

BALÁZS ENDRÉS
TDK PROJECT WORK

BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS
FACULTY OF MECHANICAL ENGINEERING
DEPARTMENT OF APPLIED MECHANICS



BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS
FACULTY OF MECHANICAL ENGINEERING
DEPARTMENT OF APPLIED MECHANICS

BALÁZS ENDRÉSZ
TDK PROJECT WORK

Image processing methods applied for balancing the Furuta pendulum

Supervisor:

Dr. Gábor Stépán

Professor

BUDAPEST, 2023

Copyright © Balázs Endrész, 2023.

Contents

List of figures	VII
1 Introduction	1
2 Design of the pendulum	2
2.1 Previous Furuta pendulum design	2
2.2 Requirements	3
2.3 Actual design	4
3 Balancing of Furuta pendulum	5
3.1 Balancing of a stick	5
3.2 Model of Furuta pendulum	7
4 Computer vision methods estimating orientation	16
4.1 Hough transform	16
4.2 Canny edge detection algorithm	17
4.3 Optical flow algorithms	21
4.4 Lucas Kanade algorithm	22
4.5 Dense optical flow algorithms	24
4.6 Feature detectors	27
5 Development of angle measurement algorithms	32
5.1 Hough transformation	32
5.2 Sweeping along the image	33
5.3 Sobel kernels	36
5.4 Vertical feature detection	39
5.5 Sift feature detectors	40
5.6 Optical flow	41
5.7 Comparison of methods	42
6 Summary	44
Összefoglaló	45
References	45
Appendix	48

Acknowledgement

The research reported in this work and carried out at BME has been supported by the Hungarian National Research, Development and Innovation Office (NKFI-KKP-133846).

I want to acknowledge and give my warmest thanks to my supervisor, Gábor Stépán for giving me the opportunity to explore the project and for his support, guidance throughout the work.

I would also like to special thanks to Richárd Wohlfart and Ákos Miklós for all the help and advices I have got in connection with the design to be able to get through all the obstacles in front of me. Without them this work would not be possible.

I am grateful for Máté Vizi giving me advices and creating the opportunity to consulting with him in connection the theoretical background and design of the structure.

I want to acknowledge, that without my friends and family this work could not be possible. I needed their support to be able to finish my work and regain my motivation all the way through the project. I have always been thankful for Csenge Dremák for making me believe in myself and helping me develop as a person all the time.

Budapest, 2023

Balázs Endrész

List of Figures

2.1	The original measurement setup of the pendulum [1]	2
3.1	Mechanical model of stick balancing	5
3.2	Mechanical model of Furuta-pendulum [2]	8
3.3	Visualization of ZOH [1]	12
4.1	Line representation in distance parameter spaces	17
4.2	Building of Department of Applied Mechanics after applying vertical Sobel kernel. Original image source is [3]	19
4.3	Building of Department of Applied Mechanics after applying Canny edge detector. Original image source is [3]	21
4.4	Aperture problem visualization [4]	23
4.5	Concept of pyramidal LK algorithm [5]	24
4.6	Farneback algorithm visualization [6]	27
4.7	Building of Department of Applied Mechanics after applying Sift feature detector. Original image source is [3]	31
4.8	Descriptor of Sift features of Figure 4.7	31
5.1	Reference rectangle object	32
5.2	Reference object with dense vertical information	32
5.3	Building of Department of Applied Mechanics [3]	32
5.4	Detected angle by Hough transform	33
5.5	Detected angle by sweep	35
5.6	Absolute angle difference by sweep method	36
5.7	Effect of vertical and horizontal kernels applying on the image	37
5.8	Sobel kernel approach, creating a linear filter for angle detection	38
5.9	Detected angle by sobel kernel detector	39
5.10	Detected angle by vertical feature detector	40
5.11	Detected angle by Sift feature detector	41
5.12	Farneback algorithm result by rotating Figure 5.3	41
6.1	Shrinkage dependece of the resultant difference function.	48
6.2	Shrinkage rate dependence of the slope.	48
6.3	Shrinkage rate dependence of the offset.	48

1 Introduction

In everyday life human balancing plays an important role. Balancing is special, because it is used not only in case of walking and running, but while standing as well. This is because of the fact that the human body is unstable, so without any control, the body would fall. This has an evolutionary advantage on escaping from dangerous situations, since letting the body tilting forward and gaining momentum is much beneficial than making efforts to move out of the equilibrium position.

In a mechanical point of view, balancing is defined as a control task to keep the body close to its vertical axis. Humans use senses from the vestibular, visual and proprioceptive systems. The sensors using by the human brain for this task consist of the labyrinth, the eyes and the receptors determining limb positions. The balancing organ, which is the so-called labyrinth, provides both angle and angular velocity signals by static and dynamic receptors [7]. The visual sensory system has also significant effect on the balancing task, but its exact way of influence has not yet explored. As researches have shown, not only the orientation of the head is measured by the eyes, but the velocity is also estimated. It is worth to mention that without a reference vertical or horizontal direction, only by eyes, the balancing task cannot be resolved. Some correlation has explored between colors and sensitivity to motion in a specific direction.

A simple, but descriptive model of human balancing is a stick balanced in the fingertip. As no special device is needed for a measurement, the effect of visual balancing and critical time delay can be evaluated [8]. Balancing models describes not only the behaviour of a human while standing and balancing, but they can also predict the effect of aging [9]. Examining how the balancing strategy of a person changes over time can help to prevent falls, which is significant as the probability of major injuries increase by age.

It is a challenging task to control an inverted and rotary inverted pendulum, due to their nonlinearity. However, due to their simplicity, they are excellent tools for observing distinct control tasks, and they can be used for educational purposes. Many engineering applications can be modelled with the help of inverted pendulums, such as the control of a rocket. A reliable option to measure the orientation of a pendulum is using encoder devices. It can model how the labyrinth works, representing measuring the change from some inner sources of the system. But as the computational power increases, new possibilities arise, for example modelling the visual sensory system.

The objective of this work is exploring possibilities of substituting encoder devices by camera, fixed at the top of a Furuta pendulum. The task is challenging, because the camera is not in a fixed position, so the reference orientation should be provided in some way. This work can help to understand the limitations of a computationally expensive operation for control tasks, where the computation leads to relevant delay in the control loop. The resultant structure is a more complex system using fewer sensors, this way, it can be considered as a more insensitive device for hardware defects.

2 Design of the pendulum

2.1 Previous Furuta pendulum design

As part of a previous research of digital control of the Furuta pendulum created by Máté Vizi [1], a laboratory device has developed. This device uses a Maxon EC 45 brushless motor having 50 W power output (Part Number: 251601), which play an essential role in the structure. The calculated control signals are fed into the motor, which creates the actuation torque on the whole structure via the arm. Besides the motor, a Maxon Spur Gearhead with 32:1 reduction ratio (Part Number: 266595) makes the motor motion even more smooth for better input resolution. The motor is controlled by Maxon ESCON Module 50/5 servo controller (Part Number: 438725) and the computation by the control law is processed by an STM32F407VG microcontroller. The STM device reads the necessary angular position information from two encoders and after the computation is done, sends a PWM (pulse-width modulation) signal towards the motor. The orientation of the pendulum in this setup is determined by an 8000 count-per-revolution optical encoder (IHC3808), while the rotation of the arm is obtained by a 400 count-per-revolution optical encoder.

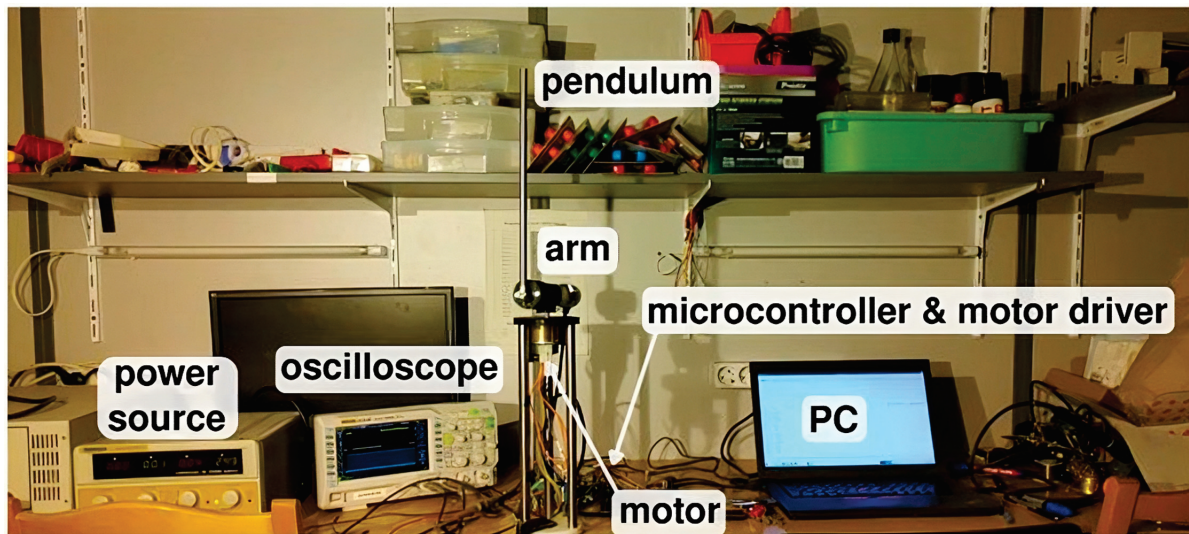


Figure 2.1: The original measurement setup of the pendulum [1]

The construction consists of a pendulum, an arm, and a mount, which provides the required height, so the pendulum does not bump into the environment beneath the level of the arm. One of the most significant part of the device is the microcontroller, which can read the input signals and give PWM output signal. Another important feature of a microprocessor is that it runs only the code without any OS (operating system). In everyday usage, the operating system is useful, since it enables to run many jobs in a parallel manner by scheduling the tasks. Keeping track of these phenomena using high-level operating systems, as Windows or Linux is impossible, because they combine multiple scheduling algorithms as round-robin with multi level feedback queue or priority

driven preemptive scheduling system and cannot be changed from user side. In case of robotic applications, when maintaining and keeping the time delay is essential, the devices using OS leads to unexpected behaviour. While the constant time lag results retarded differential equations (RDEs), in case of non-constant delay the motion can be modelled with the help of stochastic retarded differential equations (SRDEs), due to fact that the time delay can be statistically modelled.

Another crucial part of the design is related to the resolution of the encoders. As the equation of motion is independent of the position of the arm shown in Section 3.2, the resolution of the encoder measuring the pendulum angle is relevant for successful control.

2.2 Requirements

As a first step of the design, a camera should be selected, as it limits the available resolution and speed of the angle determination method.

As it is derived in Section 3, there is a theoretical critical time lag for stable balancing, which cannot be exceeded. If the camera sends the signal with too small FPS (frame-per-second) rate, then the program has not enough time to process the reaching signal and the performance is limited to the camera. In wireless applications, the time spent on sending and receiving the signal is larger than in case of versions with cable, moreover, the loss of packages can affect the performance, which is more likely in wireless cameras. On the other hand, the available FPS rate and resolution are inversely proportional, since the larger the resolution is, the larger the size of the data required to be sent from the camera.

Furthermore, in camera selection, another important aspect is that the lens should provide a distortion-free wide angle image. If the image is distorted, then more computationally expensive transformations are needed to gain a straight vertical reference line on the image. Some listed methods in Section 5 use reference objects to be able to determine the orientation of the camera. A wide angle camera can help to keep the reference object for wide arm angle in the picture.

As a third requirement, the size of the camera plays an important role. The lighter the camera is, the less it influences the moment of inertia of the pendulum, so the results are comparable with the encoder controlled case.

Though using camera with cable enables faster response and safer communication, it also affects the mechanical model. The cable influences the model by causing torsional stiffness and additional damping. The routing of the wire to the PC (personal computer) should also be resolved. Since the microcontroller possesses only relatively small computational capacity and only able to run restricted type of programming languages. The image processing is advisable to be handled in a computer and sending only the results of the computation to the STM microprocessor to resolve the scheduling problem afterwards. This option is considered as budget friendly, since a PC is available in the laboratory and

microcontrollers with high computational capacity and distinct language options are more expensive.

Considering the available programming languages, one should be selected, which is able to handle computer vision algorithms and meanwhile able to run fast enough to manage the resultant algorithm, within time of about 10 milliseconds.

2.3 Actual design

For a camera balancing task, a new design is required, since there is no specific place for holding the camera, and routing the cable creates further complications.

After considering many devices, the chosen camera is an ELP UVC 1200P, which supports USB connection, 90 FPS with distortion-free 126° lens, which meets the requirements explained in Section 2.2.

Many variations of cable routing design were considered for providing small rotational stiffness, such as designing a polyimide PCB (printed circuit board) solution or using slip rings. An easy and practical solution appeared to get rid of the outer coating of the cable, this way reducing the stiffness, and it is routing through the centre of the arm. The constraint of this method is that it only enables a few rotations of the pendulum, while for control tasks, one full rotation of the pendulum is sufficient. For further improvement, that design was selected, which is appropriate in case of redesign. It can be adopted easily for swinging-up the pendulum in the future.

The above-mentioned IHC3808 encoder ends in a shaft and shaft coupling was necessary in the previous design. Having an encoder like this, it holds the problem that the cable cannot be routed through the device. This way a new encoder was necessary, which can be mounted on a hollow shaft. This encoder is required for better comparison between the angle detection by camera and encoder, while it plays important role in the verification of the algorithm as well. As it measures the rotation of the pendulum, which is considered essential for control task, the resolution of the encoder should be selected with care. The selected encoder for this task is a 4000 count-per-revolution device (Part number: 630-AEDB-9340-B13C).

After the crucial off-the-shelf components were selected, the designed elements were modelled in CAD (computer aided design) software and manufactured by lathe operation and printed by FDM 3D printer. The further required components were bought as new bearings and screws for this new design. The assembly drawing can be seen in the Appendix.

3 Balancing of Furuta pendulum

3.1 Balancing of a stick

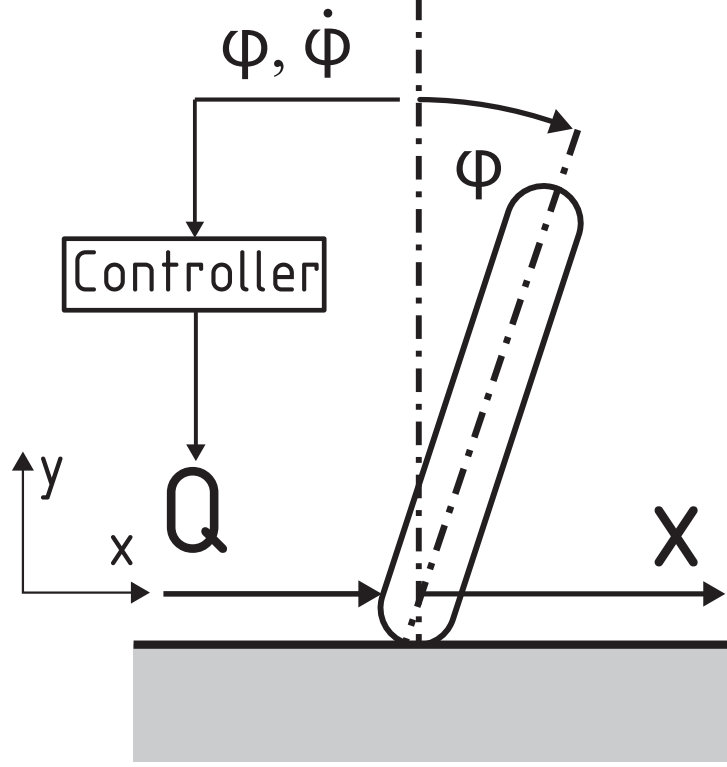


Figure 3.1: Mechanical model of stick balancing

From mechanical point of view, one of the most simple model of human balancing is a stick, which has its tip on the ground and the goal is to keep it at its upper position by acting with a control force at the lower tip. As it can be seen in [10], the control of this structure can be modelled as a *PD* controller by measuring the angle φ of the system. The equation of motion can be obtained by the Lagrangian equation of the second kind, since the system is holonomic

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_k} - \frac{\partial L}{\partial q_k} = Q_k, \quad (3.1)$$

where L is the Lagrangian function, $q_k, k = 1, \dots, n$ are the generalized coordinates and $Q_k, k = 1, \dots, n$ are the generalized forces. The Lagrangian function is constructed in the following form

$$L = T - V, \quad (3.2)$$

where T is the kinetic energy and V is the potential function. Generally the kinetic energy can be calculated in the centre of mass with a 2D motion assumption as

$$T = \sum_i \frac{1}{2} m_i v_{i_{\text{CoM}}}^2 + \frac{1}{2} J_{i_{\text{CoM}z}} \omega_i^2, \quad (3.3)$$

where $m_i, i = 1, \dots, n$ is the mass of the i^{th} rigid body, $v_{i_{\text{CoM}}}, i = 1, \dots, n$ is the velocity at the centre of mass of the i^{th} rigid body, $J_{i_{\text{CoM}z}}$ the moment of inertia of the i^{th} rigid

body with respect to the z direction, that is the perpendicular to the plane of the motion, $\omega_i, i = 1, \dots, n$ is the angular velocity of the i^{th} rigid body. In the present case

$$T = \frac{1}{2}mv_{\text{CoM}}^2 + \frac{1}{2}J_{\text{CoM}_z}\omega^2. \quad (3.4)$$

Choosing x and φ as the generalized coordinates, the velocities can be reformulated as

$$\mathbf{v}_{\text{CoM}} = \mathbf{v}_A + \omega \times \mathbf{r}_{A \rightarrow \text{CoM}} = \begin{bmatrix} 0 \\ 0 \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{\varphi} \end{bmatrix} \times \begin{bmatrix} \frac{1}{2}l \sin(\varphi) \\ \frac{1}{2}l \cos(\varphi) \\ 0 \end{bmatrix}, \quad (3.5)$$

so the velocity described by the generalized coordinates is

$$\mathbf{v}_{\text{CoM}} = \begin{bmatrix} \dot{x} + \frac{1}{2}l\dot{\varphi} \cos(\varphi) \\ -\frac{1}{2}l\dot{\varphi} \sin(\varphi) \\ 0 \end{bmatrix}. \quad (3.6)$$

The moment of inertia can be computed in case of a stick as

$$J_{\text{CoM}_z} = \frac{1}{12}ml^2, \quad (3.7)$$

where l is the length of the stick. The resultant kinetic energy function is

$$T = \frac{1}{2} \left(\frac{1}{3}ml^2 \right) \dot{\varphi}^2 + \frac{1}{2}ml\dot{x}\dot{\varphi} \cos(\varphi) + \frac{1}{2}m\dot{x}^2. \quad (3.8)$$

The potential energy function contains the effect of the gravity, the equation describing the gravitational potential is

$$V = mg\frac{1}{2}l \cos(\varphi), \quad (3.9)$$

where g is the gravitational acceleration. The presented problem is under actuated, due only the tip of the stick is pushed by a determined force, which has direct effect only in the x generalized coordinate as the function of φ other coordinate. φ is not directly controlled by any outer force.

$$\mathbf{Q} = \begin{bmatrix} 0 \\ Q(\varphi, \dot{\varphi}) \end{bmatrix} \quad (3.10)$$

The equation of motion after all the derivates were computed

$$\begin{bmatrix} \frac{1}{3}ml^2 & \frac{1}{2}ml \cos(\varphi) \\ \frac{1}{2}ml \cos(\varphi) & m \end{bmatrix} \begin{bmatrix} \ddot{\varphi} \\ \ddot{x} \end{bmatrix} + \begin{bmatrix} -mgl \sin(\varphi) \\ -\frac{1}{2}ml\dot{\varphi}^2 \sin(\varphi) \end{bmatrix} = \begin{bmatrix} 0 \\ F(\varphi, \dot{\varphi}) \end{bmatrix}. \quad (3.11)$$

As it can be observed, the equation of motion is independent of the x generalized coordinate itself, it is only dependent on its second derivative. Since x is not present in the Lagrangian function, the derivative by the generalized coordinate is

$$\frac{\partial L}{\partial x} = 0. \quad (3.12)$$

A generalized coordinate having such a property is called as "cyclic" coordinate, which causes symmetry in the system. The motion is independent of the value of the x coordinate, it only depends on its derivatives, which means on a flat surface the controller behaves identically of the position of the stick.

As it was mentioned above, the system can be controlled by using a PD control strategy. Modelling the delay in the human sensory system assuming a constant τ time lag and P and D control constants, the resultant equation is

$$Q(\varphi, \dot{\varphi}) = D\dot{\varphi}(t - \tau) + P\varphi(t - \tau). \quad (3.13)$$

Using a controller, many properties can be prescribed for each the desired behaviour such as the settling time, steady state error, robustness, but one of the most significant properties is the stability. As it is stated in [10] the stability can be examined by Routh-Hurwitz criterion in case of no delay, $\tau = 0$. The criterion is described at page 14. After linearization of the system around the equilibrium $\varphi = 0$ and eliminating $\ddot{x}$ from Equation (3.11) the following formula is resulted

$$\ddot{\varphi}(t) - \frac{6g}{l}\varphi(t) + \frac{6}{ml}(D\dot{\varphi}(t - \tau) + P\varphi(t - \tau)) = 0. \quad (3.14)$$

Using Routh-Hurwitz criterion in case of the ordinary differential equation, using the $\tau = 0$ assumption, the system is stable if

$$P > mg, \quad D > 0. \quad (3.15)$$

In the presence of time lag, the stable region shrinks, and it is asymptotically stable if

$$mg = P_{\min} < P < P_{\max}(\omega) = \left(mg + \frac{ml}{6\tau^2}\omega^2 \right) \cos(\omega), \quad (3.16)$$

where ω the unique solution of equation

$$D\omega = P\tau \tan(\omega) \quad (3.17)$$

in the region of $\omega \in (0, \frac{\pi}{2})$. Then the critical time delay is

$$\tau_{\text{cr}} = \sqrt{\frac{l}{3g}}. \quad (3.18)$$

3.2 Model of Furuta pendulum

The above described model is widespread due to its simple structure and direct measurement methods by the help of humans. The critical time delay can be measured by balancing a stick on the fingertip. Due to its relatively easy task it can be a great benchmarking task for distinct type of control strategies.

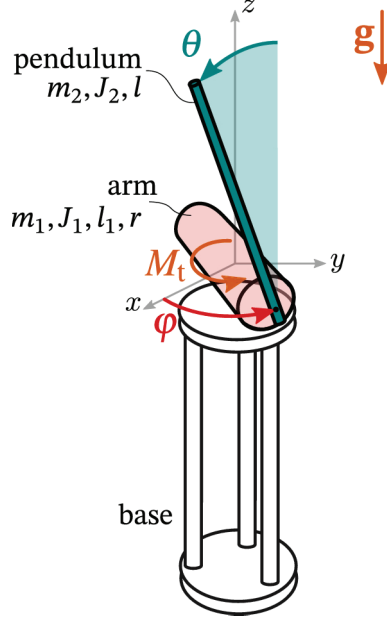


Figure 3.2: Mechanical model of Furuta-pendulum [2]

A similar, under actuated robotic system is the rotary inverted pendulum the so-called Furuta pendulum. Since the structure is similarly simple it is still able to be a benchmarking task for control [11], but the behaviour is more complex, due to the rotation around to different axis the motion is not planar anymore, and more non-linearities can occur.

In 3 dimensional cases an algorithmic way of computing the position and orientation of a point of a serial robot is using homogenous transform matrices [12], [13] with the following formalism

$${}^0\mathbf{T}_i = \prod_{k=1}^i {}^{k-1}\mathbf{T}_k, \quad (3.19)$$

where ${}^{k-1}\mathbf{T}_k$ is the transformation matrix from k^{th} link local reference frame to the $(k-1)^{\text{st}}$ link. The matrix can be constructed as

$${}^{k-1}\mathbf{T}_k = \begin{bmatrix} \begin{bmatrix} {}^{k-1}\mathbf{R}_k \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} {}^{k-1}\mathbf{p}_k \\ 1 \end{bmatrix} \end{bmatrix}, \quad (3.20)$$

where ${}^{k-1}\mathbf{R}_k$ is to rotational transformation matrix from k^{th} link local reference frame to the $(k-1)^{\text{st}}$ link, and ${}^{k-1}\mathbf{p}_k$ is the position measured in the k^{th} reference frame projected to the $(k-1)^{\text{st}}$ link.

Using this method with the Denavit-Hartenberg convention [14], which describes the joints as the serial connections of only 1 dimensional joints. Using this assumption all matrices can be created in a way that translation is described by only one local axis of the joint and the rotation is given by another one. Between two adjacent joints only rotation

around 1 axis is allowed.

To gain the equation of motion using the Lagrangian equation of the second kind, the kinetic energy can be introduced as

$$T = \frac{1}{2}J_a\omega_a^2 + \frac{1}{2}m_pv_{\text{PCoM}}^2 + \frac{1}{2}J_{\text{PCoM}}\omega_p^2, \quad (3.21)$$

where a denotes the arm, p denotes the pendulum itself. In the figure this is denoted as J_1 and J_2 , instead of p and a, respectively. J_{a_z} is calculated with respect to the rotation axis of the motor, and J_{PCoM} is the moment of inertia of the pendulum calculated with respect to the pivot axis of the arm. For further calculations, the generalized coordinates should be selected. The selected coordinates are the rotation of the arm denoted by φ , and the rotation of the pendulum with respect to the vertical orientation, which is θ .

The velocity of the centre of mass of the pendulum can be determined by

$${}^0\mathbf{r}_p = {}^0\mathbf{T}_a {}^a\mathbf{T}_p {}^p\mathbf{r}_p, \quad (3.22)$$

where the transformation matrices describing by the two generalized coordinates are

$${}^0\mathbf{T}_a = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) & 0 & 0 \\ \sin(\varphi) & \cos(\varphi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^a\mathbf{T}_p = \begin{bmatrix} 1 & 0 & 0 & r \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (3.23)$$

where l denotes the half-length of the pendulum. The local coordinate system of the arm describes the connection to the global coordinate system by rotation around axis z with the generalized φ coordinate. The local coordinate system of the pendulum can be obtained from the arm coordinate by translation along axis x by the distance of r , and simultaneously rotation around axis x by θ coordinate. So the resultant velocity can be calculated as the time derivative of the position. The resultant velocity is

$${}^0\mathbf{v}_{\text{PCoM}} = \begin{bmatrix} l \sin(\theta) \cos(\varphi) \dot{\varphi} + l \sin(\varphi) \cos(\theta) \dot{\theta} - r \sin(\varphi) \dot{\varphi} \\ l \sin(\theta) \sin(\varphi) \dot{\varphi} - l \cos(\theta) \cos(\varphi) \dot{\theta} + r \cos(\varphi) \dot{\varphi} \\ -l \sin(\theta) \dot{\theta} \end{bmatrix}. \quad (3.24)$$

The relative angular velocity can be determined similarly, by to rotational matrices

$${}^0\omega_{i_{\text{rel}}} = \prod_{k=1}^i {}^{k-1}\mathbf{R}_k {}^i\omega. \quad (3.25)$$

So the pendulum relative angular velocity can be expressed by the general coordinates as

$${}^0\omega_{p_{\text{rel}}} = {}^0\mathbf{R}_a {}^a\mathbf{R}_p {}^p\omega_p, \quad (3.26)$$

where the relative angular velocities matrices can be determined from the transformation matrices according to Equation (3.20). The resultant orientations are

$${}^0\omega_{a_{\text{rel}}} = \begin{bmatrix} 0 \\ 0 \\ \dot{\varphi} \end{bmatrix}, \quad {}^0\omega_{p_{\text{rel}}} = \begin{bmatrix} \cos(\varphi) \dot{\theta} \\ \sin(\varphi) \dot{\theta} \\ 0 \end{bmatrix}, \quad (3.27)$$

while the absolute angular velocities by adding the conductive angular velocities to the expressions are

$${}^0\omega_{\text{a_rel}} = \begin{bmatrix} 0 \\ 0 \\ \dot{\varphi} \end{bmatrix}, \quad {}^0\omega_{\text{p_rel}} = \begin{bmatrix} \cos(\varphi)\dot{\theta} \\ \sin(\varphi)\dot{\theta} \\ \dot{\varphi} \end{bmatrix}. \quad (3.28)$$

The V potential term includes the gravitational potential effect and a modelled torsional stiffness from the cable as

$$V = mg \underbrace{{}^0r_{\text{pz}}}_{l \cos(\theta)} + \frac{1}{2}k_t\theta^2. \quad (3.29)$$

The above presented Lagrangian equation of the second kind (3.1) can be extended with a dissipation term called the Rayleigh's dissipation function $\mathcal{D}$, which results

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_k} - \frac{\partial L}{\partial q_k} - \frac{\partial \mathcal{D}}{\partial \dot{q}_k} = Q_k. \quad (3.30)$$

Rayleigh's dissipation function can be interpreted as

$$\mathcal{D} = \frac{1}{2}b_1\dot{\theta}^2 + \frac{1}{2}b_2\dot{\theta}^2, \quad (3.31)$$

where b_1 and b_2 are viscous damping coefficient due to the machinery elements imperfect connections.

Similarly to the stick balancing task, the system is controlled in an under actuated manner by motor acting only on the arm

$$\mathbf{Q} = \begin{bmatrix} 0 \\ M(\theta, \dot{\theta}, \dot{\varphi}) \end{bmatrix}. \quad (3.32)$$

The resultant set of equations of motion is

$$\begin{aligned} \begin{bmatrix} J_p + l^2m & -lmr \cos(\theta) \\ -mlr \cos(\theta) & J_a + J_p + mr^2 + ml^2 \sin^2(\theta) \end{bmatrix} \begin{bmatrix} \ddot{\theta} \\ \ddot{\varphi} \end{bmatrix} + \begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \dot{\varphi} \end{bmatrix} + \begin{bmatrix} k_t & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \varphi \end{bmatrix} + \\ + \begin{bmatrix} -glm \sin(\theta) + \frac{1}{2}l^2m \sin(2\theta)\dot{\varphi}^2 + 2lmr \sin(\theta)\dot{\theta}\dot{\varphi} \\ ml^2 \sin(2\theta)\dot{\theta}\dot{\varphi} + mlr \sin(\theta)\dot{\theta}^2 \end{bmatrix} = \begin{bmatrix} 0 \\ M \end{bmatrix}. \end{aligned} \quad (3.33)$$

As it can be seen, the φ is not represented in the equation moreover

$$\frac{\partial L}{\partial \varphi} = 0, \quad (3.34)$$

so the generalized coordinate is a cyclic coordinate, the motion is independent of the exact orientation of the pendulum. So from the second equation, $\ddot{\varphi}$ can be determined

and substituting back to the first equation. By this transformation only $\dot{\varphi}$, θ and its derivatives are present.

By the linearization of the equation around the upward position ($\theta \approx 0$) from the Taylor series expansion the following substitutions can be made $\sin(\theta) \approx \theta$, and $\cos(\theta) \approx 1$. Simultaneously all the higher than first order terms can be eliminated from the equation of the motion of the pendulum. The resultant form is

$$\underbrace{\begin{bmatrix} \tilde{J}_p & -mrl \\ -mrl & \tilde{J}_a \end{bmatrix}}_{=\mathbf{M}} \underbrace{\begin{bmatrix} \ddot{\theta} \\ \ddot{\varphi} \end{bmatrix}}_{=\ddot{\mathbf{q}}} + \underbrace{\begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix}}_{=\mathbf{B}} \underbrace{\begin{bmatrix} \dot{\theta} \\ \dot{\varphi} \end{bmatrix}}_{=\dot{\mathbf{q}}} + \underbrace{\begin{bmatrix} k_t - glm & 0 \\ 0 & 0 \end{bmatrix}}_{=\mathbf{K}} \underbrace{\begin{bmatrix} \theta \\ \varphi \end{bmatrix}}_{=\mathbf{q}} = \underbrace{\begin{bmatrix} 0 \\ M \end{bmatrix}}_{=\mathbf{Q}}, \quad (3.35)$$

where $\tilde{J}_p = J_p + l^2 m$ the moment of inertia with respect to the pivot shaft in the arm and $\tilde{J}_a = J_a + J_p + mr^2$ is the combined moment of inertia of both the pendulum and arm with respect to the rotational axis of the motor. The form of the linearized equation of motion is

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{B}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{Q}. \quad (3.36)$$

The motor torque can be evaluated by formula of

$$M(\theta, \dot{\theta}, \dot{\varphi}) = NU(\theta, \dot{\theta}, \dot{\varphi}) - K\dot{\varphi}, \quad (3.37)$$

where N is the motor constant and consider the back-EMF (electromagnetic force) effect by the K term. Even though the left side of Equation (3.36) is continuous in time, the right side contains term U control input, which is provided by the microcontroller. The controller computes the acting force with finite speed. Until the calculations are not performed the controller keeps the previous control signal value, creating the so-called ZOH (zero-order-hold). Using this assumption the discrete time steps can be indicated as $t_j = j\tau, j = 0, 1, 2, \dots$

The signals are sampled at time instance t_j and the processor requires up to τ time to calculate the results. If less time is required, the remaining time can be filled with waiting time. This way the variance of τ can be avoided. So the control signal after calculations can be sent out at t_{j+1} as U_{j+1} input for the motor. After that the signal is kept constant until the consecutive input arrives at t_{j+2} . Because of the time delay, when input reaches the motor, the time lag can be considered as τ and during the ZOH, it is kept on the same value, which means that the time difference between the sampled time instance t_j and real time t increases up to 2τ . This way the average time lag is $\tau_{\text{avg}} \in (\tau, 2\tau)$, which can be considered as $\tau_{\text{avg}} = 1.5\tau$. The concept of ZOH and this phenomenon is shown in Figure 3.3. Using ZOH, the resultant input signal can be formulized using PD controller as

$$U_{j+1} = -P_1\theta_j - D_1\dot{\theta}_j + D_2\dot{\varphi}_j. \quad (3.38)$$

Since one of the coordinates is proven to be a cyclic coordinate, the state space is reduced to 3 dimensions

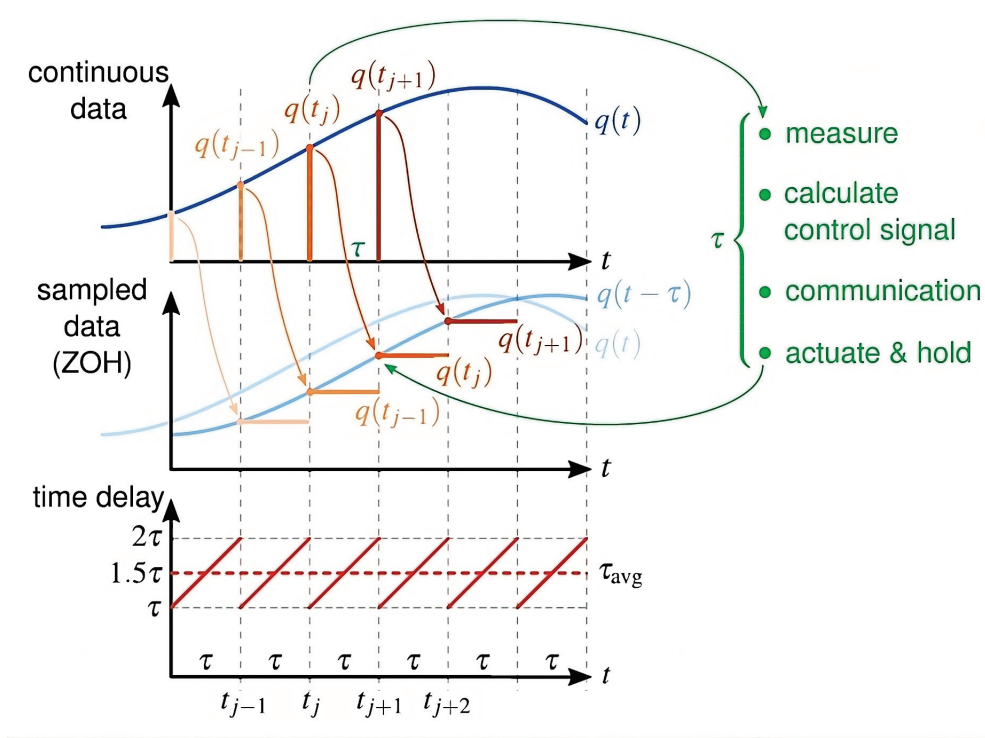


Figure 3.3: Visualization of ZOH [1]

$$\mathbf{x}(t) = \begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \\ \dot{\varphi}(t) \end{bmatrix} \quad (3.39)$$

The state space model of the Furuta pendulum can be achieved by eliminating the second derivative of the cyclic coordinate from Equation 3.35 in the form of

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}U_j, \quad t \in [t_j, t_{j+1}), \quad (3.40)$$

where the two matrices can be determined by substituting back by the cyclic coordinate second derivative $\ddot{\varphi}$ to the Lagrangian equation of the θ generalized coordinate

$$\mathbf{A} = \frac{1}{J^2} \begin{bmatrix} 0 & J^2 & 0 \\ \tilde{k}J_a & -b_1J_a & -mrl\tilde{b}_2 \\ \tilde{k}mrl & -mrlb_1 & -J_p\tilde{b}_2 \end{bmatrix}, \quad \mathbf{B} = \frac{1}{J^2} \begin{bmatrix} 0 \\ Nmrl \\ NJ_p \end{bmatrix}, \quad (3.41)$$

where $J^2 = J_aJ_p - (mrl)^2$, $\tilde{k} = k_t - mgl$ and $\tilde{b}_2 = b_2 + K$. The equation contains continuous and discrete parts at the same time. Because the model is linearized, it can be transformed into fully discrete time.

If the velocity is measured, the Equation (3.40) remains in a similar form, but in discrete form, such as

$$\mathbf{x}_{j+1} = \mathbf{A}\mathbf{x}_j + \mathbf{B}U_j. \quad (3.42)$$

The homogenous and particular solution of the system according to [1] is

$$\mathbf{x}_H(t) = e^{\mathbf{A}t}(\mathbf{x}_j + \mathbf{A}^{-1}\mathbf{B}), \quad t \in [t_j, t_{j+1}) \quad (3.43)$$

$$\mathbf{x}_P(t) = -\mathbf{A}^{-1}\mathbf{B}U_j, \quad t \in [t_j, t_{j+1}) \quad (3.44)$$

By using the extended states the states are

$$\mathbf{y}_j = \begin{bmatrix} \mathbf{x}_j \\ U_j \end{bmatrix}. \quad (3.45)$$

By the help the extended system state can be updated by the equation

$$\mathbf{y}_{j+1} = \mathbf{C}\mathbf{y}_j \quad (3.46)$$

As the encoders and camera can measure only the orientation of the pendulum and arm, the velocity should be estimated in some way. Velocity can be estimated in both cases by finite difference approximation, as

$$\dot{\theta}_j \approx \frac{\theta_j - \theta_{j-1}}{\tau}, \quad \dot{\varphi}_j \approx \frac{\varphi_j - \varphi_{j-1}}{\tau}. \quad (3.47)$$

The new input signal after the substitutions is

$$\tilde{U}_{j+1} = -P_1\theta_j - D_1\frac{\theta_j - \theta_{j-1}}{\tau} + D_2\frac{\varphi_j - \varphi_{j-1}}{\tau} = K_1\theta_j + K_2\theta_{j-1} + K_3(\varphi_j - \varphi_{j-1}). \quad (3.48)$$

The gains can be reassigned as

$$K_1 = -P_1 - \frac{D_1}{\tau}, \quad K_2 = \frac{D_1}{\tau}, \quad K_3 = \frac{D_2}{\tau}. \quad (3.49)$$

It is worth to mention, that by estimating the velocity by the sampled generalized coordinates, the state cannot be reduced to 3 dimensions, since it needs to hold the cyclic coordinate as well. As [1] explained, the dynamics of the system remained, even though the control strategy changed. By using the extended states the states are

$$\dot{\theta}_j \approx \frac{\theta_j - \theta_{j-1}}{\tau}, \quad \dot{\varphi}_j \approx \frac{\varphi_j - \varphi_{j-1}}{\tau}. \quad (3.50)$$

With this the updated extended discrete map is

$$\tilde{\mathbf{y}}_{j+1} = \tilde{\mathbf{C}}\tilde{\mathbf{y}}_j. \quad (3.51)$$

From the extended state equations the eigenvalues of $\mathbf{C}$ and $\tilde{\mathbf{C}}$ can be determined, denoted as μ_j and $\tilde{\mu}_j$, respectively. They are the so-called characteristic exponents. The stability criteria for μ_j and $\tilde{\mu}_j$ values are respectively

$$|\mu| < 1, \forall \mu, \mu \in \mathbb{C} \quad (3.52)$$

and

$$|\tilde{\mu}_j| < 1, j = 1, \dots, n-1, \tilde{\mu}_n = 1, \tilde{\mu} \in \mathbb{C}. \quad (3.53)$$

According to [1], the last characteristic exponent in case of velocity estimation appears, because the cyclic coordinate is reintroduced in the equations. Since no P_2 term is considered in the control strategy a stationary motion of the arm with constant speed is allowed, which includes the case of zero speed as well.

The above described conditions Equation (3.52) and (3.53) determines the stable region in the Argand diagram as a circle with unit radius. The widely used Routh-Hurwitz criterion determines the stability of a differential equation only by all the characteristic exponents should fall onto the left half of the complex plane. The so-called Holographic Transformation projects all the points inside the unit circle to the left half of the plane, and outside the unit circle to the right half. The transformation is the following

$$\mu = \frac{\eta + 1}{\eta - 1}. \quad (3.54)$$

After the equation is transformed, the Routh-Hurwitz criterion can be observed as

1. All the coefficients of the characteristic polynomial - which is created by the transformation - should be positive, as a necessary condition.

$$p(x) = \underbrace{a_n}_{>0} x^n + \underbrace{a_{n-1}}_{>0} x^{n-1} + \underbrace{a_{n-2}}_{>0} x^{n-2} + \dots + \underbrace{a_0}_{>0} \quad (3.55)$$

2. The sufficient condition is that the $\mathbf{H}$ matrix all principal minors should be positive as well, where $\mathbf{H}$ Hurwitz matrix is

$$\mathbf{H} = \begin{bmatrix} a_{n-1} & a_{n-3} & a_{n-5} & \dots & 0 \\ a_n & a_{n-2} & a_{n-4} & \dots & 0 \\ 0 & a_{n-1} & a_{n-3} & \dots & 0 \\ 0 & a_n & a_{n-2} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & a_0 \end{bmatrix}. \quad (3.56)$$

These conditions give stability maps for different P, D parameters. As in [1] is stated the speed estimation shrinks the stable parameter regions and the effect is more significant by increasing τ sampling time. Increasing D_2 improves the stability properties of the speed estimation case. In both cases the critical time delay can be computed. Using the encoder

based pendulum structure the critical time for measured speed and estimated speed case are

$$\tau_{\text{cr}} \approx 0.3 [\text{s}], \quad \tilde{\tau}_{\text{cr}} \approx 0.2 [\text{s}], \quad (3.57)$$

provided by [1]. As a first rule of thumb, the image processing algorithm, which is unable to estimate the angle under this time is not capable of balancing the pendulum. The largest regions in the stability chart is measured at

$$\tau = 0.025 [\text{s}],$$

so for this region should be aimed by the algorithms in the encoder based control.

The effect of the quantization can be considered negligible in case of encoder based balancing, but when the device is changed to camera, this effect may not be considered negligible. In both cases it can be assumed that using double precision floating point numbers keeps the round-off error small enough to neglect their effect. So the only non-negligible error source is from sampling. This issue comes out on the digital side, which affects the U control signal, which can be represented inside as the values are rounded to h precision, in case of both generalized coordinates and in its derivatives. The int operator rounds every value to the integer part of the number, so every positive number down to the closest integer and every negative number up to the closest integer

$$U_{j+1} = -P_1 h \text{int}\left(\frac{1}{h}\theta_j\right) - D_1 h \text{int}\left(\frac{1}{h}\dot{\theta}_j\right) + D_2 h \text{int}\left(\frac{1}{h}\dot{\varphi}_j\right). \quad (3.58)$$

This kind of digital effects can lead to chaotic motions as it is depicted in [15] and [16]. The extensive study of this micro-chaotic behaviour is out of the scope of this work.

4 Computer vision methods estimating orientation

4.1 Hough transform

To be able to estimate the orientation of the camera, one method can occur easily by determining the inclination angle of a known vertical object. By tracking on object with significant vertical features - the remaining task is to fit a line to the vertical feature and determine the angle from this information.

One method, which is able to find the desired objects is Hough Transformation. The following formulation is presented with the help of source [17]. To derive the method, let us assume that the line can be found in the form of

$$y = mx + c, \quad (4.1)$$

where m represents the slope of the line and c is the y -intercept. Considering the case, when the possible points are determined previously by an edge detection algorithm, the task is reduced to fit line to these points. While the least-square fit cannot be used here due to the fact that multiple lines can occur in the image, one possible way can be to calculate a common feature according to each point separately. Using a sampled (x_i, y_i) point from the image, by modifying Equation (4.1) the resultant formula is

$$c = -mx_i + y_i. \quad (4.2)$$

So this creates the connection between the image space ($x-y$ space) and the parameter space ($m-c$ space). Taking one point from the image space will determine one line in the parameter space, by taking into account the dependence between c and m , it can be concluded that if two points are selected from the image space, two non-parallel lines can be observed in the parameter space, which intersects at a specific point (determining an $m-c$ parameter pair, determining the line they lie on). So the more points are selected from a line in the image space, the more lines passing through the same point in the parameter space. If one outlier is selected, it will result distinct intersecting points. The calculation can be reduced by using a discretized parameter space, which can be represented by a so-called accumulator array. The initial array consists of zeros. After one point is selected, all the corresponding $m-c$ cells are increased by one. The higher a value is in the accumulator array, the more points lie in the same line on the image. Determining the strong enough local maxima, the estimated lines can be read from the array.

Though some issues can occur in connection with the parameterization with $m-c$. Since m is not bounded, relatively large accumulator array is required for exact angle determination, which is not optimal for memory and computational processes. This problem can be solved by using another kind of parameterization, where both parameters are bounded. It can be done as follows

$$x \sin(\theta) - y \cos(\theta) + \rho = 0. \quad (4.3)$$

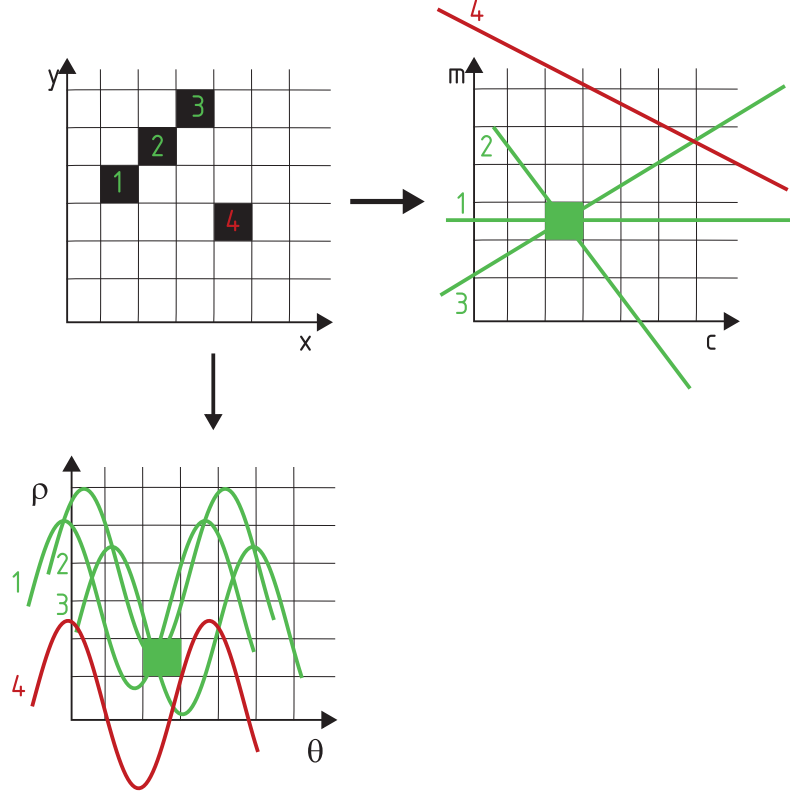


Figure 4.1: Line representation in distance parameter spaces

This case θ orientation angle is inside $[0, 2\pi)$ and ρ distance cannot be larger than the size of the image. Letting ρ being negative the angle space can be further squeezed between $[0, \pi)$.

Some related issues of the problem is, that the size of the accumulator array is due to the low resolution too many lines can fall onto the same parameter space region. In the contrary, if the resolution is too high, in the presence of noise and quantization error, potential lines can be missed out, because well described maxima cannot be found. On the other hand, lines in an image cannot be expected being perfectly straight. This problem can be solved by assigning values not only to the closest fit in the parameter space, but to a small patch of cells.

4.2 Canny edge detection algorithm

In image processing tasks, some cases it is really non-trivial how to formulate the well-known everyday life observations into mathematical schemes, such as what an edge is in the picture. In data perspective, edge is a sudden change in image intensity. One crucial assumption of the above introduced Hough transform method is that information about the contour of edges is collected beforehand the process of finding lines in the image started, to have a set of points, which are needed to observe further.

For the theoretical background of this subsection is based on [18]. The changing rate of

intensity function can be observed by calculating the derivative, which describes the local change, which can represent an edge. A rapid change results a local maxima or minima of the derivative, so the absolute value of the first derivative helps to find the edges, where the magnitude shows the strength of the edge. If the orientation of the edge is need to be considered, the gradient of the intensity function can resolve this problem, showing always the direction of the largest change. This case the magnitude (S) and orientation (θ) can be calculated as

$$S = \|\nabla I\| = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}, \quad (4.4)$$

$$\theta = \arctan\left(\frac{\partial I}{\partial y}\left(\frac{\partial I}{\partial x}\right)^{-1}\right). \quad (4.5)$$

Using discretization by finite difference approximation

$$\frac{\partial I}{\partial x} \approx \frac{1}{2\varepsilon} \left((I_{i+1,j+1} - I_{i,j+1}) + (I_{i+1,j} - I_{i,j}) \right), \quad (4.6)$$

$$\frac{\partial I}{\partial y} \approx \frac{1}{2\varepsilon} \left((I_{i+1,j+1} - I_{i+1,j}) + (I_{i,j+1} - I_{i,j}) \right), \quad (4.7)$$

where ε is the physical distance between adjacent cells. The derivative can be represented as discrete convolution in the following form

$$\frac{\partial}{\partial x} \approx \frac{1}{2\varepsilon} \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}, \quad \frac{\partial}{\partial y} \approx \frac{1}{2\varepsilon} \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix}, \quad (4.8)$$

where the indexes increase from top left corner towards the bottom right. The derivate can be computed with an unknown ε , since the whole derivative is only a scaled by this factor, so the relative magnitude is not effected by that.

Lot of approaches have arisen to approximate the derivative of the intensity function as the Roberts, Prewitt and Sobel kernels. The 3x3 Sobel kernel is constructed as

$$\frac{\partial I}{\partial x}_{\text{Sobel}} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad \frac{\partial I}{\partial y}_{\text{Sobel}} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & 1 \end{bmatrix}. \quad (4.9)$$

This approach can be expended to any matrices, where the number of rows and columns are odd. After the magnitude of the edges were computed, a threshold value should be set to define, how wild change corresponds to an edge, which creates a border between the edges and non-edge regions. A less strong change corresponds to the hysteresis threshold, determined by two thresholds ($T_0 < T_1$). This case the region is an edge, either if the value exceeds the greater threshold value (T_1), or it is between the two and a neighboring

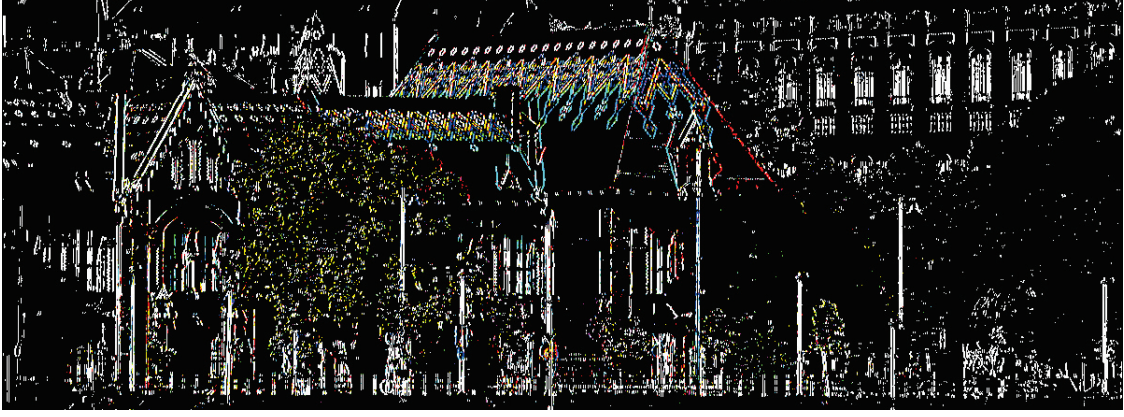


Figure 4.2: Building of Department of Applied Mechanics after applying vertical Sobel kernel. Original image source is [3]

pixel is an edge.

$$\begin{aligned}
 \|\nabla I(x, y)\| &< T_0 - \text{Not an edge} \\
 \|\nabla I(x, y)\| &\geq T_1 - \text{An edge} \\
 T_0 &< \|\nabla I(x, y)\| < T_1 - \text{An edge if at least one neighboring pixel} \\
 &\text{is assigned as an edge}
 \end{aligned} \tag{4.10}$$

Laplacian edge detection uses the second derivative of an image. In the case of a sudden change in the image (as a rising edge), the first derivative has a peak and the second derivative shows sudden change from negative to positive in a closed interval, according to the Bolzano-theorem as far as the function is continuous. This is called zero-crossing. The continuous form of the second derivative is

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \tag{4.11}$$

While gradient is able to give the location, magnitude and direction of the edge, simultaneously, the Laplacian is only able to determine the location by zero-crossing. On the other hand the Laplacian operator is linear, which is a desired property, due less online calculation is required, linear operators can be combined as it is shown later. The gradient operator loses its linearity by applying non-linear operation to calculate the magnitude and orientation described in equation (4.4) and (4.5). After discretization the equations can be formulated as

$$\frac{\partial^2 I}{\partial x^2} \approx \frac{1}{\varepsilon^2} \left((I_{i-1,j} - 2I_{i,j}) + I_{i+1,j} \right), \tag{4.12}$$

$$\frac{\partial^2 I}{\partial y^2} \approx \frac{1}{\varepsilon^2} \left((I_{i,j-1} - 2I_{i,j}) + I_{i,j+1} \right). \tag{4.13}$$

$$\tag{4.14}$$

This case convolutional mask applicable to ∇^2 can be derived as

$$\nabla^2 \approx \frac{1}{\varepsilon^2} \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \quad (4.15)$$

Problem arises, if the orientation of the edge is close to 45° , because in that case the distance between the pixels is not the assumed ε . This can be resolved by taking into account edges in different orientation and averaging after that. The resultant kernel with less sensitivity to orientation can be observed as

$$\nabla^2 \approx \frac{1}{6\varepsilon^2} \begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix}. \quad (4.16)$$

A fundamental problem in all computer-based application, that the sampled signal - in this case the image - contains noise. Noise is generally a high frequency changing quickly. The derivative is sensitive to the change, so it will amplify it. Inside the resultant signal all the characteristic features can be lost. A solution to this problem is to get rid of the noise before the derivative is calculated. One way can be of dealing with noise is to smoothen out the signal read from the sensor. Assuming a statistical behaviour of the noise a Gaussian filter is able to resolve the problem in most cases. The Gaussian filter can be represented as

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}. \quad (4.17)$$

This operator can be used both the derivatives and second derivatives. But instead of computing the Gaussian smoothened version of the image and the derivative afterwards, an operation can be saved by calculating the derivative and second derivative of the Gaussian and applying that to the image. This can be done by the fact that both the derivative and Gaussian smoothing is a linear operator

$$\nabla(G * f) = \nabla(G) * f, \quad (4.18)$$

$$\nabla^2(G * f) = \nabla^2(G) * f. \quad (4.19)$$

The latter one is called the Laplacian of Gaussian, abbreviated as LoG.

The above described algorithm are essential to understand the Canny edge detection in itself. As the first step of the algorithm, the Gaussian Sobel operator is applied, which both applies the smoothing operation and calculates the gradient. Using an above described hysteresis method in Equation (4.10) [19]. After the original implementation was created, some other versions were developed out of the base of this method. The Haralick-Canny-Type edge detector [20] drops the hysteresis threshold and computes the Laplacian along the gradient direction. The detected edges can be estimated by zero-crossing detection algorithms.

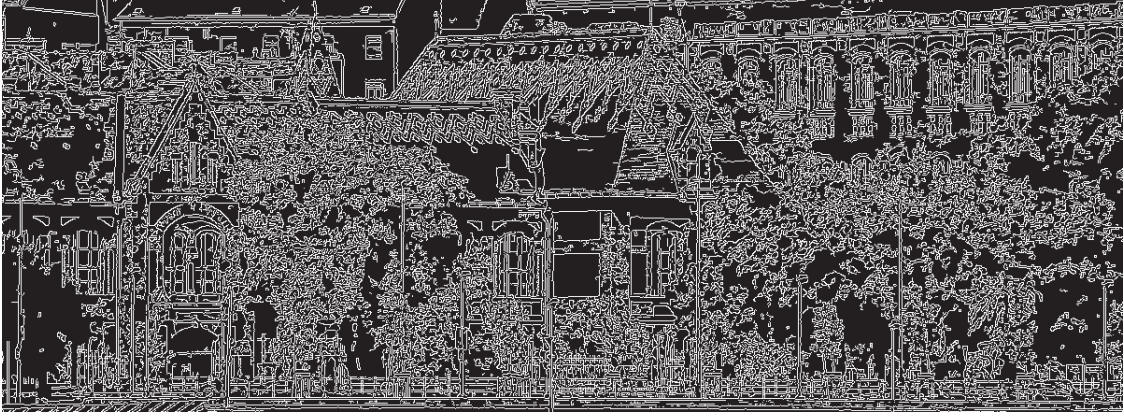


Figure 4.3: Building of Department of Applied Mechanics after applying Canny edge detector. Original image source is [3]

4.3 Optical flow algorithms

Handling a camera stably is a challenging problem, especially, if the user transfer it by hand. To compensate the errors occurring as a result of the camera motion, different approaches can be used. Both the hardware and the software can compensate this issue. From the hardware side using gimbals or sliding the camera on fixed tracks can compensate some errors, but it is still required to have solution for post-processing and when hardware can not be applied conveniently, for example in a case of a mobile phones or embedded systems, having small space requirements. One of the applications of optical flow is to help to stabilize videos.

The aim of optical flow methods is to determine the motion field of consecutive images. From the motion of pixels, not only the stabilization of video are possible, but tracking specific objects or detect the motion of the camera itself. Generally optical flow calculations are designed to minimize metrics, formulated to the task.

The algorithms operate with the brightness or color values of the image. This can be indicated as $I_1(\mathbf{x})$ and $I_0(\mathbf{x})$, where I the intensity function of a specific pixel, determined by $\mathbf{x} = (x, y)$ coordinates, index '1' indicates the recent, '0' previous image. One metrics can be the so-called sum of square differences (SSD), which is computed as

$$E_{\text{SSD}}(\mathbf{u}) = \sum_i [I_1(\mathbf{x}_i + \mathbf{u}) - I_0(\mathbf{x}_i)]^2 = \sum_i e_i^2, \quad (4.20)$$

where $\mathbf{u}_i = (u, v)$ is the displacement of a pixel in consecutive images and $e_i = I_1(\mathbf{x}_i + \mathbf{u}_i) - I_0(\mathbf{x}_i)$ is the residual error. Because the algorithm use intensity patterns of the image, changes in brightness, causes bias in the estimation process. A necessary assumption for the following derivation is that there is no change in the lighting.

4.4 Lucas Kanade algorithm

The original mathematical background was introduced by the application of stereo vision problem, where the distance of the object and the camera parameters were extracted from two different angles of an object [21].

The content of this subsection is based on [22]. While, the above described method can minimize the function up to the limit of a pixel, usually higher accuracy is needed for many applications. Lucas and Kanade introduced a method, which is able to reach sub-pixel accuracy by gradient decent on equation (4.20)

$$E_{\text{LK-SSD}}(\mathbf{u} + \Delta\mathbf{u}) = \sum_i [I_1(\mathbf{x}_i + \mathbf{u} + \Delta\mathbf{u}) - I_0(\mathbf{x}_i)]^2. \quad (4.21)$$

The equation can be reduced by linearization. To do that a natural assumption should be taken, that the displacement of a specific point should be small (preferably infinitesimally small) and simultaneously a small time step can be assumed. This assumption can be satisfied because of the camera average time step in any close-to-real time applications. So Taylor-series expansion can be carried out and after cancelling the higher order terms from the equation

$$E_{\text{LK-SSD}}(\mathbf{u} + \Delta\mathbf{u}) \approx \sum_i [I_1(\mathbf{x}_i + \mathbf{u}) + \mathbf{J}_1(\mathbf{x}_i + \mathbf{u})\Delta\mathbf{u} - I_0(\mathbf{x}_i)]^2, \quad (4.22)$$

where

$$\mathbf{J}_1(\mathbf{x}_i + \mathbf{u}) = \nabla I_1(\mathbf{x}_i + \mathbf{u}) \quad (4.23)$$

is called as the image gradient or Jacobian at $(\mathbf{x}_i + \mathbf{u})$. After linearization the optical flow constraint equation can be represented as

$$I_x u + I_y v + I_t = 0, \quad (4.24)$$

where I_x and I_y denote spatial derivate and I_t temporal derivative of the intensity function. The least-square solution of the above described equation (4.22) can be calculated as

$$\underbrace{\mathbf{J}_1^T(\mathbf{x}_i + \mathbf{u})\mathbf{J}_1(\mathbf{x}_i + \mathbf{u})}_{\mathbf{A}} \Delta\mathbf{u} = \underbrace{-e_i \mathbf{J}_1^T(\mathbf{x}_i + \mathbf{u})}_{\mathbf{b}}. \quad (4.25)$$

The resultant equation for $\Delta\mathbf{u}$ is the following

$$\Delta\mathbf{u} = \mathbf{A}^{-1}\mathbf{b}. \quad (4.26)$$

It gives the necessary condition, that $\mathbf{A}$ should be insertable. One of the occurring problem is mentioned in the early papers of optical flow is the aperture problem. This can be mathematically formulated as the described Hessian $\mathbf{A}$ matrix a rank-deficient one, whose smaller eigenvalues are close to zero, so it barely satisfies the necessary condition. In this case only the normal component of the motion can be captured reliably. This

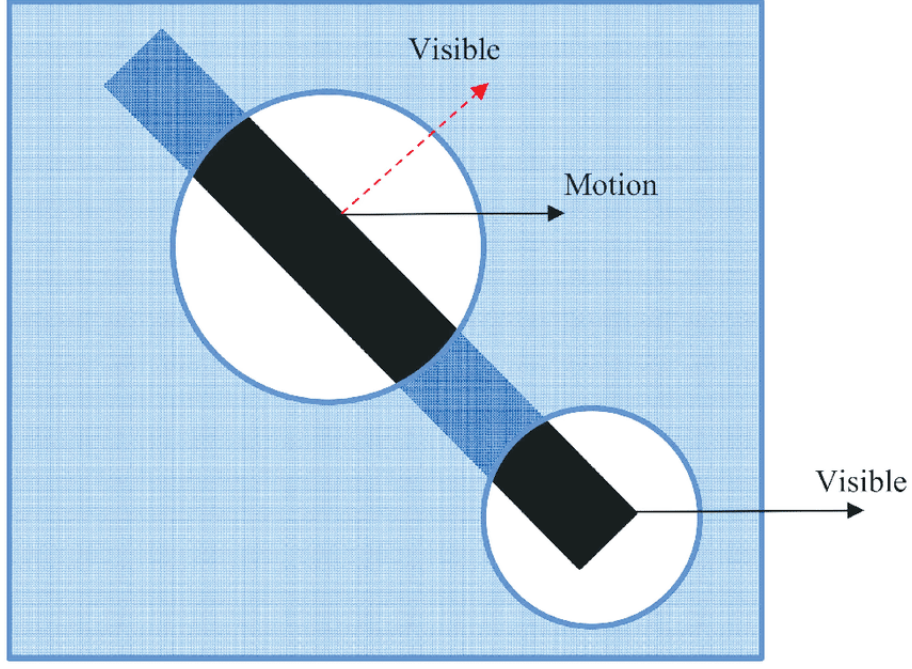


Figure 4.4: Aperture problem visualization [4]

can be mathematically concluded from equation (4.24), which describes the constraint as the equation of a straight line. From this equation the normal vector can be calculated, but the tangential component is not determined. This shows, that the equation is under constrained [23].

Having a region being locally poorly textured, for example a slanted edge with homogenous environment can cause problems during the computation, this occurs, when the gradient is prominent in one direction. This problem can be resolved by adding additional constraints to the set of defined equations as if the sudden change along an object is restricted, determining by neighboring 'similar' pixels. Another option to resolve the problem can be to apply only for sparse optical flow determination by defining rich enough points in the picture [24].

As it was described above to get equation (4.22), small displacement assumption should be applied. Although using high resolution pictures results considerably large absolute displacement. In contrary, the higher the resolution is (so more information stored inside the image), the better the estimation of the motion field is. To be able to satisfy the criteria and have a rich enough picture to work with, the pyramidal representation of the algorithm can be used. Let's assume that the 0th level representation is the original, raw image having $n_x^0 \times n_y^0$ dimensions. Any further level can be computed in a recursive

manner using the following formula

$$\begin{aligned}
I^L(x, y) = & \frac{1}{4} I^{L-1}(2x, 2y) + \\
& \frac{1}{8} (I^{L-1}(2x-1, 2y) + I^{L-1}(2x+1, 2y) + I^{L-1}(2x, 2y-1) + I^{L-1}(2x, 2y+1)) + \\
& \frac{1}{16} (I^{L-1}(2x-1, 2y-1) + I^{L-1}(2x+1, 2y-1) + I^{L-1}(2x-1, 2y+1) + \\
& + I^{L-1}(2x+1, 2y+1)) .
\end{aligned} \tag{4.27}$$

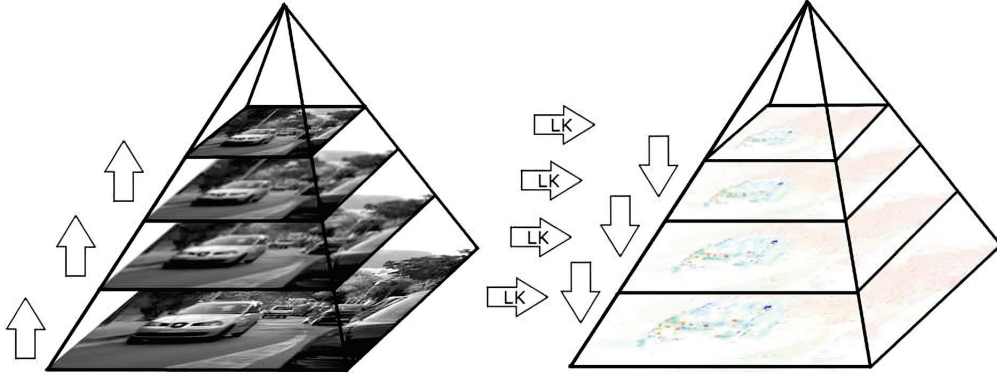


Figure 4.5: Concept of pyramidal LK algorithm [5]

The usual values of for the recursion deepness is 2, 3, 4. The effect of the pyramidal concept can be measured by this variable. If the maximal pixel difference is $d_{\max}$, then the reachable final displacement can be calculated as

$$d_{\max_{\text{final}}} = (2^m - 1)d_{\max} , \tag{4.28}$$

where m is the recursion deepness. In every step the iteration is carried in a similar scheme to the original Lucas Kanade algorithm. After the initial $\mathbf{u}$ is calculated on the deepest layer, Lucas Kanade algorithm can be applied there and iteratively fashion this using the result of the previous layer as initial value of the consecutive one.

4.5 Dense optical flow algorithms

The above introduced scheme can work reliably for motion estimation if and only if the rich points are able to describe the motion sufficiently. Although finding these points can be a computationally expensive operation correspondingly. Using dense optical flow estimation can give another option to gain optical flow information. A well-spread model of this kind is the so-called "Farneback method" [25].

The subsection below is based on [26]. The algorithm uses similar assumptions to the Lucas-Kanade algorithm, predicting the optical flow from two consecutive images.

The main idea in this method is based on the linear expansion of the so-called 3D orientation tensor, which can be give information about the local orientation. The method approximate some neighborhood of each pixel by a quadratic polynomial in the following form

$$f(\mathbf{x}) \sim \mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c, \quad (4.29)$$

where $\mathbf{A}$ is a symmetric matrix, $\mathbf{b}$ is a vector and c is a scalar. The presented coefficients can be calculated by the weighted least square fit of the neighborhood signals. The weighted logic consists of two components, certainty and applicability. Certainty is a metric of confidence in the signal value at each point, for example close to the edge, where edge effects would occur, so some pixels from outside the image would have been taken into account. To eliminate this effect, the certainty of each points outside the boundary of the image is set to zero. Similarly, the applicability represents confidence for the neighborhood, so the significance of each neighboring cell is indicated here. In orientation estimation the natural idea is creating an isotropic applicability weighting, to avoid orientation based bias. Usually the center has the largest influence on the evaluated value and effect to the value decreases radially [27].

The result of displacement can be observed by the following formula

$$f_2(\mathbf{x}) = f_1(\mathbf{x} + \mathbf{d}), \quad (4.30)$$

$$\mathbf{x}^T \mathbf{A}_2 \mathbf{x} + \mathbf{b}_2^T \mathbf{x} + c_2 = \mathbf{x}^T \mathbf{A}_1 \mathbf{x} + (\mathbf{b}_1 + 2\mathbf{A}_1 \mathbf{d}) \mathbf{x} + \mathbf{d}^T \mathbf{A}_1 \mathbf{d} + \mathbf{b}_1^T \mathbf{d} + c_1. \quad (4.31)$$

If $\mathbf{A}_1$ is non-singular

$$\mathbf{d} = \frac{1}{2} \mathbf{A}_1^{-1} (\mathbf{b}_2 - \mathbf{b}_1). \quad (4.32)$$

To create a more realistic estimation method, the global polynomial expansion can be replaced by a local approximation. Though ideally $\mathbf{A}_1(\mathbf{x}) = \mathbf{A}_2(\mathbf{x})$, but as a more practical approach the average of the two coefficient matrices can be introduced

$$\mathbf{A}(\mathbf{x}) = \frac{\mathbf{A}_1(\mathbf{x}) + \mathbf{A}_2(\mathbf{x})}{2}. \quad (4.33)$$

To have a simpler constraining equation

$$\Delta \mathbf{b}(\mathbf{x}) = \frac{1}{2} (\mathbf{b}_2(\mathbf{x}) - \mathbf{b}_1(\mathbf{x})). \quad (4.34)$$

So the local displacement can be estimated as

$$\mathbf{A}(\mathbf{x}) \mathbf{d}(\mathbf{x}) = \Delta \mathbf{b}(\mathbf{x}). \quad (4.35)$$

The displacement field can be computed in a point wise manner afterwards, though in practice the resultant field is too noisy. Taking the assumption, that there are only

small changes spatially in the displacement field, formally the following difference should be minimized

$$\sum_{\Delta \mathbf{x} \in I} w(\Delta \mathbf{x}) \|\mathbf{A}(\mathbf{x} + \Delta \mathbf{x}) \mathbf{d}(\mathbf{x}) - \Delta \mathbf{b}(\mathbf{x} + \Delta \mathbf{x})\|^2. \quad (4.36)$$

From this equation the newly obtained displacement field

$$\mathbf{d}(\mathbf{x}) = \left(\sum_{\Delta \mathbf{x} \in I} \mathbf{A}(\mathbf{x} + \Delta \mathbf{x})^T \mathbf{A}(\mathbf{x} + \Delta \mathbf{x}) \right)^{-1} \sum_{\Delta \mathbf{x} \in I} w(\Delta \mathbf{x}) \mathbf{A}^T(\mathbf{x} + \Delta \mathbf{x}) \Delta \mathbf{b}(\mathbf{x} + \Delta \mathbf{x}). \quad (4.37)$$

The given solution is unique unless the whole neighborhood is exposed to the aperture problem. Robustness can be improved by using a parameterized motion model, for example an eight parameter model as follows

$$\mathbf{d}(\mathbf{x}) = \mathbf{S}(\mathbf{x}) \mathbf{p}, \quad (4.38)$$

where

$$\mathbf{S} = \begin{bmatrix} 1 & x & y & 0 & 0 & 0 & x^2 & xy \\ 0 & 0 & 0 & 1 & x & y & xy & y^2 \end{bmatrix} \quad (4.39)$$

and

$$\mathbf{p} = [a_1 \quad a_2 \quad a_3 \quad a_4 \quad a_5 \quad a_6 \quad a_7 \quad a_8]^T. \quad (4.40)$$

So from Eq. (4.37) the equation can be reformulated as

$$\mathbf{p} = \left(\sum_i w_i \mathbf{S}_i^T \mathbf{A}_i^T \mathbf{A}_i \mathbf{S}_i \right)^{-1} \sum_i w_i \mathbf{S}_i^T \mathbf{A}_i^T \Delta \mathbf{b}_i. \quad (4.41)$$

Though this estimation can be used quite effectively to determine the displacement field, errors occur if large displacement are presented. By incorporating prior knowledge from the previously determined frames, the error can be reduced. Instead of using the same area on both images an educated guess can be taken by predicting the current location of the given points. In case of the later signal, the recent location can be evaluated by

$$\tilde{\mathbf{x}} = \mathbf{x} + \tilde{\mathbf{d}}(\mathbf{x}), \quad (4.42)$$

where $\tilde{\mathbf{d}}(\mathbf{x})$ is the prior displacement rounded to integer, representing whole pixel translation, $\tilde{\mathbf{x}}$ is the estimated location of the specific point. The coefficient parameters can be refined as

$$\mathbf{A}(\mathbf{x}) = \frac{\mathbf{A}_1(\mathbf{x}) + \mathbf{A}_2(\tilde{\mathbf{x}})}{2}. \quad (4.43)$$

To have a simpler constraining equation

$$\Delta \mathbf{b}(\mathbf{x}) = \frac{1}{2}(\mathbf{b}_2(\tilde{\mathbf{x}}) - \mathbf{b}_1(\mathbf{x})) + \mathbf{A}(\mathbf{x}) \tilde{\mathbf{d}}(\mathbf{x}). \quad (4.44)$$

Having an assumption of a zero prior displacement field, iterative scheme can be considered using dense optical flow estimation. Advantage of this approach that all the polynomial expansion coefficients should be evaluated only once, but on the other hand,

the algorithm is quite sensitive to the initial displacement settings. Having considerably large displacement there will ruin the further estimation. The above introduced pyramidal scheme can be applied here similarly, where the displacement is determined in different scales. The drawback here is that the polynomial expansion should be recomputed multiple times, which is a computationally expensive operation.

Since the resultant optical flow is dense, representing the motion by arrows cannot be visualized. Another way is representing the velocity vectors for each pixel by the HSV (Hue-Saturation-Value) scale. Where the magnitude of the velocity is given by the value parameter and direction by hue. This way, the velocity directions are represented as colors. In case of Figure 4.6 ref shows motion to the right and blue motion to the left.

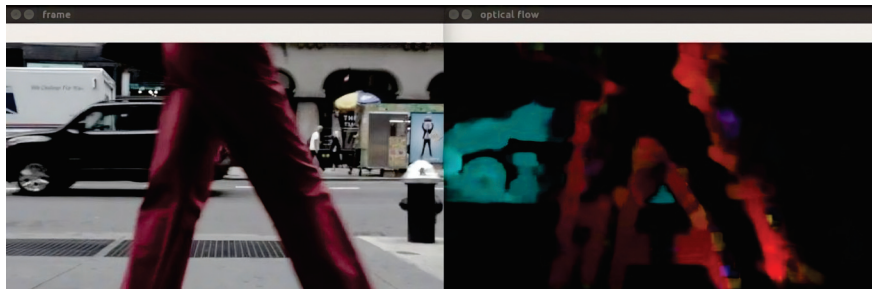


Figure 4.6: Farneback algorithm visualization [6]

4.6 Feature detectors

The subsection is base on source [23]. Observing consecutive frames of a captured video contains a lot of information about the environment. The frames can be examined alone and determine the orientation knowing some constraints about the background, such as originally vertical lines. More over as it is described above, it can be used to estimate the displacement of a specific point. For this application regions with rich source of information have to be used. On the other hand, this method can help alone to determine the orientation difference of two frames, calculating the motion from this data.

To be able to track the motion of a specific point on the image, a feature detector and matcher can be used. The intuitive formulation of measuring the difference between two point on the picture can be determined by

$$E_{\text{WSSD}}(\mathbf{u}) = \sum_i w(\mathbf{x}_i) [I_1(\mathbf{x}_i + \mathbf{u}) - I_0(\mathbf{x}_i)]^2, \quad (4.45)$$

as the weighted summed square difference of two footage, where I_0 and I_1 are the two pictures, $\mathbf{u}$ is the displacement vector and $w(\mathbf{x})$ of a spatially varying weightening window. But most of the cases' information about the exact position of the same feature in both images is not available, this is what should be evaluated from the two images. Since it is not available, another option is to compute how stable the used metric in the case of some small $\delta\mathbf{u}$ displacements on the first image, this is called as auto-correlation function, formulated as

$$E_{AC}(\Delta \mathbf{u}) = \sum_i w(\mathbf{x}_i) [I_0(\mathbf{x}_i + \Delta \mathbf{u}) - I_0(\mathbf{x}_i)]^2. \quad (4.46)$$

This metric is stable if it has a well-defined local minimum on the location of the examined feature. Using polynomial Taylor-series expansion of $I_0(\mathbf{x}_i + \Delta \mathbf{u}) \approx I_0(\mathbf{x}_i) + \nabla I_0(\mathbf{x}_i) \cdot \Delta \mathbf{u}$, this way the formula can be reduced as follows

$$E_{AC}(\Delta \mathbf{u}) \approx \Delta \mathbf{u}^T \mathbf{A} \Delta \mathbf{u}, \quad (4.47)$$

where $\mathbf{A}$ auto-correlation matrix can be represented as

$$\mathbf{A} = w \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}. \quad (4.48)$$

The inverse of $\mathbf{A}$ matrix provides a lower bound on the uncertainty in the location. Using this matrix, an eigenvalue analysis can be carried out, which results two eigenvalues (λ_0 and λ_1) and directions. The larger uncertainty belongs to the smaller eigenvalue ($\lambda_0^{-\frac{1}{2}}$), finding a local maxima to λ_0 can result a better location to the features. Besides λ_0 minimum eigenvalue, multiple other metrics was developed to quantify the structure of the $\mathbf{A}$ matrix. As

$$\det\{(\mathbf{A})\} - \alpha \text{trace}(\mathbf{A})^2 = \lambda_0 \lambda_1 - \alpha (\lambda_0 + \lambda_1)^2, \quad (4.49)$$

which is a rotationally invariant metric which down weights edge-like features in the image. A similar hyperbolic effect can be resulted by computing the harmonic mean as

$$\frac{\det(\mathbf{A})}{\text{trace}(\mathbf{A})} = \frac{\lambda_0 \lambda_1}{\lambda_0 + \lambda_1}, \quad (4.50)$$

which is smoother around $\lambda_1 \approx \lambda_0$. The basic method of all this algorithm can be formulized as follows.

1. Compute the horizontal and vertical derivative of the image using derivatives of Gaussian operator.
2. Compute the three component of the $\mathbf{A}$ auto-correlation matrix.
3. Convolved the images using a larger Gaussian kernel.
4. Calculate a measure of interest in one of the above described metrics.

A problem in connection with the local maxima search is corresponding to the fact that in case of higher contrast regions denser region will be available. The uneven distribution leads to high dependence of a specific section of the picture which is not desirable. To loosen this dependence the key points can be selected, which has higher significance than its neighbors.

In many situations localizing a key point to a fine feature may not be able to keep its properties over time. Some details are changing in time and may not be able to capture them over a specific time. A solution of the problem can be solved by a multiple resolution pyramidal structure and the matching features at the same level. This method is applicable only if the features are not go through significant changes. Most of the application cases this cannot be constrained, so another solution is required. A more generalizable solution to the problem is searching for key points, which are stable both locally and in scale. An early solution to the problem is selecting scale invariant features by seeking for extrema of Laplacian of Gaussian. A similar approach has appeared using Difference of Gaussian (DoG) operator as indicator function, which rejects the pixel if strong asymmetry in the local curvature is observed. Mathematically speaking from the Difference matrix $\mathbf{D}$, the Hessian matrix can be computed

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}. \quad (4.51)$$

A key point is rejected if the following inequality is applied

$$\frac{\text{trace}(\mathbf{H})^2}{\det(\mathbf{H})} > 10. \quad (4.52)$$

After scale invariance is resolved, the following task is ensured that the feature can be recognized in case of in-plane rotations. Instead of creating a rotationally invariant descriptor, a better approach in practice is to determine the dominant orientation of each key points. An easy way to determine this metric is two average out the gradient of a specific region around a key point. But sometimes the signed average can result only small orientation values which are hard to distinguish to each other, this way the result is unreliable. A more reliable technique is to observe the histogram of the orientations, which can be towards modified by weighting by the magnitude of the gradients.

A more general detector can be created by reaching not only orientation but affine invariance, which includes furthermore the affine deformations such as perspective foreshortening, but other continuous warping can be well approximated by affine deformations. A method to calculate affine variance is to fit an ellipse around the object and then performing operators dependent on the principle axes and ratios of the fit. Another approach is introduced as maximally stable external region (MSER) detector. The method can be used only in case of gray scaled images by observing all the gray levels of the image. The algorithm first sorts the pixels by gray level and then incrementally adding new gray level two the picture, while all connected components areas are monitored simultaneously. The region, which rate of change is minimal as the function of the threshold limit will be defined as the maximally stable region.

SIFT detector

The content of this subsection is based on [28]. The Scale Invariant Feature Transform, abbreviated as SIFT holds a lot of in common with the above described method to detect

features on an image. The first step of the algorithm is the so-called Scale space extreme detection it searches by the DoG method to identify points of interest which are scale and orientation invariant

$$\text{DoG} = (s - 1) \underbrace{\sigma^2 \nabla^2 G(x, y, \sigma)}_{\text{NLoG}}^{\text{LoG}}. \quad (4.53)$$

Since the derivative represents the changing rate of a signal, its magnitude depends on the used scale, in general the value decreases with scale. To compensate this effect σ^2 can be used[29].

The detector look at the image in distinct scales and calculated the DoG for every two consecutive image scale. After that it searches for an extreme of the absolute values of the given grid on a specific scale. All the extrema are corresponding to a specific point and with a scale value, which represents the size of the feature. The weak key points can be removed from the resultant set.

The algorithm shows how SIFT features are scale invariant. Orientation invariance applies by determining a well-defined area around the feature, which belongs to the magnitude of the extrema. Inside this square area, all the gradients are calculated, and only the orientation is taken into account due the magnitude changes by scale. From the orientations with the help of a histogram, the principal orientation can be determined, by choosing the orientation with the highest occurens.

After the features are detected in two successive images, the following task is to match them together. To be able to make the matching, a unique descriptor of the image should be evaluated. In case of SIFT descriptor the orientation histogram can help, by determining for each quadrant of the image and concatenate them. Using only orientation is beneficial since it is not only invariant to rotation and scale, but other effect such as illumination as well. The histograms ($H_n(k)$) can be matched by determining various metrics, as L2 distance

$$d_{L2}(H_1, H_2) = \sqrt{\sum_k (H_1(k) - H_2(k))^2}. \quad (4.54)$$

In this case the smaller the distance between the two images the better the match, lower bound of the metric is zero, which represents perfect match. But it can be performed by normalized correlation too

$$d_{\text{correl}}(H_1, H_2) = \frac{\sum_k [(H_1(k) - \bar{H}_1)(H_2(k) - \bar{H}_2)]}{\sqrt{\sum_k (H_1(k) - \bar{H}_1)^2} \sqrt{\sum_k (H_2(k) - \bar{H}_2)^2}}, \quad (4.55)$$

where

$$\bar{H}_n = \frac{1}{N} \sum_{k=1}^N H_i(k). \quad (4.56)$$

The larger the metric is, the better the match is. Due to it is normalized by the histograms energies, the maximal number of this match is 1, which represents the perfect match case.

Another useful metric uses the minimum value of each bins in the following form

$$d_{\text{comp}}(H_1, H_2) = \sum_k \min(H_1(k), H_2(k)). \quad (4.57)$$

The larger this metric is, the better the match is as it computes the overlaps between the two histograms.

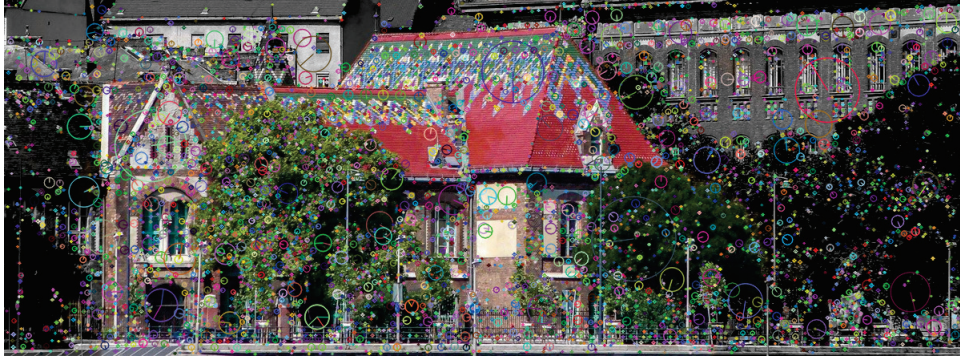


Figure 4.7: Building of Department of Applied Mechanics after applying Sift feature detector. Original image source is [3]

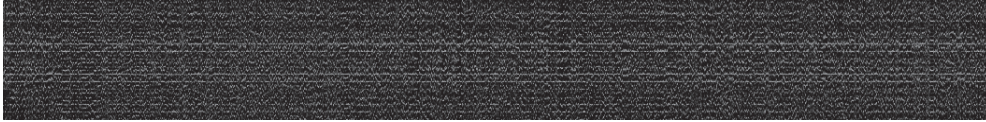


Figure 4.8: Descriptor of Sift features of Figure 4.7

5 Development of angle measurement algorithms

After the mechanics of the pendulum is introduced in Section 3.2 and the design is presented in Section 2.3, the following task is to apply the described introduced in Section 4. The algorithm is designed in C++, using OpenCV library [30] to obtain optimal implementations of the algorithms. Meanwhile, to plots was created by the help of python and Wolfram Mathematica. The test of the algorithm is done by three distinct images, depending on the specific method used.



Figure 5.1: Reference rectangle object

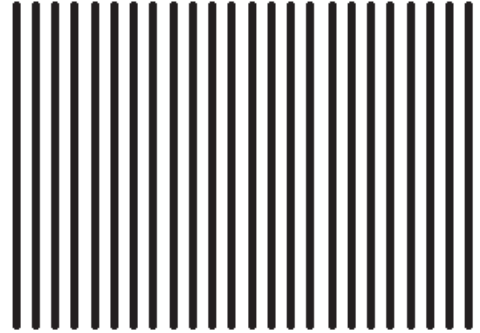


Figure 5.2: Reference object with dense vertical information



Figure 5.3: Building of Department of Applied Mechanics [3]

5.1 Hough transformation

One of the most trivial algorithm ideas for angle detection is determining the straight lines on the image. Having prior knowledge from the environment can help to select the ones, which can be defined as reference objects on the image. This kind of knowledge can be included by predefining an easily trackable object, like a rectangle as it is shown in Figure 5.1. The rectangle has two well-defined vertical edges, which can be used for angle detection. As it was described in Section 4.1, it is advantageous from the side of the algorithm, if the lines in the picture are detected by angle and distance from the origin in the parameter space. Using this method, the orientation of the spotted lines are detected.

This method is restricted only to a small fragment of the image, where the reference object can be seen. Since before Hough transformation an edge detecting algorithm should be used and the program eventually works with all the given detected points to find lines in the image. Working with the whole image is computationally expensive and creates the opportunity to sample lines with no prior knowledge about their orientation. The occurring problems can be resolved by cropping the image to a certain region of the image and tracking it simultaneously.

In the recent algorithm, the region of interest can be selected by the user. After the desired segment of the picture is determined, the two edges of the reference rectangle can be detected. By the knowledge about the position of the two edges the cropped area can be modified in a manner to keep the object inside the region of interest. If less than two edges of the rectangle can be determined and the region of interest can be extended, so creating the opportunity finding the edges in a larger scope. The major weakness of the algorithm is that if the reference is out of the scope, no prior information is available, no control is possible in the future.

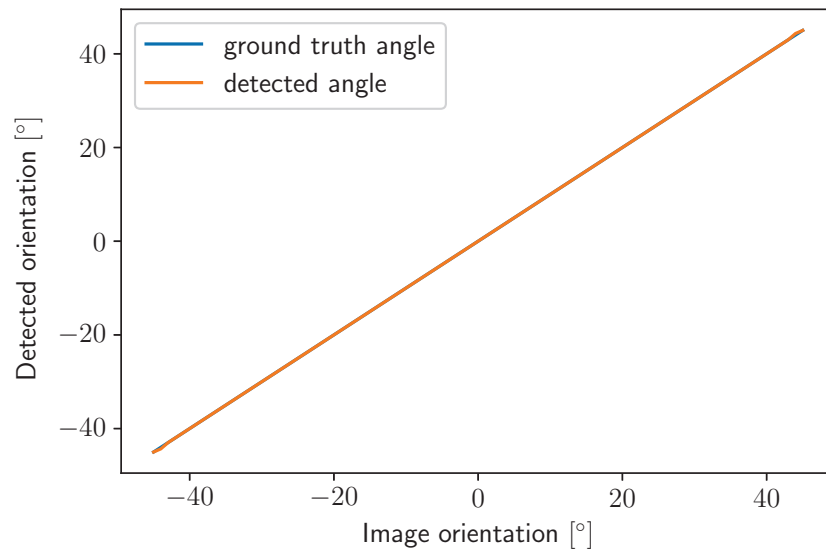


Figure 5.4: Detected angle by Hough transform

5.2 Sweeping along the image

A similar idea to the previous algorithm is using the same reference rectangle, creating a region of interest in the user and then determining the angle only by using spare information of the remained area. This can be done, by creating a reference object with high contrast, as a dark rectangle (Figure 5.1) and light background. The image can be afterwards setting a threshold to distinguish the two regions from each other. Since the brightness and lighting conditions significantly influence the information about two regions, it is advisable to let the user define these options by the help of a GUI (graphical user interface) before the exact measurement, this way ensures that the algorithm is

applicable in the present case.

In the previous algorithm, dense information from the specific segment of the image is used, by examining all the edges and lines in the image. Another occurring possibility is using convolution by a predefined kernel, using out that the examined object is a rectangle with significant vertical edges. It can be assumed that rectangle is placed in the middle of the image after cropping, so the first search can be placed there. A convolution from both the left and right sides of the image can be propagated towards the middle region of the picture. The two kernel sweep is stopped, when the kernel detects a point with enough information to be able to assign it as part of the rectangle. This detection can be carried out by assuming a weightening by the convolution. If the value of the results exceeds a predefined threshold value, then the point is saved. After sweeping in the middle of the image, the vertical side can be divided into n parts and finding the rectangle in all height. As a rule of thumb, the balancing of the pendulum can be called unsuccessful after the pendulum angle reaches 45° or -45° , so in case of all angles in that region, the significantly long edges of the rectangle remains. If the procedure of searching the rectangle at distinct height is performed in a exploring manner starting from the middle, the first few points are surely determined by the desired edge. After the middle one-third region of the image was observed, a notable change can show that this point is not connected to the desired edge. This case the algorithm stops. After a set of data points is gathered for both left and right edges, the following task is to fit a line to the data points. Because the outliers are avoided by the stop after the significant change occurs, and it can be ensured that the first few points are able to track the rectangle, a line fit by the least-square error can be applied.

The least-square method aims to find the parameters of a specific function in order to minimize the square of the error of the points. The equation of the line is

$$y = f(x, m, b) = mx + b, \quad (5.1)$$

where m determines the slope and b the offset of the line at $x = 0$ position.

The error function can be parametrized by the fitting parameters as

$$E(m, b) = \sum_i (y_i - f(x_i, m, b))^2, \quad (5.2)$$

which calculates the square of the error at every sampled point. The partial derivatives of the error function should give zero at the minimum, so the two equations are

$$\frac{\partial E}{\partial m} = \sum_i -2(y_i - mx_i - b)x_i = 0 \quad (5.3)$$

$$\frac{\partial E}{\partial b} = \sum_i -2(y_i - mx_i - b) = 0. \quad (5.4)$$

The stationary point determined by (m, b) is local minimum, if the Hesse-matrix $\mathcal{H}$ is positive definite, where $\mathcal{H}$ is

$$\mathcal{H}(m, b) = \begin{bmatrix} \frac{\partial^2 E}{\partial m^2} & \frac{\partial^2 E}{\partial m \partial b} \\ \frac{\partial^2 E}{\partial b \partial m} & \frac{\partial^2 E}{\partial b^2} \end{bmatrix} = \begin{bmatrix} \sum_i 2x_i^2 & \sum_i x_i \\ \sum_i x_i & \sum_i 1 \end{bmatrix} \quad (5.5)$$

The first principal minor is trivially positive

$$\Delta_1 = \sum_i 2x_i^2 > 0 \quad (5.6)$$

while the second principle minor is

$$\Delta_2 = 2 \sum_i x_i^2 \sum_i 1 - (\sum_i x_i)^2 > \sum_i x_i^2 \sum_i 1^2 - (\sum_i x_i)^2 (\sum_i 1)^2 > 0, \quad (5.7)$$

which is true due to the Cauchy-Schwarz-Bunyakovskij inequality.

From the two equations the slope and offset along the y axis are

$$m = \frac{\sum_i x_i y_i \sum_i 1 - \sum y_i \sum x_i}{\sum x_i^2 \sum 1 - (\sum x_i)^2}, \quad (5.8)$$

$$b = \frac{\sum_i x_i^2 \sum_i y_i - \sum y_i \sum x_i}{\sum x_i y_i \sum 1 - (\sum x_i)^2}. \quad (5.9)$$

From the slope the orientation can be determined by taking the arctangent

$$\theta \approx \text{atan} 2(m). \quad (5.10)$$

But tangent function is not determined in case of 90° , due to $\tan \rightarrow \infty$ as $\theta \rightarrow \pi$. This value can be approximated by a large number.

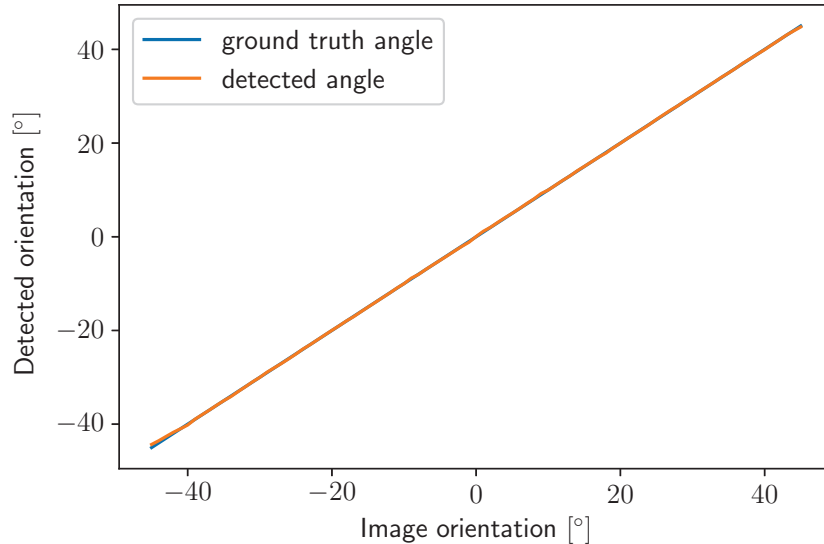


Figure 5.5: Detected angle by sweep

As it can be seen in Figure 5.5 the ground truth and detected angles are close to each other, only difference can be observed by bare eyes at the endings. To get such a fine

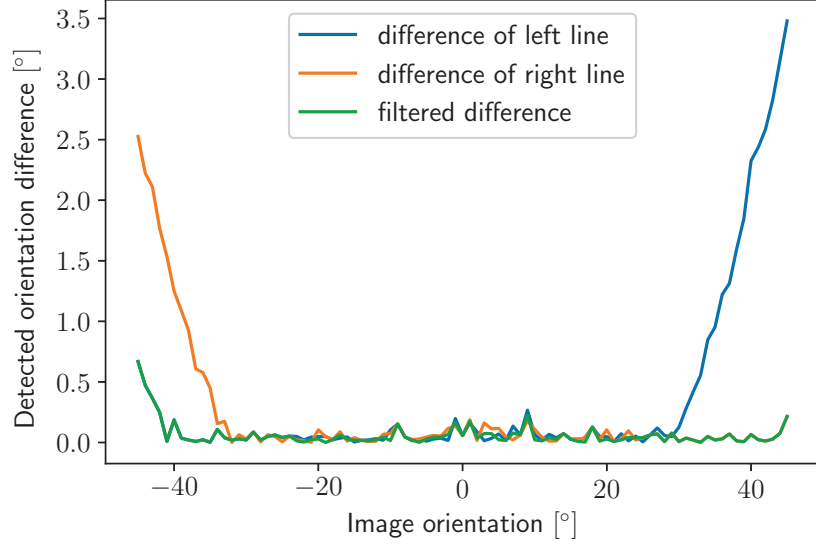


Figure 5.6: Absolute angle difference by sweep method

result, a filter should be applied to the left and right side, because if the rectangle is tilted after a specific angle, one side is unable to give fine results as it can be seen in Figure 5.6. The applied filter is

$$\begin{aligned}
 \varphi_{\text{detected}} &= \varphi_{\text{left}}, \text{ if } \frac{\varphi_{\text{left}} + \varphi_{\text{right}}}{2} < -25^\circ \\
 \varphi_{\text{detected}} &= \frac{\varphi_{\text{left}} + \varphi_{\text{right}}}{2}, \text{ if } -25^\circ \leq \frac{\varphi_{\text{left}} + \varphi_{\text{right}}}{2} \leq 25^\circ \\
 \varphi_{\text{detected}} &= \varphi_{\text{right}}, \text{ if } \frac{\varphi_{\text{left}} + \varphi_{\text{right}}}{2} > 25^\circ
 \end{aligned} \tag{5.11}$$

The weakness of the algorithm is similar to the Hough transformation detection, the whole process is developed by the assumption of the given reference object is available. So the algorithm is sensitive to noise. The boundary of the crop can be tracked and updated in every step, if the rectangle is too close to the border.

5.3 Sobel kernels

Using a reference rectangle for angle detection is a simple idea, but problems can arise in two distinct fields. In one hand, having only a specific reference object in the image can cause problems. As it is described in Section 3.2, using velocity estimation by finite difference method creates a unit characteristic multiplier, which affects the cyclic coordinate, by applying a constant angular velocity in stationary case. If the constant angular velocity is large, the camera can lose the reference object. On the other hand, using a simplest structure like that contains only a small amount of information, having some noise in the system can ruin the measurement completely.

The Sobel kernel detection algorithm aims to solve the latter problem, by using more information from a reference objective as it can be seen in Figure 5.2. Sobel kernels are popular in the field of vertical and horizontal angle detection. The kernels of these directions are given in Equation (4.9). The idea of the algorithm is creating a reference objective with more dense information as a set of vertical stripes. The detection is carried out, by observing the result of two orthogonal kernels. The resultant image of both kernels are binary thresholded, so only white and black regions are remained. The idea is, that as the camera rotates the strength of the two orthogonal directions change continuously. The captured image can be modelled by using a fixed camera and rotating environment around it. By thresholding the two images and measuring the number of remained information in case of both kernels, a change can be observed. By subtracting the results of the kernels leads to a unique description of the specific angle values. In a mathematical sense it can be represented as finding, how many points on the image the derivative along one direction is larger than a limit. By subtracting one derivative from the other remove the common information out of the metrics. As the experiments has shown, in case of vertical and horizontal Sobel kernels the resultant statistics are identical, the same points vanishes for both kernels and reflection symmetry occurs around 0° .

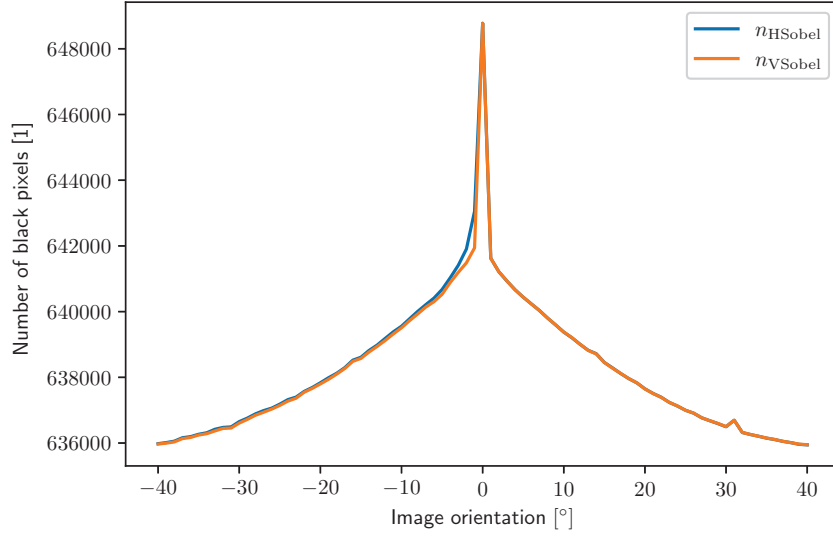
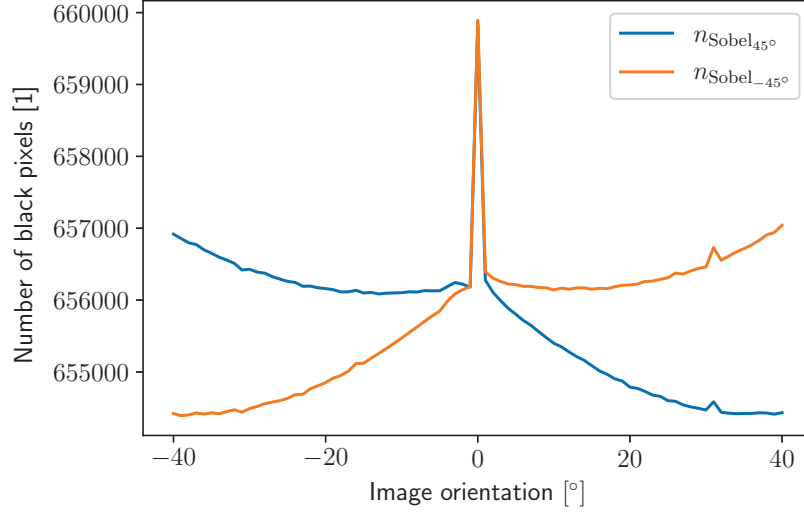


Figure 5.7: Effect of vertical and horizontal kernels applying on the image

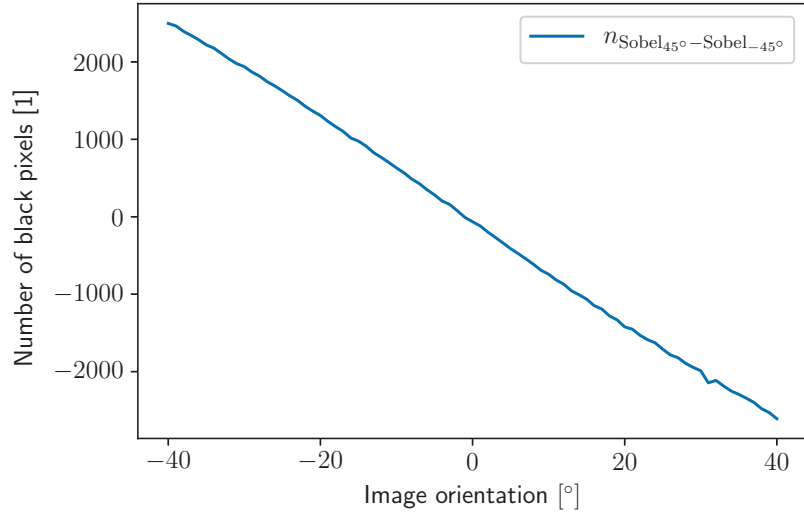
The problem can be resolved by using orthogonal kernels of 45° and -45° . This way reflection symmetry disappears and the difference of the two kernels can be computed. The kernels searching for diagonal directions are

$$K_{\text{Sobel}_{45^\circ}} = \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}, \quad K_{\text{Sobel}_{-45^\circ}} = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}. \quad (5.12)$$

The following occurring problem is that the number of pixels are dependent on the size of the image. As the camera gets closer and further from the reference objective, the region



(a) Effect of diagonal kernels applying on the image



(b) Difference of the two applied diagonal kernels

Figure 5.8: Sobel kernel approach, creating a linear filter for angle detection

of interest shrinks. The dependence on the shrinkage can be plotted as a surface plot. After the surface is plotted, shown in the Appendix in Figure 6.1, a seemingly continuous change of the slope and offset value can be observed. This values can be evaluated by fitting square root function for slope and offset as a function of shrinkage rate, as

$$m(s) = m_m \sqrt{s} + b_m, \quad b(s) = m_b \sqrt{s} + b_b, \quad (5.13)$$

where s is the shrinkage rate, m_m, m_b and b_m, b_b are the slope and offset values according to the function of linear fit m, b values, respectively, shown in the Appendix Figure 6.2 and Figure 6.3. The fitting can be carried out by least-square fit, with a similar form to the linear fit

$$m_m = \frac{\sum_i \sqrt{s_i} m_i \sum_i 1 - \sum m_i \sum \sqrt{s_i}}{\sum s_i \sum 1 - (\sum \sqrt{s_i})^2}, \quad (5.14)$$

$$b_m = \frac{\sum_i s_i \sum_i m_i - \sum m_i \sum \sqrt{s_i}}{\sum \sqrt{s_i} m_i \sum 1 - \sum s_i}. \quad (5.15)$$

$$m_b = \frac{\sum_i \sqrt{s_i} b_i \sum_i 1 - \sum b_i \sum \sqrt{s_i}}{\sum s_i \sum 1 - (\sum \sqrt{s_i})^2}, \quad (5.16)$$

$$b_b = \frac{\sum_i s_i \sum_i b_i - \sum b_i \sum \sqrt{s_i}}{\sum \sqrt{s_i} b_i \sum 1 - \sum s_i}. \quad (5.17)$$

This method is able to determine the angle of orientation with a considerably large error, but on the other hand it is more robust against occurring noise.

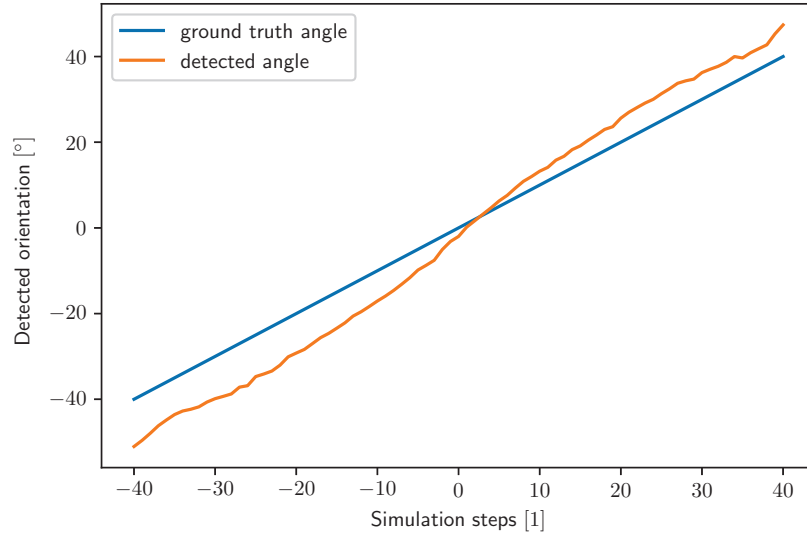


Figure 5.9: Detected angle by sobel kernel detector

5.4 Vertical feature detection

Determining the orientation is not only important in case of robotic tasks as balancing a pendulum, but other industrial applications as well such as forestry machine tilt angle detection. In case of these machines the tilt angle have significant effect, because it can use to indicate danger, if the slope under the machine is considerable large. There are distinct type of detectors for this application such as pendulum tilt indicator, or electronic tilt meters, which uses bubble level principle to sense the angle. The nowadays used gravity sensors, such as accelerometers and gyroscopes are influenced by vibration-induced errors. Using camera can be a solution against this kind of problems, investigating the environment and determining the tilt angle by static measurements. The peculiarity of the

task in a forestry application is that many vertical references are available for the camera as the trees and urban environments. A method for camera tilt angle determination is given by [31]. Since the method is applicable in a laboratory environment too, this method can be used in case of Furuta pendulum balancing as well. The algorithm is tested with Figure 5.2 as much vertical edges is located there.

The algorithm proposed in the article uses the following procedure. On a gray scaled image Sobel edge detection is applied on vertical direction. Afterwards the image is rotated by angles in the region of interest (as the algorithms should be comparable, the angle is between -45° and 45°), meanwhile the image is cropped to maintain a constant image size. The image is analyzed column-by-column and the number and length of continuous vertical lines are counted. A line is continuous if it is interrupted by a limit non-vertical pixels. The limit in case of the paper is 2 pixels. The length of the lines are stored. In the post-processing step the angle with the longest vertical lines is selected. This shows that rotation by this angle transforms the image to its vertical position. This method is robust in case of many reference vertical lines are available from the environment.

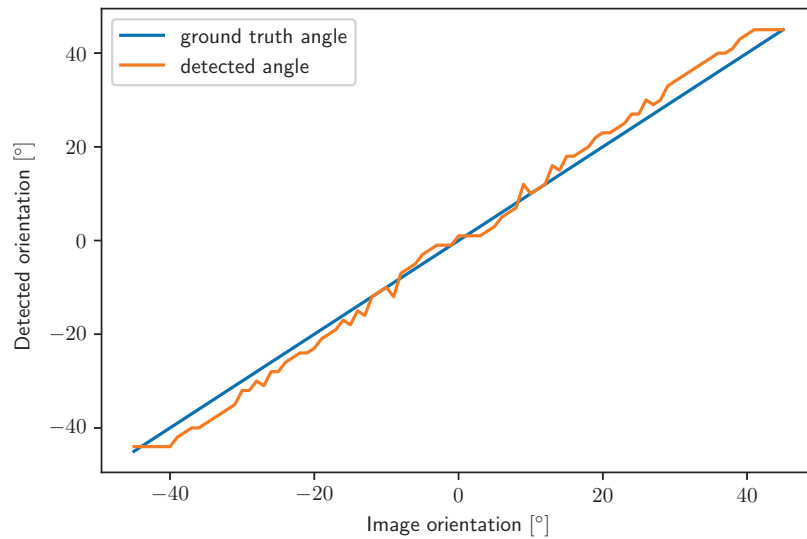


Figure 5.10: Detected angle by vertical feature detector

5.5 Sift feature detectors

Sift detectors can be used by itself for orientation detection, by having a set of reference images from the environment. While the balancing task the captured images can be matched by the features of the reference images. After that the key points can be examined by a descriptor, the principal axis of the features in the two images can be compared and their difference indicates the orientation change of the camera, where prior knowledge about the reference picture is available.

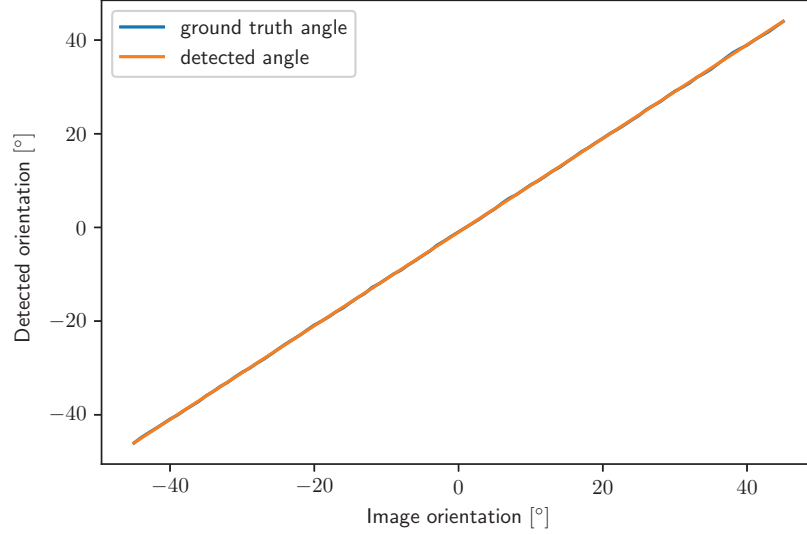


Figure 5.11: Detected angle by Sift feature detector

5.6 Optical flow

The optical flow aims measuring the velocity of specific points of the picture. As rigid motion can be assumed, the connection between the velocity of selected points of the image can be calculated as

$$\mathbf{v}_B = \mathbf{v}_A + (\boldsymbol{\omega}) \times \mathbf{r}_{A \rightarrow B}. \quad (5.18)$$

The visualized resultant image in case of rotating a segment of Figure 5.3 is the following. From the field angular velocity can be calculated. By the integral of the ω the coordinate can be calculated, having a prior knowledge about the starting position. This method is not further analyzed, due to the fact that for exact measurement the pendulum device is needed to determine the connection of angle change in the image and in real life.



Figure 5.12: Farneback algorithm result by rotating Figure 5.3

5.7 Comparison of methods

The abbreviated columns are the following: MAD - Mean Absolute Difference, MD - Maximal Difference, AVP - Average Process Time, DPT - Deviation of Process Time

Method	Resolution	MAD in $[-45^\circ, 45^\circ]$	MD $[-45^\circ, 45^\circ]$	MAD in $[-5^\circ, 5^\circ]$	MD in $[-5^\circ, 5^\circ]$	APT [ms]	DPT [ms]
Hough transformation	160×120	-	-	-	-	0.59501	0.25226
Hough transformation	320×240	-	-	-	-	1.04364	0.34987
Hough transformation	640×480	2.2563	3.5	0.28436	1	2.31715	0.83568
Hough transformation	1280×960	5.6854	45.0	7.2083	45.0	7.6057	1.8472
Sweep method	160×120	46.266	98.9534	43.764	53.849	0.13767	0.07413
Sweep method	320×240	15.704	112.45	0.656958	1.3321	0.63272	0.29553
Sweep method	640×480	2.2924	3.5082	0.41863	0.68490	0.27780	0.11936
Sweep method	1280×960	1.8472	3.4891	0.36703	0.64692	3.1233	0.7288
Sobel kernel method	160×120	7.5345	21.3647	3.3605	9.0801	1.9595	0.8536
Sobel kernel method	320×240	17.178	49.756	3.7459	7.1718	2.1789	0.93354
Sobel kernel method	640×480	2.4088	14.025	0.94204	2.4671	3.853	1.5227
Sobel kernel method	1280×960	6.5721	14.442	1.763	5.2901	10.6563	2.6001

Table 5.1: Comparison of distinct angle detection algorithms

Method	Resolution	MAD in $[-45^\circ, 45^\circ]$	MD $[-45^\circ, 45^\circ]$	MAD in $[-5^\circ, 5^\circ]$	MD in $[-5^\circ, 5^\circ]$	APT [ms]	DPT [ms]
Vertical feature detector	160×120	2.41935	6	1.4167	3	87.214	7.1679
Vertical feature detector	320×240	2.4194	4	1.9167	3	212.97	14.094
Vertical feature detector	640×480	2.2366	4	1.25	2	748.39	34.002
Vertical feature detector	1280×960	3.3226	8	0.75	1	2345.8	42.629
SIFT method	160×120	0.14137	0.49580	0.05671	0.24764	21.5277	2.6724
SIFT method	320×240	0.12141	0.4348	0.07786	0.19554	83.69	7.9114
SIFT method	640×480	0.08023	0.27050	0.07656	0.19120	259.42	12.295
SIFT method	1280×960	0.09763	0.31561	0.072867	0.13370	401.99	21.408

Table 5.2: Comparison of distinct angle detection algorithms

As it can be seen, the Hough transformation is not able to recognize any straight line in case of small resolution. The measurement of the precision of all the algorithms was done by measuring the time needed to calculate the orientation from -45° to 45° . As the most important region for balancing the pendulum is within $[-5^\circ, 5^\circ]$ region, it was measured distinctly. As it could be predicted, in case of larger resolution the precision is also larger, and the time is inversely proportional with this. As the same angular velocity amplitude was applied to the picture all over the simulation it can be seen how well the algorithm can recognize that the angular velocity is constant.

Method	Resolution	Angular velocity deviation	APT [ms]	DPT [ms]
Farneback	160×120	0.000066	5.2774	0.60979
Farneback	320×240	0.000003	10.396	2.5261
Farneback	680×480	0.0000133	82.073	14.988
Farneback	1280×960	0.00002	363.58	93.495

Table 5.3: Effect of resolution for optical flow

6 Summary

During the project work design creation, mechanical modeling and algorithm development were introduced. As the task is interdisciplinary, the careful examination of mechanical engineering, electrical engineering and computer science synergical integration is needed.

As a result, a new design was developed and manufactured, which is able to replace the encoder based approach, as it includes the necessary components for this task as well.

The mechanical modeling of Furuta pendulum and control approach with PD controller was introduced as part of the work, which creates the opportunity to develop new control strategies and fully understand the constraints and requirements of image processing algorithms for Furuta pendulum balance.

A large variety of image processing algorithms were explained and implemented in OpenCV, using the constraints given by modeling and design. New algorithms were created and application possibilities of distinct strategies were observed in case of different resolutions shown in Table 5.1, which can be used in the future as a base of sufficient method selection for angle detection in field of robotics. As it can be seen from the results, all the resultant computation time is below the critical time delay for almost all the possible resolutions. They can open the possibility of testing the balancing with the pendulum.

As a following step, the created new design is planned to be tested in the laboratory, examining the precision of the developed algorithms in application. If the methods create the opportunity, balancing of Furuta pendulum by visual angle detection can also be tested and the results can be compared to the ones with encoder measurement. The possibility of determining not only the tilt angle, but also the cyclic coordinate, arises, so in the future, controlling the pendulum with the help of camera without encoder is also considered.

The successful outcome of the described results can predict that balancing with combined encoder and camera signals can be examined in the future. This method would aim a more realistic approach of human balancing modeling. Both the angle detection and the combined with camera image processing can be tested in the future with neural networks, which are the cutting-edge technology in case of computer vision and their importance is also increasing in control theory.

References

- [1] Mate B Vizi and Gabor Stepan. Digital stability of the furuta pendulum based on angle detection. *Journal of Vibration and Control*, page 10775463221148893, 2023.
- [2] Mate B Vizi, Daniel M Horvath, and Gabor Stepan. Routh reducibility and controllability of unstable mechanical systems. *Acta Mechanica*, 233(3):905–920, 2022.
- [3] URL <https://www.mm.bme.hu/ipari-partnerek>.
- [4] Parvin Zarei Eskikand, Tatiana Kameneva, Michael R Ibbotson, Anthony N Burkitt, and David B Grayden. A possible role for end-stopped v1 neurons in the perception of motion: a computational model. *PloS One*, 11(10):e0164813, 2016.
- [5] Krzysztof Blachut, T Kryjak, and M Gorgon. Hardware implementation of multi-scale lucas-kanade optical flow computation algorithm—a demo. In *Proceedings of the 2018 Conference on Design and Architectures for Signal and Image Processing (DASIP), Porto, Portugal*, pages 10–12, 2018.
- [6] Maxim Kuklin (Xperience.AI). Optical flow in opencv (c++/python): Learnopencv, Jul 2021. URL <https://learnopencv.com/optical-flow-in-opencv/>.
- [7] Gabor Stepan. Delay effects in the human sensory system during balancing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367(1891):1195–1212, 2009.
- [8] Martin Lakie and Ian D Loram. Manually controlled human balancing using visual, vestibular and proprioceptive senses involves a common, low frequency neural process. *The Journal of physiology*, 577(1):403–416, 2006.
- [9] Arthur D Kuo. An optimal state estimation model of sensory integration in human postural balance. *Journal of neural engineering*, 2(3):S235, 2005.
- [10] G. Stépán and L. Kollár. Balancing with reflex delay. *Mathematical and Computer Modelling*, 31(4):199–205, 2000. ISSN 0895-7177. doi: [https://doi.org/10.1016/S0895-7177\(00\)00039-X](https://doi.org/10.1016/S0895-7177(00)00039-X). URL <https://www.sciencedirect.com/science/article/pii/S089571770000039X>. Proceedings of the Conference on Dynamical Systems in Biology and Medicine.
- [11] Samuel Barrios Valero and Péter Galambos. Design and implementation of a furuta pendulum device for benchmarking non-linear control methods.
- [12] Giovanni Legnani, Federico Casolo, Paolo Righettini, and Bruno Zappa. A homogeneous matrix approach to 3d kinematics and dynamics — i. theory. *Mechanism and Machine Theory*, 31(5):573–587, 1996. ISSN 0094-114X. doi: [https://doi.org/10.1016/0094-114X\(95\)00100-D](https://doi.org/10.1016/0094-114X(95)00100-D). URL <https://www.sciencedirect.com/science/article/pii/0094114X9500100D>.

- [13] Giovanni Legnani, Federico Casalo, Paolo Righettini, and Bruno Zappa. A homogeneous matrix approach to 3d kinematics and dynamics—ii. applications to chains of rigid bodies and serial manipulators. *Mechanism and Machine Theory*, 31(5):589–605, 1996. ISSN 0094-114X. doi: [https://doi.org/10.1016/0094-114X\(95\)00101-4](https://doi.org/10.1016/0094-114X(95)00101-4). URL <https://www.sciencedirect.com/science/article/pii/0094114X95001014>.
- [14] Jacques Denavit and Richard S Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. 1955.
- [15] Gy Haller and G Stépán. Micro-chaos in digital control. *Journal of Nonlinear Science*, 6:415–448, 1996.
- [16] Gábor Csernák and Gábor Stépán. Sampling and round-off, as sources of chaos in pd-controlled systems. In *2011 19th Mediterranean Conference on Control & Automation (MED)*, pages 1319–1324. IEEE, 2011.
- [17] S.K. Nayar. Boundary Detection. In *Monograph FPCV-2-2, First Principles of Computer Vision*, Columbia University, New York, Jun 2022.
- [18] S.K. Nayar. Edge Detection. In *Monograph FPCV-2-1, First Principles of Computer Vision*, Columbia University, New York, May 2022.
- [19] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.
- [20] Ron Kimmel and Alfred M Bruckstein. Regularized laplacian zero crossings as optimal edge integrators. *International Journal of Computer Vision*, 53:225–243, 2003.
- [21] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI’81: 7th international joint conference on Artificial intelligence*, volume 2, pages 674–679, 1981.
- [22] J.-Y. Bouguet. Pyramidal implementation of the lucas kanade feature tracker. 1999. URL <https://api.semanticscholar.org/CorpusID:9350588>.
- [23] Richard Szeliski. *Computer vision algorithms and applications*. Springer, London; New York, 2011. ISBN 9781848829343 1848829345 9781848829350 1848829353. URL <http://link.springer.com/book/10.1007%2F978-1-84882-935-0>.
- [24] S.K. Nayar. Lucas-kanade method. In *Video Series of Reconstruction II, First Principles of Computer Vision*, Columbia University, New York, May 2021.
- [25] Gunnar Farneback. Fast and accurate motion estimation using orientation tensors and parametric motion models. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 1, pages 135–139. IEEE, 2000.

- [26] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Image Analysis: 13th Scandinavian Conference, SCIA 2003 Halmstad, Sweden, June 29–July 2, 2003 Proceedings 13*, pages 363–370. Springer, 2003.
- [27] Gunnar Farnebäck. *Polynomial expansion for orientation and motion estimation*. PhD thesis, Linköping University Electronic Press, 2002.
- [28] S.K. Nayar. Sift Detector. In *Monograph FPCV-2-3, First Principles of Computer Vision*, Columbia University, New York, Aug 2022.
- [29] Tony Lindeberg. Feature detection with automatic scale selection. *International journal of computer vision*, 30:79–116, 1998.
- [30] Itseez. Open source computer vision library. <https://github.com/itseez/opencv>, 2015.
- [31] Jaroslav Matej. Determination of forestry machine’s tilt angle using camera and image processing. *Computers and Electronics in Agriculture*, 109:134–140, 2014. ISSN 0168-1699. doi: <https://doi.org/10.1016/j.compag.2014.09.011>. URL <https://www.sciencedirect.com/science/article/pii/S0168169914002257>.

Appendix

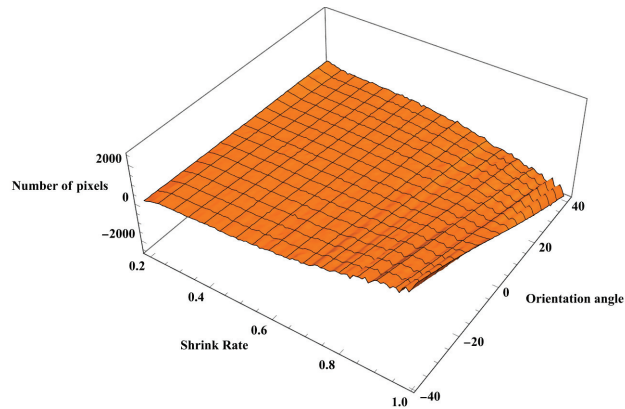


Figure 6.1: Shrinkage dependence of the resultant difference function.

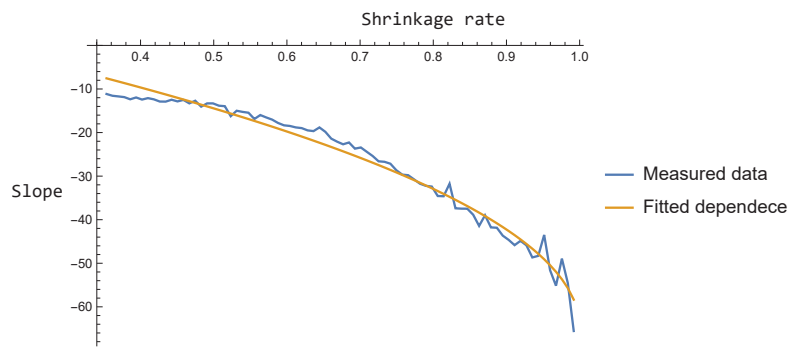


Figure 6.2: Shrinkage rate dependence of the slope.

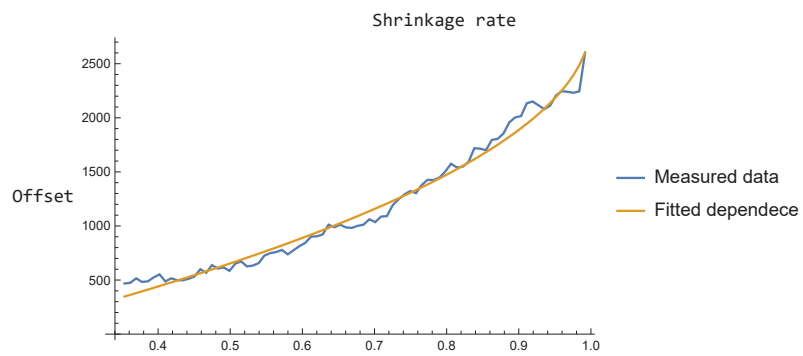


Figure 6.3: Shrinkage rate dependence of the offset.

QTY.	ITEM NO.	Part number	Dimensions	Material	Mass
1	1	Arm shaft	∅10x92	Steel	0.029 kg
1	2	Sleeve casing	∅50x23	Polyamid	0.039 kg
1	3	Back casing	∅50x15	Polyamid	0.028 kg
3	4	Threaded rode	∅6x132	Steel	0.024 kg
1	5	Y guide	40x50x20	Polyamid	0.017 kg
6	6	Plain washer	4	Steel	0.000 kg
1	7	Ball bearing	19x10x4	Steel	0.000 kg
1	10	Ball bearing	15x10x4	Steel	0.000 kg
1	13	Spacer	∅12x10	Polyamid	0.001 kg
2	14	Seeger ring	10	Steel	0.000 kg
2	15	Cable gland	M8x1	Steel	0.002 kg
1	16	Rode-cable gland connecting part	∅12x10	Polyamid	0.003 kg
1	17	Upper casing layer	∅50x1	FR4	0.003 kg
1	18	Encoder			0.004 kg
1	19	Encoder ring	M8	Steel	0.002 kg
6	20	Hex nut	M4	Steel	0.001 kg
8	21	Screw for plastics	M3	Steel	0.001 kg
2	22	Screw for plastics	M2	Steel	0.000 kg
1	23	DIN 976-1 - M3 x 6 - A		Steel, Mild	0.000 kg

Designer:	Product:	Size-scale:	BME DEPARTMENT OF APPLIED MECHANICS
Balázs Endrész	Assembly	1 : 1	
Date:	Assembly:	Proj. mode	Drawing number:
2023. 10. 14.	Furuta pendulum		MM-EB-001
Checked by:	Material:	Mass:	File name:
		0.218 kg	