Bachelor Project

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Radio Engineering

Identification, detection and parameter estimation of physical movement events measured by multi-sensor network

Filip Bobuski

Supervisor: Prof. Ing. Jan Sýkora, CSc.
May 2018

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Bobuski Filip** Personal ID number: **459917**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Radioelectronics**

Study program: **Open Electronic Systems**

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

Identification, Detection and Parameter Estimation of Physical Movement Events Measured by Multi-Sensor Network

Bachelor’s thesis title in Czech:

Identifikace, detekce a odhad parametrů mechanických pohybů měřených multi-senzorovou sítí

Guidelines:

Student will design detection and parameter estimation algorithms that allow identification, detection, and parameter estimation of physical movement events and also the global aggregated estimation function (e.g. the movement trajectory in 3D). The measurement input is provided by a network of mechanical/physical movement/acceleration/etc sensors. The work should include a measurement of the properties of sensor which will serve for identifying the sensor statistical observation model. Also, the work should identify required properties of the sensor network (e.g. the number on sensors, their relative orientation, distances, etc) to guarantee that the global aggregated estimation target (e.g. full 3D trajectory) is achievable with given target precision. Algorithms and processing should be first designed for a generic environment and sensor and statistical observation model a then in a second stage applied on some practical suitably chosen scenario.

Bibliography / sources:

[1] S. M. Kay: Fundamentals of statistical signal processing-estimation theory, Prentice-Hall 1993
[2] S. M. Kay: Fundamentals of statistical signal processing-detection theory, Prentice-Hall 1998

Name and workplace of bachelor’s thesis supervisor:

prof. Ing. Jan Sýkora, CSc., Department of Radioelectronics, FEE

Name and workplace of second bachelor’s thesis supervisor or consultant:

Date of bachelor’s thesis assignment: **31.01.2018** Deadline for bachelor thesis submission: **25.05.2018**

Assignment valid until: **30.09.2019**

prof. Ing. Jan Sýkora, CSc. Supervisor’s signature	Head of department’s signature	prof. Ing. Pavel Ripka, CSc. Dean’s signature

III. Assignment receipt

The student acknowledges that the bachelor’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor’s thesis, the author must state the names of consultants and include a list of references.

.

Date of assignment receipt	Student’s signature

Acknowledgements

I would like to thank my professor Jan Sýkora, for exceptional mentoring and guidance he provided me throughout the making of this thesis. I would also like to thank my family and friends who helped me in my studies.

Declaration

I declare, that I have produced the submitted work independently and that I have provided all the information sources used in accordance with the Methodological Guideline on Ethical Principles in the Preparation of Graduate Final Theses.

Prague, May 23, 2018

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, 23. května 2018

Abstract

This thesis has as a goal to introduce basic concepts of signal processing and motion tracking together with construction and programming of a real working measuring device. Specifically maximum likelihood estimation, data acquisition and processing of signals from accelerometer and gyroscope. It describes common types of sensors, explains their inner workings and in the practical part uses gained information to integrate them into a process of trajectory estimation successfully. It also lays a basis for possible further research and sets a direction for possible improvements of implemented estimators.

Keywords: estimation, ML, motion, accelerometer, gyroscope

Supervisor: Prof. Ing. Jan Sýkora, CSc.

Abstrakt

Cílem práce je seznámení se základními pojmy zpracování signálu a sledování pohybu společně s konstrukcí a naprogramováním reálného měřícího zařízení. Konkrétněji uvádí aplikace maximálního odhadu pravděpodobnosti a popisuje zpracování signálů ze senzorů mechanického pohybu. Představuje běžné typy snímačů, vysvětluje jejich vnitřní fungování a v praktické části využívá získané informace k jejich úspěšnému začlenění do procesu odhadování trajektorie. Rovněž vytváří základ pro případný další výzkum a stanoví směr pro případné zlepšení implementovaných odhadů.

Klíčová slova: estimace, ML, pohyb, accelerometr, gyroskop

Překlad názvu: Identifikace, detekce a estimace parametrů fyzického pohybu měřeného sítí senzorů.

Contents

Figures

Tables

Chapter 1

Introduction

As our technological capabilities progress and electronic devices getting smaller and cheaper, personal motion and fitness tracking devices are progressively more common. The primary motivation of this thesis is to look at what obstacles we will face when attempting to create one of such devices from the standpoint of signal processing. We will be focussing primarily on trajectory estimation. We will use data from three-axis accelerometer and three-axis gyroscope fixed on an object moving in a testing trajectory. In most cases, the raw output of motion tracking sensors is not of sufficient quality for immediate use and have to be revised in various ways according to its applications. It is common that sensor we are given has imperfections caused by manufacturing defects of the measuring apparatuses and low quality of electronic components controlling them. The goal of this thesis is to investigate how these devices work, how to compensate for their inadequacies and subsequently we will try to apply basic maximum likelihood estimators to mitigate the inaccuracies in our signals. However, first, we will get acquainted with internal construction and workings of used sensors and basic concepts of signal estimation.

Chapter 2

Theoretical part

2.1 Accelerometers

Accelerometers are mechanical or electromechanical devices, which measures acceleration experienced by an object due to static or dynamic forces acting on it. Static forces include gravity, while dynamic forces include vibrations and movement. There are several types of accelerometers available based on different physical phenomena.

2.1.1 Strain gauge accelerometer

This type of accelerometer was one of the first commercially used. The discovery of the strain gauge is independently credited to both A. Ruge, Massachusetts Institute of Technology (MIT), April 3, 1938, and E. Simmons, Caltech, September 1936 [1]. This type of sensor is composed of a cantilever which is mounted onto the casing on one end, a reference mass attached to the other free end and bounded strain gauges mounted on the cantilever beam. Usually, the housing is filled with a viscous fluid to provide damping, for increasing their usable frequency response and, at the same time, decreasing their fragility. When the sensor is subjected to acceleration, the inertia of the mass will cause the cantilever to flex, which subsequently causes deformation in the strain gauges and proportional change of their resistance, which can be measured as a change of voltage on the output. Figure 2.1 illustrates the device. Frequency constraints of these sensors limit their application in measuring high-frequency vibrations and short duration impulses. Also, use of dumping fluid makes them temperature sensitive.

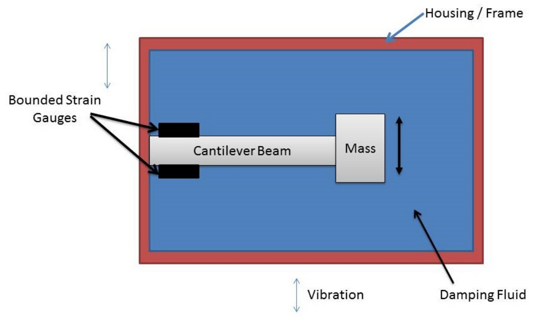


Figure 2.1: Strain gauge accelerometer design [8]

2.1.2 Piezoelectric accelerometer

Demand for a sensor which can handle high g overload, have flat frequency response to high values and high-temperature range, satisfies piezoelectric accelerometer. It has no moving parts. They came into commercial use in the late 1940s and early 1950s. In 1880 Pierre Curie & Gaybriel Loppmann discovered that when a force is applied onto a crystal of tourmaline, quartz or topaz, the electric charge appears on them [10]. The piezoelectric material consists of atoms with positive and negative charges, called ions. Molecules containing these ions have non-zero electric dipole moments. Areas, where the dipoles have same orientations, are called domains. These domains are non-centrosymmetric, i.e., lacking a center of symmetry[10]. In normal circumstances, these domains, have random direction, but when force is applied, the deformation of the crystal structure causes the non-centrosymmetric domains to move, that leads to net movement of the positive and negative ions with respect to each other and results in a polarisation of the crystal. Because of this process, uncompensated opposite bounded charge at the ends of the crystal, proportional to the force applied, will appear. This charge is then converted into a usable voltage signal. Piezoelectric sensors are better suited for dynamic measurements such as vibrations, but with additional signal conditioners, they can measure static accelerations.

2.1.3 MEMS accelerometers

Micro-ElectroMechanical Systems (MEMS) were first proposed in the 1960s, but not commercialized until the 1980s. [11] The idea was to make tiny mechanical systems, which could be connected to other electronics directly on the same chip. These structures are created using the same photolithography techniques as an integrated circuit. Because of that MEMS sensors are smaller, lighter and cheaper. This type of sensor works similarly to those mentioned before. It includes a moving element with a specified mass, which, due to force applied, will shift relative to the outer structure. This displacement is proportional to external acceleration, which can be measured by several methods. They are based on piezoelectric, capacitive, piezoresistive, optical, and electromagnetic principles. Capacitance-based accelerometers are, however, the most successful and popular mechanism in the MEMS domain due to the relatively simple design and passive operation. They could be used as both sensors and actuators. The method of sensing used, change of capacitance, is insensitive to temperature changes. Also, it is independent of the base material. A differential capacitance readout circuit is usually used. The capacitance is counted with respect to both of the sides of the moving structure, shown on the figure 2.2. The motion increases the capacitance of capacitor on one side and simultaneously decreases the capacitance of the other. As a result, it is possible to maintain the output signal linearity and to ensure noise compensation, which allows tiny displacements (within fractions of a nanometre) to be detected.

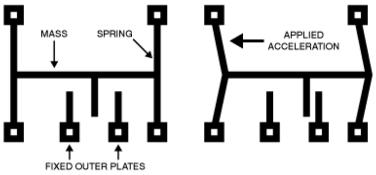


Figure 2.2: Simplified workings of a capacitance accelerometer. [9]

2.2 Gyroscopes

Gyroscopes are devices, which can sense and measure angular velocity. Many types of gyroscopes exist, depending on physical principle on which they operate. We have mainly mechanical gyroscopes, optical gyroscopes and Micro-electromechanical system (MEMS) gyroscopes.

2.2.1 Mechanical gyroscopes

The primary effect on which a gyroscope relies is that an isolated spinning mass tends to keep its angular position stable with respect to an inertial reference frame. Rotational movement can then be sensed from a relative variation of the angle between the rotation axis of the mass and a fixed direction on the frame of the gyroscope.

2.2.2 Mechanical rate gyroscopes

This type of gyroscopes measures the velocity of angular motion. When is gyroscope rotating around z-axis and angular velocity in the direction of x-axis is imposed on it, a proportional torque in y-axis appears. By restricting the movement of the spinning structure with a spring, with known stiffness, we can measure the imposed angular speed, by measurement of the angle assumed by the gyroscope in an xz plane.

2.2.3 Optical gyroscopes

The underlying operating principle of almost all optical gyroscopes is the Sagnac effect. The Sagnac principle, states that two counter-propagating optical beams propagating in a ring structure change their relative phase, if the ring is rotating, thus it is possible to relate the phase change to the angular speed of the ring. This phenomenon named after French physicist Georges Sagnac, manifests itself when two light beams are made to follow the same path, but in opposite directions. This can be achieved by series of mirrors or fiber optics. If the optical closed loop is rotating, the interference fringes are displaced compared to those measured in a loop at rest. The amount of displacement is proportional to the angular velocity of the loop. The effect is caused by different times it takes each beam to make the distance needed. The times can be expressed as

$$t_{\pm} = \frac{2\pi r \pm r\Omega t_{\pm}}{v}.$$

Where t_+ time taken by beam propagating in the direction of rotation, t_- time taken by beam propagating against the direction of rotation,Ω is angle velocity of the rotation and v is speed of light. Hence the times are

$$t_{\pm} = \frac{2\pi r}{v \pm r\Omega}.$$

It follows, that the time difference Δt will be

$$\begin{aligned}\Delta t &= t_+ - t_- \\ &= \frac{2\pi r}{v - r\Omega} - \frac{2\pi r}{v + r\Omega} \\ &= \frac{4\pi r^2\Omega}{v^2 - r^2\Omega^2}.\end{aligned}$$

Because usually $v \gg r^2\Omega^2$ then

$$\Delta t = \frac{4A\Omega}{v^2}.$$

Where A is area of the closed loop. Phase difference $\Delta\phi$ can be obtained as

$$\begin{aligned}\Delta\phi &= \Omega\Delta t \\ &= \frac{2\pi\Delta t}{T} \\ &= \frac{2\pi\Delta t}{\frac{\lambda}{v}} \\ &= \frac{2\pi v}{\lambda}\frac{4A\Omega}{v^2} \\ &= \frac{8\pi A\Omega}{v\lambda}.\end{aligned}$$

Where λ is wavelength of light.

2.2.4 MEMS gyroscopes

MEMS gyroscopes generally use a vibrational movement of a mechanical element for detection of angular velocity. They are doing so, by sensing Coriolis force given by following formula.

$$F_C = -2m\boldsymbol{\omega} \times \boldsymbol{v} \tag{2.1}$$

It can be demonstrated on a situation depicted in figure 2.3. On the first picture where $\tau = t_0$, we have a red ball at the center of a disk with a diameter of r, rotating with an angular velocity of ω and a target at the edge. At time $\tau = t_0$ the ball will start to move with a speed of v in the direction pointed to the target. On the picture, b) $\tau = t$ we see that the ball reached the edge of the disk at a point where the target used to be. From the point of view of spectator situated outside the rotating disk, in the time t the ball traveled in the straight line, and a target moved the distance Δs which corresponds with a change in angle $\Delta\varphi$

$$\Delta s = r\Delta\varphi = r\omega\tau = v\omega\tau^2. \tag{2.2}$$

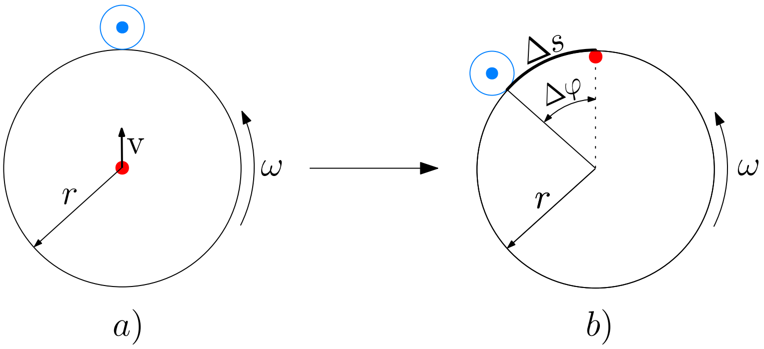


Figure 2.3: Illustration of Coriolis force

From the view of a spectator at the center of rotating disk everything else, apart from the disk and target, is rotating in the opposite direction with angular velocity ω. The movement of the ball is not in a straight line anymore. Instead, it is moving along a curve, and the ball leaves the disk with a deflection from the target Δs. Newton's first law of motion states that every object in a state of uniform motion tends to remain in that state of motion unless an external force is applied to it. So that means that some force, perpendicular to the balls motion had to influence its trajectory. Let us mark it as F_C, then following statement holds true.

$$\Delta s = \frac{1}{2}\frac{F_C}{m}\tau^2 \tag{2.3}$$

Because for both spectators the ball has to end up in the same point in space, both distances from equation 2.2 and 2.3 have to be equal.

$$\begin{aligned} v\omega\tau^2 &= \frac{1}{2}\frac{F_C}{m}\tau^2 && (2.4)\\ F_C &= 2m\omega v && (2.5)\end{aligned}$$

We derived formulation for a magnitude of Coriolis force when the angle between vectors v and ω is $\frac{\pi}{2}$. We can establish ω as a vector quantity and arrive at the expression 2.1. In the case of the gyroscopic sensor, the direction of Coriolis force will be orthogonal to both angular velocity $\boldsymbol{\omega}$ and movement of the vibrating structure $\boldsymbol{v}$. For known vibrational velocity of the structure, this force will cause a displacement proportional to the angular velocity of the gyroscope. Displacement magnitude can be sensed by several methods, including capacitance change between moving and a stationary part of the sensing apparatus.

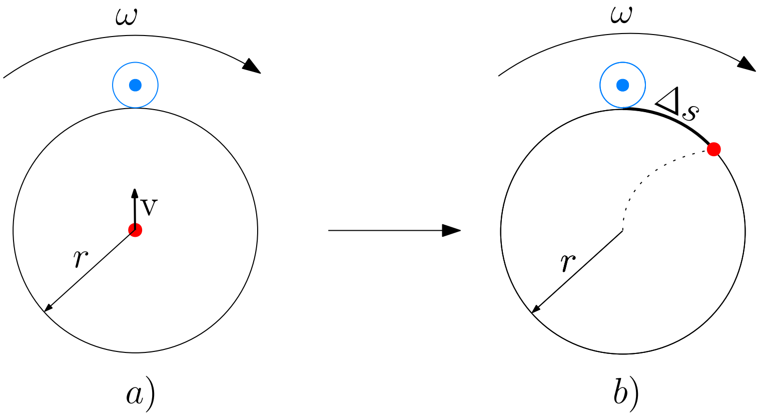


Figure 2.4: Illustration of Coriolis force

2.3 Trajectory

The subject of this work is to estimate a trajectory of physical movement using accelerometers and gyroscopes. For that, we will need to be able to convert 3-axis acceleration and rotation data to x, y, z coordinates of individual points in the space of trajectory travelled. Acceleration is a change of velocity across a given time.

$$\frac{dv}{dt'} = a \tag{2.6}$$

$$dv = adt^{'} \tag{2.7}$$

$$\int dv = \int_{t_0}^{t} adt^{'} \tag{2.8}$$

$$v \overset{1}{=} a(t - t_0) + v_0 \tag{2.9}$$

The equality in (2.9) holds true given $a = const$ for $\Delta t = t - t_0$. Than with the same process we obtain s(t).

$$\frac{ds}{dt'} = a(t - t_0) + v_0 \tag{2.10}$$

$$\int ds = \int_{t_0}^{t} (a(t - t_0) + v_0) dt^{'} \tag{2.11}$$

$$s(t) = \frac{1}{2} a(t - t_0)^2 + v_0(t - t_0) + s_0 \tag{2.12}$$

For expression 2.12 same equality condition holds true as in 2.9. This will be our equation for trajectory calculations. The rotational motion can be expressed by rotational matrices. For clarity of explanation, let us take just two-dimensional case. Lets suppose that we have complex number $z = a + ib$. It can be represented in the complex plane as a point with its real and imaginary part constitutes its coordinates on real and imaginary axes. If we want to rotate this number by some angle α. We could multiply z by complex exponential $e^{j\alpha}$, which can be rewritten as

$$e^{j\alpha} = \cos(\alpha) + i \sin(\alpha),$$

$$e^{j\alpha}(a + ib) = \cos(\alpha) + i \sin(\alpha)(a + ib) \tag{2.13}$$

$$= [a\cos(\alpha) - b\sin(\alpha)] + i\,[a\sin(\alpha) + b\cos(\alpha)]\,. \tag{2.14}$$

Now, if we profess the real and imaginary part as x a y coordinates. We can perceive this as a "yaw" - a rotation around z-axis. Rewritten in a matrix form

$$\begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}.$$

Because "yaw" doesn't affect z-axis, for 3D case the expression can be modified

$$\begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}.$$

Similar process can be applied for the remaining x and y axes. Resulting in "roll" and "pitch" matrices.

$$\mathrm{R_x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix}, \quad \mathrm{R_y} = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix},$$

$$\mathrm{R_z} = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation of a point $A = [A_x, A_y, A_z]$ in 3D space by angles α, β, γ would then look as follows

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \mathrm{R_x R_y R_z} \begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix}.$$

2.4 Estimation

2.4.1 Introduction

On paper or in mathematical equations we can control what data or signal we will work with and creating ideal circuits with which we will manipulate those entities. But in a real world, we, unfortunately, do not have such a luxury. Because atoms, from which everything in our world is composed, including electrical devices, are always in motion, if not cooled to $T = 0K$, every signal we send or receive will be corrupted with noise caused by this motion. Another big source of noise is our sun, which ceaselessly bombards our planet Earth with charged particles interfering with our signal and devices working with them. That means that we can no longer count on receiving the same signal we have sent, we are unable to know and predict all the influences and contaminants our signal encounters, and therefore, we can no longer predict how exactly it will look like. We can only try to collect as much information as we can about the signal and medium through which it propagated and based on its quantity and quality; we can try to guess parameters of the received signal.

This is where estimation theory comes into play. One example of estimation problem can be radar.

2.4.2 Radar

When we need to know the presence and position of an object, we send an electromagnetic pulse of known length and frequency. The potential object then reflects this pulse and an echo is detected by our antenna τ_0 seconds later. From the equation $\tau_0 = \frac{2R}{c}$ we can then determine probable position of the object. Because only a small part of our signal will hit the object and even smaller will make it back to our antenna, the echoed signal can be significantly decreased in amplitude, the estimation problem can become very difficult. Very same principles are used with sonar. There are two types of sonar technology. Passive sonar is just a microphone listening for the noises in the environment. Active sonar is very similar to radar, by using its own sound signal often called a "ping." Needed information is then estimated from the parameters of the echo.

In all these applications we are faced with a need to extract a value of parameters based on our signals. Because we will work with digital computers we will model our data set as vector $x\left[n\right], n = 0, 1, ..., N-1$ dependent on an unknown parameter θ.

In the process of building and estimator, one of the first steps is to model our data mathematically. Because of its random nature, we will do so with probability density functions (PDFs). We must choose a PDF which is consistent with the problem and mathematically tractable.

2.4.3 Example: DC level

Let us suppose that we measure a constant DC voltage A, with a digital voltmeter. Its output will be vector $x\left[n\right], n = 0, 1, ..., N-1$ of values, contaminated with noise imposed on them by the voltmeter. This noise is a random variable we must define by some PDF. We will model it as a white Gaussian noise (WGN), this assumption is justified by the need of constructing a tractable model so that that closed-form estimators can be found. Because of its tractability, WGN is usually chosen whenever it is possible unless there is strong evidence to the contrary. So our data looks as follows

$$x\left[n\right] = A + w[n]. \tag{2.15}$$

where samples $w[n]$ have a normal distribution $\mathscr{N}(0, \sigma^2)$ with zero mean and variance σ^2

$$p(x[n]|A) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}\left(x[n] - A\right)^2\right). \tag{2.16}$$

So our data is defined as deterministic value with stochastic variable modelled by its PDF. But lets say that the true voltage is variable in time to a minor extent. We do not know what is the exact development of such signal. We can incorporate a prior knowledge and model it as a random variable with its own PDF. Such an approach is termed *Bayesian* estimation. Then the estimate of θ is taken as a realization of a random variable. This type of model would be described with joined PDF.

$$p(x, \theta) = p(x|\theta)p(\theta) \tag{2.17}$$

Where $p(x|\theta)$ is conditional PDF, which sets the probability of x conditioned by prior knowledge of θ and $p(\theta)$ is the distribution of θ.

2.5 Linear Models

This class of data models greatly simplifies finding an optimal estimators. In matrix notation the model is written as

$$\boldsymbol{x} = \mathbf{H}\boldsymbol{\theta} + \boldsymbol{w},$$

where $\mathbf{H}$ is a known matrix called observation matrix and the $\boldsymbol{w}$ vector has the statistical characterization $\boldsymbol{w} \sim \mathscr{N}(\mathbf{0}, \sigma^2\boldsymbol{I})$.

2.6 Maximum Likelihood estimator

Maximum likelihood (ML) estimator has a goal to find the input value in stochastic input-output model, that maximizes the probability of observed output. In another words for a given fixed output, it varies potential inputs

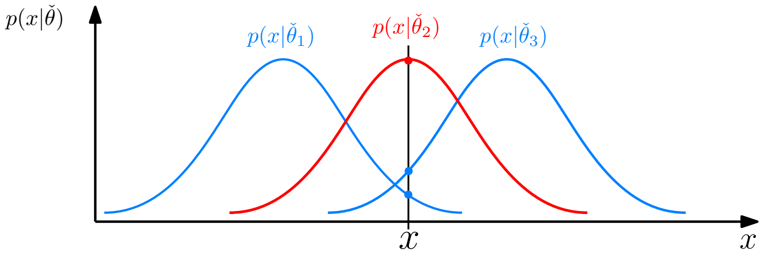


Figure 2.5: ML estimator principle.

and chooses the one that maximizes the probability of receiving said output. As a metric it has a stochastic input-output model.

$$p(x|\check{\theta})$$

Criterion of ML estimator 2.18 is directly related to the observation model. This type of estimator, is overwhelmingly the most popular approach to obtaining practical estimators.

$$\hat{\theta} = \arg\max_{\check{\theta}} p(x|\check{\theta}) \tag{2.18}$$

2.6.1 Maximum likelihood estimator in Linear AWGN observation

For linear AWGN observations where $\boldsymbol{x} = \mathbf{H}\boldsymbol{\theta} + \boldsymbol{w}$ where $\mathbf{H} \in \mathbb{C}^{N\times L}, L < N$ and $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{C_w})$. ML estimator has a relatively straightforward implementation. Lets begin with PDF of Gaussian noise, given by 2.19.

$$p_{\mathbf{w}}(\boldsymbol{w}) = \frac{1}{\pi \det(\mathbf{C_w})} \exp(-\boldsymbol{w}^H \mathbf{C_w^{-1}} \boldsymbol{w}). \tag{2.19}$$

Where $\mathbf{C_w} = E\left[\boldsymbol{w}\boldsymbol{w}^H\right]$ is covariance matrix. As mentioned before, ML estimator is maximizing likelihood function $p(x|\check{\theta})$. This will be achieved by common method of finding the extreme of a function by taking a derivative a making it equal to zero. The same result will be obtained by maximizing a log-likelihood function $\ln p(x|\check{\theta})$. Final result wont change because $\ln(x)$ is strictly increasing function.

$$\begin{aligned}\hat{\boldsymbol{\theta}} &= \arg\max_{\check{\boldsymbol{\theta}}} p(\boldsymbol{x}|\check{\boldsymbol{\theta}}) \\ &= \arg\max_{\check{\boldsymbol{\theta}}} \ln p(\boldsymbol{x}|\check{\boldsymbol{\theta}}) \\ &= \arg\max_{\check{\boldsymbol{\theta}}} \ln p(\boldsymbol{x} - \mathbf{H}\boldsymbol{\theta}) \end{aligned} \tag{2.20}$$

Log-likelihood function will have following form

$$\ln p(\boldsymbol{x}|\hat{\boldsymbol{\theta}}) = -\ln \pi^N \det(\mathbf{C_w}) - \left[(\boldsymbol{x} - \mathbf{H}\boldsymbol{\theta})^H \, \mathbf{C_w^{-1}} \, (\boldsymbol{x} - \mathbf{H}\boldsymbol{\theta})\right]. \tag{2.21}$$

Next we will make derivative, but we have to be careful because we do not know if our function satisfies Cauchy-Riemann equations and use generalized derivative.

$$\frac{\tilde{\partial}}{\tilde{\partial}\boldsymbol{\theta}} p(\boldsymbol{x}|\boldsymbol{\theta}) = \frac{\tilde{\partial}}{\tilde{\partial}\boldsymbol{\theta}} \left[(\boldsymbol{x} - \mathbf{H}\boldsymbol{\theta})^H \mathbf{C_w^{-1}} (\boldsymbol{x} - \mathbf{H}\boldsymbol{\theta}) \right] \tag{2.22}$$

After the derivation and simplification of 2.22 we get

$$\frac{\tilde{\partial}}{\tilde{\partial}\boldsymbol{\theta}} p(\boldsymbol{x}|\boldsymbol{\theta}) = -\left(- \left(\boldsymbol{x}^{\boldsymbol{H}} \mathbf{C_w^{-1}} \mathbf{H} \right)^T + \left(\mathbf{H}^{\boldsymbol{H}} \mathbf{C_w^{-1}} \mathbf{H} \right)^T \boldsymbol{\theta}^* \right).$$

Now for locating the maximum we set $\frac{\tilde{\partial}}{\tilde{\partial}\boldsymbol{\theta}} p(\boldsymbol{x}|\boldsymbol{\theta}) \Big|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}} = 0$,

$$\hat{\boldsymbol{\theta}} = \left(\mathbf{H}^{\boldsymbol{H}} \mathbf{C_w^{-1}} \mathbf{H} \right)^{-1} \mathbf{H}^H \mathbf{C_w^{-1}} \boldsymbol{x}. \tag{2.23}$$

As a result we get nice compact implementation of ML estimator 2.23.

Chapter 3
Practical part

3.1 MPU-9250

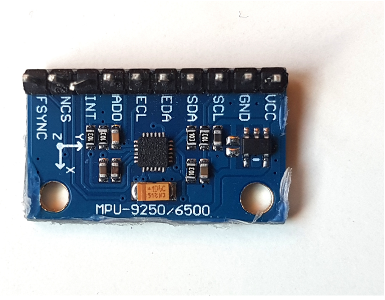


Figure 3.1: MPU-9250 on a breakout board

Throughout our experiments, we will be working with an MPU-9250 sensor from InvenSense shown above on image 3.1. It is a multi-chip module (MCM) consisting of two dies integrated into a single QFN package. One die houses the 3-axis gyroscope and the 3-axis accelerometer. The other die houses the AK8963 3-axis magnetometer from Asahi Kasei Microdevices Corporation. Hence, the MPU-9250 is a 9-axis MotionTracking device that combines a 3-axis gyroscope, 3-axis accelerometer, 3-axis magnetometer and a Digital Motion Processor.[3] It has 16-bit analog-to-digital converters for digitizing the outputs of individual sensors. For precise measuring of both fast and slow movements, we can choose from several user-programmable ranges.

3.1.1 Accelerometer sensitivity ranges

Range	**LSB/g**
$\pm 2g$	16384
$\pm 4g$	8192
$\pm 8g$	4096
$\pm 16g$	2048

3.1.2 Gyroscope sensitivity range

Range	LSB/(°/s)
±250	131
±500	65.5
±1000	32.8
±2000	16.4

Where **LSB** stands for *Least significant bit.* Note that with the increasing range the precision of the sensor decreases. For our experiments, we will mostly use the ±$2g$ range, if not stated otherwise.

3.1.3 Measuring equipment

Measuring device we constructed for this purpose is composed of three main parts. The primary part of the apparatus is Arduino Uno, a microcontroller board based on the ATmega328P connected to an MPU-9250 sensor on a breakout board. Wiring between the boards is displayed in figure 3.2.

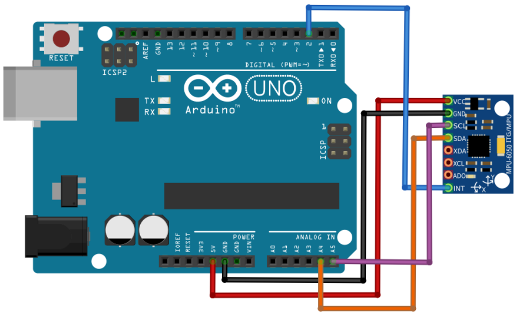


Figure 3.2: Wiring between Arduino Uno and MPU-9250 [6]

It communicates with a previously mentioned MPU-9250 sensor via I2C protocol on its pins A4 and A5 and interrupt pin connected to digital pin 2. The third part of our device is a micro SD card module. The connection between Arduino and SD module is shown in figure 3.3. Their communication is carried out by SPI protocol.

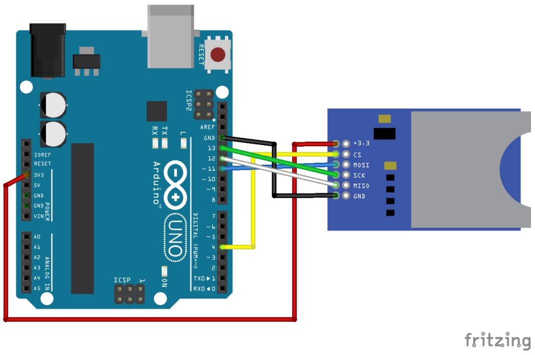


Figure 3.3: Wiring between Arduino Uno and SD card module. [7]

Sd card module will be used for data storage because average time between read events with SD card module is about $5ms$ and we encountered troubles with managing these transfer speeds via USB cable. Real photo of the created device is shown in figure 3.4.

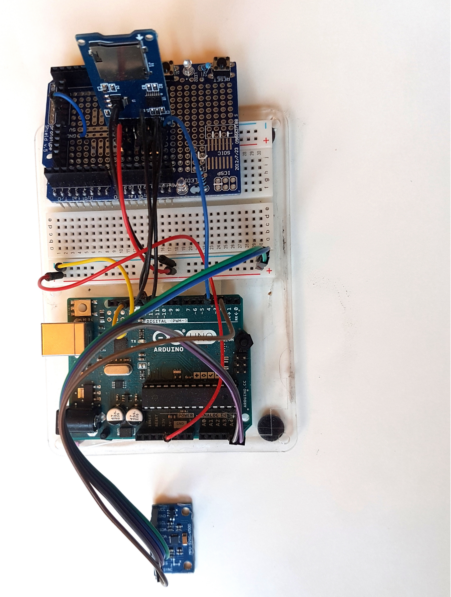


Figure 3.4: Arduino Uno connected to the sensor at the bottom, and SD card module at the top.

Program in Arduino checks roughly every $5ms$ all three accelerometer and gyroscope axes and stores their data in a text file on the SD card together with information about time duration between each read event in milliseconds. Data is stored in 7 columns. First three are accelerometer data for x, y and z-axis second three for gyroscope data and seventh column is for time information. This device will be used throughout this study for all experiments including noise inspection and model construction, mentioned in following sections, in addition to trajectory data measurement and storage.

3.2 Sensor model

Our goal is to try approximate a trajectory of general movement. But before we even begin to think about an estimator, model of our working sensor has to be made. As mentioned in the previous section 2.5 about linear models, it would be advantageous, if our observation model was linear. For that to be true, we need to fit our model into a form

$$\boldsymbol{x} = \mathbf{H}\boldsymbol{\theta} + \boldsymbol{w}.$$

We begin with a reflection on how model of our sensor can be most accurately modelled. Generally, any sensor can display signs of non-linearity in its full range of possible values due to manufacturing imperfections. By the same cause, axes of the individual sensing apparatuses could be misaligned (not have exactly 90° between each other), so the sensors could sense a fraction of a value measured on one axis on remaining axes. We can model this effect by an arbitrary function $g()$. An output of the sensor will most likely be combined with noise. We will model this noise as additive white Gaussian noise and hope reality will not deviate much significantly from this assumption. So from this first contemplation, a preliminary model can be made.

$$x = g(\theta(t)) + w \tag{3.1}$$

3.2.1 Noise inspection

As a first step in verifying the validity of our presumed model, we will try to inspect existence and nature of the noise. We will do so with static experiment. For data acquisition, we use Arduino Uno, open-source microcontroller board developed by Arduino.cc and all the data post-processing is done in MATLAB software.

Thinking behind this experiment is that, if the sensor remains stationary throughout the test, its output should be possible to interpret as constant value A covered with WGN.

$$x[n] = A + w[n]$$

Based on this assumption, the reasonable estimator of the constant A is

$$\hat{A} = \frac{1}{N}\sum_{n=0}^{N} x[n]. \tag{3.2}$$

Because for the mean of such estimator following statements holds true.

$$\begin{aligned} E\left(\hat{A}\right) &= E\left(\frac{1}{N}\sum_{n=0}^{N} x[n]\right) \\ &= E\left(\frac{1}{N}\sum_{n=0}^{N} A + w[n]\right) \end{aligned}$$

Then due to linearity properties of the expectation operator

$$= \frac{1}{N}\sum_{n=0}^{N} E\left(A\right) + E\left(w[n]\right).$$

When assumed Gaussian noise with $\mathcal{N}(0,\sigma^2)$ then $E\left(w[n]\right) = 0$ and

$$E\left(\hat{A}\right) = A.$$

Meaning that on average this estimator will yield the true value. Subsequently, this value can be subtracted from the data leaving remaining samples of WGN with $\mathcal{N}(0,\sigma^2)$. A sensor will be placed on a stable surface, it is also necessary,

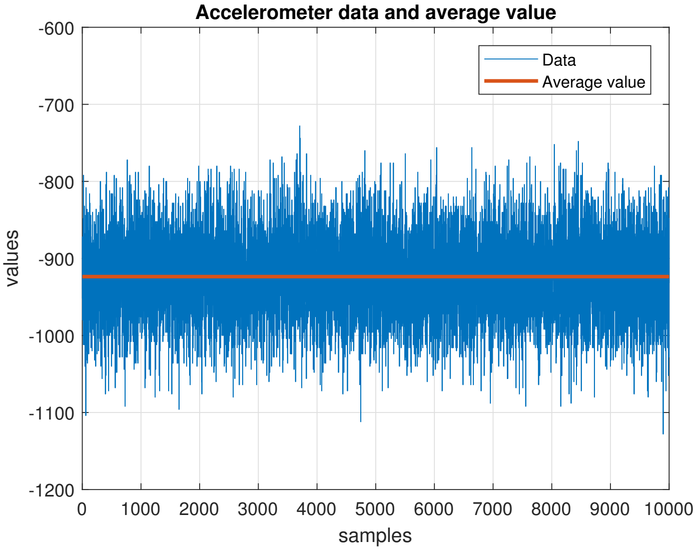


Figure 3.5: Noise with its calculated average value

that the sensor will not move throughout the experiment. Then sufficient amount of data is collected. In our case 100000 samples. Figure 3.5 shows one chosen signal and its average value. Consequently we can make a histogram of remaining samples. In figure 3.6 there are histograms for each axis of accelerometer and gyroscope.

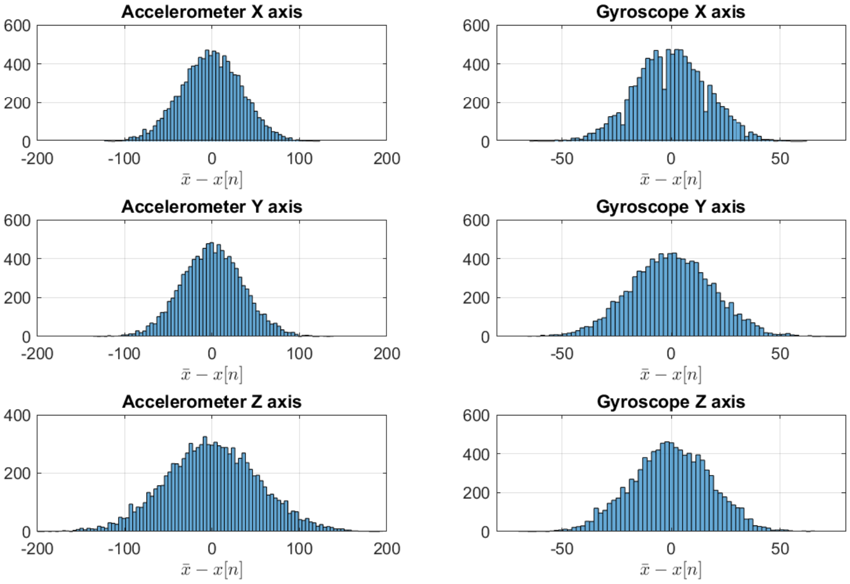


Figure 3.6: Histograms from collected samples for each axis of accelerometer and gyroscope.

From visual inspection of 3.6 we can indeed assume Gaussian noise. We can assure ourselves of its *whiteness* by calculating autocorrelation function of the data, which should be dirac at the zero shift. It seems reasonable to assume, from figure 3.7, our measured noise as AWGN.

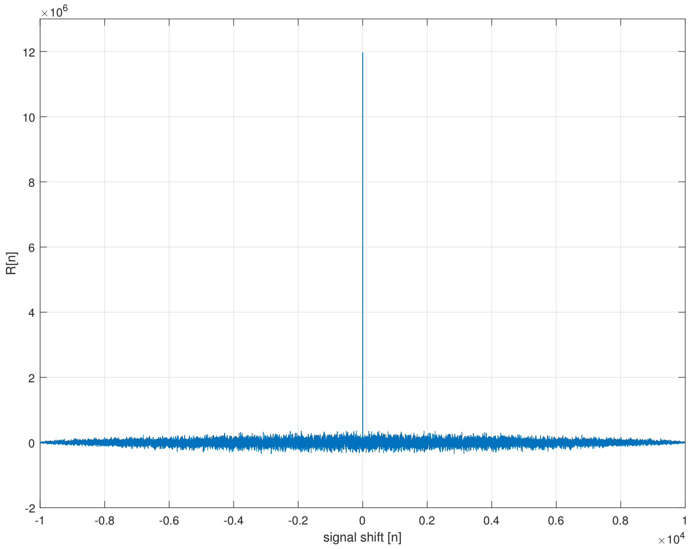


Figure 3.7: Autocorrelation function of collected noise.

3.2.2 Observation Matrix

The last remaining part of our observation model is the function $g()$. Note that in this section we presuppose only translational movement and work just with accelerometers. When looking at the datasheet for our sensor (MPU-9250), we see that non-linearity of readings should be around $\pm 0.1\%$. It is then reasonable to assume the linear behaviour of the sensor without loss of significant accuracy. Final part of our model is taking care of parameters of the sensor like axis scaling, cross-axis talk, and permanent offset. For determination of axis scaling a simple experiment comes to mind. Just setting sensor on a flat surface and measuring if $a = 1g$ is measured. But this is by no means an accurate measurement because we can not know if the surface on which the sensor is placed is truly level and therefore if the acceleration we measure is not

$$a = g\cos(\alpha), \text{ where } \alpha \neq 0.$$

Even if we somehow ensure a true level of the surface, we still can not ascertain if the sensor itself is mounted on its printed circuit board properly or how sensing apparatuses are situated in the casing themselves. The inability of precise determination of these parameters render any such experiment inaccurate and calls for a different approach. After a consultation with engineers from the department of measurements, all attempts for creating measuring apparatuses such as centrifuges or free fall machines were abandoned for the same problems of unknown inaccuracies of such constructions. In any given time interval we will be given data from all three axes of the accelerometer. Those can be modeled as

$$\mathbf{A} = \mathbf{S}(\mathbf{V} - \mathbf{O}). \tag{3.3}$$

Where **A** is the true value and **V** is the outputted value. Parameters of our interest can be modelled as a matrices **S** and **O**.

$$\mathbf{S} = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix}, \mathbf{O} = \begin{bmatrix} O_x \\ O_y \\ O_z \end{bmatrix} \tag{3.4}$$

Where diagonal elements of **S** are scale factors along the three axes and the other are representing cross-axis talks. The matrix **O** describes constant offsets of each axis. if we take S matrix as symmetric ($S_{xy} = S_{yx}, S_{xz} = S_{zx}, ...$) we then have nine unknown variables. If we construct nine or more equations for those variables, we could try to derive them numerically.

3.2.3 Parameter optimization

In this process, we will use findings and methods found in [4], with slight alterations. We will use the fact that in ideal state the modulus of the acceleration, from all three axes, should be equal to gravity acceleration.

$$\sqrt{a_x^2 + a_y^2 + a_z^2} = g$$

For computing the model parameters we, set our sensor in $N \geq 9$ different orientations (we chose $N = 20$). For each orientation we define an error e_k which is equal to the square difference between the modulus of measured accelerations and g

$$e_k = a_x + a_y + a_z - g^2. \tag{3.5}$$

Equation 3.5 can be rewritten as

$$e_k = \sum_{i=x,y,z} \left(\sum_{j=x,y,z} \left[S_{i,j} \left(V_{j,k} - O_j \right) \right]^2 \right) - g^2, \tag{3.6}$$

$V_{j,k}$ are accelerometers output for the k-th orientation. All we need to do next is to combine e_k^2 into a cumulative error function.

$$E(S_{xx}, S_{yy}, S_{zz}, S_{xy}, S_{xz}, S_{yz}, O_x, O_y, O_z) = \frac{\sum_{n=1}^{N} e_k^2}{N} \tag{3.7}$$

Which gives us non-linear function of desired parameters. Be means of minimizing function 3.7 with respect to **S** and **O** we should obtain parameters which best fit our model in the least squares sense. As initial values we used datasheet of our sensor displayed below in 3.1.

S_{xx_o}	S_{yy_o}	S_{zz_o}	S_{xy_o}	S_{xz_o}	S_{yz_o}	O_{x_o}	O_{y_o}	O_{z_o}
1	1	1	0.02	0.02	0.02	0	0	0

Table 3.1: Initial values for optimization.

As a minimazing function in MATLAB we use `fminsearch`. Results can be seen in *Optimisation.m* file present in a list of attached MATLAB files. Graph of iterative progress is illustrated in figure 3.8. Calculated values are in 3.2

S_{xx}	S_{yy}	S_{zz}	S_{xy}	S_{xz}	S_{yz}	O_x	O_y	O_z
1.0928	1.0400	1.0893	0.0158	0.0108	0.0185	-158.9248	3.2768	26.2144

Table 3.2: Resulting values of optimization.

where O_x, O_y, O_z are converted to sensor data format (16384 instead of $1g$)

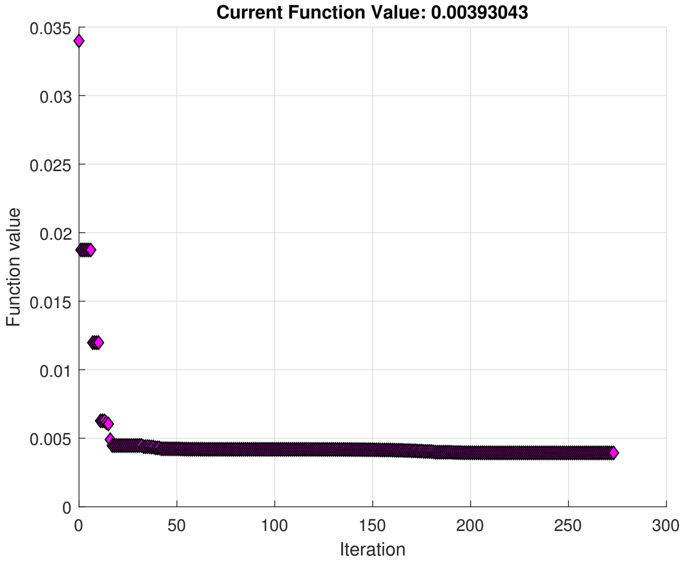


Figure 3.8: Graph of iterational improvements of `fminsearch` function.

So our observation model is as follows

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} 1.0928 & 0.0158 & 0.0185 \\ 0.0158 & 1.0400 & 0.0185 \\ 0.0185 & 0.0185 & 1.0893 \end{bmatrix}^{-1} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} + \begin{bmatrix} -158.9248 \\ 3.2768 \\ -26.2144 \end{bmatrix} + \begin{bmatrix} w_{1,n} \\ w_{2,n} \\ w_{3,n} \end{bmatrix}. \quad (3.8)$$

But this model works only for translational movement because it utilizes only accelerometers and no gyroscopes so no rotational motion can be detected. If we want to generalize our model for general motion in other words combination of translation and rotation, we also need to create a model for our gyroscopes.

3.2.4 Gyroscope Model

Unfortunately, there is no definite natural usable constant or simple experiment, as with accelerometers and gravitational constant, that would allow us to determine parameters of a gyroscope simply. Nevertheless, from the previous test described in section 3.2.1, we already know parameters of the noise present in gyroscope readings. We can approximate constant offsets of used gyroscopes with equation 3.2, where the constant A will represent our offset. Because MEMS accelerometers (described in section 2.2.4) use vibrations in their operation, they will be influenced by gravitational force as well. To mitigate those shifts we collected 10000 samples for 10 different random orientations and averaged them into one value. Resulting offset values are displayed in 3.3.

Gyro X	Gyro Y	Gyro Z
50.22	−167.16	−155.86

Table 3.3: Calculated offset values of gyroscope.

For lack of means for finding out otherwise, we will assume zero crosstalk and absolute linearity. So model of gyroscope sensor will have the following form

$$\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix} = \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}\begin{bmatrix}\theta_4\\ \theta_5\\ \theta_6\end{bmatrix} + \begin{bmatrix}50.22\\ -167.16\\ -155.86\end{bmatrix} + \begin{bmatrix}w_x\\ w_y\\ w_z\end{bmatrix}. \tag{3.9}$$

where $\theta_4, \theta_5, \theta_6$ are, in order, representing angle velocity in $x, y,$ and z axis. Now, when we have models of both sensors we can construct comprehensive model for of acceleration.

$$\mathbf{A}_{\mathrm{n}} = \mathbf{R}_{\mathrm{x}}\mathbf{R}_{\mathrm{y}}\mathbf{R}_{\mathrm{z}}\boldsymbol{\theta}_{\mathrm{n}} \tag{3.10}$$

where $\mathbf{R}_{\mathrm{x}}, \mathbf{R}_{\mathrm{y}}, \mathbf{R}_{\mathrm{z}}$ are rotation matrices, $\boldsymbol{\theta}_{\mathrm{n}}$ vector of accelerations in all three axes in n-th sample and $\mathbf{A}$ is final vector of acceleration.

3.3 Trajectory estimation

A vital part of finding a usable and preferable estimator is choosing a clear way of accessing one's performance. Given our goal, to estimate the real trajectory of a moving object, our choice is obvious. We will compare correct and estimated trajectory. MATLAB will be used for data processing and plotting.

To transform accelerometer and gyroscope data to a trajectory, we will use equation 2.12. For a discrete time and three-dimensional vector, we can rewrite 2.12 to

$$\begin{bmatrix}x_n\\ y_n\\ z_n\end{bmatrix} = \frac{1}{2}\begin{bmatrix}\theta_{1,n}\\ \theta_{2,n}\\ \theta_{3,n}\end{bmatrix} t_n^2 + \begin{bmatrix}v_{1,n}\\ v_{2,n}\\ v_{3,n}\end{bmatrix} t_n + \begin{bmatrix}x_{n-1}\\ y_{n-1}\\ z_{n-1}\end{bmatrix}. \tag{3.11}$$

On the left side we have coordinates of a new calculated position $\theta_{1,n}, \theta_{2,n}, \theta_{3,n}$, represents acceleration in x, y, z in time n, $v_{1,n}, v_{2,n}, v_{3,n}$ are velocities in time

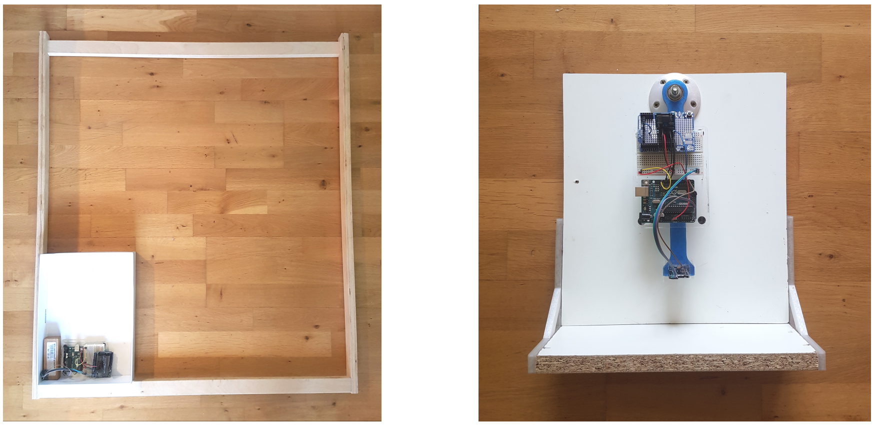
n and $x_{n-1}, y_{n-1}, z_{n-1}$ are coordinates of a point of position in space in time $n-1$. Velocities will be calculated from modified equation 2.9

$$\begin{bmatrix} v_{1,n} \\ v_{2,n} \\ v_{3,n} \end{bmatrix} = \begin{bmatrix} \theta_{1,n} \\ \theta_{2,n} \\ \theta_{3,n} \end{bmatrix} t_n + \begin{bmatrix} v_{1,n-1} \\ v_{2,n-1} \\ v_{3,n-1} \end{bmatrix}. \tag{3.12}$$

As initial conditions for position and velocity, we will choose

$$\begin{bmatrix} x_{1,0} \\ y_{2,0} \\ z_{3,0} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} v_{1,0} \\ v_{2,0} \\ v_{3,0} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

We will use two trajectories whose parameters are known, as a reference. In the scope of this work, we will focus solely on a two-dimensional trajectory, for the sake of available time and resources. Models and equations will be constructed and stated for a three-dimensional case nonetheless. Square with side 0.6 m long and a circle with a radius of $r = 0.22$ m. For reproducibility of the experiment, we will construct guiding apparatuses for each trajectory. For a square, we chose a wooden frame and a box into which sensor will be secured, and guide it through corners. The centrifuge will be constructed for a circular trajectory, ensuring strictly circular path. Both constructions are shown on the figure 3.9. Square will test quick changes in acceleration and circle cooperation between the two sensors. These will be our references for ideal trajectories. Because these are two-dimensional movements, with the intention of zero movements in the z-axis, we will substitute samples of z-axis acceleration with zeros. Because our sensor cannot distinguish between gravitational acceleration, we would have to subtract $1g$ of acceleration from the z-axis, and if rotational movement along x or y-axis were present, we would have to track these changes and dynamically adjust our compensation.

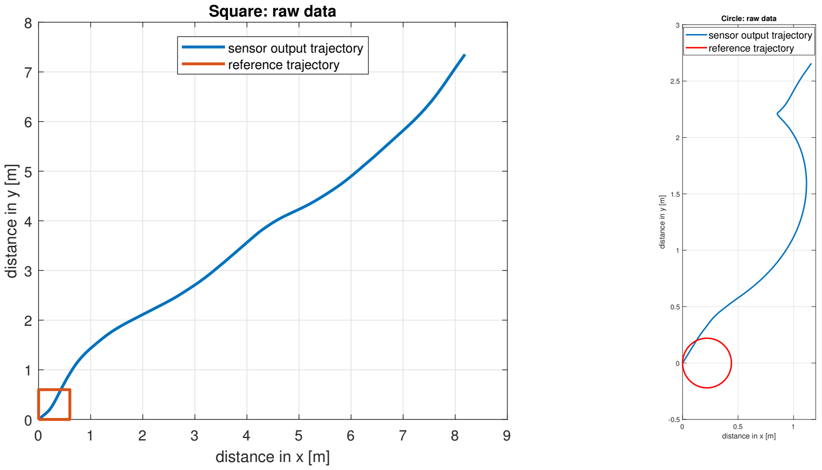


(a) Square trajectory

(b) Circle trajectory

Figure 3.9: Constructed guiding structures

This introduces an excess problem which is beyond the scope of this study. Let us first plot raw data, outputted from our sensors. Output for a square is shown in figure 3.10 a, circle in figure 3.10 b. At first glance, we can see that reference and outputted trajectories differ significantly. It is caused mostly by angle α of a sensor relative to the gravitational force, which will cause offsets dependent on this angle. An accelerometer cannot distinguish between acceleration caused by a change of velocity from that caused by gravity. The situation is illustrated by image 3.11 With square trajectory, we moved the sensor on a relatively smooth surface, and movement of the sensor was only translational, so no rotational movement. This means that when the surface is not level, the sensor will recognize constant offsets on its x and y-axes throughout the experiment. This problem can be mitigated, by initial calibration. If we let the sensor sit at rest before we start moving it along the trajectory, we can estimate those offsets by averaging samples at rest and subtract calculated values from the signals. On the figure 3.12 is displayed



(a) Square trajectory

(b) Circle trajectory

Figure 3.10: Raw data plotted

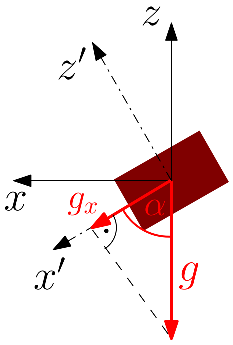
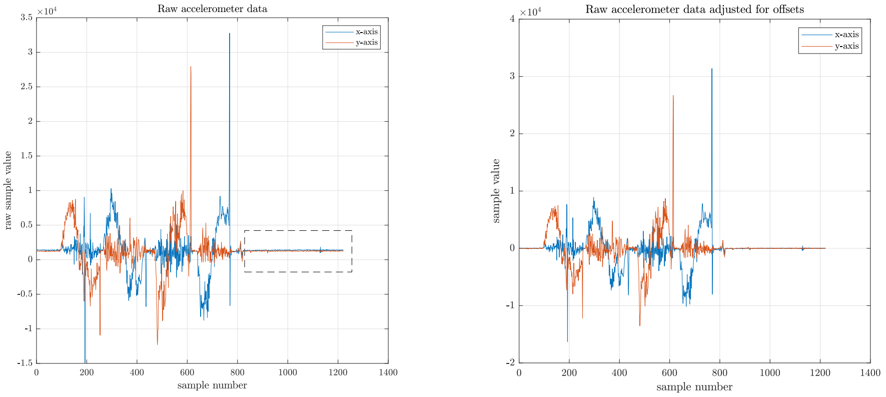


Figure 3.11: Angle of accelerometer

unprocessed output of accelerometer sensor, and with a dashed rectangle, there is enclosed the part used for offset estimation. In this case, we use data after the trajectory have been drawn, but the endpoint is identical with the starting point so it should not make a difference.



(a) With offsets

(b) With offsets subtracted

Figure 3.12: Adjusting for offsets

Calculated values which are stated in the table 3.4 below, were subtracted from the data and adjusted signals plotted. Resulting trajectory can be seen on 3.13

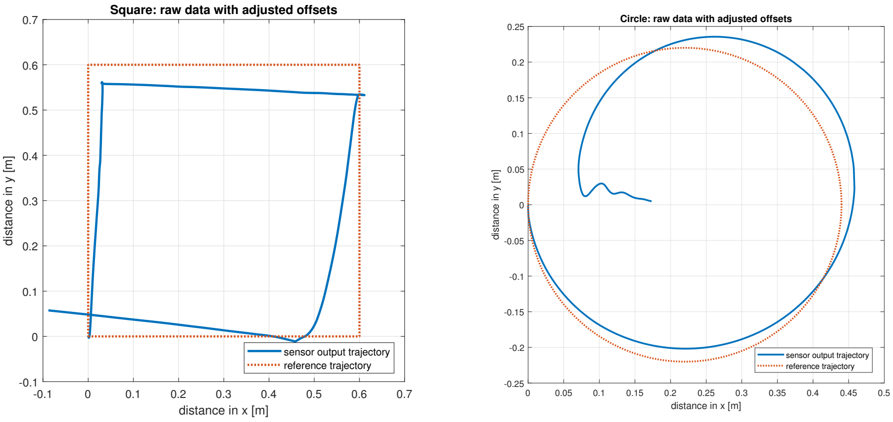
axis	X	Y
offset	1381	1257

Table 3.4: Calculated offsets

We can see that shape of calculated trajectory improved significantly. It is still inaccurate, but we can make out the overall shape. When we try the same approach to the circular trajectory, we will find that it has a similar effect.

3.3.1 ML Point estimator

Lets try to improve our trajectory estimate a bit further with ML estimator introduced in section 2.6. We will use equation 2.23 and redress it for our sensor models. For our accelerometer and gyroscope models, corresponding matrices **H** and **C** from equation 3.9 and 3.11 will be substituted in ML estimator 2.23. Our x will be vectors



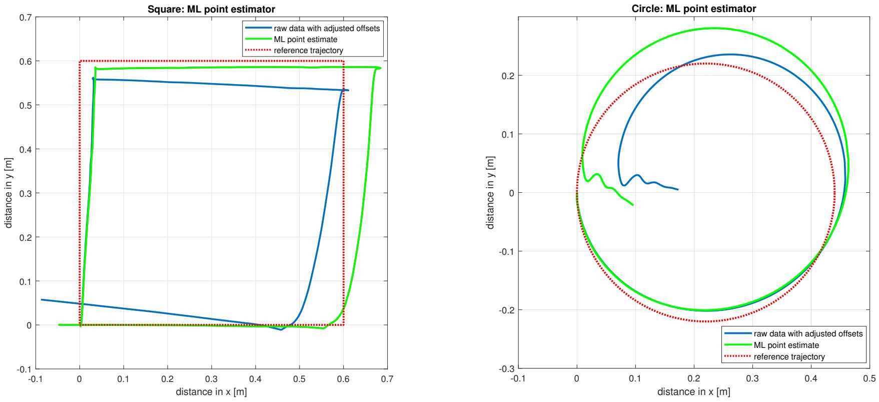
(a) Square trajectory

(b) Circular trajector

Figure 3.13: Effect of offset adjustment

$$a_n = \begin{bmatrix} a_{x,n} \\ a_{y,n} \\ a_{z,n} \end{bmatrix}, \quad g_n = \begin{bmatrix} g_{x,n} \\ g_{y,n} \\ g_{z,n} \end{bmatrix},$$

which have one acceleration and angular velocity sample for each axis for one given moment in time n. We will loop through the whole recorded signals and replace each recorded sample with its estimation point by point. Created estimators are provided in appended files *ML_point_estimator_circle.m* and *ML_point_estimator_square.m*. The result can be seen in figure 3.14.



(a) ML point estimator: square

(b) ML point estimator: circle

Figure 3.14: Effects of ML point estimator

We can see that trajectory estimate yielded slightly better result than raw data with offset adjustment, but we must take into consideration that our

z-axis signal is strict zero. If it were not the case, it would influence our result significantly through cross-talk.
We will try to improve our estimate further and attempt to smooth out our data. As we can see in figure 3.15, there are a few unusually significant peak values which could be caused by error or temporary malfunction. To get rid of those, we will utilise ML block-wise estimation.

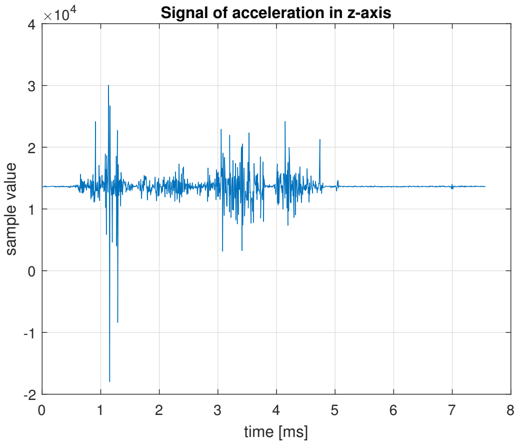


Figure 3.15: Undesirable peaks in signal

3.3.2 ML estimator for block-wise constant dynamic model

In this section one important assumption, regarding dynamic behaviour of studied signals, will be made. We will state that signal is for the duration of $n \in \mathbb{N}$ samples constant in value. For this objective we have to modify the model of observation to the following form

$$\begin{bmatrix} a_{x,1} \\ a_{x,2} \\ . \\ . \\ a_{x,n} \\ a_{y,1} \\ a_{y,2} \\ . \\ . \\ a_{y,n} \\ a_{z,1} \\ a_{z,2} \\ . \\ . \\ a_{z,n} \end{bmatrix} = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{xx} & S_{xy} & S_{xz} \\ . & & \\ . & & \\ S_{xy} & S_{yy} & S_{yz} \\ S_{xy} & S_{yy} & S_{yz} \\ . & & \\ . & & \\ S_{xz} & S_{yz} & S_{zz} \\ S_{xz} & S_{yz} & S_{zz} \\ . & & \\ . & & \end{bmatrix} \left[\begin{bmatrix} C_1 \\ C_2 \\ C_3 \end{bmatrix} + \begin{bmatrix} O_x \\ O_y \\ O_z \end{bmatrix} \right] + \begin{bmatrix} w_x \\ w_x \\ . \\ . \\ w_y \\ w_y \\ . \\ . \\ w_y \\ w_z \\ . \\ . \\ w_z \end{bmatrix}. \tag{3.13}$$

which takes n samples from each axis, ordered in one column. Observation matrix is composed of three sections with n identical rows. We will use simplified version for two dimensions with consideration to our trajectories. Gyroscopic data are to be estimated first and then use them for acceleration estimation. To gain insight into how trajectory is affected by this estimation, we will iterate through increasingly longer n for one variable, with the other one constant and vice versa. Then we will inspect its influence on trajectory. Created estimators are provided in appended files *ML_constant_estimator_circle.m* and *ML_constant_estimator_square.m*. Example of comparison between original and estimated signal is shown in figure 3.16.

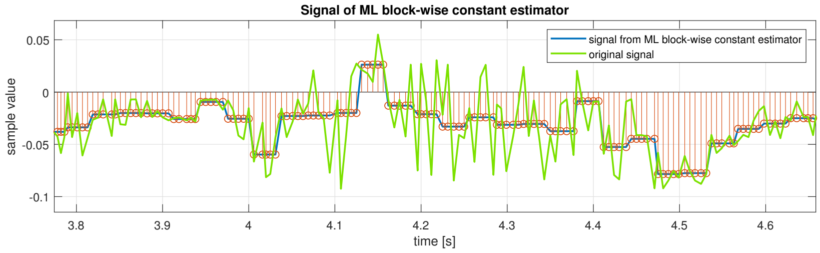
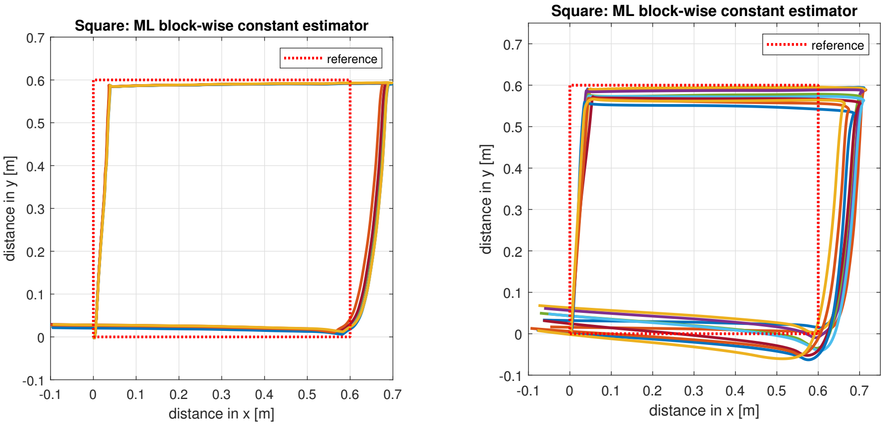


Figure 3.16: Comparison between original and estimated signal with ML constant estimation for $n = 5$

Results of our iterations are illustrated in figure 3.17.



(a) Estimating angular velocity

(b) Estimating acceleration

Figure 3.17: Trying different numbers of samples estimated into one constant. For $n \in \{2, 7, 12, 17, 22, 27, 32, 37, 42, 47\}$

As we can see from 3.17 estimating angular velocity does not have a significant effect on the shape of trajectory, which is understandable given the fact that this trajectory was achieved strictly with translational movement. On the other hand estimation of acceleration has a much greater effect. From closer inspection it becomes apparent that for $n > 9$ the trajectory becomes too deformed and the best choice seems to be $n = 5$ (3.18).

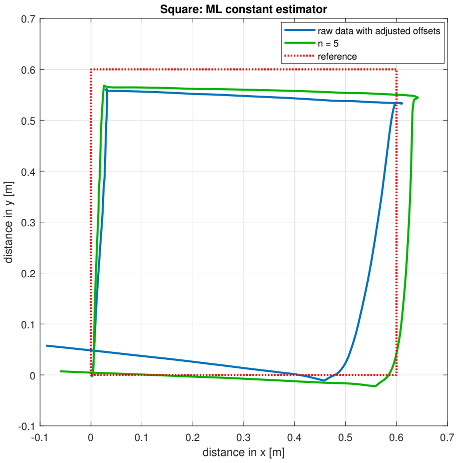
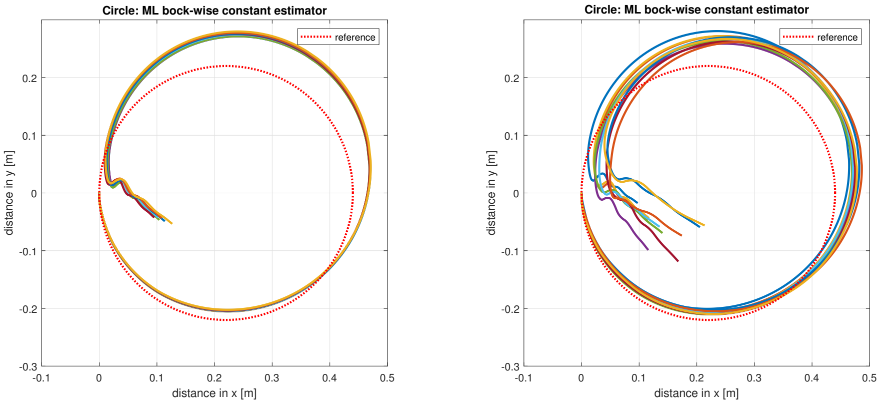


Figure 3.18: Comparison between raw data with adjusted offset and ML block-wise constant estimation for $n = 5$

Our best estimate of square trajectory is then achieved with $n = 1$ for angular velocity signals and $n = 5$ for acceleration. Let's try the same process with a circular trajectory from the figure 3.19. We can see the same tendencies as with the previous case. Again using different values of n for angular velocity have almost no effect and estimating the acceleration had some influence, but if it leads to a better approximation of true trajectory is questionable.



(a) Estimating angular velocity

(b) Estimating acceleration

Figure 3.19: Trying different numbers of samples estimated into one constant. For $n \in \{5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$

3.3.3 ML estimator for block-wise linear dynamic model

In this section, we will modify our assumption about the dynamic behaviour of studied signals. We will presume that our signal is made up of lines, given by an expression

$$x = \mathrm{A}t + \mathrm{B}. \tag{3.14}$$

Let us say we have a set of n samples to be estimated. In our data, we have time saved as a duration between samples. To position m-th sample properly in the timeline, we need to accumulate all previous $m-1$ sample durations to obtain the proper time. Then we put these values into variables $t_1, t_2, .., t_n$ used in the model 3.15 below. Then after the parameters A_n and B_n are calculated, we will sample linear function specified by these parameters at previously calculated moments of time and added as substitutes for the original samples from the outputted signal. A simplified two-dimensional model will be used, as with the previous model.

$$\begin{bmatrix} a_{x,1} \\ a_{x,2} \\ \vdots \\ a_{x,n} \\ a_{y,1} \\ a_{y,2} \\ \vdots \\ a_{y,n} \\ a_{z,1} \\ a_{z,2} \\ \vdots \\ a_{z,n} \end{bmatrix} = \begin{bmatrix} S_{xx}\cdot 0 & S_{xx}\cdot 1 & S_{xy}\cdot 0 & S_{xy}\cdot 1 & S_{xz}\cdot 0 & S_{xz}\cdot 1 \\ S_{xx}\cdot t_1 & S_{xx}\cdot 1 & S_{xy}\cdot t_1 & S_{xy}\cdot 1 & S_{xz}\cdot t_1 & S_{xz}\cdot 1 \\ & & \vdots & & & \\ S_{xx}\cdot t_n & S_{xx}\cdot 1 & S_{xy}\cdot t_n & S_{xy}\cdot 1 & S_{xz}\cdot t_1 & S_{xz}\cdot 1 \\ S_{xy}\cdot 0 & S_{xy}\cdot 1 & S_{yy}\cdot 0 & S_{yy}\cdot 1 & S_{yz}\cdot 0 & S_{yz}\cdot 1 \\ S_{xy}\cdot t_1 & S_{xy}\cdot 1 & S_{yy}\cdot t_1 & S_{yy}\cdot 1 & S_{yz}\cdot t_1 & S_{yz}\cdot 1 \\ \vdots & & & & & \\ S_{xy}\cdot t_n & S_{xy}\cdot 1 & S_{yy}\cdot t_n & S_{yy}\cdot 1 & S_{yz}\cdot t_1 & S_{yz}\cdot 1 \\ S_{xz}\cdot 0 & S_{xz}\cdot 1 & S_{yz}\cdot 0 & S_{yz}\cdot 1 & S_{zz}\cdot 0 & S_{zz}\cdot 1 \\ S_{xz}\cdot t_1 & S_{xz}\cdot 1 & S_{yz}\cdot t_1 & S_{yz}\cdot 1 & S_{zz}\cdot t_1 & S_{zz}\cdot 1 \\ \vdots & & & & & \\ S_{xz}\cdot t_n & S_{xz}\cdot 1 & S_{yz}\cdot t_n & S_{yz}\cdot 1 & S_{zz}\cdot t_n & S_{zz}\cdot 1 \end{bmatrix} \begin{bmatrix} A_1 \\ B_1 \\ A_2 \\ B_2 \\ A_3 \\ B_3 \end{bmatrix} + \begin{bmatrix} w_x \\ w_x \\ \vdots \\ w_x \\ w_y \\ w_y \\ \vdots \\ w_y \\ w_z \\ w_z \\ \vdots \\ w_z \end{bmatrix} \tag{3.15}$$

Example of comparison between original and signal attained via ML line estimator is shown in figure 3.20. Created estimators are provided in appended files *ML_line_estimator_circle.m* and *ML_line_estimator_square.m*.

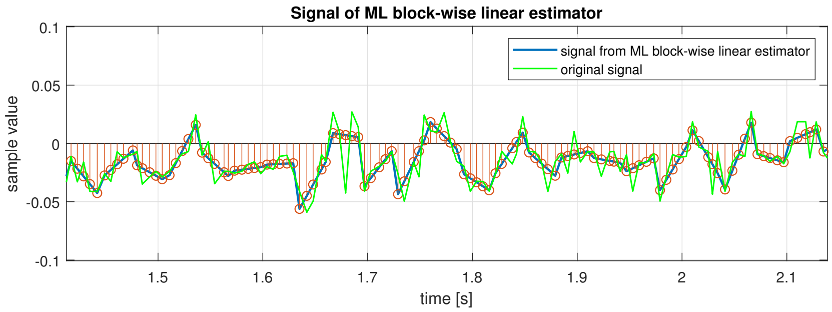
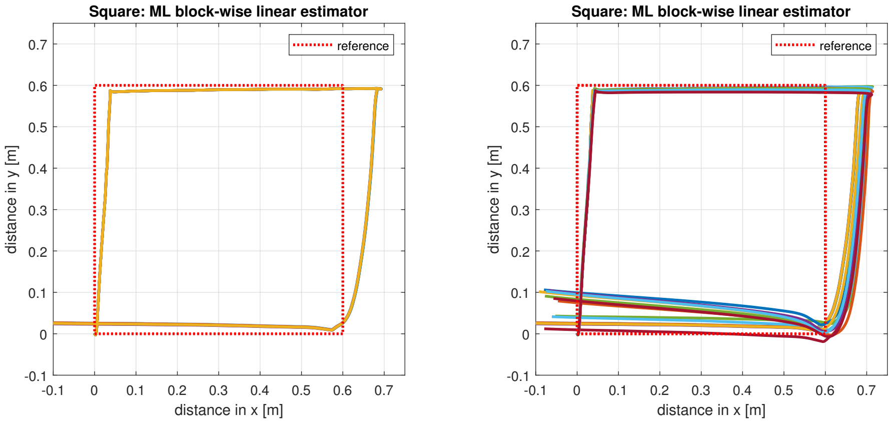


Figure 3.20: Comparison between original and estimated signal with ML line estimation for $n = 5$

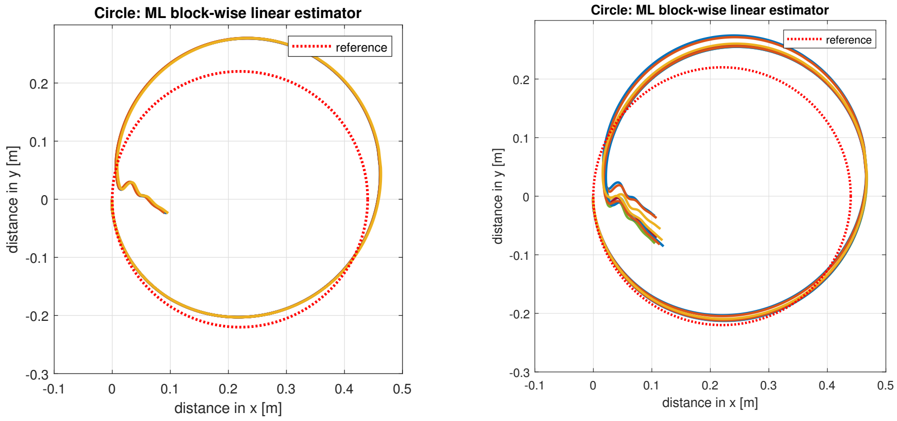
We will recreate the same experiment as with the previous model. Several different values of n will be tried first on the signal of angular velocity, with lowest possible value for acceleration estimate held constant and vice versa. The lowest value of n for ML line estimator is $n = 2$ because it needs at least two points to determine parameters of a line properly. Results are displayed on 3.21 and 3.22.



(a) Estimating angular velocity

(b) Estimating acceleration

Figure 3.21: Trying different numbers of samples estimated into one line. For $n \in \{5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$



(a) Estimating angular velocity

(b) Estimating acceleration

Figure 3.22: Trying different numbers of samples estimated into one line. For $n \in \{5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$

From a comparison with figure 3.19 from the previous estimator, we can assume very comparable results. Estimator again leads to minor changes in trajectory.

3.3.4 ML estimator for block-wise quadratic dynamic model

Our last estimator will attempt to fit through n samples a quadratic function given by expression.

$$x = At^2 + Bt + C$$

We will again shift our samples to zero, calculate their position in time and insert those values into time variables used in modified observation model 3.16 below.

$$\begin{bmatrix} a_{x,1} \\ a_{x,2} \\ \vdots \\ a_{x,n} \\ a_{y,1} \\ a_{y,2} \\ \vdots \\ a_{y,n} \\ a_{z,1} \\ a_{z,2} \\ \vdots \\ a_{z,n} \end{bmatrix} = \begin{bmatrix} S_{xx}\cdot 0 & S_{xx}\cdot 0 & S_{xx}\cdot 1 & S_{xy}\cdot 0 & S_{xy}\cdot 0 & S_{xy}\cdot 1 & S_{xz}\cdot 0 & S_{xz}\cdot 0 & S_{xz}\cdot 1 \\ S_{xx}\cdot t_1^2 & S_{xx}\cdot t_1 & S_{xx}\cdot 1 & S_{xy}\cdot t_1^2 & S_{xy}\cdot t_1 0 & S_{xy}\cdot 1 & S_{xz}\cdot t_1^2 & S_{xz}\cdot t_1 & S_{xz}\cdot 1 \\ & & \vdots & & & & & & \\ S_{xy}\cdot t_n^2 & S_{xy}\cdot t_n & S_{xy}\cdot 1 & S_{yy}\cdot t_n^2 & S_{yy}\cdot t_n 0 & S_{yy}\cdot 1 & S_{yz}\cdot t_n^2 & S_{yz}\cdot t_n & S_{yz}\cdot 1 \\ S_{xy}\cdot 0 & S_{xy}\cdot 0 & S_{xy}\cdot 1 & S_{yy}\cdot 0 & S_{yy}\cdot 0 & S_{yy}\cdot 1 & S_{yz}\cdot 0 & S_{yz}\cdot 0 & S_{yz}\cdot 1 \\ S_{xy}\cdot t_1^2 & S_{xy}\cdot t_1 & S_{xy}\cdot 1 & S_{yy}\cdot t_1^2 & S_{yy}\cdot t_1 0 & S_{yy}\cdot 1 & S_{yz}\cdot t_1^2 & S_{yz}\cdot t_1 & S_{yz}\cdot 1 \\ & & \vdots & & & & & & \\ S_{xy}\cdot t_n^2 & S_{xy}\cdot t_n & S_{xy}\cdot 1 & S_{yy}\cdot t_n^2 & S_{yy}\cdot t_n 0 & S_{yy}\cdot 1 & S_{yz}\cdot t_n^2 & S_{yz}\cdot t_n & S_{yz}\cdot 1 \\ S_{xz}\cdot 0 & S_{xz}\cdot 0 & S_{xz}\cdot 1 & S_{yz}\cdot 0 & S_{yz}\cdot 0 & S_{yz}\cdot 1 & S_{zz}\cdot 0 & S_{zz}\cdot 0 & S_{zz}\cdot 1 \\ S_{xz}\cdot t_1^2 & S_{xz}\cdot t_1 & S_{yz}\cdot 1 & S_{yz}\cdot t_1^2 & S_{yz}\cdot t_1 0 & S_{yz}\cdot 1 & S_{zz}\cdot t_1^2 & S_{zz}\cdot t_1 & S_{zz}\cdot 1 \\ & & \vdots & & & & & & \\ S_{xz}\cdot t_n^2 & S_{xz}\cdot t_n & S_{xz}\cdot 1 & S_{yz}\cdot t_n^2 & S_{yz}\cdot t_n 0 & S_{yz}\cdot 1 & S_{zz}\cdot t_n^2 & S_{zz}\cdot t_n & S_{zz}\cdot 1 \end{bmatrix} \begin{bmatrix} A_1 \\ B_1 \\ C_1 \\ A_2 \\ B_2 \\ C_2 \\ A_3 \\ B_3 \\ C_3 \end{bmatrix} + \begin{bmatrix} w_x \\ w_x \\ \vdots \\ w_x \\ w_y \\ w_y \\ \vdots \\ w_y \\ w_z \\ w_z \\ \vdots \\ w_z \end{bmatrix} \tag{3.16}$$

Example of comparison between original and signal attained via ML block-wise quadratic estimator is shown in figure 3.23. Created estimators are provided in appended files *ML_quad_estimator_circle.m* and *ML_quad_estimator_square.m*.

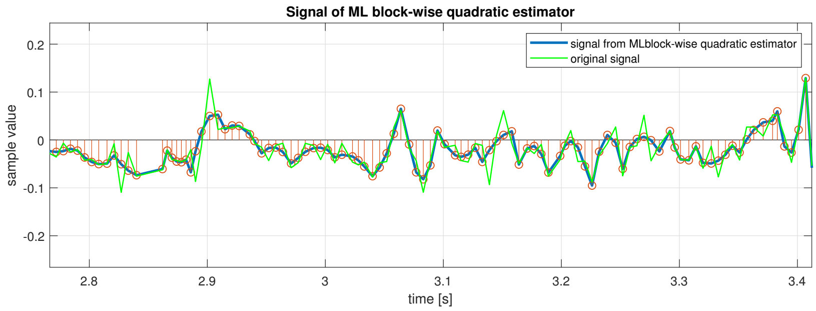
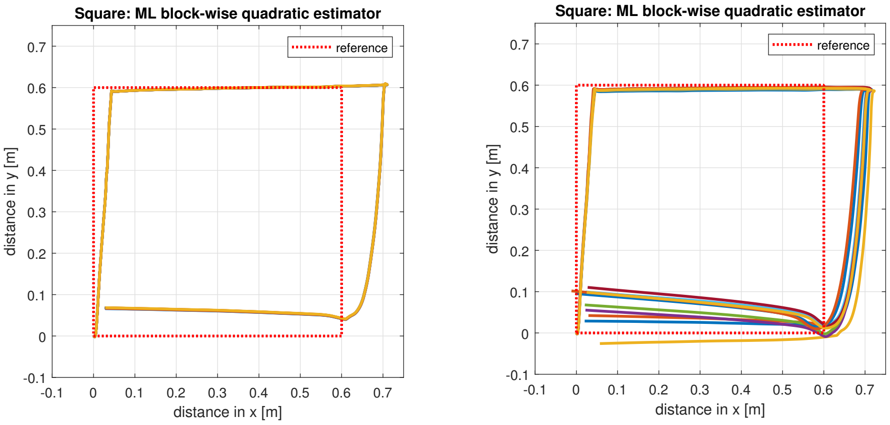


Figure 3.23: Comparison between original and estimated signal with ML block-wise quadratic estimation for $n = 5$

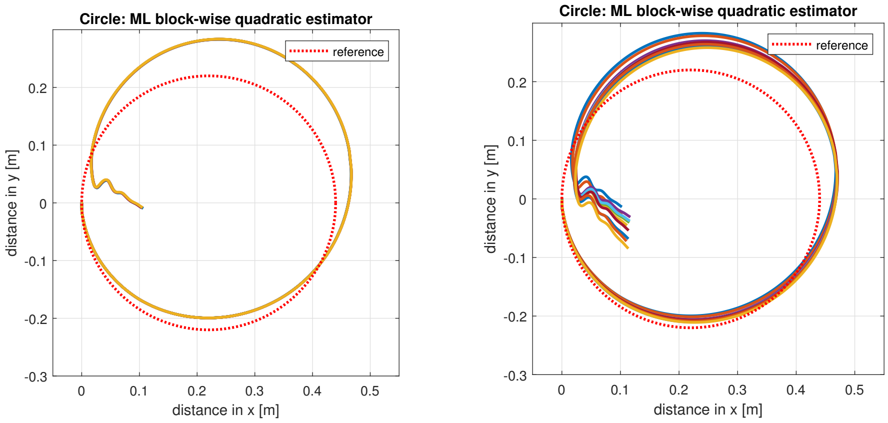
Again we will test performance of this estimator with same experiment as in previous sections. Resulting trajectories are displayed in figure 3.24 and 3.25 below.



(a) Estimating angular velocity

(b) Estimating acceleration

Figure 3.24: Trying different numbers of samples estimated into one quadratic function. For $n \in \{5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$



(a) Estimating angular velocity

(b) Estimating acceleration

Figure 3.25: Trying different numbers of samples estimated into one quadratic function. For $n \in \{5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$

Once more no major improvement was achieved. This could be perhaps because of very high Signal-to-noise ratio of our signals, caused by very good quality of our sensor, which renders any additional smoothing out of our data ineffective.

3.4 Notes on further progress

From our experiments is clear, that our ML estimators do not significantly affect the trajectory. Perhaps it could be beneficial to verify our hypothesis about the too good quality sensor and try an experiment with lower signal-to-noise ratio. This can be achieved by moving through the trajectory slower, thereby lowering the amplitude of the signal, or add some additional noise to the signal. Vibration from a mobile phone placed close to the sensor could be a convenient arrangement. Such an experiment could sufficiently test the capabilities of used estimators. As we can see, for example, on the figure 3.23, our estimators can work out parameters of a continuous function and sample it, but these functions are discontinuous between each other, which can lead to artificial spikes in the signal. Further thought could be put into estimators which can so to say "respect the history" of the signal and remove mentioned discontinuities. One of the possibilities can be application of AR/MA or ARMA models on the data sets. In the ideal case we would use Kalman filtering which takes into account whole history of the data, tries to correct itself and changes its own parameters throughout the measurement.

Chapter 4

Conclusion

The assignment of this thesis was to get acquainted with a chosen motion sensor, construct and program a real measuring device for data acquisition with it. Then analyse its properties and create an observation model based on them along with the design of basic estimators of the exact trajectory of mechanical movement. We divided this task into several steps. In the first section, we studied different types of accelerometers and gyroscopes, their design and principles on which they operate, to understand their behaviour. Several different types were presented and described including MEMS type sensors which were later used in practical part of this study. Following section was devoted to an explanation of basic principles for mechanical movement such as trajectory calculations, rotational matrices and fundamentals of estimation theory, its application, linear models and Maximum likelihood estimators. In the second section, we inspected properties of our actual chosen sensors, constructed and explained our motion tracking device and basic workings of its program. Subsequently we hypothesised its observation model. Then we began with measurements of noise parameters present in attained signals, which led to confirmation of AWGN channel. Model of our accelerometer was achieved by modified autocalibration method devised by I. Frosio, F. Pedersini, N. A. Borghese and published in [4]. A gyroscopic model was calculated from datasheet information and measured offset values. Two reference trajectories were chosen, square and circle, each testing different capabilities of sensors, and guiding apparatuses constructed for corresponding paths. Trajectory estimations were achieved by preliminary compensation of gravitational acceleration, which caused offsets in our measurements, realised by average of values from calibration phase of the outputted signal, followed by Maximum Likelihood estimators of several different modifications, which attempted to estimate given signal as a series of constants and linear or quadratic functions composed of variable number of samples. Introduction of calibration phase significantly improved the accuracy of attained trajectory. ML estimators led just to minor changes in trajectory, whose beneficial effect was questionable. Suboptimal performance o ML estimators regardless their type or intensity was most likely caused by high signal to noise ratio of collected signals, produced by exceptional quality and high sensitivity of used sensors. All points of the assignment have been successfully met.

Bibliography

[1] J.E. Starr, J. Dorsey and C.C. Perry, *Fifty years plus of accelerometer history for shock and vibration (1940–1996)*, IMEKO XIWorld Congress, Houston, TX, October 1988

[2] Passaro, Vittorio M. N. et al. *Gyroscope Technology and Applications: A Review in the Industrial Perspective.* Sensors (Basel, Switzerland) 17.10 (2017): 2284. PMC. Web. 13 May 2018.

[3] InvenSense *MPU-9250 Product Specification* 2016

[4] I. Frosio, F. Pedersini, N. A. Borghese *Autocalibration of MEMS Accelerometers* IEEE transactions on instrumentation and measurement, vol. 58, No. 6, June 2009

[5] Kay, Steven M., *Fundamentals of Statistical Signal Processing, Volume I: Estimation theory*, Prentice Hall, 2009

[6] Image: *Arduino and MPU-9250 wiring* `https://maker.pro/storage/LuePqRP/LuePqRP4SZEwZVOOiKOtAwz8yLXRET6Z3Bql0M33.png`

[7] *Arduino and SD card module wiring*
`https://i2.wp.com/randomnerdtutorials.com/wp-content/uploads/2017/08/sd-card-adaptor-fritzing_bb.png?ssl=1`

[8] *Image of a straing gauge accelerometer*
`http://1.bp.blogspot.com/-Q47bBsOknBA/UHfFldZrjcI/AAAAAAAAAjg/4I96OsIx5o4/s1600/Strain+gauge+Accelerometer.jpg`

[9] *Image of a capacitance accelerometer*
`https://eu.mouser.com/images/microsites/mems-art-fig02.png`

[10] *Materials Classroom*
`http://classroom.materials.ac.uk/casePiez.php`

[11] *MEMS*, Winston Churchill - Engineering and Technology History Wiki
`http://ethw.org/MEMS`

Appendix A

List of MATLAB files

A.1 Estimators

- ML_point_estimator_circle.m
- ML_point_estimator_square.m
- ML_constant_estimator_circle.m
- ML_constant_estimator_square.m
- ML_line_estimator_circle.m
- ML_line_estimator_square.m
- ML_quad_estimator_circle.m
- ML_quad_estimator_square.m

A.2 Functions

- Accmatdefine.m
- Accmatdefine_const.m
- Accematdefine_lin.m
- Accmatdefine_quad.m
- Gyrmatdefine.m
- Gyrmatdefine_const.m
- Gyrmatdefine_quad.m
- lineSampler.m
- timeAccum.m
- timeAccum_quad.m

- quadSampler.m
- rotation.m

A.3 Other files

- optimisation.m
- variation.m
- DATA_variance.txt
- square.txt
- circle.txt
- autocalibration.txt