility to receive the time stamp to get precise Δt, in other words, to get accurate time between two samples. In addition, Arduino sends the button state. This button allows our system to receive additional bool value that is defined by user.

Axes of all sensors have essentially the same origin but magnetometer has a different right-handed axis system than accelerometer and gyroscope. Thus the measured magnetic field vector has to be rotated. We obtain one united vehicle-fixed coordinate system RPY (Figure 3.4).

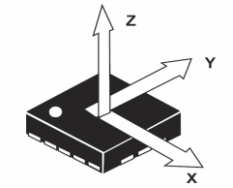

Figure 3.1 Direction of detectable angular rates

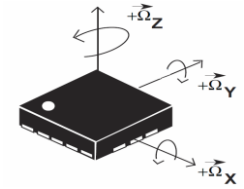

Figure 3.2 Direction of detectable accelerations

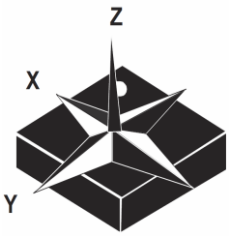

Figure 3.3 Direction of detectable magnetic field

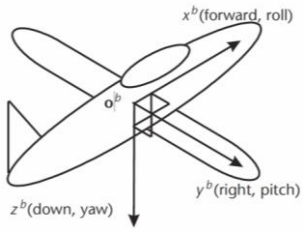

Figure 3.4 Body-fixed RPY (roll-pitch-yaw) axis

Arduino sends just a raw data from converters. Based on the register settings, the acceleration is measured in range of ±2 g and the sensitivity is then 0.061 mg/LSB. The rotation rate sensor is set to range of ±500 dps, the sensitivity is then 17.5 mdps/LSB. The magnetometer measures magnetic field in range of ±4 gauss and the sensitivity is then 0.146 mGauss/LSB. Resolution of all sensors is 16 bits. Typical magnetic field at places where the measurements were performed is about 48.897 nT (488.977 mGauss).

Finally, we get 11 values from 9-DOF device per sample (this is what the device sends: [mag_x, mag_y, mag_z, gyr_x, gyr_y, gyr_z, acc_x, acc_y, acc_z, time, button], and those sets are sent with frequency of about 86 Hz.

The required sampling frequency is given by the spectrum of the measured data during typical and particular movements. For the walk and staying still the sampling frequency about 80 Hz is satisfactory.

3.4 SENSOR CALIBRATION AND COMPENSATION

Sensor calibration is the process determining the parameters of the compensation model. Sensor compensation is the process recovering the sensor inputs from the sensor outputs. We calibrated and compensated sensors, appropriate steps and are described in details in my dissertation. We are not able to determine the heading of the IMU unless we have data from the calibrated magnetometer and we know the magnetic declination (δ) of the place where the measurement was performed. The rotation to the flat position must be applied to the calibrated magnetometer data in order to determine the heading correctly in 3D. Then, Figure 3.5 shows calibrated magnetometer data. The sin(x) and cos(x) function that is formed when the IMU rotates along a single axis by 360° with starting and ending heading equal δ – the *x*-axis points straight to magnetic north.

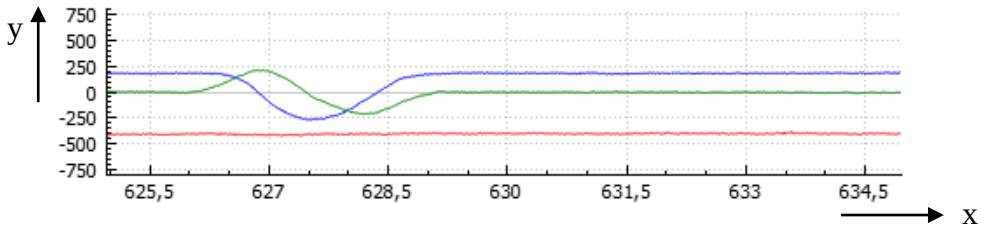

Figure 3.5 Calibrated magnetometer data (360° rotation), x-axis in [s], y-axis in [mGauss] .

Recalibration may be done also manually during the measurement. Nevertheless it is necessary to ensure the conditions for the calibration. The measurement of the still-shake-still state of the IMU is shown in Figure 3.6.

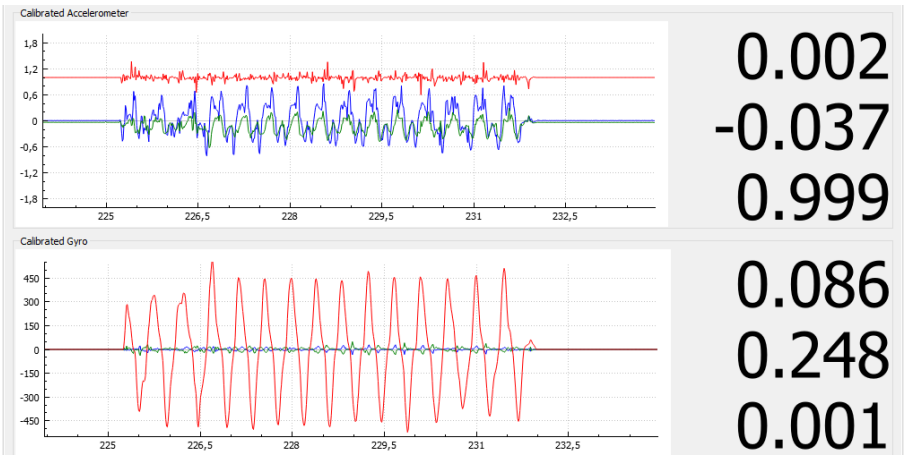

Figure 3.6 Data from calibrated sensors while shaking, x-axis in [s], y-axis in [g], [°/s].

As you can see, the rotation rate and the acceleration show "what is going on" with the IMU in the body coordinate frame in time. The IMU is placed on the table but it is not perfectly in flat position (in a horizontal plane). The *y*-axis of the gyroscope has the value of 0.248 °/s after the movement. This is the drift and it has to be suppressed for further processing. Thus, the calibration of the gyroscope is necessary during the measurement. This is exactly what the "soft" calibration performs.

3.5 NEURAL NETWORKS

The new proposed approach is primarily based on the artificial neural network that is designed in order to determine state – "what it is going on". In this work, recognition of two states are presented. The first case is that the IMU walks, the second case is that the IMU is static regardless of its orientation. In principal, further states may be added, for example jogging, running, driving, riding, shaking, flying, falling etc.

The very important piece of information is that when the ANN determines the state of the IMU incorrectly, there are two cases of the wrong decision.

1. The ANN determines the walk and the IMU is still
2. The ANN determines that the IMU stays still and the IMU walks.

The first case does not bring complications, however it is undesirable, since the ANN does not improve the positioning. The second case is unacceptable, the orientation of the moving IMU is recalculated as it is "still" (from the actual accelerometer data). The further processing of data after the incorrect determination of the orientation causes the error with a very high severity.

3.6 PRINCIPLE OF ANN

I decided to create time-delayed neural network, since the static values of one sample do not have any predictive value. Figure 3.7 shows principle of used ANN type, FFTD (feed-forward time-delayed) network, [37].

To obtain the training data we recorded a walk with stops. The person holding the IMU used the button to determine if he is walking or is standing still. Then, the neural network was trained with the input set consisting of measured data and the button state as the target output.

The script in MATLAB was written to automatic creation, training and simulation of FFTD ANN. While the train function is given, the time delay (the number of previous samples used) and the number of neurons in hidden layer changes. The training set is divided to three blocks - training part, validation part and test part. The goals has to be set appropriately.

Parameter and architecture properties such as training, performance, divide, adaptation, transfer function etc. also have a significant impact on ANN output, please see dissertation for details.

Figure 3.7 FFTD ANN principle

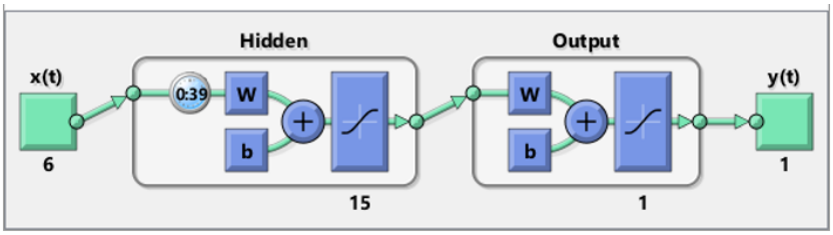

Figure 3.8 FFTD ANN structure used.

The issue is extensively described in my doctoral thesis, including particular functions and algorithms. Figure 3.9 shows the response of the ANN that was evaluated as the most suitable ANN for our purpose. The comparison of many TD ANNs is performed in the document, also the examples of other ANN types applied on our problematics are presented.

As the best-input parameters seem to be a vector of raw data from the accelerometer and a vector of raw data from the gyroscope, both in all three axes. The magnetometer data were discarded due to important dependence on the surrounding magnetic strength.

The ANN may be also trained by adjusted data from the accelerometer and gyroscope, e.g. by gravitational vector length as the first input parameter and rotation vector length as the second input parameter. Nevertheless, trained network showed worse result. Two additional ANNs were created to demonstrate the function of the ANN in INS. One of them is trained to decide on the state "still" or "anything else" and is very useful when we check the functionality of the ANN behind the PC. The second one is trained for state changes only.

The training of the ANN with chosen structure (Figure 3.8) is relatively computationally complex, see [38] and [39]. It had been trained on PC (16 GB RAM, Intel® CORE™ 4 x i5-4690K CPU @ 3.50GHz, SSD) using MATLAB™.

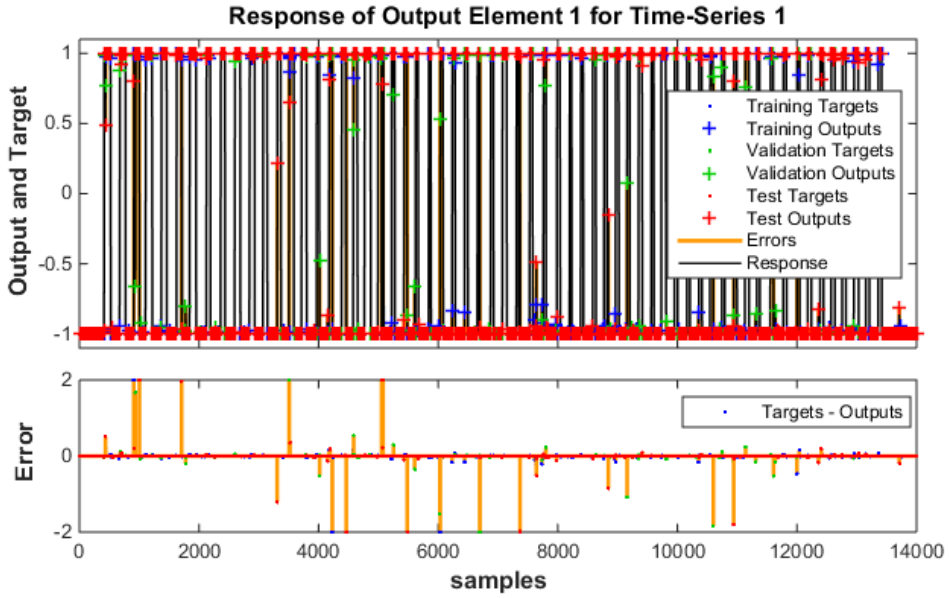

Figure 3.9 Time-series response, epoch 73, TDNN_dividerand_3_TS.

TABLE 3.1 Comparison of networks – training parameters.

Name of ANN	Duration* [h]	Epochs	Performance	Reliability
TDNN_divideblock_1	4:49:57	140	0.006440	0.993560
TDNN_dividerand_1	7:50:28	215	0.007870	0.992130
TDNN_divideblock_2	5:48:42	1000	0.006900	0.993100
TDNN_dividerand_2	0:29:49	86	0.000940	0.999060
TDNN_divideblock_3	1:52:03	367	0.000999	0.999001
TDNN_dividerand_3	0:27:15	81	0.000997	0.999003

* PC: 16 GB RAM, Intel® CORE™ 4 x i5-4690K CPU @ 3.50GHz, SSD, using MATLAB™ (used for all following experiments)

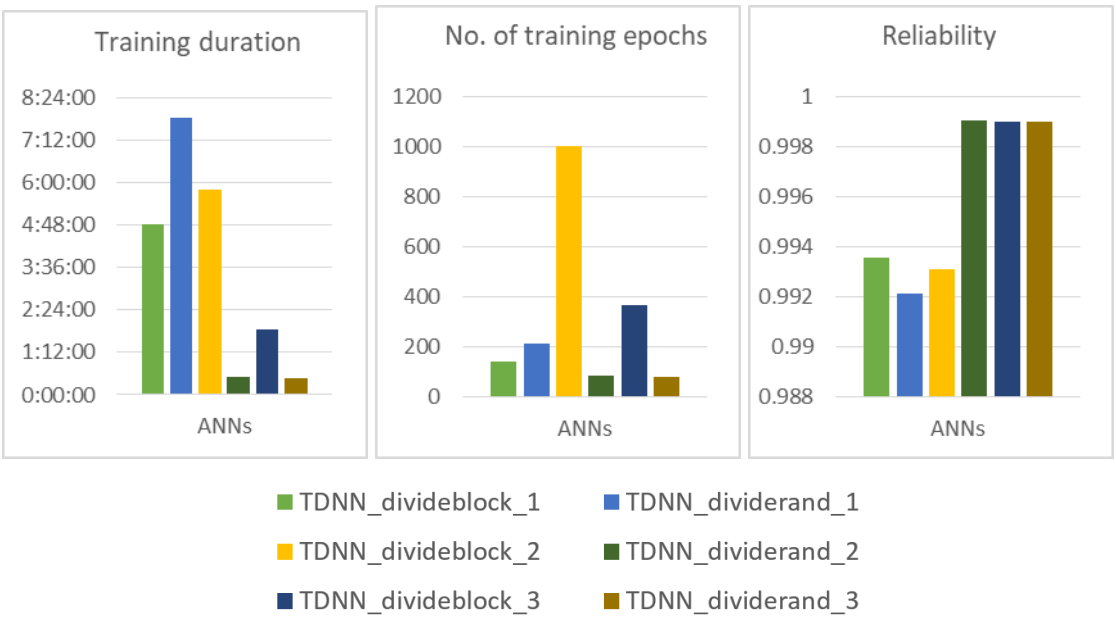

Figure 3.10 Graphical comparison of trained feed forward time delayed networks.

TABLE 3.2 Comparison of networks II – training parameters

Name of ANN	Duration [h]	Epochs	Performance	Reliability
TDNN_divideblock_1_TS	0:01:26	29	$6.23 \cdot 10^{-22}$	≈ 1
TDNN_dividerand_1_TS	0:00:46	27	$9.77 \cdot 10^{-22}$	≈ 1
TDNN_divideblock_2_TS	0:01:11	28	$5.62 \cdot 10^{-22}$	≈ 1
TDNN_dividerand_2_TS	0:02:00	28	$5.99 \cdot 10^{-22}$	≈ 1
TDNN_divideblock_3_TS	0:14:36	90	0.00187	0.99813
TDNN_dividerand_3_TS	0:09:46	73	0.00229	0.99771

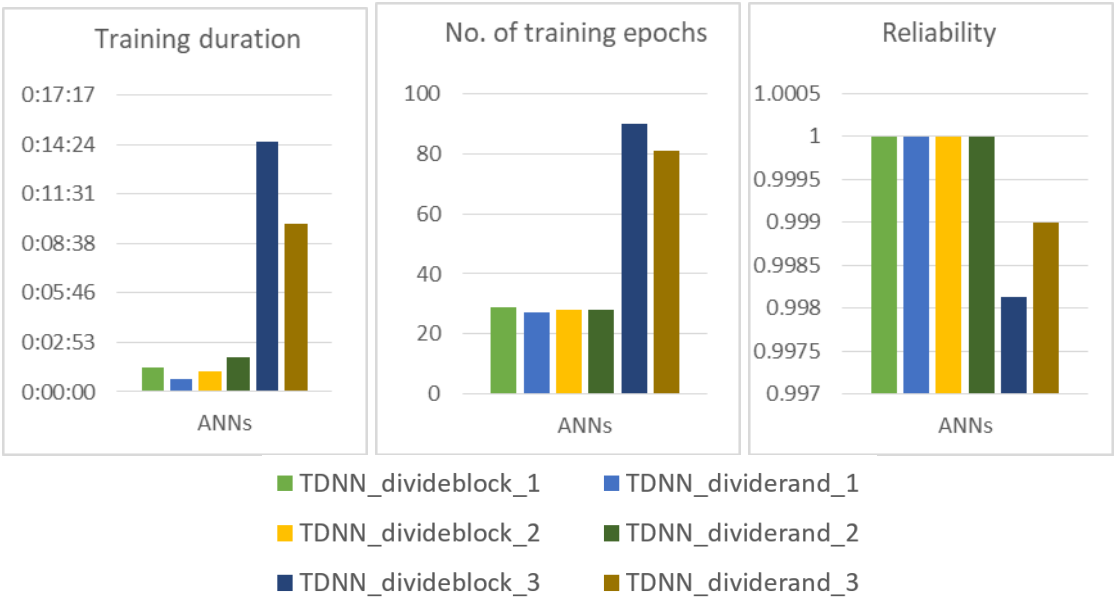

Figure 3.11 Graphical comparison of trained feed forward time delayed networks - TS

The duration, final number of epochs and achieved performance of some of the particular networks trainings are shown in TABLE 3.1, TABLE 3.2, Figure 3.10 and Figure 3.11. The reliability is shown as a complement of the performance to 1.

4 DATA PROCESSING

Thanks to the ANN we know the state of the IMU. Provided we trust it, the attitude of the IMU in time can be determined with higher accuracy. Whenever the state defined by the ANN is "still", the accurate (absolute) tilt of the IMU can be recalculated. The only property that still remains to be absolutely determined is heading.

The determination of heading comes from magnetometer data. The value that has to be checked before heading determination is total magnetic field. If the value is too low or too high, there are other influences than Earth's magnetic field, and the heading cannot be determined by the magnetometer. In that case we have to rely on integrated data from gyroscope.

In the second case, when the value of total magnetic field falls within the given range, the heading can be computed. However, it can be done after the acceleration data tilt compensation – derotation into the flat level. After the proper magnetometer data derotation the heading value can be determined only from *x*-axis and *y*-axis of magnetometer data. The absolute attitude of the IMU can be finally determined.

4.1 THE STATE IS "STILL"

The neural network decides that the IMU state is "still". Then, there are more ways, as described in theoretical part, how to rotate the IMU back into flat level (horizontal plane). Figure 4.1 clearly shows all three Euler stages of derotation from IMU (RPY) coordinates into inertial, ENU coordinates.

At first, the derotation of heading needs to be done by an angle $-\theta$. This rotation is solved separately after the derotation into flat position. Rotations by angles $-\Psi$ and $-\Phi$ define the tilt of the IMU in RPY coordinates and through them the IMU coordinates from RPY coordinate system into relative ENU coordinates can be converted.

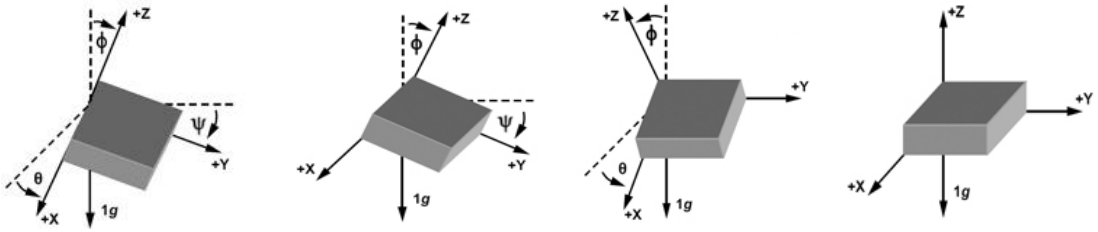

Figure 4.1 Derotation from the RPY (body) coordinates to relative ENU coordinates.

The Euler angles or direction cosine matrix (DCM) method seems to be easy to use. Nevertheless, the gimbal lock is the fundamental problem for the INS. For this reason the application of quaternions must be implemented. In proposed algorithms, the rotation around z axis is denoted by symbol Ψ, the rotation around y axis is denoted by symbol θ, and the rotation around x axis is denoted by symbol Φ.

From accelerometers, after axis alignment, compensation and calibration, the accelerations in particular axes are obtained, the vector $\vec{a} = (acc_x, acc_y, acc_z)\vec{a} = (acc_x, acc_y, acc_z)$ for each time step. In the source code, the calibrated accelerometer output is stored in three-dimensional vector `acc` with

components **acc.x**, **acc.y** and **acc.z**. In each cycle the computation of α is performed. It expresses the angle between the acceleration vector **acc** and last acceleration vector (**accXlast, accYlast, accZlast**). The vector of the last acceleration $\overrightarrow{acc}_{last}$ is always $(0,0,1)$ in order to get absolute attitude. That means it is set to the value that is present on the accelerometer output in case that the IMU is placed horizontally. The angle $\alpha^{\boldsymbol{\alpha}}$ is used in formulas (4.1) – (4.4) and based on that we are able to determine the quaternion, in my source code called orientation (Figure 4.2).

$$\alpha = \text{acos}(accXlast \cdot acc.x + accYlast \cdot acc.y + accZlast \cdot acc.z) \tag{4.1}$$

$$rotVecX = \frac{1}{\sin(\alpha)} \cdot (accYlast \cdot acc.z - accZlast \cdot acc.y) \tag{4.2}$$

$$rotVecY = \frac{1}{\sin(\alpha)} \cdot (accZlast \cdot acc.x - accXlast \cdot acc.z) \tag{4.3}$$

$$rotVecZ = \frac{1}{\sin(\alpha)} \cdot (accXlast \cdot acc.y - accYlast \cdot acc.x) \tag{4.4}$$

```
orientation = QQuaternion::fromAxisAndAngle(p, q, r, angle)
//p = rotVecX; q = rotVecY; r = rotVecZ;
//angle = -alfa*180/M_PI);
```

Figure 4.2 Quaternion ***orientation*** *determination.*

For further application, the quaternion **orientation** equals **ORI**.
The θ and the φ angle are subsequently determined from quaternion by formulas (4.5) and (4.6):

$$\theta = -\text{asin}\,(2 \cdot ORI.x \cdot ORI.z - 2 \cdot ORI.y \cdot ORI.scalar) \tag{4.5}$$

$$\varphi = \text{atan2}\left(\frac{A}{cos(\theta)}, \frac{B}{cos(\theta)}\right) \tag{4.6}$$

where:

$$A = 2 \cdot ORI.y \cdot ORI.z + 2 \cdot ORI.x \cdot ORI.scalar$$

$$B = 1 - 2 \cdot ORI.x \cdot ORI.x - 2 \cdot ORI.y \cdot ORI.y$$

This algorithm derotates the IMU to the nearest flat level (by the smallest angle). It means that there may occur some undesirable rotation around z-axis during this derotation. It is defined as **zrot** and expressed by (4.7). This must be subtracted (the IMU is rotated by the **zrot** quaternion in opposite direction) from computed attitude, see the part of the source code, Figure 4.3.

Now the IMU's tilt is defined by the quaternion **rotToFlat**. Then the last rotation around the *z*-axis by the heading angle is performed to the orientation of the IMU into the ENU coordinate system. The acceleration in the ENU coordinates is then expressed by the vector **accDerot**. Once the **ORI** is defined, the acceleration **acc** may be rotated by the **ORI** quaternion conjugation. After that, the full size of gravitational acceleration occurs in *z*-axis and thus the gravitational acceleration (1 g) can be subtracted in the *z*-axis to getting of the IMU inertial acceleration only.

$$zrot = \operatorname{atan2}\left(\frac{A}{cos(\theta)}, \frac{B}{cos(\theta)}\right) \tag{4.7}$$

where: $A = 2 \cdot ORI.y \cdot ORI.x + 2 \cdot ORI.z \cdot ORI.scalar$

$B = 1 - 2 \cdot ORI.z \cdot ORI.z - 2 \cdot ORI.y \cdot ORI.y$

```
// attitude from accelerometer
 float alfa, cosAlfa;
 float rotVecX, rotVecY, rotVecZ;
 float accXlast=0, accYlast=0, accZlast=1;
 acc.normalize();
 cosAlfa=accXlast*acc.x()+accYlast*acc.y()+accZlast*acc.z();
 alfa=acos(cosAlfa);
 rotVecX=1/sin(alfa)*((accYlast*acc.z()-accZlast*acc.y()));
 rotVecY=1/sin(alfa)*((accZlast*acc.x()-accXlast*acc.z()));
 rotVecZ=1/sin(alfa)*((accXlast*acc.y()-accYlast*acc.x()));
// remove parasite
 orientation = QQuaternion::fromAxisAndAngle(0,0,-1,zrot)*orientation;
 rotToFlat = orientation;
 orientation = QQuaternion::fromAxisAndAngle(0,0,1,heading)*orientation;
// derotation
 QVector3D accDerot = orientation.rotatedVector(acc);
 QVector3D magFlat = rotToFlat.rotatedVector(mag);
```

Figure 4.3 A part of c-code for derotation.

As it was written above, when the IMU is still, we can compute the heading from the magnetometer. When the measured data are rotated to flat level (**`magFlat`**, Figure 4.3) and the result of absolute magnetic field meets the conditions (the outcome has to be found within the interval ⟨44985.24;52808.76⟩ nT $\langle 44985.24; 52808.76 \rangle\ nT$, this is ± 8 % from the standard environment where the measurements were performed, obtained as an average value from long-term measurement).

The computation of the heading from magnetometer in 2D is shown in relation (4.8); the vector **`magFlat`** is used. Because data measured by magnetometer are transformed into flat level, the determination of the heading (δ) in 2D is correct. When the magnetometer data are not derotated into level, another formula for declination determination has to be used.

$$\begin{aligned}
&\delta = \frac{\pi}{2} - atan\left(\frac{magFlat.x}{magFlat.y}\right) && when\ magFlat.y > 0 \\
&\delta = 3 \cdot \frac{\pi}{2} - atan\left(\frac{magFlat.x}{magFlat.y}\right) && when\ magFlat.y < 0 \\
&\delta = \pi && when\ magFlat.y = 0 \ \ and \ \ magFlat.x < 0 \\
&\delta = 0 && when\ magFlat.y = 0 \ \ and \ \ magFlat.x > 0
\end{aligned} \tag{4.8}$$

This gives us the heading (azimuth), the direction of the IMU *x*-axis due to the magnetic north (inertial *x*-axis), [42]. To be correct, the geodetic declination must be added to get the heading regarding to the geographic north. Of course, it has to be adjusted for different locations by different values, which are determined in map charts [40]. For Brno, Kohoutovice, the declination reaches 4.16 ° [41]. This is positive, east declination and thus the value has to be subtracted from the calculated true azimuth.

4.2 THE STATE IS "WALKING"

ANN decides that the IMU state is "walking". The accelerometer measures the gravitational acceleration mixed with inertial acceleration that it needs to be separated. At first, the transfer from RPY to ENU coordinates is performed. This is quite difficult since the gyroscope has almost full confidence. Then the full size of gravitational acceleration occurs in *z*-axis again and 1 g can be subtracted to get the inertial acceleration only.

The accelerometer data cannot be used for the attitude determination and we have to use only the data that was measured by the gyroscope. Figure 4.4 shows how the quaternion **deltaFrame** is defined and how the quaternion **orientation** is rotated by **deltaFrame** to find out the new **orientation.**

```
// attitude from gyro
    QQuaternion deltaFrame;
    float q0=1, q1=0, q2=0, q3=0, gy, gz, gx;
        gx=gyr.x()/180*M_PI;
        gy=gyr.y()/180*M_PI;
        gz=gyr.z()/180*M_PI;

    float qDot1,qDot2,qDot3,qDot4;
        qDot1 = 0.5 * (-q1 * gx - q2 * gy - q3 * gz);
        qDot2 = 0.5 * (q0 * gx + q2 * gz - q3 * gy);
        qDot3 = 0.5 * (q0 * gy - q1 * gz + q3 * gx);
        qDot4 = 0.5 * (q0 * gz + q1 * gy - q2 * gx);

        q0 += qDot1 * frameTime;
        q1 += qDot2 * frameTime;
        q2 += qDot3 * frameTime;
        q3 += qDot4 * frameTime;
        deltaFrame = QQuaternion(q0,q1,q2,q3);

orientation = orientation * deltaFrame;    //successive attitude determ.
headG = headingFromQuat(orientation) *180/M_PI; //heading after the mov.
orientation = QQuaternion::fromAxisAndAngle(0,0,-1,headG)*orientation;
```

Figure 4.4 Single step of successive attitude determination.

The matrix **dcm** (direct cosinus matrix, DCM) is created from recalculated quaternion **orientation** (after the **deltaFrame** effect application). All over, the θ and the φ angle are determined. The rotation around the *z*-axis, **zrotation,** is computed by equation (4.8). The last rotation, the rotation around the *z*-axis, is then performed. The acceleration in the ENU coordinate frame is then expressed by the **accDerot** vector.

$$zrotation = atan2\left(\frac{A}{cos(\theta)}, \frac{B}{cos(\theta)}\right) \tag{4.9}$$

where:

$$A = 2 \cdot ORI.y \cdot ORI.x + 2 \cdot ORI.z \cdot ORI.scalar$$

$$B = 1 - 2 \cdot ORI.z^2 - 2 \cdot ORI.y^2$$

The heading is computed the same way as in the case when the IMU is "still". In this case, when the IMU state is "walking" there is not parasite rotation present. On the other hand, the heading is computed only from the integrated data that was measured by the gyroscope as well as the tilt and this leads to growing inaccuracy with time, as mentioned above. We can improve the accuracy by replacing heading calculated from the gyroscope by absolute heading from the magnetometer.

4.3 PROBLEMS AND THEIR SOLUTIONS

Because of data history is needed in process, the type of the ANN cannot be from group of the classification ANNs. It results in a fact that the continuous output is in the range from -1 to +1 (in classification ANN the output is exactly -1 or +1, see Chapter 5 in dissertation). To distinguish of two states (walking and staying still) a decision border has to be defined. In addition, the neural network output is filtered by a KF and due to this filtering, some delay in network decision on state occurs. The movement is then evaluated later and the orientation is determined in a bad way.

To avoid this, two additional steps are performed:

1. Auxiliary condition has to be met to classify the IMU state as "still", see (4.10), where δ_{acc} is the actual deviation in [g] of the measured acceleration and (4.11), where δ_{gyr} is defined by maximum value of rotation rate. This condition block is called **SillyStatus** filter and it returns "0" when any of the values was out of required range in last N samples, otherwise it returns "1". Tolerance of acceleration deviation δ_{ACC} was set to 0.01 g and the rotation rate tolerance was set to 2 °/s.

$$\delta_{acc} = \sqrt{\left(acc_x{}^2 + acc_y{}^2 + acc_z{}^2\right)} - 1 \tag{4.10}$$

$$\delta_{gyr} = \max(gyr_x,\ gyr_y,\ gyr_z) \tag{4.11}$$

2. The second step is to delay the data processing by N samples in order to be able to "see the future". Whenever the state changes to "walking", the walk processing algorithm runs using N samples in advance and thus the KF delay and ANN delay are eliminated. This ensures that the **orientation** quaternion is not absolutely computed from accelerometer while the IMU is already moving.

Another issue is heading computation when the IMU's *x*-axis points upwards or downwards. When such a situation occurs, the heading is not defined and the **orientation** (particularly the rotation around ENU *z*-axis) is computed fully from rotation rate sensor, regardless of whether the IMU is "still" or not.

5 SOFTWARE FOR THE IMS – TRACKER

As a suitable environment Qt has been chosen. Qt is the leading independent technology for cross-platform development. An application for data processing and visualization was created including visualization. Besides the main.cpp file, further C++ source files have been developed:

Main source codes for processing:

receiver.cpp	derotator.cpp	filehandler.cpp
calibrator.cpp	integrator.cpp	brain.cpp

Visuals source codes:
- calstatus.cpp
- glcubevidget.cpp
- qcustomplot.cpp
- tcompass.cpp

Utils source codes:
- brain.cpp
- pplotgroup.cpp
- putils.cpp

ANN source codes:
- genericnetwork.cpp
- mod.cpp

User interface source codes:
- cubedialog.cpp
- customtab.cpp
- packformater.cpp
- settings.cpp
- vizualizer.cpp

The pointer with an arrow defines the deviation from the North and the number of consequently incoming samples in last state (still, walk, other IMU modes), see Figure 5.1. Application allows to track the IMU body in 3D and it shows information about the number of samples per second (SPS), the position, the velocity and the actual orientation quaternion value in time (Figure 5.2).

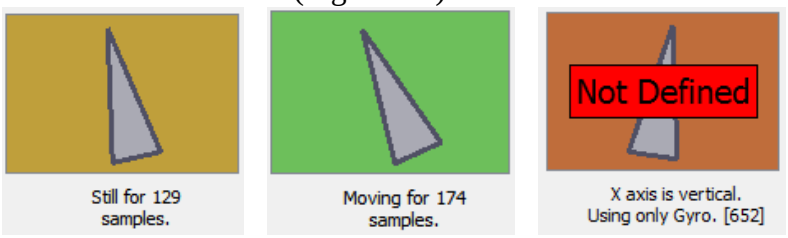

Figure 5.1 State pointers.

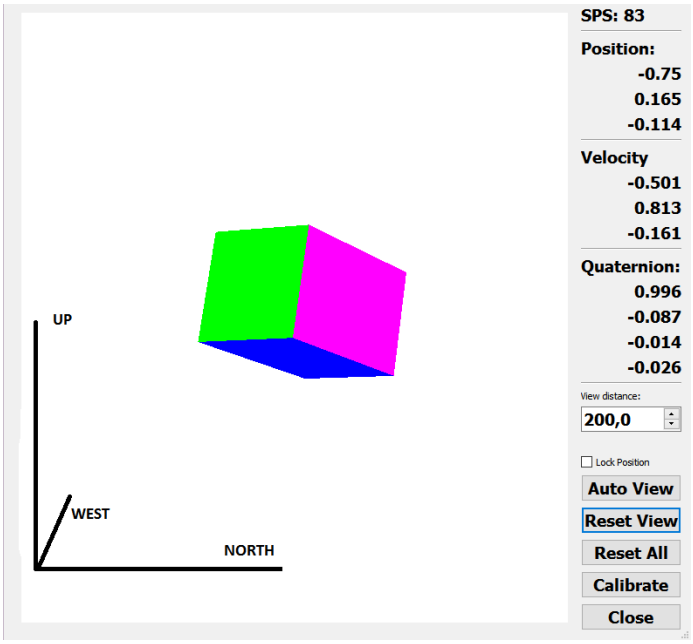

Figure 5.2 Cube view.

All functions that may be plotted in the application:

Raw Acc.	Raw accelerometer data	Cal. Acc.	Calibrated accelerometer data
Raw Gyro	Raw gyroscope data	Cal. Gyro	Calibrated gyroscope data
Raw Mag.	Raw magnetometer data	Cal. Mag.	Calibrated magnetometer data
ENU ACC	Linear acceleration in ENU	ENU KF ACC	Linear acceleration in ENU after KF
Euler angles	Euler angles of actual orientation	Velocity	Velocity in ENU coordinate frame
Neural output	ANN output, ANN output after KF, combined state (with SillyStatus)	Position	Position in ENU coordinate frame

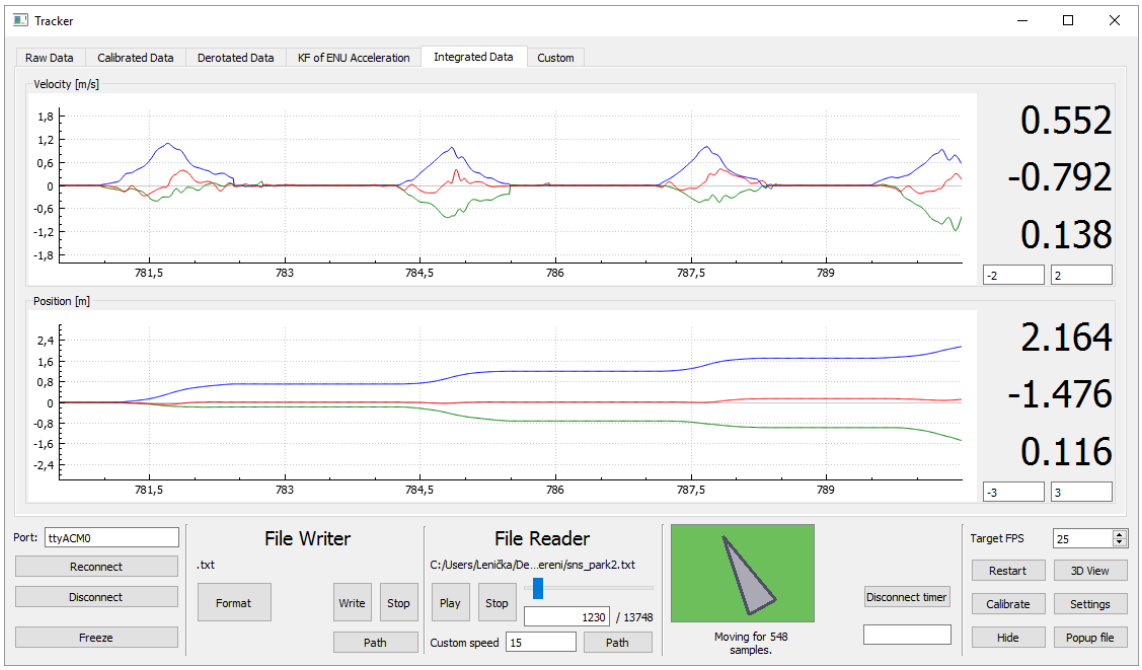

Figure 5.3 Integrated data.

Figure 5.3 represents two of twelve possible output functions processed by the developed TRACKER software – computed velocity and position in time during the measurement.

6 ONE OF EXPERIMENTAL MEASUREMENTS

Step and Stop motion, the IMU is held in the hand

Step and Stop measurement contain both, steps and the still phases. The measurement was performed in Brno, Cerna pole. The true shape of the trajectory is a square with a side length of about a = 4 m (A => B => C => D). This trajectory repeats for 5 minutes. The detailed positions are shown in Figure 6.1. These data are summarized in TABLE . The direction to the North (x-axis of the IMU) is positive. The direction to the East (y-axis of the IMU) is negative.

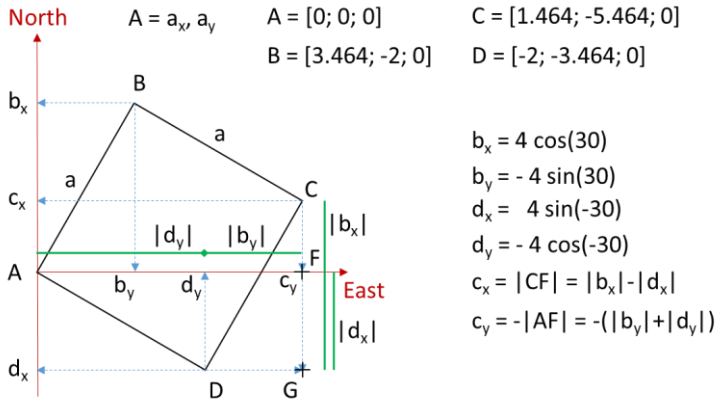

Figure 6.1 Detailed positions of Step and Stop measurement.

The direction to the East (*y*-axis of the IMU) is negative. That is why the minus is present in calculations when the distance in *y*-axis is computed.

TABLE 6.1 Positions of Step and Stop measurement.

square part	length sum [m]	position [m]	azimuth [°]
1. A => B	4	A = [0.00; 0.00; 0]	30
2. B => C	8	B = [3.464; 2.0; 0]	120
3. C => D	12	C = [1.464; 5.464; 0]	210
4. D => A	16	D = [-2.0; 3.464; 0]	300

The first three steps are presented in following figures. In this example you can observe the rising inaccuracy in time. The SillyStatus state (red coloured) and neural network output (raw output is green coloured, output after Kalman filtration is blue coloured) is shown in Figure 6.2. The *x*-axis of all graphs represents time in [s].

In Figure 6.3 the velocity in ENU coordinate system in UOG mode is shown. The corresponding position in time is depicted in Figure 6.6. Again, the velocity in ENU coordinates in shown in Figure 6.4, however the estimation in time is adjusted by SillyStatus filter. The corresponding position in time to this result is depicted in Figure 6.7. The last one mode is shown in Figure 6.5. This is the velocity in ENU coordinates estimated when the ANN is applied. The corresponding position shows Figure 6.8

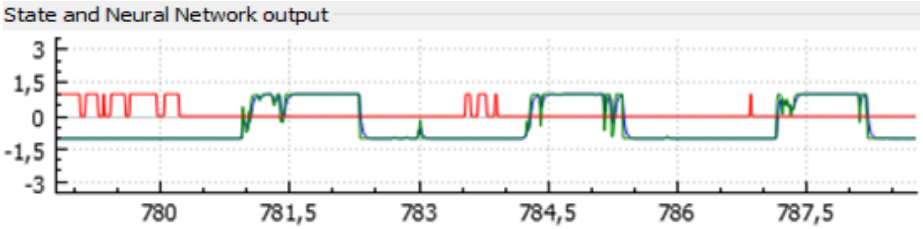

Figure 6.2 SillyStatus state (red) and ANN output state in time [s].

TABLE 8.8 in dissertation thesis shows the estimated outputs in all three modes, UOG mode, SillyStatus mode and the mode with our ANN in process. The expected values, if available, are shown in the last column.

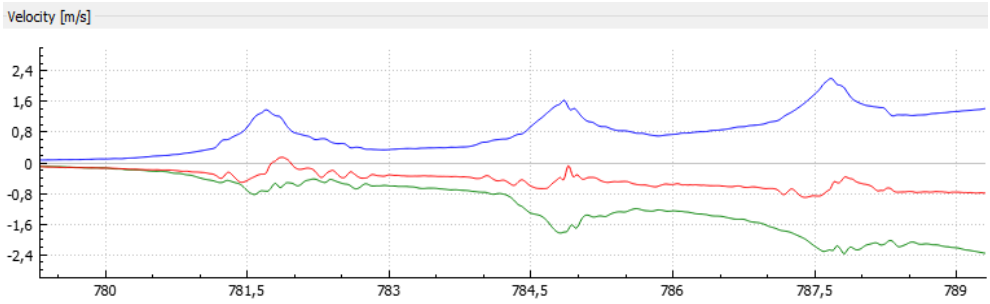

Figure 6.3 ENU velocity estimated in UOG mode, x-axis represents meas. time in [s].

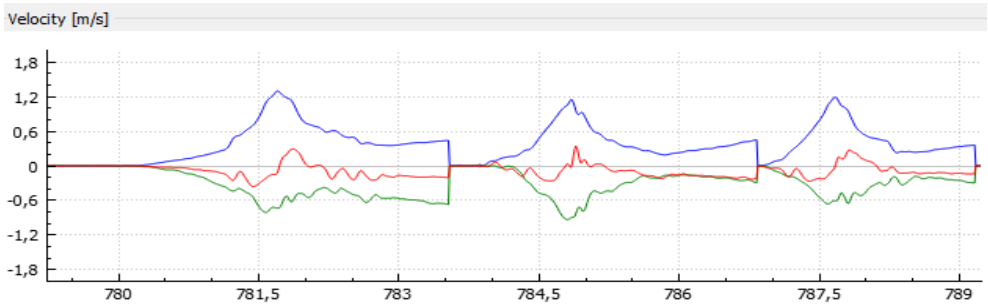

Figure 6.4 ENU velocity estimated with SillyStatus filter, x-axis represents meas. time in [s].

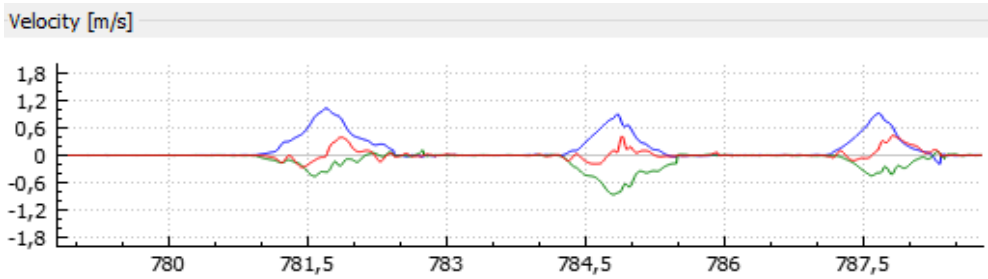

Figure 6.5 ENU velocity estimated with ANN in process, x-axis represents meas. time in [s].

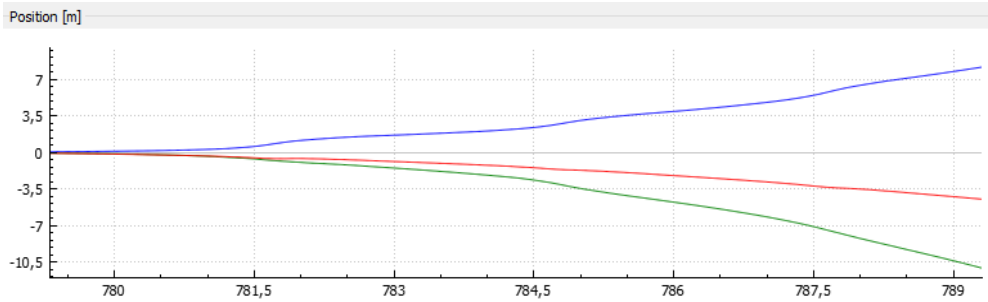

Figure 6.6 ENU position estimated in UOG mode, x-axis represents meas. time in [s].

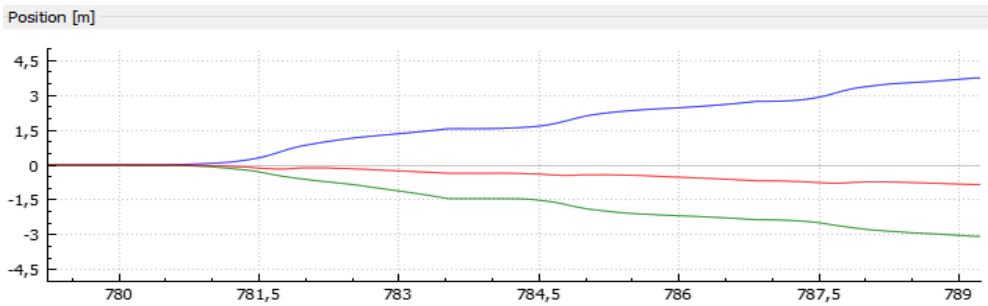

Figure 6.7 ENU position estimated with SillyStatus filter, x-axis represents meas. time in [s].

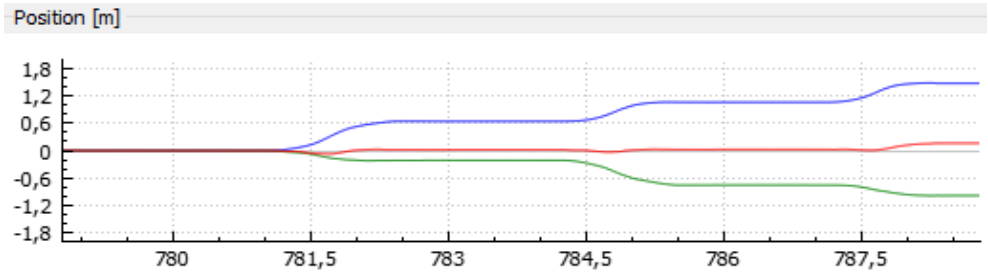

Figure 6.8 ENU position estimated with ANN in process, x-axis represents meas. time in [s].

Inaccuracies arise when rotations by 90 degrees occurs. The applied ANN is not trained to rotations around z axis without human steps. The graph of distances from the (0, 0, 0) position in ENU coordinate system is shown in Figure 6.9. Figure 6.10 shows the reconstructed 2D trajectory (East-North view) and Figure 6.11 shows the reconstructed 3D trajectory (East-North-Up view) of this measurement using the inertial measurement system only. Graphs show 60 seconds of measurement.

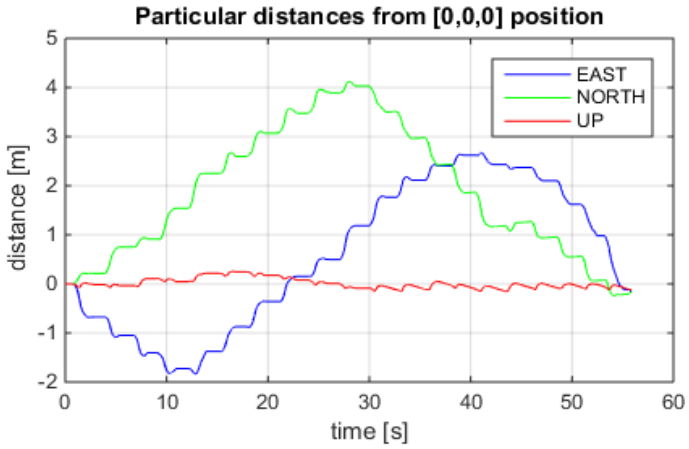

Figure 6.9 Particular distances from [0,0,0] position in ENU coordinates.

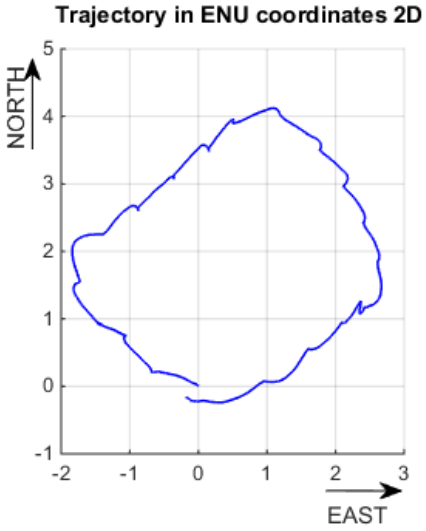

Figure 6.10 Trajectory in ENU coordinates, 2D, [m].

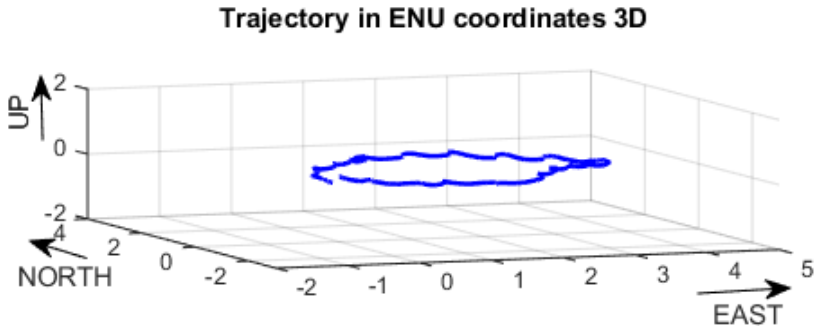

Figure 6.11 Trajectory in ENU coordinates, 3D, [m].

Other experimental measurements are fully described in detail in my dissertation.

7 CONCLUSIONS

In proposed dissertation thesis, the processing of inertial navigation sensor data is presented. As the new approach method I decided to estimate the state of the IMU by an artificial neural network without any support of auxiliary or global positioning system. It ensures that the data from the inertial sensors are processed typically (with the integration disadvantages) only for a vital period. The orientation of the IMU is fully derivate from inertial sensors.

The correction of the IMU orientation is performed during data processing when the ANN decides that the IMU is still. That leads to more accurate positioning based on DR, regardless of the environment (indoor, outdoor, underground, etc.). As the IMU

hardware, Arduino UNO was chosen in combination with ST Nucleo expansion board, which contains all used inertial sensors.

Special positioning system software called Tracker was developed in C++ programming language using Qt framework. It also offers graphical environment for the user. It process the data from the IMU and presents various intermediate and final results. The system also allows to record data into a file in adjustable format – raw/calibrated/derotated sensor data, Euler angles, heading, velocity and position in all available modes. A window with 3D IMU model is also available.

Proposed ANN was designed in MATLAB^{TM} software and estimates the state of the IMU based on the previous 40 values from inertial sensors, the type of the ANN is time-delayed feed-forward. It does not take the data from magnetometer into account, because of the magnetic field typically extremely fluctuates. The output of the ANN defines the state of the IMU – „walking“ or „staying still“, which is applied in data processing to improve positioning.

Such a system works very precisely in case that the IMU stays still on the table or stays still in the hand. In those cases, the error in positioning reached about 2 millimetres in the case the IMU was lying on the table and about 20 centimetres in case the IMU was held in the hand, after 2 minutes of acquisition.

The very interesting results were achieved when the IMU was held in the hand and the user performed a walk that often changes with still phases. Such a motion can be seen for example in a museum or in an art gallery. In these cases proposed system achieves very small positioning errors compared to the systems based purely on DR method. As shown in Chapter 8.3 in the dissertation, the INS achieved the error of only 2 meters after 2 minutes of measurement in 2D (horizontal positioning). The error in vertical *z*-axis reached up to 5 meters and that was caused by subtraction of the inaccurately determined earth’s gravitational acceleration constant.

In situations when the ANN decides that the IMU is still, the system is recalibrated and the cumulative error caused by integration is reset. Thus the position during discontinuous walking is effectively estimated with low error. When the walking motion is present during the measurement only, this method fails and the INS works as a simple DR system (however, in a real world a man must stop anytime).

In this dissertation thesis, proposed method based on ANN state recognition has been successfully validated by experiments focused on pedestrian movements. Anyway, more applications can be found in a human life in which this method could improve positioning, for example in specific professions, military applications or different types of vehicles. It opens new opportunities in future research for specific applications where the suitable artificial neural network structure have to be investigated and properly trained or modified with wider classification group (more types of movements).

REFERENCES

[1] Titterton, D. H. and J. L. Weston. *Strapdown inertial navigation technology*. 2nd ed. Stevenage: Institution of Electrical Engineers, c2004. ISBN 08-634-1358-7.

[2] Woodman, O. *An introduction to inertial navigation.* Technical Report 696, University of Cambridge, 2007. UCAM-CL-TR-696. ISSN 1476-2986.

[3] Beomju, S. *et al*. Indoor 3D pedestrian tracking algorithm based on PDR using smartphone. In *Control, Automation and Systems (ICCAS), 2012 12th International Conference on* , vol., no., pp.1442-1445, 17-21 Oct. 2012.

[4] Takai, M. and T. Ura. A model based self-diagnosis system for autonomous underwater vehicles using artificial neural networks. *Proceedings of IEEE/ASME International Conference on Advanced Intelligent Mechatronics.* Tokyo, Japan: IEEE, 1997, p. 82. DOI: 10.1109. ISBN 0-7803-4080-9.

[5] Nordlund, P-J. and F. Gustafsson. Recursive estimation of three-dimensional aircraft position using terrain-aided positioning. *In: IEEE International Conference on Acoustics Speech and Signal Processing.* IEEE, 2002, II-1121-II-1124. Orlando, 2002. DOI: 10.1109/ICASSP.2002.5743996. ISBN 0-7803-7402-9.

[6] Chiang, K.-W. and N. El-Sheimy. INS/GPS Integration Using Neural Networks for Land Vehicle Navigation Application. *Proceedings of the 15th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS 2002)*, pp. 535-544. Portland, Oregon, 2002.

[7] Chiang, K.-W.; El-Sheimy, N.; Lachapelle, G. An Adaptive Neuro-fuzzy Model for Bridging GPS Outages in MEMS-IMU/GPS Land Vehicle Navigation. *Proceedings of the 17th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2004)*, pp. 1088-1095. Long Beach, California, 2004.

[8] Toth, C.; Grejner-Brzezinska, D.A.; Moafipoor, S. Pedestrian Tracking and Navigation Using Neural Networks and Fuzzy Logic. *2007 IEEE International Symposium on Intelligent Signal Processing*, Alcala de Henares, 2007, pp. 1-6. DOI: 10.1109/WISP.2007.4447525.

[9] Moafipoor, S.; Grejner-Brzezinska, D.A.; Toth, C.K. Multi-sensor personal navigator supported by adaptive knowledge based system: Performance assessment. *2008 IEEE/ION Position, Location and Navigation Symposium*, Monterey, CA, 2008, pp. 129-140. DOI: 10.1109/PLANS.2008.4570049.

[10] Zheping, Y.; Dongnan, Ch.; Zhi, Z.; Chao, D. Dead reckoning error compensation algorithm of AUV based on SVM. *OCEANS 2011 MTS/IEEE KONA*, pp.1-7, Sept. 2011. Waikoloa, HI, 2011. DOI: 10.23919/OCEANS.2011.6107037.

[11] Wind frame and body-fixed frames. ECI-ECEF and geodetic coordinate frames - Scientific Figure on ResearchGate. [accessed Nov 11, 2015]. Available from: http://www.researchgate.net/figure/278680835_fig1_Figure-4-4.4-ECI-ECEF-and-geodetic-coordinate-frames.

[12] Grewal, M. S.; Andrews, A. P.; Bartone, Ch. G. *Global navigation satellite systems, inertial navigation, and integration*. Third edition. Hoboken: John Wiley, 2013. ISBN 978-1-118-44700-0.

[13] Aggarwal, P.; Syed, Z.; Niu, X.; El-Sheimy, N. A Standard Testing and Calibration Procedure for Low Cost MEMS Inertial Sensors and Units. *2008. Journal of Navigation*, 61, pp 323-336. DOI: 10.1017/S0373463307004560.

[14] Hamilton, W. R. and Ch. J. Joly. *Elements of quaternions*. [3d ed.]. New York: Chelsea Pub. Co, 1969. ISBN 08-284-0219-1.

[15] Altmann, S., L.; Andrews, A., P.; Bartone, Ch. *Rotations, quaternions, and double groups*. New York: [Ny] udg. Mineola, New York: Dover, 2005. ISBN 04-864-4518-6.

[16] Yang, Y. Spacecraft attitude determination and control: Quaternion based method. *In Annual Reviews in Control*. Volume 36, Issue 2, 2012, Pages 198-219. ISSN 1367-5788.

[17] Kuipers, J., B. *Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality*. Princeton: Princeton University Press, 2002. ISBN 978-0691102986.

[18] Tanikic, D. and V. Despotovic. *Artificial Intelligence Techniques for Modelling of Temperature in the Metal Cutting Process.* Metallurgy - Advances in Materials and Processes. InTech, 2012. DOI: 10.5772/47850. ISBN 978-953-51-0736-1.

[19] Sumathi, S.; Hamsapriya, T.; Surekha, P. *Evolutionary intelligence: an introduction to theory and applications with Matlab.* Online-Ausg. Berlin: Springer, 2008. ISBN 978-354-0753-827.

[20] Hassoun, M. H. *Fundamentals of artificial neural networks*. Cambridge, Mass.: MIT Press, c1995, 511 p. ISBN 02-620-8239-X.

[21] Tejmlová, L. and J. Šebesta. Design of wideband Wilkinson dividers using neural network. In *Proceedings of 23rd International Conference Radioelektronika 2013*. 2013. p. 204 - 208. ISBN 978-1-4673-5517-9

[22] Tomás, L.-P., and L. Kaelbling. *Techniques in Artificial Intelligence (SMA 5504),* lecture 2, Fall 2002. Massachusetts Institute of Technology: MIT OpenCourseWare, MIT Course Number 6.825. http://ocw.mit.edu (Accessed 4 Feb, 2015). License: Creative Commons BY-NC-SA.

[23] Ji, Y.; Zhang, M.; Liu, G.; Liu, Z. Positions research of agriculture vehicle navigation system based on Radial Basis Function neural network and Particle Swarm Optimization. *2010 Sixth International Conference on Natural Computation*. Yantai, Shandong, 2010, pp. 480-484. DOI: 10.1109/ICNC.2010.5583145.

[24] Hu, Y.; Xu, J.; Zhong, H.; Wu,Y. Fusion model of vehicle positioning with BP neural network. *Proceedings of the 2003 IEEE International Conference on Intelligent Transportation Systems, 2003, pp. 643-648 vol.1.* DOI: 10.1109/ITSC.2003.1252031.

[25] Omura, Y.; Funabiki, S.; Tanaka, T. A monocular vision-based position sensor using neural networks for automated vehicle following. Power Electronics and Drive Systems, 1999. PEDS '99. Proceedings of the IEEE 1999 International Conference on, 1999, pp. 388-393 vol.1. DOI: 10.1109/PEDS.1999.794594.

[26] Touretzky, D. and K. Laskowski. Artificial Neural Networks: Neural Networks for Time Series Prediction. *In Engineering Applications of FPGAs*, pp 117-150. Springer, Cham. ISBN 978-3-319-34113-2.

[27] Bishop, G. and G. Welch. An introduction to the Kalman filter. *Proceedings of SIGGRAPH 2001 course 8, In Computer Graphics, Annual Conference on Computer Graphics & Interactive Techniques.* Los Angeles, CA, USA. SIGGRAPH 2001 course pack edition, 2001.

[28] Simon, D. *Optimal state estimation: Kalman, H [infinity] and nonlinear approaches.* Hoboken, N.J.: Wiley-Interscience, 2006. ISBN 9780471708582.

[29] Maybeck, P. S. The Kalman filter: An introduction to concepts. Autonomous Robot Vehicles. I. J. Cox and G. T. Wilfong. New York, Springer-Verlag: 194- 204, 1990. ISBN 0387972404.

[30] Wu, Z.; Yao, M.; Ma, H.; Jia, W. Improving Accuracy of the Vehicle Attitude Estimation for Low-Cost INS/GPS Integration Aided by the GPS-Measured Course Angle. In *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 553-564, June 2013. DOI: 10.1109/TITS.2012.2224343.

[31] Hide, Ch.; Moore, T.; Smith, M. Adaptive Kalman Filtering for Low-cost INS/GPS. *Journal of Navigation*, 56, pp 143-152. DOI:10.1017/S0373463302002151.

[32] Asada, H. *Identification, Estimation, and Learning*. *Spring 2006,* Massachusetts Institute of Technology: MIT OpenCourseWare, MIT Course Number 2.160. http://ocw.mit.edu (Accessed 4 Feb, 2015). License: Creative Commons BY-NC-SA.

[33] Tejmlová, L. *Fusion methods for GNSS/INS using neural networks for precision navigation.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2013. 24 p. Dissertation preview. Supervisor: doc. Ing. Jiří Šebesta, Ph.D.

[34] Barrios, C. and Y. Motai. Improving Estimation of Vehicle's Trajectory Using the Latest Global Positioning System With Kalman Filtering. *In IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 12, pp. 3747-3755, Dec. 2011. DOI: 10.1109/TIM.2011.2147670.

[35] Borenstein, J.; Everett, H. R.; Feng, L; Wehe, D. *Mobile robot positioning: Sensors and techniques*. J. Robotic Syst., 14: 231–249, 1997. DOI: 10.1002/(SICI)1097-4563(199704) 14:4<231::AID-ROB2>3.0.CO;2-R.

[36] Sukkarieh, S.; Nebot, E. M.; Durrant-Whyte, H. F. A high integrity IMU/GPS navigation loop for autonomous land vehicle applications. In *IEEE Transactions on Robotics and Automation*, vol. 15, no. 3, pp. 572-578, Jun 1999. DOI: 10.1109/70.768189.

[37] Time delay neural network. MathWorks: Accelerating the pace of engineering and science, [online]. 2016 [cit. 2016-07-26]. Available from: https://www.mathworks.com/help/nnet/ref/timedelaynet.html.

[38] Tejmlová, L. and J. Šebesta. Fusion methods for INS using neural networks for precision navigation. In *International Conference on Indoor Positioning and Indoor Navigation*, 2013, Montbéliard, France. Piscataway, N.J.: IEEE, p. 1-5. ISSN 2162-7347.

[39] Sheela, K. G. and S. N. Deepa. Review on Methods to Fix Number of Hidden Neurons in Neural Networks. *Mathematical Problems in Engineering*, vol. 2013, Article ID 425740, 11 pages, 2013. doi:10.1155/2013/425740.

[40] NOAA. National Geophysical Data Center: *National Centers for Environmental Information* [online]. [cit. 2015-4-16]. Available from: http://www.ngdc.noaa.gov/geomag-web.

[41] Calculation of Magnetic Declination. *Find Magnetic Declination* [online]. [cit. 2016-04-04]. Available from: http://www.geosats.com/magdecli.html//www.geosats.com/magdecli.html.

[42] Henke, D. *Magnetometer Reading to Compass Heading*, [online]. Small Golden SceptreTechnology, 2016 [cit. 2016-07-26]. Available from: http://mythopoeic.org/magnetometer/.

OWN PUBLICATIONS

Tejmlová, L.; Šebesta, J.; Zelina, P. Artificial Neural Networks in an Inertial Measurement Unit. In *Proceedings of 26th International Conference Radioelektronika 2016*. 2016. p. 176 - 180. ISBN 978-1-5090-1673-0.

Tejmlová, L. and J. Šebesta. Fusion methods for INS using neural networks for precision navigation. In *International Conference on Indoor Positioning and Indoor Navigation. 2013*. p. 814 - 815. ISBN 978-1-4673-1954-6.

Tejmlová, L. and J. Šebesta. Design of wideband Wilkinson dividers using neural network. In *Proceedings of 23rd International Conference Radioelektronika 2013*. 2013. p. 204 - 208. ISBN 978-1-4673-5517-9.

Tejmlová, L. *Ultra-Wide Band Power Divider for Mobile Frequency Bands*. ElectroScope – Online magazine (www.electroscope.zcu.cz), [online], 2012,(5 p.). ISSN\~1802-4564.

Tejmlová, L. Ultra-Wide Band Pulse Generator and Positioning System. In *Proceedings of the conference Vsacký Cáb 2012*. 2012. s. 1-6. ISBN: 978-80-214-4579- 6.

Zelinová, L. UWB generátor a system pro určení polohy. Elektrorevue – Online magazine (http://www.elektrorevue.cz), [online], 2012, č. 21, s. 1-5. ISSN: 1213- 1539.

CURRICULUM VITAE

PERSONAL DETAILS

Name	Lenka Tejmlová Ing., b. Zelinová
Date of Birth	24.9.1986
Address	Uzbecká 10, 625 00, Brno
Telephone	+420702089559
E-mail	tejmlova.lenka@gmail.com
Nationality	Czech
Field	Electrotechnics and communications

PROFESSIONAL EXPERIENCE

from August 2016	SAP ČR, Vyskočilova 1481/4, Praha 4 – Michle, 140 00 Software developer, department of AIS
April 2013 – Dec. 2014	Dept. of Radioelectronics, Brno University of Technology Technická 12, 61600 Brno, Czech Republic Research into wireless channels for intra-vehicle communication and positioning (GACR no 13-38735S) Christoph Mecklenbrauker, Ales Prokes
Job description:	Publications on current research (VaV) - UWB technologies, ultra wideband power dividers - cooperation with Škoda auto a.s. - ANNs for precision tracking systems (Doctoral Thesis) - Lessons and leadership (bachelor's and master's theses)
Feb. 2012 – Nov. 2013	ŠKODA AUTO a.s., Tř. Václava Klementa 869, 293 60 Mladá Boleslav R&D – SoL device, in-car implementation
Oct. 2009 – July 2011	ABB, Vídeňská 117,619 00 Brno, CZ Switchgear Design, working in a team Electrical Engineer
June 2007 – Oct. 2007 June 2008 – Oct. 2008 June 2009 – Oct. 2009	Giannos Rhodopoulos, Adrina Beach hotel, Skopelos, Greece Electrical maintenance services Network management and technical support
Feb. 2007 – May 2007	Honeywell, Technická 13, 616 00 Brno, CZ Research and Development PCB for the testing of pressure sensor
June 2006 – Nov. 2006	Mediaservis s.r.o, Moravské nám. 13, 602 00 Brno, CZ Active telemarketing Operator, communication with customers

EDUCATION

Sept. 2011 – Sept. 2015	Brno University of Technology, Antonínská 1, 601 90 Brno
Faculty:	Faculty of Electrical Engineering and Communication
Program title:	Electrical Engineering and Communication Department of Radio Electronics
Study level:	Doctoral, PhD.
Study form:	full-time study
Sept. 2009 – June 2011	Brno University of Technology, Antonínská 1, 601 90 Brno
Study level:	Master's, Master's degree, Ing.
Sept. 2006 – June 2009	Brno University of Technology, Antonínská 1, 601 90 Brno
Study level:	Bachelor's, Bachelor's degree, Bc.

SKILLS

LANGUAGE SKILLS:

NATIVE LANGUAGE:	Czech	
LANGUAGES:	English	- upper-intermediate/advanced
	German	- basic
	Russian	- basic
	Greek	- basic

DRIVER'S LICENSE: B (January 2005) – daily active

TECHNICAL SKILLS:
ABAP, Matlab, QT (C-code)
– own source codes, application development
Orcad PSpice, HFSS, CST studio suite
– hardware development, HW testing
CAD, Eplan, Cadelec
– switchgear designing
MS Office

CHARACTERISTICS:
reliabe, organized, self-reliant, cooperative and hardworking
punctual and accurate
willing to learn, fast learner

ABILITIES:
analytical thinking, good communication skills
ability to work in a team as well as self-sufficiently
excellent planning and organising abilities
advanced time management skills
creation of presentations, documentations, articles etc.

INTERESTS:
healthy lifestyle
fantasy books
TV documentaries (Discovery, National Geographic)
board games
culture

ABSTRACT

The dissertation is focused on inertial navigation systems and dead reckoning positioning. The issue in the problematics is that the dead reckoning systems and inertial navigation systems are inaccurate for medium-term and long-term application due to cumulative errors, assuming that the positioning is not supported by another external system. The dissertation shows possible approaches to the issue of more accurate positioning system based only on the inertial sensors. Basically we are talking about 9-DOF inertial measurement unit that allows sensing the global acceleration, rotation rate and magnetic field strength in three particular axes. The new approach brings artificial neural networks into data processing, where proper neural network is able to recognize the character of motion leading to improvement in positioning. The description of the proposed method includes an analytical procedure of its development and, if possible, the analytical performance assessment. Proposed artificial neural networks are modelled in MATLABTM and they are used for the determination of the state of the inertial unit. Due to this determination, the position of the inertial measurement unit is evaluated with higher accuracy. An application using Qt framework was developed to create an evaluation system with user interface for standard inertial measurement unit. The designed system based on artificial neural networks was verified by experiments using real sensor data.

ABSTRAKT

Disertační práce je zaměřena na oblast inerciálních navigačních systémů a systémů, které pro odhad polohy používají pouze výpočty. Důležitým faktem v dané problematice je vysoká nepřesnost určení polohy při střednědobém a dlouhodobém využívání takového systému díky kumulativní chybě za předpokladu, že inerciální systém není podpořen žádným dalším přídavným systémem. V disertační práci jsou uvedeny možné přístupy k této problematice a návrh na zvýšení přesnosti určování polohy pouze na základě inerciálních senzorů. Základem inerciální měřicí jednotky je systém s 9 stupni volnosti, který umožňuje snímat celkové zrychlení, rychlost rotace a sílu magnetického pole, jednotlivě ve třech osách. Klíčovou myšlenkou je zařazení umělých neuronových sítí do navigačního systému tak, že jsou schopny rozpoznat charakteristické rysy pohybů, a tím zvýšit přesnost určení polohy. Popis navrhovaných metod zahrnuje analytický postup jejich vývoje a tam, kde je to možné, i analytické hodnocení jejich chování. Neuronové sítě jsou navrhovány v prostředí MATLAB^{TM} a jsou používány k určení stavu inerciální jednotky. Díky implementaci neuronových sítí lze určit pozici jednotky s řádově vyšší přesností. Aby byl inerciální polohovací systém s možností využití neuronových sítí demonstrativní, byla vyvinuta aplikace v prostředí Qt. Navržený systém a neuronové sítě byly použity při vyhodnocování reálných dat měřených senzory.