Mälardalen University
School of Innovation, Design and Engineering
Västerås, Sweden

Project Course in Robotics and Embedded Systems,
DVA473 and DVA474
Autumn 2018

Project AUTOBIKE

Autonomous Bicycle Platform

Examiner: Mikael Ekström
Mälardalen University, Västerås, Sweden

Project Manager: Therése Eriksson
Mälardalen University, Västerås, Sweden

Project Team: Mohamed Mahmoud Abdelnaiem
Tom Andersson
Gustav Carlstedt
Niklas Persson
Mälardalen University, Västerås, Sweden

Research partners:

February 15, 2019

Abstract

The field of bicycle dynamics has fascinated the scientific community since the early 19th century. Studies have analytically proven that under the correct circumstances a bicycle will be able to self-stabilise without the assistance of a human rider. With the arrival of computers, this could also be seen in software simulations, and during the late 20th and 21st century the goal of achieving a real-life bicycle capable of self-stabilising has become more and more attainable with improvements to processing units, sensor systems, and real-time operating systems. The purpose of Project AutoBike is to create an autonomous, self-stabilising bicycle to be used for the validation of traffic safety systems for other vehicles. In these validation tests, it is of great importance that the bicycle behaves in a natural way as if a human is riding it. Described in this paper is the design and development of this bicycle, with all of the necessary subsystems needed in order to reach the set goals. These subsystems include the creation of the entire electrical system, a mechanical system for mounting it, as well as the software to integrate the electronics and process sensor information. Models of the developed bicycle have been created and used for simulation of its balancing capabilities; these have then been compared to experimental tests of the real bicycle platform. Conclusions regarding the project point to several of the set goals being reached, as well as possible improvements that can be implemented.

Acknowledgements

We would like to thank our supervisor Mikael Ekström and Henrik Falk for their endless support and trust in us students.

Additionally, we would like to thank the previous AutoBike team members, university staff, and our research partners for the help and feedback given to us during the project.

We are also grateful for the help on the control loop given by Alessandro Papadopolous at Mälardalen University.

Lastly we thank Würth Electronics for their sponsorship of various electronic components.

Abbreviations

ADRC Active Disturbance Rejection Controller
CAD Computer Aided Design
CSFC Cascade State Feedback Control
CoG Center of Gravity
DoF Degrees of Freedom
ESC Electronic Speed Control
FPGA Field-Programmable Gate Array
FSMC Fuzzy Sliding Mode Controller
GPI General Proportional Integration
IMU Inertial Measurement Unit
LQR Linear Quadratic Regulator
NI National Instruments
NPN Negative Positive Negative
PD Proportional-Derivative
PDB Power Distribution Board
PID Proportional-Integral-Derivative
PPM Pulse-Position Modulation
PWM Pulse-Width Modulation
RC Radio Controller
RT Real-Time
RTOS Real-Time Operating System
SPI Serial Peripheral Interface
UI User Interface
VI Virtual Instrument

Table of Contents

1 Introduction

Author: Therése Eriksson

The history of the bicycle goes back to 1817, with the German invention of the draisine being the first two-wheels, human-powered transportation system. While many unverified bicycle inventions have been claimed to exist before this [1], it wasn't until this year that a wider study of this phenomenon took off. Ever since this innovative means of transport entered the scene, hobbyists and researchers alike have been engrossed with the idea of how this works, and how the system stays balanced. Several studies performed in 19th century proved that balancing a system with two wheels in the correct speed interval is possible [2], but it wasn't until 1897 and 1899 that Carvallo[3] and Whipple[4] respectively proved this analytically supported by mathematical linearised equations and models. By steering the bicycle into a fall, Whipple found that this caused it to self-stabilise with the correct forward speed. According to the models of both Whipple and Carvallo, a bicycle is constructed out of four essential, fixed parts. These are the front wheel, the fork connected to the handlebar, the main frame, and the rear wheel. All other parts are considered non-essential. This system is in itself statically unstable [5], dynamically falling over, but is capable of dynamic stability in several ways. A common belief is that it is possible through the effect of torque-induced precession on the front wheel, as claimed by Yetkin and Ozguner [6]. This means that the gravity produces a downward force while the normal force causes a counter-reaction. The torque that is created as a consequence from this is what causes the top of the bicycle to precess. This theory is however questioned by Papadopolous et al. in [7] and by Åström et al in [8]. Another way to achieve dynamic stability is through adequate forward velocity, or through the outside influence of a cyclist using the standstill technique.

Research in the latter half of the 20th century has been aided by the advent of the computer, and access to computer software has proved to spark a new interest in the modelling of dynamic systems. With the general ease of using software to solve the equations of motion, as well as for simulation and modelling, it came to be a prominent subject for numerous theses [9], and the first computer simulation modelling a bicycle was presented in the works by Roland and Massing [10] [11].

When dealing with vehicle dynamics there are various Degrees of Freedom (DoF) associated. For a mobile robot, six important parameters to read and control are the x-, y-, and z-coordinates (the latter in the case of for example flying robots such as unmanned aerial vehicles), as well as the roll, pitch, and yaw axes [12]. Bicycles, as a rigid body, have six DoF, as well as three internal ones. These last three represent the steering angle and the angle of the two wheels. A basic version of a dynamic model for a vehicle is the bicycle with two DoF, where lateral and yaw motions are included. This simplification is a direct consequence of holonomic and nonholonomic constraints, as well as the fact that several of the remaining DoF do not affect the motion of the bicycle. However, when balancing a bicycle, the common considerations are the forward motion, the lean angle, and the steering angle [13]; a graphical representation of the latter two can be seen in Figure 1 and Figure 2 below. This is based on the fact that the bicycle will never under normal circumstances change its position in the z-direction or perform a motion along the pitch axis.

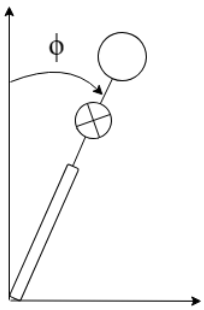


Figure 1: A graphical representation of the lean angle ϕ of the bicycle.

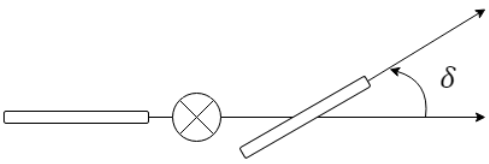


Figure 2: A graphical representation of the steering angle δ of the bicycle.

The advantages of a bicycle are many. Firstly, bicycles are very energy-efficient and cheap compared to vehicles such as cars and trucks. Secondly, they are generally quite small, and thus capable of navigating within small spaces and places that are challenging to reach with larger vehicles. This makes constructing a self-stabilising bicycle an ideal challenge for a student project. Additionally, achieving control of a bicycle could serve as an introduction to more complex control systems. This project was established as an initiative between Mälardalen University and Chalmers University in the autumn of 2017, with the aspiration of prototyping an autonomous bicycle fit for use in conjunction with other autonomous vehicles. In this specific case, the purpose is to test the safety systems in the autonomous vehicles manufactured by Volvo Cars with the help of the autonomous bicycle so that it can identify normal bicycle behaviour and verify these systems under controlled circumstances. However, it was suggested by the team from the 2017 iteration that the system should be redesigned using a new bicycle [14], thus, in the autumn of 2018, the project was restarted. The goals set for this iteration of the project is to by remote control have the bicycle balance, have the bicycle autonomously move in a straight line from point A to point B in an indoor environment while balancing itself, and holding a specified velocity of 15-15.5 km/h. Additionally, the bicycle should be capable of carrying a dummy load of 3 kg. Lastly, a computer simulation of the bicycle balancing is to be created in order to easier test the system. The aspiration is to achieve both remote control and autonomy for the bicycle, however, the primary target is the former of the two. This will be achieved by creating a sufficient electrical-, sensor-, and mechanical system, backed by corresponding software controlling the separate modules. Areas out of scope for this project iteration is trajectory tracking for the bicycle giving it a location within the environment, as well as turning which needs to be implemented only after balancing has been achieved.

The report detailing the work on project AutoBike is structured in the following way. Section 2 provides an overview of related work and research projects in the field of autonomous bicycle balancing. Section 3 details the structure and method of the bicycle system for this project, and presents some discussions and comparisons between different approaches. Section 4 holds a discussion and overview of the dynamic model, as well as details regarding the simulation of the balance of the bicycle. Section 5 includes a review of the hardware required, including a complete specification of the selected bicycle. In section 6 the design of the power supply system is presented along with details regarding noise handling. Speed control of the bicycle is documented in section 7, while the mechanical subsystems designed to mount the electronics is presented in section 8. The software detailing the integration of the electronic system is presented in section 9. Section 10 discusses the constructed and implemented controllers both in simulation and onto the bicycle platform. Section 11 holds the information regarding the test setup and the other test specification. Section 12 reviews the experimental results as well as a discussion regarding these. Lastly, section 13 wraps up the project report with a conclusion drawn from these results, as well as holding a subsection detailing the future work that could be implemented on the bicycle.

2 Related work

Author: Therése Eriksson

The bicycle system is nonlinear and very influenced by outside disturbances that can be hard to model. When designing a controller for the stabilisation of it, this needs to be a large concern in order to properly diminish the effects of the disturbances [15]. In more recent times, the research regarding the balancing of a bicycle has been divided into two camps according to Huang[16]: control through regulating of lean angle and controlling through direct steering. The former group constitutes of control methods such as using mechanical flywheel balancers, gyroscopic balancers, or different constructions of pendulums; the latter uses steering to directly control the lean angle of the bicycle itself. This section will present an examination of the specific methods used as well as the simulated or experimental results of applying them.

2.1 Balance with lean regulator

Control of a bicycle by means of using a lean regulator of some kind has been used in several adaptations of self-balancing bicycles in the past 20 years of research as can be seen in the following sections. These methods are concerned with providing an extra control parameter in the form of the lean angle of the bicycle, turning an underactuated system into a fully actuated system [17]. One popular method for achieving this is through the use of different kinds of pendulums, mimicking the behaviour of a human cyclist. Another common method is mounting a flywheel using gyroscopic precession motion on to the bicycle. The motion combines with the spinning flywheel to create stabilising torque to oppose the rolling motion. Presented below are the different methods within this category suggested and implemented in related research projects in order to achieve stabilisation of a bicycle.

2.1.1 Control through pendulum balancer

Åström et al. note that at low speeds bicycles display a very intricate behaviour and that this is amplified when the bicycle is running at zero speed [8]. However, stabilisation was achieved by Yamakita et al. at zero speed through the use of a version of pendulum called a balancer [18]. By deriving dynamic equations from Lagrange's and using a setup involving one inverted pendulum in the centre of the bicycle frame, stability was achieved in multiple simulations. The author has since produced several experimental studies where the balance of a bicycle has been accomplished through the use of both a pendulum balancer setup and with a flywheel balancer setup [19] [20].

Hwang et al. employed two approaches for the implementation of their Fuzzy Sliding Mode Controller (FSMC) when analysing self-stabilisation of an electric bicycle [21]. The first approach was concerned with achieving control of the Centre of Gravity (CoG) of the bicycle through the application of a pendulum, and the other approach with controlling the steering angle. This combination mimics the human bicycle movements in that the pendulum acts like the human CoG while the steering control simulates the handlebar turning the human performs while riding. The controller was tested in simulations and proved to be able to manage uncertainties and the control of the fluctuating speed of the bicycle.

A balancing bicycle system using an inverted pendulum based on gyroscopic precession was designed by Jin et al. [22]. The system presents new advances compared to traditional pendulum-based designs in regards to the stabilisation of the equilibrium as well as to the system response time. Using this combined with a Proportional-Integral-Derivative (PID) controller using simulation-determined control parameters, the bicycle was tested both in normal and disturbance-filled experimental environments. The normal experiments validated the quick response and convergence of the system, while the disturbance-filled environment tested the robustness of the system. Stabilisation showed to be possible in both environments, but testing in more complex settings was suggested.

2.1.2 Control through flywheel and gyroscopic balancer

Aphiratsakun and Techakittiroj have achieved an autonomous, self-balancing bicycle using several flywheels, and measuring the lean angle with a gyroscope [23]. Building upon an earlier version of the bicycle that focuses on balancing only, this version integrates a new goal to achieve both balancing and tracking of it autonomously. The bicycle proved to achieve a very stable lean control of $\pm 2°$, however, the tracking was unstable as the bicycle drifted off its trajectory by a few meters.

To achieve self-stabilisation at bicycle standstill, Tamayo-León et al. used an Active Disturbance Rejection Controller (ADRC) method [15] with Cascade State Feedback Control (CSFC). Extended state observers are used to estimating the system states and the internal and external noise, which trains the system to reject these disturbances. This method gives an estimation and a rejection of the disturbances to the system, with resulting simulations showing a robust control scheme with quick response times for stabilisation.

Two flywheels, one upper and one lower, controlled the balancing for the bicycle platform constructed by Suebsomran by rotation with a set velocity [24]. A Proportional-Derivative (PD) controller was used to track the lean angle of the bicycle; this was done by regulating the pitch angle of the two flywheels. Results from simulation showed that the error was quite low at the

steady state and that balancing of the bicycle could be achieved. The real bicycle platform was a rear-wheel driven model but was not used for testing and validation.

Hsieh et al. designed and constructed a stabilising bicycle using a gyroscopic balancer and an implemented FSMC controller [17]. They identified the rotation angle of the flywheel as the most important aspects of the balance of the system and used this and the lean angle in the design of their sliding surface balancing index. The bicycle proved to be robust to outside disturbances such as sideways forces in both simulations as well as in experimental studies of the real platform. Balance was achieved even with zero forward speed using this method.

2.2 Balance without lean regulator

Control of a bicycle without the addition of balancing equipment is another popular category in recent research. These methods mainly revolve around using closed-loop feedback in the form of PID controllers of various configurations to control the steering of the handlebar and in extension the lean angle of the bicycle. This is the method used by Baquero-Suárez et al. for the stabilisation of their self-driving bicycle [25]. They managed to achieve forward speed control through the use of a PI controller and lean angle control through the use of a two-stage observer-based ADRC approach. ADRC proved here to be an effective method and could be validated even in real experiments on their constructed prototype bicycle.

In a study performed by He et al. a constant-velocity steering controller was designed and tested both in simulations and on a prototype bicycle [26]. By implementing both feedback and feedforward control, the controller can both get rid of the error produced between the current and desired state, and stay balanced at the natural stable state. The results from the simulation show that the system quickly reaches the set reference angle, and both straight-forward and turning instructions were shown to be successful on the experimental prototype bicycle.

A research project including Huang et al. used a front-wheel driven bicycle to analyse and prove self-balancing motion using precise kinematics analysis and through the use of the steering angle and the angular velocity of the front wheel [27]. They concluded from the simulations performed in MATLAB [28] that the steering angle followed the lean angle of the bicycle closely and that the ability to adjust the lean angle of the bicycle is limited by the velocity, which has to be large enough.

A Linear Quadratic Regulator (LQR) was used for stabilisation of an electric bicycle in the work by Anjumol and Jisha [29]. The required steering angle to balance the bicycle is gained by access to the states of the bicycle, which is determined by the lean angle and the lean velocity, as well as from the desired trajectory of the bicycle. Simulations of this system were carried out in MATLAB, and these showed that stabilisation was possible and that the method employed was more robust than conventional controllers.

An autonomous bicycle project was started in the fall of 2017 at Mälardalen University with the goal of producing a bicycle that could self-stabilise using lean and steering angle [14]. Several different controllers were constructed yet the platform itself was limited since the bicycle was a front-wheel driven model, and finding a suitable dynamic model that fit the platform was troublesome for the team. Ideas from this work can however be built upon in terms of the sensor system, mechanical solutions, as well as ideas for the different controllers.

2.3 Drive mode

In most research performed on autonomous self-stabilising bicycles, the model used for both simulations and for the experimental platforms has been rear-wheel driven bicycles. Baquero-Suérez et al. used it to successfully prove that their ARDC control method approach worked in both simulations and experiments on their bicycle platform [25]. He et al. showed that by using a rear-wheel driven bicycle in combination with a no-regulator method involving a constant-velocity controller, both forward and turning control could be achieved [26]. Wang et al. [30] added electric motors to the rear wheel of a normal mountain bicycle, and by mounting a gyro-balancer to the luggage carrier, used this as a platform to test their balance control design with. Both of their controllers proved to be successful in experiments where both the tracking error as well as the lean angle error

converged. Only one research team was found to have utilised a front-wheel driven model of a bicycle as their test platform to achieve self-stabilisation [27].

2.4 Mounting of steering motor

The research has focused on solutions that have been successfully implemented in reality, as well as the solutions that have been implemented in the previous iteration of AutoBike. The motor mounts that have worked in reality have used a strap solution to control the steering angle of a bicycle. Baquero-Suarez et. al. described a strap solution that connects the handlebar to a servo motor which proved to be a robust enough method to get the bicycle to balance [25]. In the previous iteration of this project Forsberg et. al. tested two methods, one with two cogwheels connected with a plastic chain and another with the two cogwheels connected to each other. When testing, none of the two methods showed promising results. The plastic chain broke and the two connected cogwheels could not give a good enough precision [14]. The placement of the motor mount can be either at the top tube of the frame as presented by Baquero-Suarez et. al. [25] or at the head tube of the frame as Forsberg et. al. [14] presented.

3 Method

Author: Therése Eriksson

In the previous section, an overview of recent work in the realm of self-stabilising bicycles was presented. The following section contains a related discussion regarding applicability to this project in conjunction with the presented theories and approaches.

3.1 Balancing methods

Designing a bicycle without a regulator is a challenging task. For instance, balancing the bicycle at very low or zero velocities is very hard or even impossible to achieve. However, mounting and using a regulator like a flywheel or some version of a pendulum is an intricate process, and requires a lot of time and work. The approach of using a flywheel has a disadvantage since it transforms the appearance of the bicycle to the extent that it no longer looks like a bicycle, and this is of utmost importance in this project. This is because the bicycle needs to be recognisable to the automatic trucks and their safety systems; if it becomes conditioned to think that this is what a bicycle looks like, it will perform a faulty validation process and thus not be able to recognise bicycles driven by humans. This is an advocation for the pendulum method since by using this approach the system will resemble a human riding a bicycle. The pendulum movement itself closely simulates the human body movement when the person leans to the side to control the bicycle. However, the disadvantage with using a pendulum is that additional motors need to be added which adds to the complexity of the system. The advantages of not using a regulator, and instead relying on control through a steering mechanism, is that it requires very little extra mounting. No large flywheel or pendulum needs to be added to the bicycle, which only requires a steering motor and a corresponding mounting solution for it. The downside is that it requires a powerful enough balancing controller that can stabilise the bicycle fast enough, however, given the availability of different methods this poses less of a challenge compared to using a regulator. The chosen method is, therefore, to go with steering control of the bicycle.

3.2 Drive mode

In the previous iteration of project AutoBike from the autumn semester 2017, the bicycle was a front-wheel driven model with the motor located in the hub of the front wheel. This proved complex to work with, and there were significant difficulties in finding a dynamic model suitable for this kind of bicycle. Since only one research team has been able to accomplish self-stabilisation of a robot bicycle with front-wheel drive [27], there is not enough evidence to suggest that this is a suitable method for balancing a bicycle. However, there has been a lot of work and research into the use of rear-wheel drive bicycles as a base for autonomous bicycle robots, with many successful

attempts in both simulations and real experiments as shown by numerous research teams such as Baquero-Suárez et al. [25], He et al. [26], and Wang et al. [30]. Additionally, it is not feasible to create a new dynamic model for a front-wheel driven bicycle for the project since this would take up too much time and resources. On this basis, the selected bicycle for the project iteration this year will be a rear-wheel drive model.

4 Dynamic model and simulation

Author: Gustav Carlstedt

Before being able to balance the physical bicycle, it is advisable to create a simulation model. The two simulation models that are used for this project is a mathematical model and one based on a CAD model. This section will describe how both models were implemented and how the simulation software was chosen.

4.1 Dynamic model

Author: Gustav Carlstedt and Niklas Persson

To design a controller that controls the lean angle by regulating the steering of the bicycle, a dynamic model needs to be derived. The dynamic model of a bicycle is complex and has been researched for over a century. Two commonly used models are the point mass model, presented by Hand et al. and the linear Whipple model [31] [4]. However, there exist plenty more models, but as reported by Hand et al. the same equations as the Whipple or point mass model is usually derived, sometimes with small exceptions. In this section, the Whipple and point mass model are investigated and based on the result, a suitable dynamic model is implemented.

4.1.1 Whipple

The Whipple model is often applied with the simplified linear model presented by Meijaard et al [32]. The bicycle is modelled as four rigid bodies with revolute joints connecting the bodies with each other. The derived linear equations are fourth order differential equations, illustrated in eq. 1.

$$\mathbf{M}\ddot{\mathbf{q}} + v\mathbf{C_1}\dot{\mathbf{q}} + \left[g\mathbf{K_0} + v^2\mathbf{K_2}\right]\mathbf{q} = \mathbf{f}, \tag{1}$$

The equations of the the Whipple model relies on a series of assumptions and simplifications. For example, the air drag and rolling resistance are neglected, the revolute joints can move without friction, and the bicycle design is simplified with the four rigid bodies. Further, a constant velocity is considered and constant zero lean and steer angle; these simplifications enables a linear model to be derived.

The model is utilised in the work by Baquero-Suárez et al. where they manage to balance a bicycle using the equations derived by Meijaard et al. [25].

4.1.2 Point mass model

The point mass model balances the bicycle by performing calculations based on the placement of CoG and the forward velocity [31]. It can be used with either a linear or a nonlinear model. Getz and Marsden demonstrate that a bicycle using the point mass model is simplified so that the wheels are neglected and all of the mass is placed in the bicycles CoG [33]. This means that the bicycle can be approached as an inverted pendulum problem according to Limbeer and Sharp [34] as well as Getz and Marsden. As a follow up to this, Sharam et al. proved that a linear model can be used for both small and large changes in angle [35]. So far, no research papers that have been read have presented successful results on a real platform using the point mass model.

4.2 Reasoning

Before reasoning about the models the following simplifying assumptions are made:

- The bicycle is non-holonomic in the lateral and longitudinal directions
- The bicycle is holonomic in the normal direction
- The bicycle is operating on a flat surface
- Both of the wheels have ground contact at all times
- Left-right symmetry of the bicycle

In the paper by Åström et al. it is mentioned that the Whipple model is well suited for autonomous bicycles since it captures the main dynamics of a bicycle [8]. Going through papers applying different dynamic models the number of successful experiments using the Whipple model also works in favour of it. Also, last year Forsberg et al. concluded in the simulation that the point mass model performed better for a front-wheel driven bicycle compared to a rear-wheel driven bicycle [14]. Considering the stated assumptions and the findings in the paper by Åström et al., this project utilises the Whipple model with the simplified linear equations presented in the paper by Meijaard et al. However, in case of forward acceleration, the model can only be seen as an approximation since the linear model only holds for constant forward velocity. The steering and lean angle are also to be considered as small. As an alternative to the linear model, a nonlinear model is developed using computer software.

4.3 Implementation of the Whipple model

Author: Tom Andersson

First, a Cartesian coordinate system is defined with the origin at the ground contact point of the rear wheel with the z-axis pointing downwards, x-axis pointing in the forward direction, and y-axis pointing outwards as seen in Figure 3.

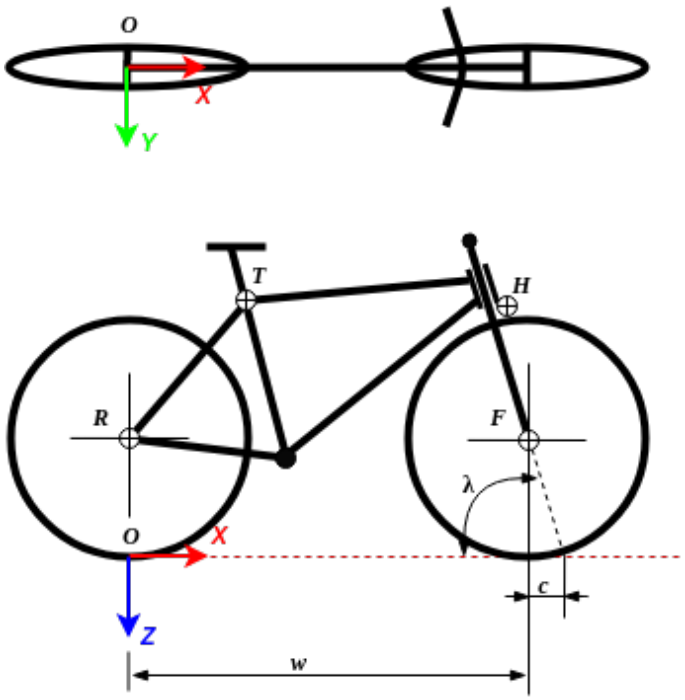


Figure 3: In the figure, the trail (c), head angle (λ) and wheelbase (w) are also illustrated to give a clarification of these parameters. The letters R, T, H, and F are the annotations for the different parts.

The model is linearised at the equilibrium point where lean and steer angles are equal to zero i.e. $\varphi = 0$ and $\delta = 0$, and with the assumptions that the bicycle is moving at a constant velocity. The reformulated Whipple model is displayed in eq. 2, also mentioned earlier in eq. 1, as two coupled differential equations:

$$\mathbf{M}\ddot{\mathbf{q}} + v\mathbf{C_1}\dot{\mathbf{q}} + \left[g\mathbf{K_0} + v^2\mathbf{K_2}\right]\mathbf{q} = \mathbf{f}, \tag{2}$$

where the $\mathbf{q} = \begin{bmatrix}\varphi & \delta\end{bmatrix}^T$ are the lean and steering angles and $\mathbf{f} = \begin{bmatrix}T_\varphi & T_\delta\end{bmatrix}^T$ are the lean and steering torques. The damping matrix $v\mathbf{C_1}$ incorporates the gyroscopic torques from the rear wheel speed, lean rates, and steering rates. M is a symmetric matrix describing the inertia properties of the bicycle. The stiffness matrix $\left[g\mathbf{K_0} + v^2\mathbf{K_2}\right]$ includes a gravitational term, forward velocity and gyroscopic and centrifugal forces. The matrices $\mathbf{M}$, $\mathbf{C_1}$, $\mathbf{K_0}$, and $\mathbf{K_2}$ are defined in [13, Appendix A]. Since matrix $\mathbf{M}$ is a symmetric and invertible matrix, the two coupled differential equations applied with the $\mathbf{M}$ inverse becomes:

$$\ddot{\mathbf{q}} + \mathbf{M^{-1}}v\mathbf{C_1}\dot{\mathbf{q}} + \mathbf{M^{-1}}\left[g\mathbf{K_0} + v^2\mathbf{K_2}\right]\mathbf{q} = \mathbf{M^{-1}}\mathbf{f}. \tag{3}$$

Equation 3 is rewritten in state space form below.

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}(v)\mathbf{x} + \mathbf{B}\mathbf{u},\\ \mathbf{y} &= \mathbf{C}\mathbf{x},\end{aligned} \tag{4}$$

with,

$$\begin{aligned}\mathbf{A}(v) &= \begin{bmatrix}\mathbf{0} & \mathbf{I}\\ \mathbf{M^{-1}}\left[g\mathbf{K_0} + v^2\mathbf{K_2}\right] & \mathbf{M^{-1}}v\mathbf{C_1}\end{bmatrix},\\ \mathbf{B} &= \begin{bmatrix}\mathbf{0}\\ \mathbf{M}^{-1}\end{bmatrix}, \qquad C = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\end{bmatrix},\end{aligned} \tag{5}$$

where $\mathbf{x} = \begin{bmatrix}\varphi & \delta & \dot{\varphi} & \dot{\delta}\end{bmatrix}^T$ are the lean angle, steering angle, and their corresponding angular velocities. The system inputs $\mathbf{u} = \begin{bmatrix}T_\varphi & T_\delta\end{bmatrix}^T$ are torques applied to the handlebar and the frame.

Table 1: Parameters used in differential equations for the ELis bicycle. The first section of the table refers to the whole bicycle, while the next four sections of the table refers to the four rigid bodies. All values are extracted from a CAD model of the bicycle.

Bicycle parameters

Parameter	**Symbol**	**Value**
Total mass $[kg]$	m_{tot}	22.3900
Wheel base $[m]$	w	1.0805
Trail $[m]$	c	0.0869
Gravity $[m/s^2]$	g	9.8200
Head angle $[deg]$	λ	72.9490°
Rear Wheel (R)		
Radius $[m]$	r_R	0.3490
Mass $[kg]$	m_R	6.7800
Mass moments of inertia $[kg \cdot m^2]$	I_{Rxx}, I_{Ryy}	0.2444, 0.4850
Rear frame (B)		
CoG position with respect to O $[m]$	x_B, z_B	0.7973, -0.6330
Mass $[kg]$	m_B	10.5100
Mass moments of inertia $[kg \cdot m^2]$	$\begin{bmatrix} I_{Bxx} & 0 & I_{Bxz} \\ 0 & I_{Byy} & 0 \\ I_{Bxz} & 0 & I_{Bzz} \end{bmatrix}$	$\begin{bmatrix} 0.4521 & 0 & -0.3845 \\ 0 & 1.0919 & 0 \\ -0.3845 & 0 & 0.6655 \end{bmatrix}$
Front frame (H)		
CoG position with respect to O $[m]$	x_H, z_H	1.1740, -0.7320
Mass $[kg]$	m_H	3.1100
Mass moments of inertia $[kg \cdot m^2]$	$\begin{bmatrix} I_{Hxx} & 0 & I_{Hxz} \\ 0 & I_{Hyy} & 0 \\ I_{Hxz} & 0 & I_{Hzz} \end{bmatrix}$	$\begin{bmatrix} 0.1154 & 0 & 0.0367 \\ 0 & 0.1193 & 0 \\ 0.0367 & 0 & 0.0255 \end{bmatrix}$
Front wheel (F)		
Radius $[m]$	r_F	0.3490
Mass $[kg]$	m_F	1.9900
Mass moments of inertia $[kg \cdot m^2]$	I_{Fxx}, I_{Fyy}	0.1004, 0.2003

Using the values in Table 1 the mass matrix $\mathbf{M}$, the damping matrix $\mathbf{C_1}$, and the two components in the stiffness matrix $\mathbf{K_0}$, and $\mathbf{K_2}$ becomes:

$$\begin{aligned} \mathbf{M} &= \begin{bmatrix} 7.8581 & 1.2895 \\ 1.2895 & 0.5675 \end{bmatrix}, \quad \mathbf{C_1} = \begin{bmatrix} 0 & 8.9893 \\ -0.6961 & 2.0172 \end{bmatrix}, \\ \mathbf{K_0} &= \begin{bmatrix} -11.9901 & -1.8582 \\ -1.8582 & -0.5742 \end{bmatrix}, \quad \mathbf{K_2} = \begin{bmatrix} 0 & 12.2820 \\ 0 & 1.7917 \end{bmatrix}. \end{aligned} \tag{6}$$

4.4 Simulation environment

Author: Therése Eriksson and Gustav Carlstedt

This section contains a review of the different simulation environments under consideration for the dynamic modelling of the bicycle system. The three environments are Gazebo, V-REP, and ADAMS [36, 37, 38].

4.4.1 Background

Before performing experiments of the stabilisation ability of the bicycle, simulations should be performed to test the balance control of the bicycle in a known environment. There are several options when it comes to simulation software, depending on the problem specifications. In this project, it was preferential that the engine could work within a Windows system, or less preferential possible to work within a Linux system. It had to be simple to set up and come with an intuitive user interface that was quick to learn, and it had to be easy to work with. Finally, it has to be compatible with MATLAB Simulink.

4.4.2 Selection

The three main contenders for suiting the needs of the project was Gazebo, V-REP, and ADAMS. Gazebo is an open source simulation engine available exclusively for Linux systems, while V-REP is a commercial product available for free with an academic license which has support for Windows, Linux, and Mac systems. ADAMS is a simulation program for model testing, available in both Windows and Linux, and is compatible with both MATLAB and LabVIEW for testing of the model. Studies by Ivaldi et al. have found that while Gazebo seems to be more used when compared to V-REP, V-REP is the most well-regarded among its users [39]. When comparing the three simulators, Nogueira concluded that Gazebo requires several external tools in order to compete with V-REP [40]. ADAMS is the software used by Baquero-Suárez et al. which successfully managed to balance a bicycle in reality [25]. According to Ivaldi et al., ADAMS was found to be a commonly used software for mobile robotics applications. It is possible to import a CAD model from SolidWorks to ADAMS as well as extract a model plant for MATLAB Simulink from ADAMS and co-simulate the controller with the simulation. A model can be built using LabVIEW Control Design and Simulation Module and LabVIEW MathScript RT Module and then evaluated alongside ADAMS to get the expected behaviour. Drawing from these specifications, the selected option of the three alternatives was ADAMS and this was used as a robotic simulation environment for the bicycle balancing. This is based on its user-friendly interface, its availability in Windows, as well as its compatibility with both MATLAB and LabVIEW.

4.5 SolidWorks model description

Solidworks has been used to calculate mass properties for the Whipple model's dynamic parameters, and also to provide a simulation model of the bicycle. It is for these purposes very important to be precise when measuring the bicycle. To follow the Whipple model the bicycle has been divided into four parts: rear wheel, frame, fork, and front wheel, see Figure 4. Each part has been carefully measured and weighed. The measurements are, compared to the real bicycle, within an error of approximately $\pm 1\ mm$. The weights have been measured in kilogram with two decimal points accuracy and added to the corresponding part in the model, see Table 2a.

In addition to the CAD model a motion study has to be created for the model to be ready for the simulation environment. In the motion study all motors, forces, frictions, and contact surfaces are defined. The rear wheel motor is defined as a rotary motor with constant velocity where as the steering motor is defined as a torque acting on the steering axis of the bicycle. A gravity force is added in positive z-direction, pointing down. A contact surface is added between both wheels and the ground to prevent the bicycle from penetrating the ground.

Figure 4: The SolidWorks CAD model of the bicycle used for extracting mass properties and running simulations.

4.6 ADAMS simulation model

Author: Tom Andersson and Gustav Carlstedt

The CAD model from SolidWorks is exported to ADAMS. Friction, contact surfaces, design, and weight properties carry over from SolidWorks to ADAMS. However, it proved to be more difficult to implement advanced properties in SolidWorks than in ADAMS, so the exported model needed to be modified in ADAMS with additional information. The parameters used to model the friction and contact forces between the ground and the bicycle tires can be viewed in Table 2b. The modelled materials are rubber against tarmac, which is the contact of the wheels against ground [25]. In ADAMS an initial velocity is added to all parts, this is to avoid an acceleration in the beginning of each run. The rear wheel is set to an initial angular velocity and the rest of the bicycle parts are set to an initial velocity.

Using co-simulation, ADAMS can create a nonlinear plant which can be controlled in MATLAB Simulink. Before the plant can be created, the inputs and outputs need to be defined.

The bicycle plant has three inputs; steering torque, rear wheel angular velocity, and lean angle disturbance. The bicycle plant also has five outputs; lean angle, steering angle, distance travelled, drift angle, and velocity.

All plant I/Os are calculated using ADAMS runtime functions and are applied/extracted in a manner that is in consistent with reality. The lean angle disturbance force is included to simulate environmental abnormalities. Lean angle is calculated around a coordinate system located in the same position as the Inertial Measurement Unit (IMU) on the real bicycle. Steering angle is obtained by calculating the front forks coordinate system in correlation to the bicycle frame. Drift angle is used to give an understanding of how much the bicycle deviates from its initial course.

Table 2: Table 1(a) displays the specific weights of the separate bicycle, electronic, and mechanical components that are included in the total bicycle system. Table 1(b) displays the environmental impacting forces upon the bicycle system and their coefficients.

(a)

Bicycle Weights		
Components	kg	In CAD
Fork	2.75	Yes
Handlebar	0.3	Yes
Cogwheel	0.06	Yes
Total:	3.11	
Frame	5.86	Yes
Steering motor mount	1.33	Yes
Component mount	0.17	Yes
Battery	2.56	Yes
Brake motor	0.18	Yes
Protection bar	0.35	Yes
IMU + Mount	0.09	No
Cogwheel	0.06	Yes
ESC with cable	0.16	No
Total:	10.76	
Front wheel	1.99	Yes
Total:	1.99	
Back wheel + motor	6.31	Yes
Magnet holder + magnets	0.32	Yes
Brake disc	0.15	Yes
Total:	6.78	
All part total:	22.64	
CAD model total:	22.39	

(b)

Rubber against Tarmac	
Impact normal force	
Stiffness	1×10^8 N/m
Damping	1×10^4 Ns/m
Penetration depth	1×10^{-4} m
Force exponent	2.2
Coulomb Friction force	
Static coefficient	0.72
Dynamic coefficient	0.72
Stiction transition velocity	0.2 m/s
Friction transition velocity	1.0 m/s

4.6.1 Simulink co-simulation

Using ADAMS and MATLAB enables the possibility to co-simulate the created controllers. The MATLAB Simulink plant generated from ADAMS is added to the work space as the dynamic model of the simulated bicycle. The controller is created in MATLAB Simulink environment using the I/Os specified in ADAMS. When the controller is finished it can be evaluated either by itself in MATLAB or in a co-simulation with ADAMS running alongside the MATLAB script, showing a graphic representation of the bicycle behaviour during run-time.

5 Hardware

In this section, an overview of the hardware such as the steering motor, selected microcontroller, and the various sensors used on the bicycle is explained. Additionally, a motivation of the selected and purchased bicycle is presented.

5.1 Bicycle

Author: Tom Andersson, Gustav Carlstedt, Therése Eriksson, Mohamed Mahmoud, Niklas Persson

As the elasticity is more noticeable in a female bicycle model, compared to a male model, a male model is chosen for the prototype bicycle [8]. To ease the derivations of the Whipple dynamic model a bicycle with as few accessories as possible is needed. Two other requested properties of the bicycle are to have the battery mounted on the frame, and to have the bicycle rear-wheel driven; this is also for the Whipple model and according to the discussion in section 3.2. Finally, having access to two bicycles with different setups would be beneficial in order to compare which one is optimal.

Crescent ELis

The Crescent ELis [41] has a rear-wheel drive electrical engine located in the rear wheel, a rechargable 36V 11Ah battery located on the main frame, and a control panel on the front fork. The bicycle, which is a male model, comes in three sizes: 51, 55, and 59 cm. The downsides to the ELis model is that its adjustable handlebar is quite limited, and the Bafang motor controller has no official documentation.

Crescent ELton

The Crescent ELton [42] also sports a rear-wheel drive electrical engine, however it is located in the middle of the chainset of the bicycle. It has the same specifications as the ELis for its battery, but in this design it is semi-integrated into the main frame. The control panel design is also semi-integrated into the front fork where the panel itself is part of the top side of it. This bicycle is only available in size 55 cm, and in the male model version. Similarily to the ELis, the handlebar adjustment is limited, and the Bafang motor controller has no documentation.

5.1.1 Selection of bicycle

The two bicycles that have been considered in this iteration of the project are the Crescent ELis electrical bicycle in size 51 and the Crescent electrical bicycle ELton in size 55. Size is based on the desire for the smallest one possible, since it will lower the CoG which is desirable in terms of stability [43]. ELis and ELton have similar specifications, however the placement of the motor for ELton might be more accessible since it is located in the chainset. Both the bicycles have quite limited handlebar adjustment, meaning that it might be tricky to mount a cog wheel. However, an adjustable stem adaptor is possible to fit on the bicycles, solving this issue. In the end, only the ELis bicycle was purchased, since it was the smallest and had the least integrated battery. This makes the bicycle easy to work with and suits the dynamic model well. The ELton was not purchased but is a possible candidate for a second version of an autonomous bicycle.

5.2 Disassemble of the bicycle

Author: Niklas Persson

After the bicycle was purchased a lot of its peripherals were removed, such as the mudguard, pedals, buzzer, front brake, saddle, chain, and chainset. They are removed since they at this point do not contribute in the work towards the riderless bicycle. However, a few key components are kept such as the brake disk, battery, motor and of course the two wheels, steering axis, handlebar, and the frame. These components were kept since a few of them are included in the dynamic model, such as the wheels and the frame. The others were kept since they were used in the electrical and mechanical systems on the bicycle.

5.3 Evaluation of old AutoBike

Author: Tom Andersson and Mohammed Mahmoud Abdelnaeim

A full analysis of the previous system components has been conducted, except for the IMU which is discussed in section 5.6.1. The steering motor, the encoder, as well as the motor driver that is used to drive the steering motor, was tested and proved to show full functionality; more about this is discussed in section 5.5. The 36V to 24V DC/DC converter was tested in the integrated system. The battery from the previous AutoBike was not used but instead kept as a reserve part. Below is a list of the electrical components that showed potential for being reused or at least saved as a reserve part.

- Steering DC motor
- Motor encoder
- Motor gearbox
- Motor driver for steering motor
- DC/DC converter (36V to 24V)
- Power Distribution Board (PDB)

All of the mentioned components were reused except the motor driver for steering motor. More detailed specifications regarding the listed components can be found in the previous years' report [14].

5.4 Microcontroller

This section will present a motivation along with a discussion regarding whether to use a National Instruments (NI) roboRIO or continue using the Raspberry Pi 3 as the microcontroller for the AutoBike project [44].

Raspberry Pi 3 [45] is a small credit card size computer designed for the DIY hobbyists. The advantage with this system on a chip circuit is that it is possible to install practically any operating system, including a Linux kernel. It is also quite cheap, making it available to projects of varying budget sizes, and has a small weight which might be of importance to the dynamic model. The roboRIO is a development board specialised for robotic applications, featuring a Field Programmable Gate Array (FPGA) [46] along with a normal processor and a real-time module [47]. It is 30 times more expensive but it houses the ability for real-time implementation.

The previous AutoBike project described a significant data loss when reading the motor encoder [14]. The Raspberry Pi 3 as the main computer was not sufficient for this purpose; it lacked the computational power and speed. Looking at the previous hardware setup and report, the system lacks real-time capabilities, which the roboRIO can provide. The drawback with the roboRIO is its hefty price. However, in this case, the price was irrelevant since a roboRIO was available courtesy of Mälardalen University. In conclusion the roboRIO was chosen due to is computational power, computational speed, integrated FPGA, real-time capabilities and its availability through Mälardalen University. These abilities combined are beneficial when reading the continuous data output from the encoder.

5.5 Steering motor

In this section, the alternatives for steering motors are discussed. The comparisons are made between using a DC motor, a servo motor, or a stepper motor.

5.5.1 Steering system analysis

The previous steering system on the old AutoBike consists of the Maxon motor DCX 32 [48] (DC motor), Planetary gearhead GPX 32 [49] and the HEDS 5540 encoder [50]. DC motors are overall very reactive and fast, however, the limited torque from the motor needs to be compensated with a gearbox. The DC motor has a nominal torque of 108mNm which is not sufficient for the purpose of the project; this is why the previous iteration invested on a gearbox with a reduction ratio of 111:1.

With the datasheet parameters for the different components, the nominal torque and the speed of the motor combined with the gearbox are calculated. Eq. 7 shows the nominal torque when the motor is combined with the gearbox.

$$T = nominal\,motor\,torque * Reduction\,ratio = 0.108mNm * 111 \approx 12Nm \tag{7}$$

The nominal speed when the motor is combined with the gearbox; this is shown in eq. 8.

$$S = \frac{Motor\,speed}{Reduction\,ratio} = \frac{7710}{111} = 69.5RPM = 417^{\circ}/s \tag{8}$$

A solution is the servo motor HS-1005SGT [50] which is an alternative implementation in place of the current steering system. A servo motor can be both reactive and strong, however, they are very expensive and need specific drive modules from the same manufacturer to work properly [51]. The suggested servo that is discussed here is quite powerful and fast, but still, the performance is less in comparisons to the current steering system since the servo has a maximum torque of 10.89Nm. Stepper motors' reaction time is overall too slow when compared to servos and DC motors; thus the conclusion is to exclude stepper motors in order to achieve a reactive steering system. When comparing the characteristics of the servo motor with the DC motor steering system, both the torque and the speed are lower for the servo motor. The characteristics of the servo are calculated when no load is applied to the system, while for the DC motor the values are calculated for the nominal load. Various stepper motors are available with high torque but low speeds, and with inefficient energy consumption compared to the other motor types. A practical aspect of the servo motor is that it is designed for radio controller enthusiasts, therefore the technical information for the servo is very limited compared to the industrial components. The conclusion is therefore, to keep the currently available steering motor, gearhead, and encoder if possible.

5.5.2 Testing of the chosen motor

The chosen motor, encoder, and gearbox have been thoroughly tested in order to guarantee that the torque and speed are efficient enough for rotating the steering axis as required by the software. In order to drive the steering motor, a motor driver is essential. Multiple tests were performed on the motor driver Cytron Md30c, and the results were not satisfactory. The Cytron motor driver can control our motor with a simple Pulse Width Modulation (PWM) signal that is sent from the roboRIO and a digital bit which indicates the direction. Only the duty cycle and direction can be controlled which is limiting. Tests were performed on the duty cycle in order to show a relationship between angular velocity and duty cycle; this is shown in Figure 5.

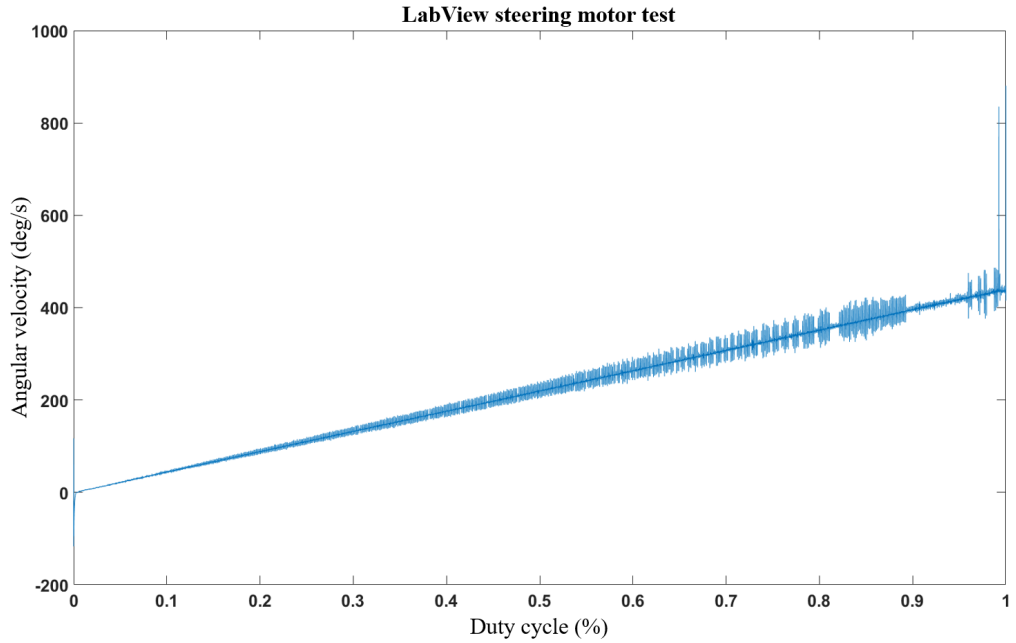


Figure 5: This graph shows the relationship between angular velocity and the duty cycle when no load is applied to the motor.

As can be seen in Figure 5, it is possible to control the velocity of the motor, but that is all. In order to balance the bicycle with the currently used control theory methods, we need the functionality to inspect the current flow to the motor in order to be able to regulate the centripetal force that is created when the steering is turned. The motor controller needs to be accurate and predictable thus the Cytron driver is not sufficient for our purpose.

A driver under the name JSP-090-20 which has both speed and torque-control functionality with a PID loop integrated into the system that optimises current and voltage control. The PID values can be self-tuned or manually tuned with the Copley controls software "CME2" [52] which allows more control over the motor behaviour. The torque is controlled by a PWM signal but additional functionality is to control it with a digital to analogue converter ± 10 V which allows for a wider setpoint range. But achieving zero torque was not possible with the current DC motor setup, instead, the DC motor is driven by velocity control.

5.6 Sensors

Keeping track of the state of the bicycle is vital to the ability to control and balance it . To be able to do this, several sensors have been used in order to collect and relay data to the roboRIO for processing.

5.6.1 IMU

One important state of the bicycle is its orientation in the room, that is, its lean angle [25]. To get access to this information an IMU is facilitated. This sensor is a composition of three separate sensors; these are an accelerometer, a gyroscope, and a magnetometer. Below is an evaluation regarding the choice of the IMU. When discussing which IMU to use, there are multiple parameters that need to be considered such as; measurement rate, measurement range, gyroscope bias, gyroscope random walk and the roll, pitch, and yaw accuracy. Something to consider is that there is not one type of IMU; there exist multiple types. The most common ones are Fibre Optic Gyro (FOG), Ring Laser Gyro (RLG) and Micro Electro Mechanical Systems (MEMS) [53].
There are three IMUs that are considered adequate for the purpose of the project; MPU-6000 [54], MPU-9250 [55] which are of type MEMS, and the Saab 8088 000-112 gyroscope which is a FOG [56]. The MPU-6000 and MPU-9250 IMU are quite similar and are developed by InvenSense for aerial applications, such as quadcopters. However, the MPU-6000 has a high vibration tolerance with a lower sample rate while MPU-9250 is sensitive to vibrations but has a sample rate twice as fast. Finally, there is the Saab developed FOG which was salvaged from a previous project. This gyroscope is developed for military use; missiles and other types of weapon stabilisation. The FOG is by far the most reliable gyroscope of the proposed three gyroscopes. However, the amount of surrounding electronics are too extensive in the sense of development time in order to facilitate the precision of the FOG. The specifications that are listed for the MPU-9250 and MPU-6000 are very similar overall; the MPU-9250 is not as vibration tolerant compared to the MPU-6000, but it has higher clock speed on the Serial Peripheral Interface (SPI) bus. Additionally, the MPU-9250 can be salvaged from the previous AutoBike iteration. Therefore, the MPU-9250 is used.

5.6.1.1 Installation of MPU-9250

From the electronic standpoint, the wiring is pretty straightforward. However, to make the Sparkfun MPU-9250 [55] communicate with an SPI protocol the connection on the jumper pads JS2 needs to be de-soldered and soldered on to the empty pad, see Figure 6. The connections from the IMU to the roboRIO can be viewed in Table 3.

Table 3: A table describing the connections from the MPU9250 to the roboRIO.

MPU-9250	**roboRIO**
GRD	GRD
VDD	3,3V
SCI	SCLK
SDA/SDI	MOSI
SDO	MISO
CS	CS0
VDDIO	3,3V

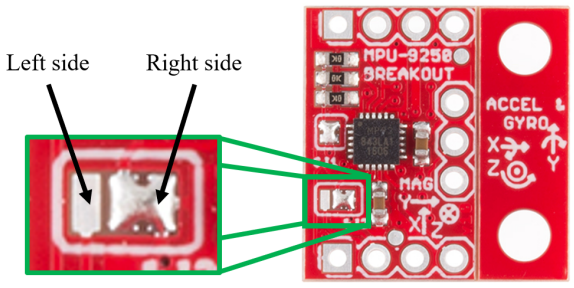


Figure 6: A visual representation of the de-soldering and soldering areas on the Sparkfun MPU9250, the jumper on the right side need to be de-soldered and solder on the left side.

5.6.2 Current sensor

In order to effectively secure the safety of the battery and the electrical system it is crucial to monitor the current flow and, if needed, save the history in a log. One current sensor module of type PmodISNS20 [57] is implemented; the current sensor monitors how much current that is drawn from the battery. The sensor is put in series with the positive power line which allows measuring of current; the information is then sent over the SPI bus to the roboRIO. An image of how the system looks can be seen in Figure 7.

Figure 7: Picture of the current sensor PmodISNS20; this sensor is wired in series between the battery and the electrical system.

5.6.3 Hall sensor

To be able to measure the speed of the bicycle a speed measurement sensor needs to be implemented. The original electrical bicycle sensor uses an internal hall sensor embedded in the rear wheel bicycle motor. However, this sensor proved to be very hard to access due to the lack of documentation provided by the manufacturer, therefore an external hall sensor was implemented.

The sensor of choice is the Honeywell Negative Positive Negative (NPN) Hall effect sensor [58]. Being an NPN sensor, the sensor needs a pull-up resistor to the signal cable, in practice the V_{DD} to the sensor is connected to the signal output with a $68\ k\Omega$ resistor, see Figure 8. The output signal is a linear analogue signal, however, the signal is only used to sense a magnet passing by; this means that the magnet sensed by the sensor will drive output signal V_{DD} to ground, which is interpreted as a digital signal instead of original analogue signal.

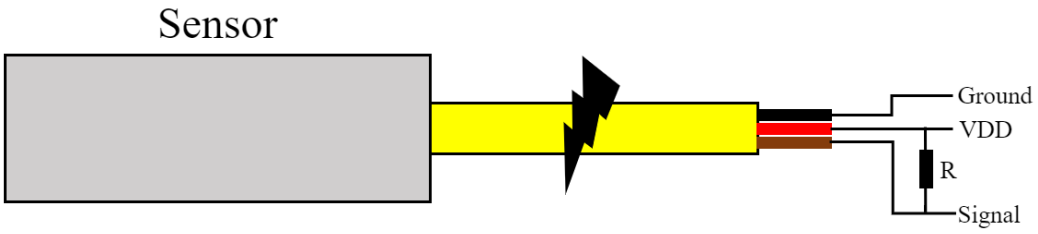


Figure 8: The rod housed Hall sensor, with corresponding wiring schematics. The black wire is the ground, the red wire is the input voltage, and the brown wire is the signal. Observe the resistor R which acts as a pull-up resistor for the NPN sensor.

The sensor is mounted at the back of the frame perpendicular to the rear wheel. The main reason for this location is that it grants an easier way to test the Hall sensor, it also makes it easier to mount the magnets since it provides good mounting options. Magnets on the back wheel act as measuring points; the magnets themselves are of a cup design, where the centre of the magnet is the north magnetic poles and the exterior material the south magnetic poles, see Figure 9. The structure of the magnet will generate two signal pulses for each magnet the Hall sensor passes.

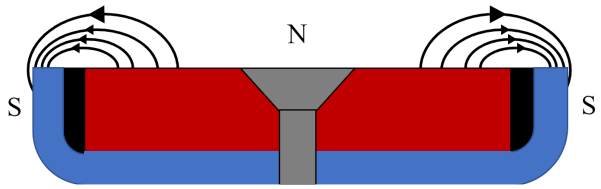


Figure 9: A cross section of a cup magnet and its corresponding magnetic field.

5.7 Safety switch

If the software fails to engage its safety procedures, a mechanical structure needs to take control to ensure that the handlebar doesn't keep spinning. The first idea was to implement a mechanical stop to limit the swing of the handlebar. However, this idea was scrapped since the force of the swing might deform the mechanical structure which might be time-consuming to repair.

The chosen design is a simple circuit header that gets pulled out from its socket when the handlebar goes approximately 75° in either direction. The mechanics to pull the header are similar to the steering of a soapbox car, where to steer the car you pull the string either left or right to turn. In this case, the steering motor does the steering and the string simply pulls the header when the handlebar goes out of range.

The motor controller needs main power, ground, a PWM signal, direction signal and three ground wires which enables amplifier, positive current flow and negative current flow. The amplifiers enable signal is favourable for safety usage, disconnecting the enable signal will make the controller instantaneously deactivate; cut the power to the motor. By routing the amplifier enable signal through the green header the system can mechanically switch off the power to the motor if the header is pulled. Figure 10 shows the header with the circuitry and Figure 11 shows the soapbox string setup on the front fork.

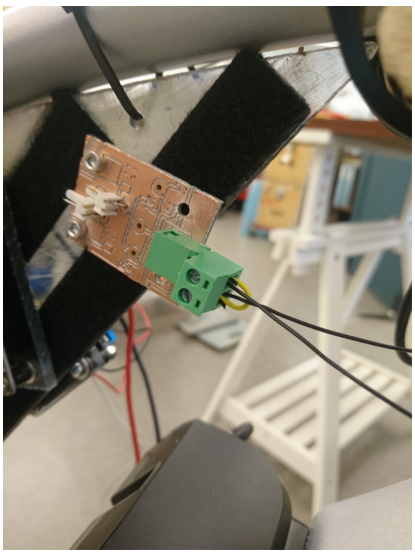


Figure 10: The image shows the green header, in which the amplifier enable signal travels through.

Figure 11: The soapbox string acting as a pulling force on the header, breaking the amplifier enable signal to the motor controller.

6 Power supply

Author: Tom Andersson and Mohammed Mahmoud Abdelnaeim
This section discusses how the power is harnessed from the integrated battery and how it is converted to the different voltages for the various subsystems.

6.1 Power harness from integrated battery

There are many ways to supply the external electrical system on the bicycle; one way is to buy Li-Po batteries and supply the PDB or one can use the existing battery on the bicycle to harness directly from it. To just simply draw power from the battery is not feasible, the motor controller that is wired to the battery needs to be connected. To make this work, wires are soldered to the power cables that are between the motor controller and the battery; the signal cables are not modified in order to allow the motor controller to communicate with the battery. Figure 12 shows how the power cables are wired between the battery and the motor controller.

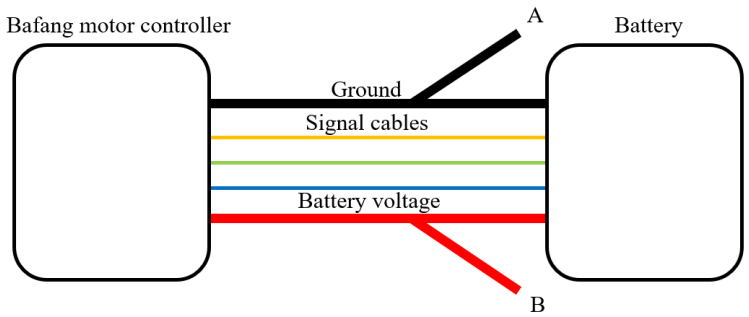


Figure 12: This figure shows the wiring diagram between the battery and the motor controller, wire **A** is the ground to the PDB and **B** is the VDD to the PDB.

6.2 DC voltage supply - 5, 15, & 24 Volt

Initially, the old AutoBike was stripped down for reusable parts (see section 5.3). For the power distribution, two components were salvaged. First is the PDB; this board supplies the voltage regulators with 36V DC. There are also safety aspects to it such as a fuse and a dormant power monitor system. The second part was a 24V DC supply board for the steering motor. The old system lacks supply boards with the feature to convert 36V DC to 15V DC and 5V DC.

The supply voltage PCBs have similar functionality; the boards are for the DC/DC conversion from battery voltage to 5V, 15V and 24V. The main difference is the DC/DC converter component. The conversion to 5V uses the off the shelf model S36SE05003NRFB [59], 15V conversions use the UWE-15/5-Q48N-C model [60], and 24V conversion uses the I6A4W010A033V-001-R [61]. For stability, there are multiple capacitors integrated on all the PCBs with the purpose of stabilising the input and output voltages. The PCB designs can be found in Appendix A. Figure 49 to 54 are the respective Multisim and Ultiboard schematics for the 5, 15 and 24V conversion as well as their respective component table.

6.3 Electrical system noise handling

Author: Mohamed Mahmoud Abdelnaeim
All the power supply systems are troubling since the wiring creates electrical noise due to the magnetic field generated from the current. The current design of the electrical system is therefore separated into two sections which are mounted on opposite sides of the bicycle. On the left side of the bicycle shown in Figure 13 are all of the noise sensitive electrical components which are connected to a bus or needs a noise-free digital signal. On the right-hand side shown in Figure 14 are all the high current components; PDBs, H-bridge, and motor controller. Both sides are quite separated but there are three lines that connects both sides. The first one is the signal cable

which connect the roboRIO to the motor driver that controls the steering motor, the second is a similar connection but for the H-bridge that controls the brake motor, and finally there is the power supply cable that supplies the roboRIO. The power supply that is supplying the roboRIO is the main source of noise on the digital sensitive side, this was confirmed with voltage readings through an oscilloscope. Therefore ferrite cores are integrated to reduce the incoming high-frequency noise. One part is how the ground wires are routed, digital ground and the analog ground are separated. All of the DC/DC converters are galvanically isolated by choice which showed to be a rather beneficial implementation when measuring the noise with the use of an oscilloscope.

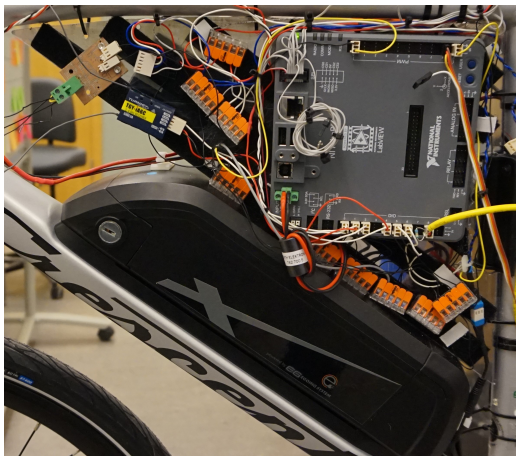


Figure 13: Pictured is the left hand side of the Plexiglass plate mounted to the frame of the bicycle. This side is holding all of the noise sensitive components.

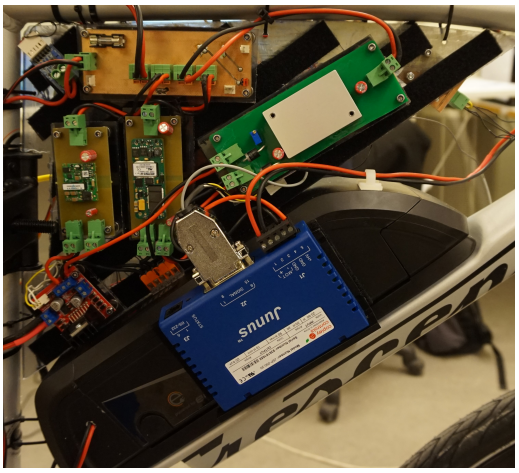


Figure 14: Pictured is the right hand side of the Plexiglass plate mounted to the frame of the bicycle. This side is holding all of the high current components.

7 Speed Control

Author: Tom Andersson and Mohammed Mahmoud Abdelnaeim

In this section, the three methods to manipulate and control the rear wheel motor is presented. The first two methods use the Crescent integrated Bafang motor controller to manipulate the speed of the bicycle. The last and final implementation uses an external motor controller to bypass the Bafang motor.

7.1 Bypassing the pedal sensor

The original system uses the pedal sensor which sends a PWM signal each time the pedal is triggered by a user; however, this not a feasible solution since the bicycle must be autonomous [14]. In order to trigger the motor without the aid of the pedal sensor, the triggering signal of the sensor needs to be investigated. After experimentation with a waveform generator, it is concluded that the triggering signal is a PWM signal with a frequency of at least 4Hz and with an amplitude of 5V. The PWM signal does not correlate to a specific velocity; it simply triggers the motor with an effect depending on the chosen assist level. The assist levels velocity correlates according to Table 4. The connector wiring for testing the system is shown in Figure 15.

Table 4: The velocity test was done while the bicycle was lifted above floor level to avoid friction; a PWM signal with a 20% duty cycle and 4Hz frequency was used as pedal triggering signal.

Assist level	Velocity (km/h)	m/s
0	0	0
1	16.5	4.6
2	17.2	4.8
3	21.5	6.0
4	24.7	6.9
5	28.1	7.8

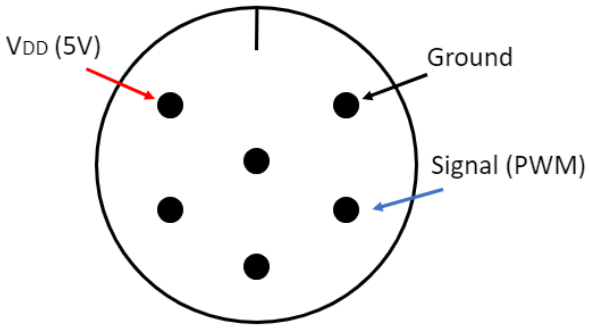


Figure 15: A representation of the pedal sensor connector and wiring schematic.

7.2 Walk assist remote triggering

The minimum speed of the bicycle is 16.5 km/h as shown in Figure 4 when triggering the bicycle with assistance level 1. This velocity is too high for testing; a speed that is low and stable is beneficial when testing the system since lower velocities make it easier to run along the bicycle. The bicycle offers a walk-assist mode, meaning that the bicycle can assist the person holding it by the push of a button. This enables the bicycle to run at a stable velocity of approximately 6km/h. This mode is triggered when the user is holding the ”gear decrement” button, but this can be bypassed with a relay module and a digital signal from the roboRIO. The normally open relay is basically replacing the button; when the relay module is triggered by a digital signal from roboRIO, the circuit is closed and the bicycle activates walk assist mode, this idea originates from previous iteration of the Autobike project [14]. The schematic is shown below in Figure 16.

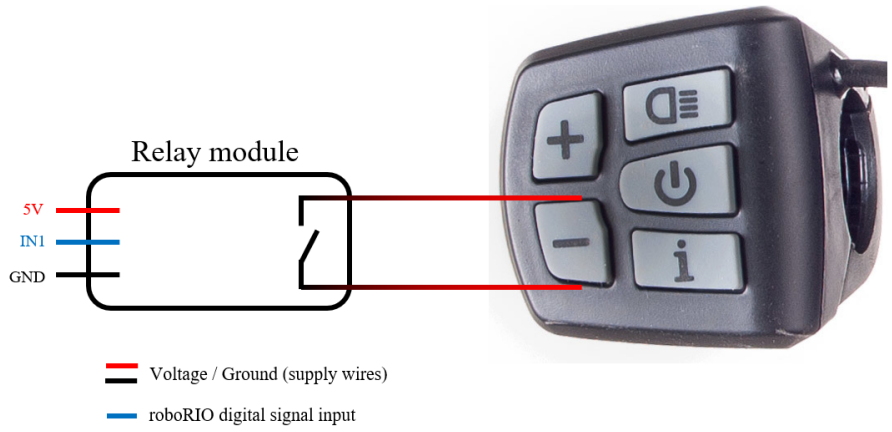


Figure 16: An electrical wiring schematic of how the walk assist is triggered with a digital signal instead of a physical button.

7.3 Exchange the Bafang motor controller

The problem with the Bafang system is that it is designed to assist the rider. The Bafang motor controller only drives the motor at a fixed speed when the rider uses the pedals. The speed of the motor is based on the gear selection on the handlebar display. From a control perspective this is problematic since the controller can't fine-tune the velocity to a reference speed; therefore the existing Bafang motor controller was switched out to an Electronic Speed Controller (ESC).

By using the HV60 ESC [62] the enclosed Bafang motor controller can be completely removed and the brushless rear wheel motor can be controlled with a PWM signal of a 500 Hz with a duty cycle 50% to 75%. This enables the AutoBike to dynamically change the velocity to the requested reference speed and removes the heavy oscillations in the velocity.

8 Mounts and constructions

Author: Gustav Carlstedt
This section will describe how the mounts and constructions have been designed and where they are located on the bicycle.

8.1 Steering motor mount

When developing an autonomous bicycle one of the most important mechanical structures is the mount of the steering motor. This structure has to be very tight and well strapped. This means that if the construction is crooked or loose in any way it may cause the steering angle to act unreliably. It also has to be constructed in such a way that small variations from the DC motor have an effect on the actual steering angle. If the mechanism does not react to small changes from the DC motor it could lead to the bicycle not being able to balance. The idea is to build a motor mount that enables the bicycle to react to small changes in the steering angle, and also be robust enough so the motor stays in place when the motor performs quick changes in direction [25].

8.1.1 Turn arrangement

It is crucial for the steering angle precision that the motor mount is robust, straight, and doesn't rattle. The solution that will be used will have a strap attached around two cogwheels. The cogwheels will have the same diameter; this is because the motor is geared so it is at this stage not desired to change the relation between them.

8.1.2 Motor mount

For the specified mount, a motor with a gearbox and encoder has been selected. This entails that the placement of the mount has to be at the head tube of the frame to ensure stability and simplify the construction. Another important aspect is to be able to tighten the strap when in place. This means that the motor mount has to have a dynamic solution that is strong enough to maintain a fully stretched strap.

The motor mount is connected in the front of the bicycle frame with two U-bolts. The cogwheels and strap a has been custom made to give a tight fit around the motor rotor and the bicycle fork. To make sure that the strap can be tightened when in place, four nuts can be adjusted to push the two cogwheels further away from each other. For the motor to stay in place when switching direction it had to be capsuled with two U-bolts. In Figure 17 is the previous iterations design and in Figure 18 is the new and improved design of the motor mount.

Figure 17: Image of the previous iterations motor mount with the plastic chain in place.

Figure 18: The steering motor mount with both cogwheels in place and the strap well tightened.

8.2 Brake motor

The bicycle is equipped with an automated brake system as a safety procedure, to be able to brake in case of a crash to ease the fall. To activate the brake on the rear wheel a 12V DC motor and a wired disc brake are used. The motor winds the wire around its rotor to tighten the disc brake and thereby slow down the speed. If the motor tightens the wire too hard it is a risk that the brake system will break; this is controlled with a motor controller of type L298 [63] which response to a PWM signal sent from the roboRIO. The mount of the motor is placed where the chainset of the bicycle usually is located. This is because it is easy to create a tight and stable mount for the motor in that location and also to pull the wire from the brake system to the rotor. The brake system can be seen in Figure 20.

8.3 Speed measurement

To measure the speed of the bicycle a hall sensor is placed at approximately 2 mm distance from the fixed magnets that are mounted on the rear wheel. The magnets have been evenly distributed around a 3D-printed plate located in the centre of the back wheel, see Figure 19.

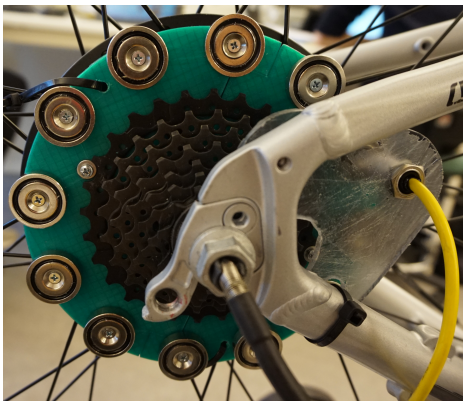


Figure 19: Twelve evenly distributed magnets used for speed measurement.

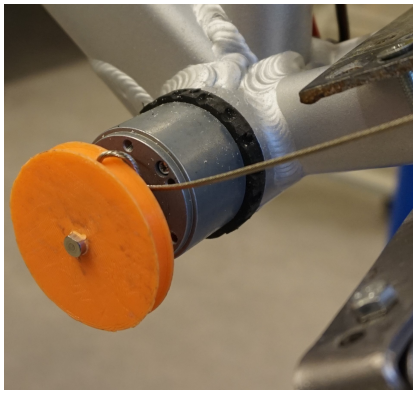


Figure 20: Image of the mounted brake system. The hydraulic brake originally included with the bicycle was switched out for a mechanical brake.

8.4 Component mounting

Author: Therése Eriksson and Gustav Carlstedt

The bicycle has to include a way to mount all external electronic components such as the roboRIO and different circuit boards. The team agreed upon four requirements for the placement of the components based on convenience and the dynamic model related works, and is as follows:

- It has to be centred around the frame to favour the dynamic model
- All components have to be able to be connected in an efficient manner
- It has to protect the electronics in case of a fall
- It still has to look like a bicycle after construction

Many bicycles have a luggage carrier which can fit a common square-shaped component box. But the bicycle acquired for this project has been selected to favour the dynamic model and therefore stripped from all redundant accessories. In order to comply with this reasoning the component mount should be designed to fit somewhere on the main frame. The mounting should be placed in such a way to keep the centre of gravity for the whole bicycle as low as possible; this will be beneficial when attempting to drive autonomously.

8.4.1 Current placement

A triangular shaped box in Plexiglas was created and mounted in the middle of the main frame. The box is located around the centre of gravity and it is thus possible to divide the weight evenly on both sides of the centre of the bicycle. The design includes a centre plate with both sides covered in Velcro; this solution enables the components to be placed in any formation. All of the components are strategically placed on the triangular shaped box in order to minimise the noise and to simplify the troubleshooting of the electrical system. All wire lengths are customised and placed in a manner to make the electrical components as accessible as possible and easier to replace.

To protect the electronics, from a fall during testing, a steel beam has been mounted on the main frame. The steel beam has two useful areas; the first is to take the hit if the bicycle would fall over and the second is to use it as a handle. Figure 21 is a photo of the final construction; tennis balls has been attached to the edges to soften the impact and give a softer grip.

8.4.2 Future placement

The mounting of the electronics on the triangular plate is not a viable solution since it presents problems for the radar cross-section. Therefore it is suggested that the electronic components are moved, and two possible placements are presented here. The first solution would be to add a bicycle basket in front of the bicycle fork and place the electronics inside of it. This is a logical placement since it is a normal accessory to commercial bicycles. The downsides to this would be that the access to the components would be restricted since they would be more enclosed in the basket, as well as greatly affecting the dynamic model of the bicycle. The second solution would be to place the electronics inside of the dummy intended for placement on top of the bicycle. The advantage of this is that they would be completely hidden away, making the bicycle system mimic a human cyclist very closely. The downside would be that the dummy is flammable, as well as slightly unstable in its placement, which places the electronics at a risk to break and/or present inaccurate readings. Since the dummy is unavailable as of now this solution is not yet implementable, and thus the first solution is the one recommended.

8.5 IMU

Author: Tom Andersson and Therése Eriksson

8.5.1 Current placement

The current location of the IMU sensor is where the bicycle rider usually sits. The bicycle seat has been removed and replaced with a wooden pole inserted into a metal tube and then sawed off to make the IMU mounting surface parallel to the ground. This placement is initially selected to reduce the noise in the IMU readings because the idea was that the acceleration data produced from the IMU would get a higher signal to noise ratio further away the sensor is the rotational axis. However, there are other forces acting on the system to consider. The current mount offers a horizontal position which makes the IMU coordinate system to match the system of the bicycle, but make the IMU reading less accurate due to the distance from the ground. The IMU mount and the previously mentioned safety bar can be viewed in Figure 21.

8.5.2 Future placement

The mounting of the IMU is in need of a future relocation since the dummy provided by Volvo will need to sit in the current placement of it. An advisable placement is to have the IMU as close to the rotational axes as possible and to be placed horizontally to produce correct readings. One option would be to construct a robust mount for it below the chainset of the bicycle to satisfy these specifications; this, however, needs to be out of the way from the braking system already located inside of the chainset.

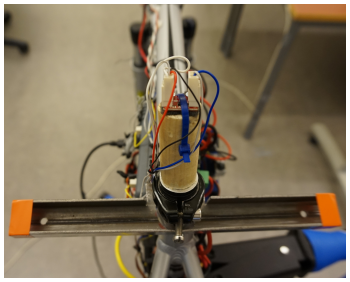


Figure 21: Collision and grip pole mounted on the main frame of the bicycle, with the main intention to protect the electronics components in case of a fall. The component that can be seen on top of the wooden stick is the IMU.

8.6 Support wheels

Author: Gustav Carlstedt

When testing the bicycle there is a need for support wheels. The support wheels are there to save a fall, so the main focus is to make them as robust as possible while at the same time be light and not too clumsy. In the previous iteration of the project Forsberg et. al tried with a few design iterations, but none that could save a fall [14].

Using the knowledge gained from the less successful constructions the new design is a bit sturdier. The improvement to the structure was to decrease the width between the wheels and add a stronger support beam between the wheels. In Figure 22 and 23 is a comparison of the previous and current design.

Figure 22: The final design of the support wheels built from the previous bicycle iteration.

Figure 23: The new and improved version of the support wheels mounted on the bicycle.

9 Software

Author: Niklas Persson

The bicycle is equipped with multiple sensors and actuators which needs to be read, processed and controlled. The main modules are listed below

- Steering motor encoder
- Steering motor controller
- IMU
- Current sensors
- Brake controller
- Speed control
- Remote control

Since a roboRIO is used as the main processing unit in the project, the code is written using National Instruments LabVIEW software development environment [44][64]. The reason to use LabVIEW is mainly that it is easy to use and there exists a lot of support and documentation for it. It is also the official language for the roboRIO. To get a better understanding of what LabVIEW is and how to best utilise it for our purpose some online training was performed via the National Instruments website and the LabVIEW Core modules [65]. In this project, I/O operations are made from both the FPGA and Real-Time (RT) target of the roboRIO [46] [47]. Since there is not any hybrid mode on the roboRIO, it is quite a challenge to set up I/O on both the RT target and the FPGA and have it work simultaneously. The solution used in this project was to copy and modify the customised FPGA personality and use the compiled bitfile for both the project and in the 'Open FPGA VI Reference.VI', as described in this forum post [66]. This procedure is explained in detail in the software manual.

9.1 Overview

To let the user know which part of the code is currently executing the communication LED on the roboRIO is used. In the initialisation phase of the software, the FPGA is started and the correct bitfile is loaded. When the communication LED turns red the code is ready for the IMU bias calculations. To execute the bias calculations, the SWD (channel 7) needs to be switched. After the bias calculations are done, the communication LED turns green and the main loop and IMU

starts to execute. When the brake command from the radio controller is sent, the communication LED is turned off. An overview of the LabVIEW modules can be seen in Figure 24.

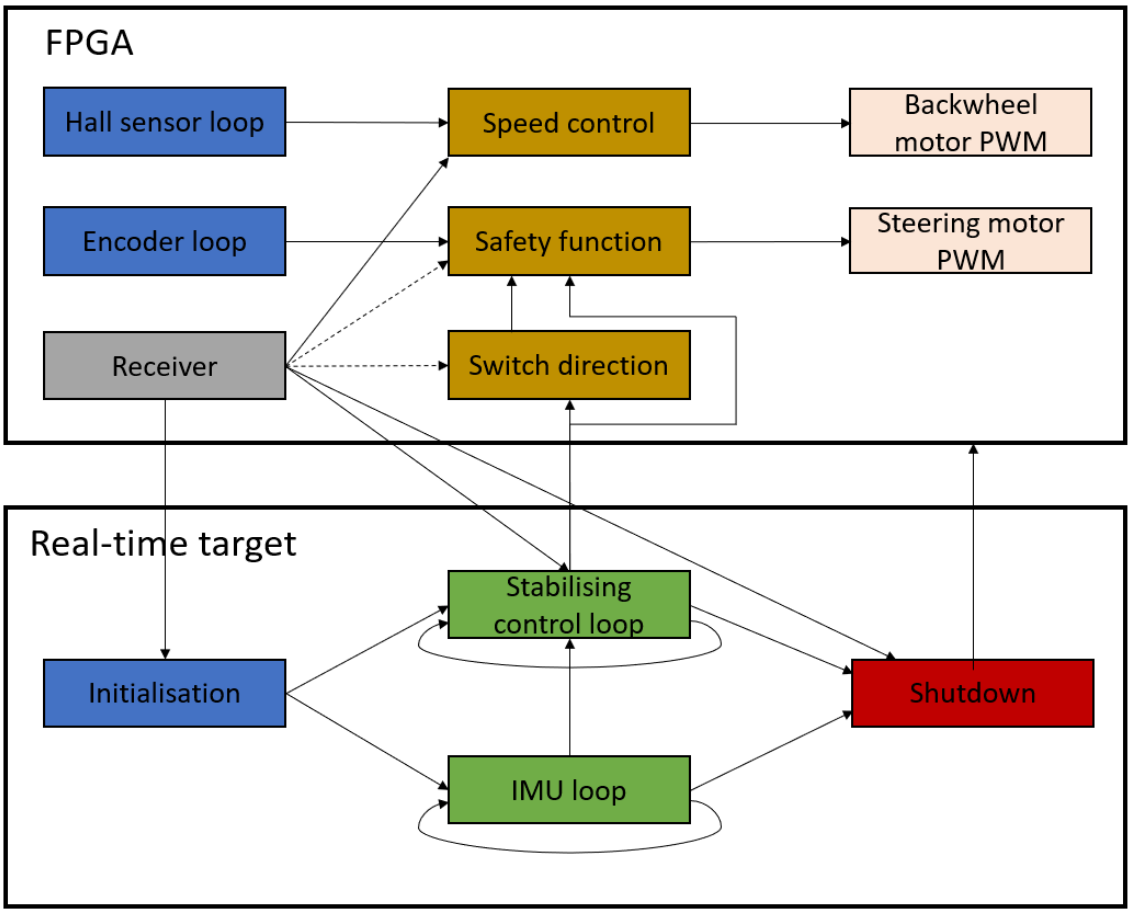


Figure 24: Overview of the different VI's and FPGA loops. The user can decide if the bicycle should run the stabilising control loop or if the steering should be controlled by the user through the radio controller. The dotted line represents an option to control the bicycle using either the closed loop or directly from the radio controller. In the shutdown phase, all motors are turned off, the brake is activated, and the FPGA is aborted.

9.2 Steering motor encoder

The output from the encoder is a quadrature phase signal, which is made out of two square waves shifted 90 degrees from each other. When the motor is running at maximum speed the two square waves from the encoder have a frequency of approximately 65 KHz each. Because of the Nyquist theorem, the decoder has to run at a rate of at least 130 KHz. To make sure that no pulses are missed, and to utilise the power of the roboRIO, the encoder is implemented in a single cycle timed loop on the FPGA. To send data between the FPGA and RTOS multiple Read/Write Control VIs from the FPGA palette are utilised, National Instruments describes the VI in [67]. Another solution would be to send the data through direct memory access FIFOs, however, since only the latest data is of interest the Read/Write Control VIs are sufficient. The position of the handlebar is sent both to the RT system to be used as an input for the stabilising control loop, and to another loop on the FPGA which acts as a safety function and do not allow the handlebar to pass ±45°. To send data between the different loops on the FPGA, local variables are utilised.

9.3 Steering motor controller

To control the steering motor, a PWM signal is generated on the FPGA target with a duty cycle determined by the control loop used for stabilising the bicycle. To control the direction of the motor, a digital high or low is sent through a digital I/O from the FPGA. As mentioned in the previous section a safety function is also present and if the position of the handlebar is ±45° the steering motor gets a PWM signal with a duty cycle of 0%. For the motor to start function normally again, the switch direction signal has to go from high to low or vice verse depending on its current value.

9.4 IMU

The raw data from the IMU can be transferred to the roboRIO with either SPI or I2C communication. Since the transfer speed of SPI is faster than I2C for the IMU and also easy to set up for multiple slave devices, SPI is chosen as the communication protocol [68]. On the master device, in this case, the roboRIO, the raw data is read on RT target utilising the SPI express VI. To be able to transfer data, a digital Chip Select (CS) signal has to be set low before transmission and set high after the transmission is completed. With different CS, it is possible to communicate with multiple slaves using the same communication bus.

To initialise the communication with the IMU, a number of registers and offsets are configured on the IMU from the roboRIO. The 16-bit raw data is utilised to compute a gyroscope and accelerometer value using eq. 9

$$Value = \frac{(Rawdata - Bias)}{Sensitivity} \tag{9}$$

The bias is calculated in the initialisation phase of the code, where 1000 readings from the gyroscope and accelerometer are averaged. The computed average value is called a bias value, and is used to minimise uncertainties in the measurements [14]. A bias value is produced for each axis for both the accelerometer and gyroscope. When the bias is calculated it is important that the bicycle is in an upright position standing as still as possible. The sensitivity values are found in the MPU9250 data sheet and are set using the GYRO_FS_SEL and ACCEL_FS_SEL for the gyroscope and accelerometer respectively. The gyroscope sensitivity is currently set to 131 and the accelerometer sensitivity is 8192.

To acquire the roll angle, the values from the accelerometer and gyroscope are merged and the rapid changes of the accelerometer and the slow changes from the gyroscope are neglected using a complementary filter. It could also be possible to use a Kalman filter, however, it is generally harder to implement and requires more computational power compared to the complementary filter [14]. To compensate for both the centripetal and lateral acceleration of the IMU, the acceleration in both y and z-direction needs to be modified according to eq. 10 and eq. 11 with the terms explained in Table 5.

$$a_y^{modified} = a_y + h_{IMU}\ddot{\phi} - \left(\frac{v^2}{b^2} tan^2(\delta)\sqrt{b^2 cot^2(\delta) + c^2}\right) cos(\phi) \tag{10}$$

$$a_z^{modified} = a_z + \left(\frac{v^2}{b^2} tan^2(\delta)\sqrt{b^2 cot^2(\delta) + c^2}\right) sin(\phi) \tag{11}$$

Table 5: Terms used in eq. 10 and eq. 11 to modify the accelerometer values to include compensation for the centripetal and lateral acceleration of the IMU. Most of the values changes with time, however, some are measures from the bicycle geometry. The $\ddot{\phi}$ term is the derivative of the gyroscope reading from the roll angle.

Term	Explanation	Value	Unit
a_z	Accelerometer readings in z direction	Varies over time	g
a_y	Accelerometer readings in z direction	Varies over time	g
h_{IMU}	IMU placement above ground	0.85	m
ϕ	Previous roll angle	Varies over time	rad
$\ddot{\phi}$	Previous roll acceleration	Varies over time	rad/s^2
v	Forward velocity	Varies over time	m/s
b	Wheel base	1.0805	m
δ	Steering angle	Varies over time	rad
c	Horizontal distance from the center of the rear wheel to the IMU	0.43	m

From Table 5 it is important to note that the previous roll acceleration, $\ddot{\phi}$ is derived from a derivative of the gyroscope reading of the roll angle. However, the gyroscope reading is quite noisy and therefore needs to be filtered. The filter used is presented in eq. 12, with $a_1 = 0.8$ at the moment, and the discrete derivative is illustrated in eq. 13 with $dt = 0.005$s

$$\dot{\phi}_{Filtered} = a_1 \dot{\phi}_{PreviousFiltered} + (1 - a_1)\dot{\phi}_{Gyroscope} \tag{12}$$

$$\ddot{\phi} = \frac{\dot{\phi}_{Filtered} - \dot{\phi}_{PreviousFiltered}}{dt} \tag{13}$$

The complementary filter in eq. 14 illustrates the calculations used to extract the roll angle of the bicycle, with $a_2 = 0.95$.

$$Roll_{new} = a_2(x_{gyroscope} + Roll_{old}) + (1 - a_2)Roll_{acc} \tag{14}$$

From the equation, one can see that the values from the gyroscope are integrated over time. If the value of a_2 is increased, more rapid changes are excluded from the accelerometer; however, more drifting from the gyroscope is included. The $Roll_{acc}$, which is the contribution from the accelerometer to the roll angle is calculated using eq. 15

$$Roll_{acc} = atan2(a_y^{modified}, a_z^{modified}) \frac{180}{\pi} \tag{15}$$

The equations used to derive the roll angle from the IMU was given to the team by associates at Chalmers University.

9.5 Brake controller

When the main loop has finished executing, the PWM to the forward motor and the steering motor are set to 0% duty cycle. In addition to turning off the motors, the brake motor is turned on for three seconds to assure that the bicycle comes to a complete stop. At the moment, the brake is only activated when the bicycle is shut down, but it would be possible to integrate it into the main loop to have the bicycle brake to tune the velocity of the bicycle. After the brake motor has finished executing the FPGA execution is also aborted.

9.6 Current sensors

Author: Mohamed Mahmoud Abdelnaeim

The current sensors communicate with the roboRIO over the SPI protocol, but with separate CS signals from the IMU. The raw data consists of 16 bits per frame, however, the data of interest

are only 12 bits long and complemented with 4 stuffed bits. The data is converted to a mA value using eq. 16. To tune the readings from the current sensors, the number inside the parentheses, in this case, 2042, should be changed.

$$Current = (Rawdata - 2042)\frac{1000}{89.95} \tag{16}$$

9.7 Speed control

Author: Mohamed Mahmoud Abdelnaeim and Niklas Persson

There are 12 magnets mounted on the back wheel of the bicycle; each magnet triggers two peaks when passing by the hall sensor which makes it basically an irregular PWM signal that is read by the FPGA on the roboRIO. By simply measuring the time it takes between the magnets the velocity is extracted by using the formula in eq. 17:

$$v = \frac{2\pi r}{dt * n}3.6 \tag{17}$$

The numerator is the circumference of the back wheel; this divided by the time, dt, between the magnets times the number of magnets n gives the velocity in m/s which is converted to km/h. The speed control works in two stages; when the error between the reference speed and the current speed is more than ±1 km/h the increase or decrease in duty cycle to the motor is regulated with only a P controller. However, when the error is less than ±1 km/h, the P controller is switched to a PI controller to ensure a small steady-state error, and to reach the reference speed in reasonable time. The different gains for the controller can be seen in Table 6. The low value for the P controller is because the speed sensor updates much slower compared to the loop time for the speed controller, which will make the duty cycle increase too fast if a higher P value is used. However, when switching to the PI controller this problem is solved by using the calculated dt value from the hall sensor to control the loop execution time for the speed control. The control structure can be seen in Figure 25.

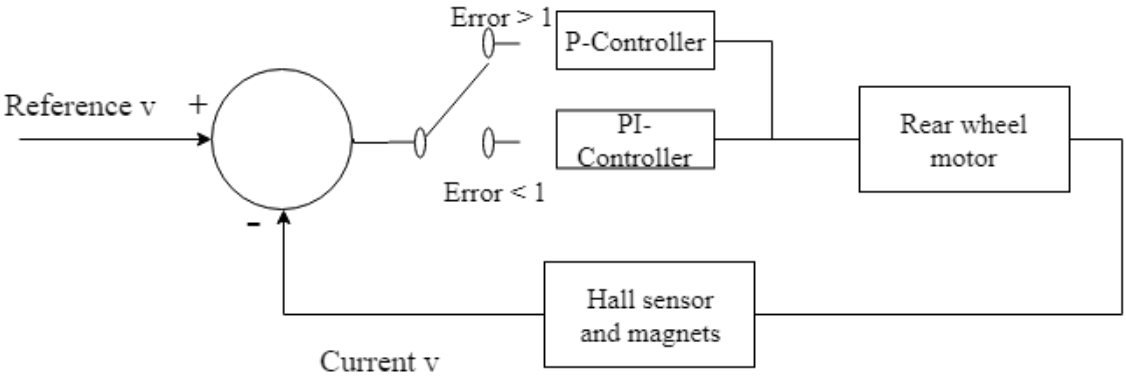


Figure 25: The control structure for the rear wheel motor, where two different controllers are utilised depending on the magnitude of the error.

Table 6: The different gains for the two controllers used to regulate the duty cycle for the rear wheel motor.

Control parameters for back wheel motor		
	K_p	K_i
P controller	0.000001	-
PI controller	40	15

9.8 Remote control

Author: Tom Andersson and Niklas Persson

To be able to remotely control some functionality of the bicycle, a TGY-i6S [69] remote control is used together with a TGY-iA6C [70] receiver. From the receiver, a Pulse Position Modulation (PPM) signal is sent to the roboRIO, where it is demodulated on the FPGA. The PPM signal starts with a 4-10 ms sync signal, and followed by the eight channels and their corresponding data; one period of the signal can be seen in Figure 26. When the signal has been demodulated, it is possible to see that if a switch is moved, the corresponding channels' high pulse will be longer or shorter, depending on the position of the switch.

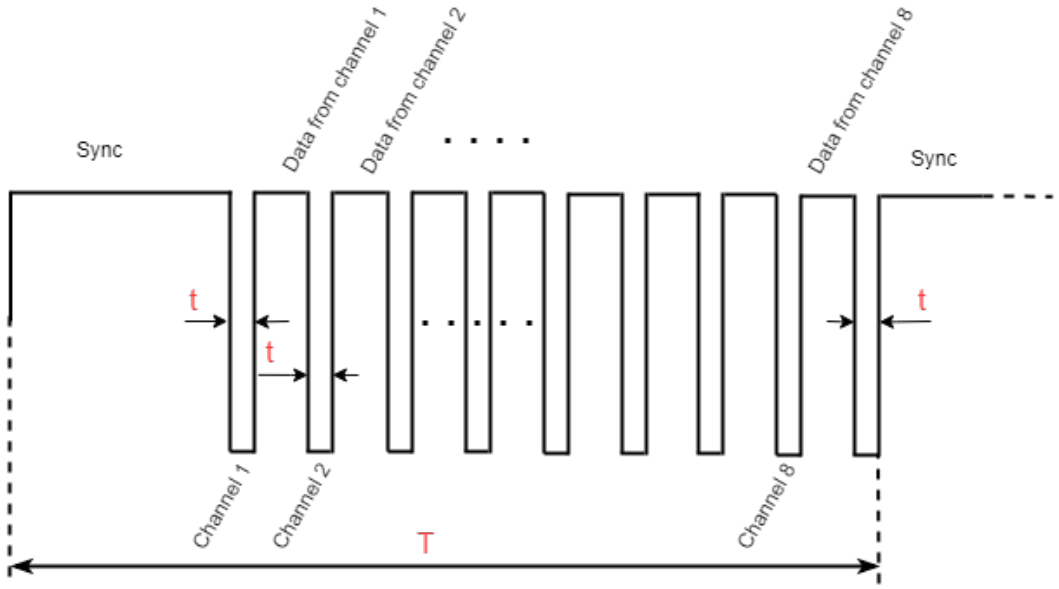


Figure 26: Shows a PPM frame, T is the total time of one frame and t is the time between each channels data transfer

9.8.1 Remote control function assignment

The remote controller purpose is to be able to remotely brake and shut down the bicycle if needed. However, switching between walk assist mode and cruise control, manually adjusting the steering, and IMU calibration functionaries are just as important. The receiver has a capacity of 8 channels; the first four channels are predefined as the joysticks, while the remaining channels are allocated for auxiliary channels. These are configurable through the controller User Interface (UI). The auxiliary channels are configured to the switches SWA, SWB, SWC, and SWD, see Figure 27. Switches SWA and SWD have two states whereas the SWB and SWC have the three switchable states. SWA is the emergency stop, while SWB has two functionalities. The first is the walk-assist and the second is the cruise control. With SWC the user can determine how the steering motor should be controlled, either by the right-hand side Radio Controller (RC) joystick or by the control loop. SWD is the calibration switch; it is used at the beginning to obtain the biased values for the IMU.

Figure 27: The TGY-i6s controller, the switches SW- A,B,C, and D are located on top of the controller, the right stick is used for manual movement of the steering motor.

10 Controller

Author: Tom Andersson

In this section, the different control structures will be explained in detail. In section 10.1 the construction and implementation of a two stage closed loop controller is explained. Followed with section 10.2 where a reinforcement learning method is implemented.

10.1 Active Disturbance Rejection Controller

The control method used for the bicycle is the ADRC proposed by Baquero-Suárez et al. [25]. The proposed method suggests the use of two General Proportional Integration (GPI) observers as disturbance observer to control the bicycle. The outer controller uses a stabilising controller which takes the lean angle and steering torque as inputs to create a desired steering angle. The desired steering angle is compared with the actual steering angle of the bicycle to calculate an error. This steering angle error is processed in the tracking controller which outputs a desired steering torque to the bicycle, see Figure 28.

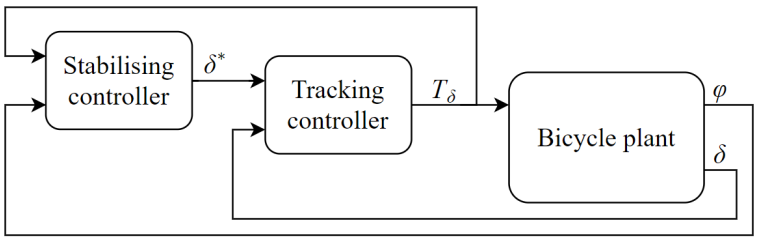


Figure 28: An overview of the control structure, where φ is the lean angle, δ is the steering angle, δ^* is the desired steering angle, and T_δ is the inputted steering torque to the bicycle plant.

The outer and inner disturbance observers are derived from the coupled Whipple model in eq. 3. Simplifying the system gives:

$$\ddot{\mathbf{q}} + \bar{\mathbf{C}}\dot{\mathbf{q}} + \bar{\mathbf{K}}\mathbf{q} = \bar{\mathbf{M}}\mathbf{f}, \tag{18}$$

where the bar matrices are

$$\bar{\mathbf{C}} = \mathbf{M}^{-1} v\mathbf{C_1} = \begin{bmatrix} \bar{c}_{11} & \bar{c}_{12} \\ \bar{c}_{21} & \bar{c}_{22} \end{bmatrix},$$
$$\bar{\mathbf{K}} = \mathbf{M}^{-1}\left(g\mathbf{K_0} + v^2\mathbf{K_2}\right) = \begin{bmatrix} \bar{k}_{11} & \bar{k}_{12} \\ \bar{k}_{21} & \bar{k}_{22} \end{bmatrix},$$
$$\bar{\mathbf{M}} = \mathbf{M}^{-1} = \begin{bmatrix} \bar{m}_{11} & \bar{m}_{12} \\ \bar{m}_{21} & \bar{m}_{22} \end{bmatrix}.$$

Expanding eq. 18 gives two differential equations:

$$\ddot{\varphi} + \bar{c}_{11}\dot{\varphi} + \bar{c}_{12}\dot{\delta} + \bar{k}_{11}\varphi + \bar{k}_{12}\delta = \bar{m}_{11}T_\varphi + \bar{m}_{12}T_\delta, \tag{19}$$
$$\ddot{\delta} + \bar{c}_{21}\dot{\varphi} + \bar{c}_{22}\dot{\delta} + \bar{k}_{21}\varphi + \bar{k}_{22}\delta = \bar{m}_{21}T_\varphi + \bar{m}_{22}T_\delta. \tag{20}$$

10.1.1 Outer control loop

Using eq. 19 the outer control loop can be derived. Since the steering motor is the only regulator on the bicycle, the lean torque is considered a disturbance ξ_ϕ. The steering angle and angular velocity are lumped together and substituted with u_ϕ, visible in eq. 21.

$$\ddot{\varphi} = \underbrace{-(\bar{c}_{12}\dot{\delta} + \bar{k}_{12}\delta)}_{u_\varphi} - \bar{c}_{11}\dot{\varphi} - \bar{k}_{11}\varphi + \bar{m}_{12}T_\delta + \underbrace{\bar{m}_{11}T_\varphi}_{\xi_\varphi}. \tag{21}$$

The simplified eq. 21 yields the outer loop control law if the estimations $\hat{\varphi}$ and $\hat{\xi}_\varphi$ are accurate. Meaning that $(\dot{\varphi} - \hat{\dot{\varphi}}) = 0$ and $(\xi_\varphi - \hat{\xi}_\varphi) = 0$. Then eq. 21 becomes the following auxiliary control law

$$u_\varphi = -\alpha_1\hat{\varphi} - \alpha_0\dot{\varphi} - \bar{m}_{12}T_\delta - \hat{\xi}_\varphi. \tag{22}$$

The auxiliary control law forces the bicycles lean angle to zero if the gains α_1 and α_0 are selected in such a way that the characteristic polynomial in eq. 23 has its roots on the left hand side of the complex plane s.

$$s^2 + (\alpha_1 + \bar{c}_{11})s + (\alpha_0 + \bar{k}_{11}) = 0. \tag{23}$$

By Laplace transforming the substitution in eq. 21, the transfer function between the u_φ and the the desired steering angle δ^* can be obtained:

$$u_\varphi = -(\bar{c}_{12}\dot{\delta} + \bar{k}_{12}\delta) \quad \overset{\mathscr{L}}{\Longrightarrow} \quad \delta^* = \frac{-1}{\bar{c}_{12}s + \bar{k}_{12}}u_\varphi. \tag{24}$$

The lean torque disturbance $\hat{\xi}_\varphi$, the lean angle velocity $\hat{\dot{\varphi}}$, and the lean angle $\hat{\varphi}$, are estimated by the following disturbance observer.

$$\begin{aligned} \dot{\hat{\mathbf{x}}} &= \mathbf{A_o}\hat{\mathbf{x}} + \mathbf{B_o}u_\varphi + \mathbf{E_o}T_\delta + \mathbf{L_o}(\varphi - \hat{\varphi}), \\ \hat{\varphi} &= \mathbf{C_o}\hat{\mathbf{x}}, \end{aligned} \tag{25}$$

where,

$$\dot{\hat{\mathbf{x}}} = \begin{bmatrix} \hat{\varphi} \\ \hat{\dot{\varphi}} \\ \hat{\xi}_\varphi \end{bmatrix}, \qquad \mathbf{A_o} = \begin{bmatrix} 0 & 1 & 0 \\ -\bar{k}_{11} & -\bar{c}_{11} & 1 \\ 0 & 0 & 0 \end{bmatrix}, \qquad \mathbf{B_o} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix},$$
$$\mathbf{E_o} = \begin{bmatrix} 0 \\ -\bar{m}_{12} \\ 0 \end{bmatrix}, \qquad \mathbf{C_o} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.$$

The gain vector $\mathbf{L_o}$ is selected in such a way that the eigenvalues of $(\mathbf{A_o} - \mathbf{L_o}\mathbf{C_o})$ are located on the left hand side of the complex plane s.

10.1.2 Inner control loop

Similarly to the outer controller loop, the inner controller loop also uses a disturbance observer and a control law. Since the inner control loop is fed with a desired steering angle from the stabilising controller, the control law need to be reformulated in terms of $\ddot{e}_\delta = (\delta - \delta^*)$. Modifying the eq. 20 generates the following steering dynamics.

$$\ddot{e}_\delta = \bar{m}_{22} T_\delta - \bar{c}_{22}\dot{e}_\delta - \bar{k}_{22} e_\delta - \bar{c}_{21}\dot{\varphi} - \bar{k}_{21}\varphi + \xi_\delta, \tag{26}$$

where,

$$\xi_\delta = \bar{\xi}_\delta - \ddot{\delta}^* - \bar{c}_{22}\dot{\delta}^* - \bar{k}_{22}\delta^*. \tag{27}$$

Using the steering dynamics reformulated in terms of steering error (e.g. eq. 26) the control law for the tracking controller can be constructed.

$$T_\delta = \frac{1}{\bar{m}_{22}}\left(\gamma_1 \hat{\dot{e}}_\delta - \gamma_0 e_\delta + \bar{c}_{21}\hat{\dot{\varphi}} - \bar{k}_{21}\varphi + \hat{\xi}_\delta\right). \tag{28}$$

Similarly to the α_1 and α_0 the γ_1 and γ_0 needs to be selected in such a way that the characteristic polynomial in eq. 29 has its roots on the left hand side of the complex plane s.

$$s^2 + (\gamma_1 + \bar{c}_{22})s + (\gamma_0 + \bar{k}_{22}) = 0. \tag{29}$$

The estimation of the disturbance signal $\hat{\xi}_\delta$, the steering angle error $\hat{e}_\delta$, and the steering velocity error $\hat{\dot{e}}_\delta$, are produced by the following disturbance observer.

$$\begin{aligned} \dot{\hat{\mathbf{z}}} &= \mathbf{A_i}\hat{\mathbf{z}} + \mathbf{B_i} T_\delta + \mathbf{E_{i_1}}\hat{\dot{\varphi}}_\delta + \mathbf{E_{i_2}}\dot{\varphi}_\delta + \mathbf{L_i}(e_\delta - \hat{e}_\delta), \\ \hat{e_\delta} &= \mathbf{C_i}\hat{\mathbf{z}}, \end{aligned} \tag{30}$$

where,

$$\dot{\hat{\mathbf{z}}} = \begin{bmatrix} \hat{e}_\delta \\ \hat{\dot{e}}_\delta \\ \hat{\xi}_\delta \\ \hat{\dot{\xi}}_\delta \\ \hat{\ddot{\xi}}_\delta \\ \hat{\dddot{\xi}}_\delta \end{bmatrix}, \qquad \mathbf{A_1} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -\bar{k}_{22} & -\bar{c}_{22} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad \mathbf{B_1} = \begin{bmatrix} 0 \\ \bar{m}_{22} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

$$\mathbf{E_i1} = \begin{bmatrix} 0 \\ -\bar{c}_{21} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \qquad \mathbf{E_i2} = \begin{bmatrix} 0 \\ -\bar{k}_{21} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \qquad \mathbf{C_o} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

The gain vector $\mathbf{L_i}$ is selected in such a way that the eigenvalues of $(\mathbf{A_i} - \mathbf{L_i C_i})$ are located on the left hand side of the complex plane s.

10.1.3 Constructing the controller

Both the outer stabilising controller and the inner tracking controller has been defined, the nested control structure can be viewed in Figure 29. The control structure is in continuous time, hence in the s domain. However, the control loop needs to be converted into discrete time (i.e. the z-domain) in order to run on the roboRIO, which is easily done through MATLAB and Simulink [71]. Observe that both the inner and outer controllers with their corresponding disturbance observers need to be converted to discrete time. This means that the roots and eigenvalues need to be located inside the unit circle.

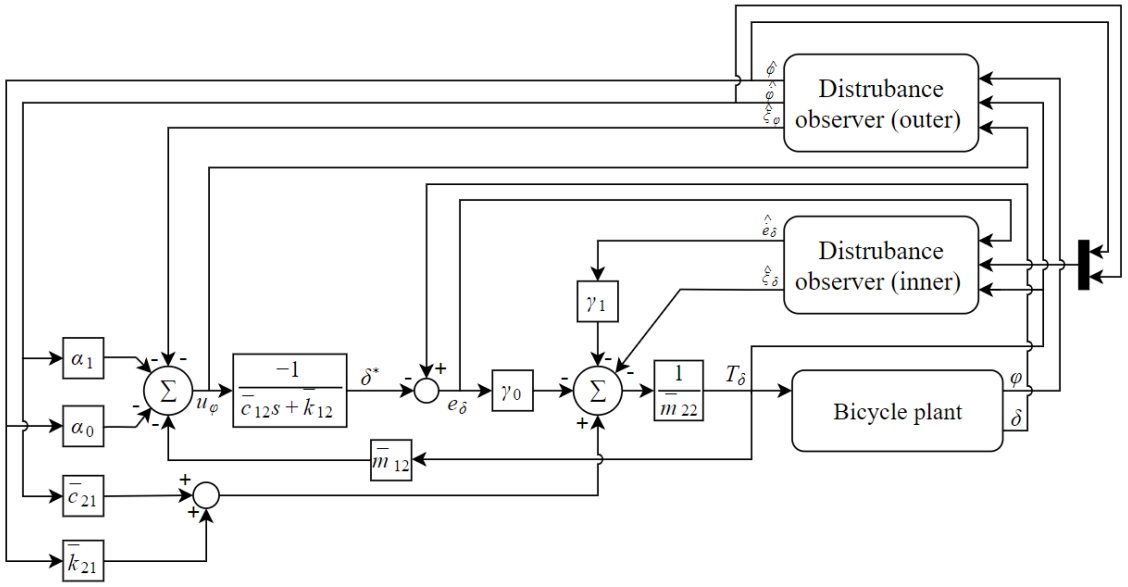


Figure 29: The complete ARDC control structure in continuous time and without saturation. In practice, the input torque T_δ needs to be saturated to the physical limitation of the steering motor. The same goes for the desired steering angle δ^*, the steering angle needs to be within a certain range, since the handlebar motion is periodic. Observe that the bicycle plant block is a generic block, in order to control a specific bicycle model this block needs to be substituted with a corresponding model plant.

10.2 Reinforcement learning controller

Author: Therése Eriksson

As a supplementary method to the closed-loop control method, a reinforcement learning controller based on Q-learning [72] in a stochastic environment [73] was created in MATLAB/Simulink. This method was also one alternative in design of controllers for the previous iteration of project AutoBike [14]. Q-learning is a method where an agent learns to reach a goal through interaction with its environment. It does this by producing a Q-table that it can use to find the optimal action given a system state. The goal itself could be to either maximise or minimise a certain parameter such as time or distance and in this case, the sought measurement is the maximum time where the absolute value of the lean angle is minimised. An important aspect of the implementation of the Q-learning algorithm is the existing trade-off between exploitation and exploration. With high exploitation, the chance to find new optimal state-action pairs is diminished, while a high exploration means a more uniform selection of possible actions. This means that there is a risk of selecting a less optimal action over a more optimal action. When the exploration rate increases, this risk also increases. The algorithm implemented on the supplementary controller uses a Boltzmann approach to overcome this issue, by using the Boltzmann distribution function [74]. This gives each state-action pair a probability based on eq. 31 to be selected.

$$\pi(s_t, a) = \frac{e^{Q_t(s_t,a)/T}}{\sum_{i=1}^{m} e^{Q_t(s_t,a)/T}} \tag{31}$$

T denotes the temperature value, which is the parameter which decides the ratio of exploitation versus exploration. When T = 0, the agent does not explore other states but only relies on exploitation of known state-action pairs. The current state is denoted by s_t, the action is denoted by a, and Q is the Q-table holding the state-action pairs. This Q-table is updated through iterations by the equation seen in eq. 32.

$$Q(s_t, a_t) = (1 - \alpha) \cdot Q(s_t, a_t) + \alpha \cdot (r_t + \gamma \cdot \underset{a}{max} Q(s_{t+1}, a)) \tag{32}$$

The variable α is the learning rate which decides how much weight to add to new rewards in comparison to old rewards. This value is set to $1/N$ where N is the number of times that state-action pair has been visited. The reward given when moving from one state to another is denoted by r, and γ is the discount factor which assigns importance either to immediate rewards or future rewards depending on value. In the implemented controller this value was set to 0.9 to prioritise long-term rewards instead of short-term. The rewards were set to 1 for positive reinforcement and to 0 for negative reinforcement.

The complete control structure can be seen in Figure 30. This gives an overview of the flow of information within the control system for the Q-learning algorithm. An initial torque is set and sent into the plant of the bicycle where lean angle ϕ, lean velocity $\dot{\phi}$, steering angle δ, and steering velocity $\dot{\delta}$ is retrieved. The state of the bicycle s is calculated from these parameters, and an appropriate action a based on this is selected along with a new torque for the next state calculation. A reward r is decided based on this action, which is then used to update the Q-table as described in eq. 32.

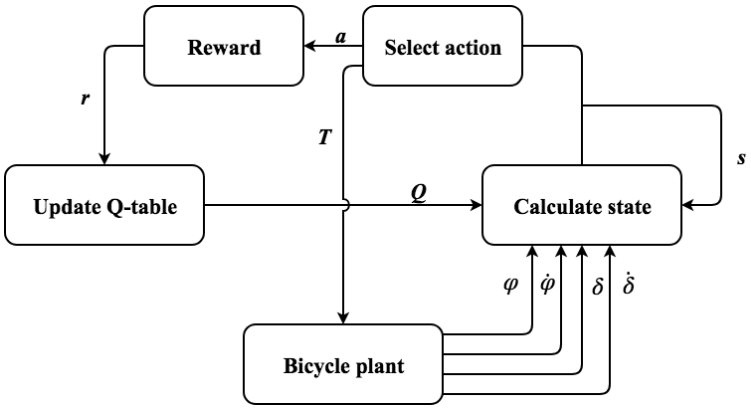


Figure 30: The control structure for the Q-learning controller. The initial torque is set to 0.001 initially and is then updated throughout the simulation. The Q-table starts out with zero in every cell. The bicycle plant is varied depending on dynamic model.

The final combination of the parameters of the bicycle forms a total of 124 states. The lean angle is divided into 31 uniformly distributed sections between -4 and 4 degrees which is the allowed interval before the bicycle is considered as having lost its balance. Similarly, the steering angle is checked for each iteration if it is within ±15 degrees. The lean velocity and the steering velocity are checked if they have a positive or negative sign to see which correction to the balance of the bicycle will occur. These states were modified throughout the implementation of the controller, and can further be optimised along with the parameters regarding exploration/exploitation, discount factor, as well as the rewards specifications.

11 Testing

Experiments were conducted to validate the behaviour of the implemented speed control, steering motor and the readings from the IMU. In this section, the experimental setups are described for the different subsystems of the bicycle.

11.1 Experimental setup for validation of speed controller

Author: Niklas Persson

Three experiments were conducted to validate the speed controller. In the first experiment the back wheel was tested without friction; the bicycle was placed in a rack and the back wheel spun freely in the air. The reference velocity was varied throughout the experiment to see how the control loop reacted to different velocities. In the second experiment the bicycle rolled on a roller,

purchased from Cykelkraft, and the roller could be set to three different resistances [75]. The resistance was set to a constant 1, while the speed reference varies as it did in the first experiment. The experimental setup for the third experiment was the same as in the second, however, in the third experiment, the roller resistance varies with time while the speed reference was kept constant at 12 km/h. The two different experimental setups can be seen in Figure 31 and Figure 32, along with the analogue resistance switch for the roller in Figure 33 and its mapping in Figure 34.

Figure 31: The first experimental setup to validate the speed control. In the first experiment the bicycle stands on a rack and the back wheel can revolve without friction.

Figure 32: To further validate the speed controller, the back wheel was placed on a bicycle roller where the resistance of the roller can be changed during runtime.

Figure 33: The analogue switch for the roller where the resistance can be changed during runtime.

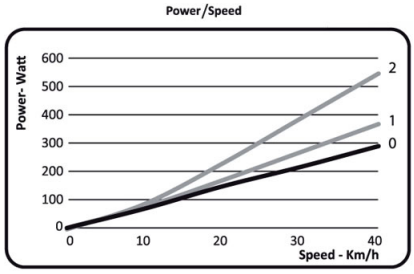


Figure 34: How the resistance switch was mapped against power at the different positions of the switch.

11.2 Experimental setup for validation of steering motor

The steering motor was controlled through a PWM signal which sets the rpm of the motor. To validate the functionality three experiments were conducted where the desired rpm was set and the actual rpm was measured. The software used to control the Junus was used for both setting the rpm and to measure the actual rpm of the motor [52]. While testing the motor the rubber band between the two cogwheels were detached, making the steering motor rotate freely. In the first experiment, the reference rpm was set to 3000. The second experiment has a reference rpm of 7500. During the third experiment, reference rpm was switched during run-time from 3500 to -3500 rpm, which means that the motor switches direction.

11.3 Experimental setup for validation of IMU

Author: Gustav Carlstedt

To be sure that the IMU output was accurate, it had to be validated. The validation setup had to be built in a way that is similar to reality, see Figure 35. For this case, that meant it had to be fixed placed on top of a pendulum with known height. A servo motor was used to move the pendulum to the desired angle. To measure the time of movement a laser pointer sheds light onto a photoresistor to mark the end of a movement.

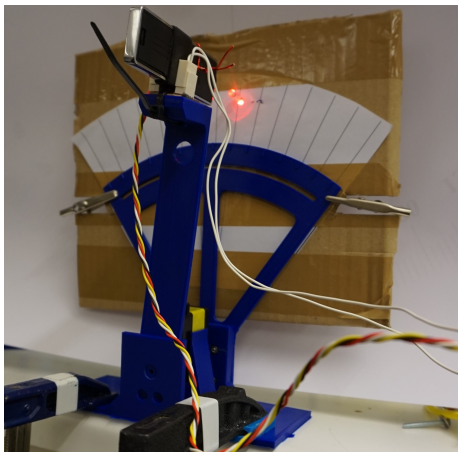


Figure 35: This figure shows how the experimental setup used for validating the IMU roll angle.

11.4 Real bicycle experimental setup

When testing a balancing algorithm on the real bicycle the same initialisation steps should be done no matter the method. Also, all experiments conducted were held indoors with the bicycle at a constant speed either on a roller or on a flat surface. For the bicycle to be considered ready for a test run, all subsystems have to be tested to validate that they work in synchronisation with each other. This was done using the radio controller while the bicycle was hanging in the air. To get the bicycle up to the desired speed for the real tests two people have to assist it. The bicycle has to be kept straight and steady while accelerating up to the given constant speed. During the accelerating phase, the balancing algorithm is not activated. It is up to a third person, handling the remote controller, to activate the balancing algorithm. This is done when the third person considers the bicycle to be straight and steady. After the activation of the balancing algorithm, it is important that the two people assisting the bicycle let go.

12 Experimental results

Author: Niklas Persson

In this section, the results are presented from the different experiments followed by a small discussion. First, the results from the experiments described in sections 11.1, 11.2, 11.3 are presented, followed by results from the different controllers. Lastly, the results from the complete system test of the bicycle are presented.

12.1 Speed control test results

The results from the three experiments of the speed control can be seen in Figure 36, 37, and 38 respectively, where the red line represents the reference speed and the blue line represents the actual speed of the rear wheel. In the third experiment, the green line indicates the value of the resistance switch of the roller. All three experiments have a duration of 120 s with a sample rate of 100 Hz.

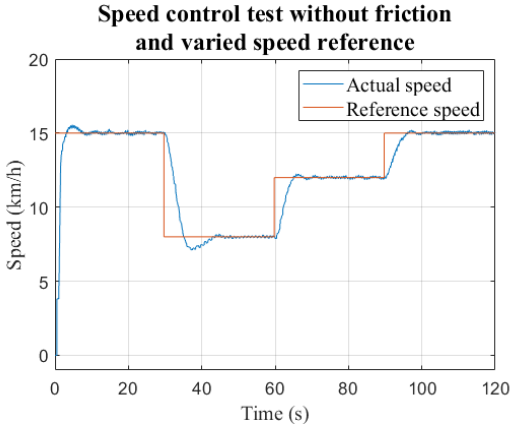


Figure 36: The result of the first experiment of the speed control where the rear wheel revolves in the air without friction. The speed reference is varied over time according to the red line.

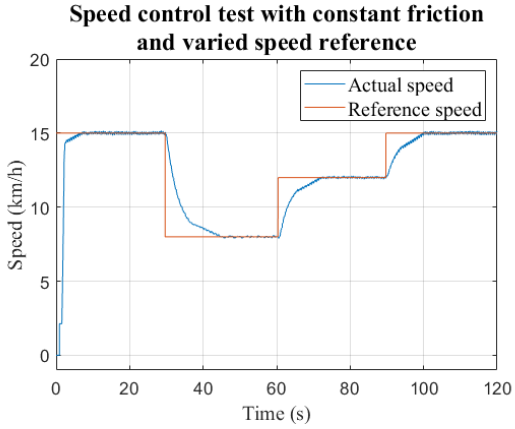


Figure 37: Result from the second experiment of the speed control. The variations of the speed reference are the same as in the first experiment, but in this experiment friction is present as well.

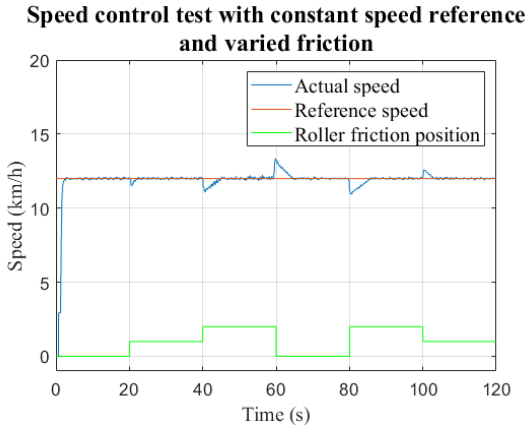


Figure 38: In the third experiment, the roller resistance is varied over time, but the speed reference is kept at 12 km/h throughout the experiment. The green line represents the value of the analogue switch on the roller.

12.1.1 Discussion

It is quite a lot of acceleration when starting the bicycle and the reference speed is set to 15 km/h. When performing stabilising experiments of the bicycle, the test personal need to be aware of this. Another solution would be to tune the different gains of the controllers, which would be necessary if the bicycle should balance during the acceleration phase as well. The speed controller manages to settle the speed of the bicycle at the different reference points with small steady-state errors, which is a promising result for balancing the bicycle at a constant speed.

12.2 Result IMU

The result from the two IMU experiments can be seen in Figure 39 and Figure 40 where the red line represents the reference angle and the blue line is the actual angle. When performing the experiments, the sampling speed was 50 Hz.

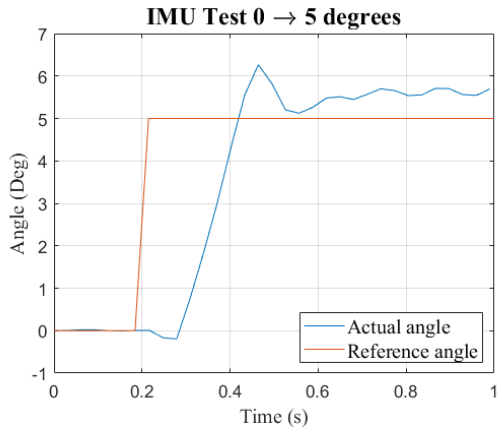


Figure 39: Results from the first roll angle experiment, where the reference is set to 5 degrees. Both the delay and the accuracy of the IMU are of interest.

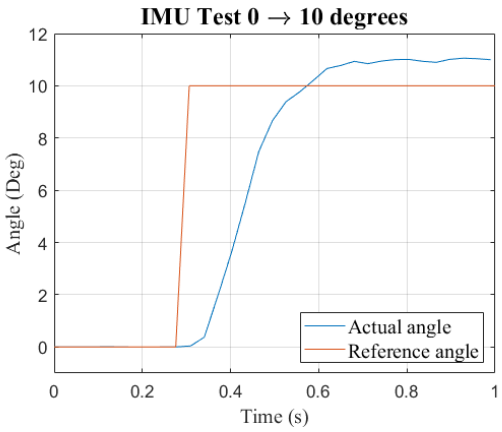


Figure 40: In the second experiment to validate the IMU, the reference roll angle is set to 10 degrees. The blue line is the actual angle the red line represents the reference angle.

12.2.1 Discussion

According to the results, the IMU readings are unreliable. However, this needs to be investigated further as the experimental setup needs to be improved to give a more reliable result. For example, the servo motor which is used in the experimental setup should be replaced with a quality servo motor to ensure the angles are correct. In that case, the laser could also be discarded. At the moment, the laser is also a source of error since it is hard to decide exactly where the photodiode picks up the light from the laser beam. Another improvement to the roll angle measurement system would be to replace the current IMU with an IMU which processes the data internally and outputs a roll angle.

12.3 Result steering motor

The result from the three steering motor experiments can be seen in Figure 41, 42,43 respectively.

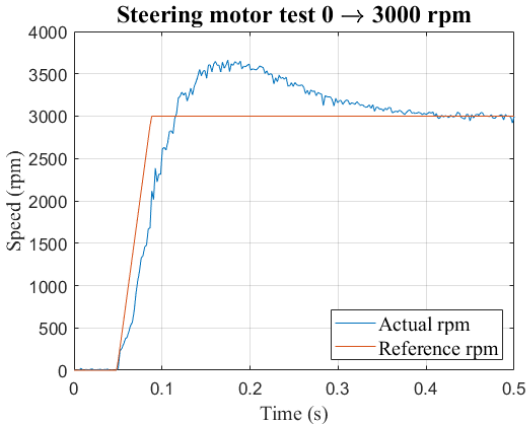


Figure 41: The result from the first experiment of the steering motor. The blue line is the reference, which is set to 3000 rpm, and red line is the actual rpm of the motor.

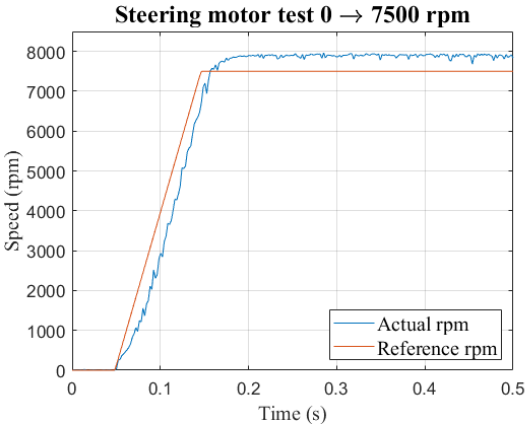


Figure 42: The graph illustrates how the steering motor behaves when the reference rpm is set to 7500. The data is collected with a sample rate of 500 Hz.

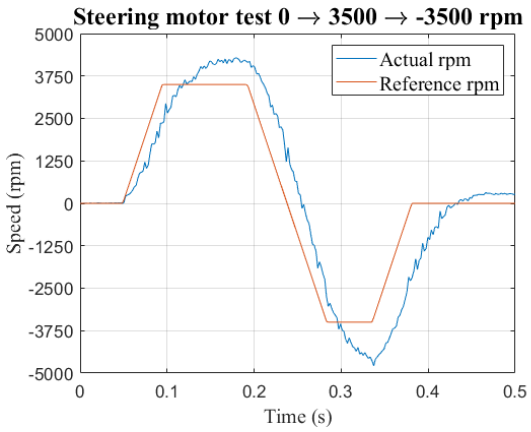


Figure 43: The result from the third experiment of the steering motor where the reference rpm is switched during run-time. In reality this means that the steering motor has switched direction.

12.3.1 Discussion

As can be seen from the results the overshoot is quite large, especially in the first experiment. However, when the motor is steering the bicycle a new commanded duty cycle will arrive every 2 ms which means that the overshoot will not be noticed. The delay of the steering motor might have a bigger impact on the actual bicycle, and this will need to be taken into consideration when modelling and controlling the bicycle. The results could be improved by tuning the motor with Junus software, and this is something that needs to be investigated further.

12.4 Result closed loop controller

Author: Tom Andersson

The result for the closed loop controller in continuous time is shown below. The outer observer gain vector from state space in eq. 25 is: $\mathbf{L_o} = [1.557 \cdot 10^3 \ 5.386 \cdot 10^5 \ 4.148 \cdot 10^7]^T$ with corresponding α_0 and α_1 assigned with gains 22.812 and 4.367. The inner observers gain vector are set to $\mathbf{L_i} = [1.099 \cdot 10^2 \ 4.693 \cdot 10^3 \ 1.094 \cdot 10^5 \ 0.22 \cdot 10^5 \ 0.22 \cdot 10^4 \ 0.3 \cdot 10^2]^T$ with γ_0 and γ_1 selected to 448.438 and 38.538.

12.4.1 ADAMS model

The simulation result of the ADRC acting on the ADAMS model in continuous time, can be viewed in Figure 44 and 45. The experimental result is recorded when the bicycle is moving at a speed of 12 km/h and with no external disturbance applied.

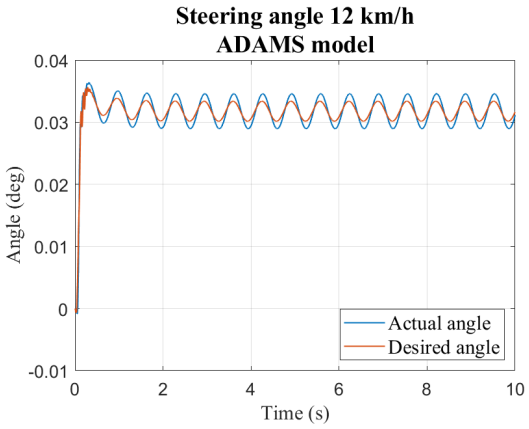


Figure 44: The result of the steering angle of the bicycle where it is moving with a speed of 12 km/h. The blue line is the ADAMS model output steer angle and the red line desired steering angle estimated through the stabilising controller.

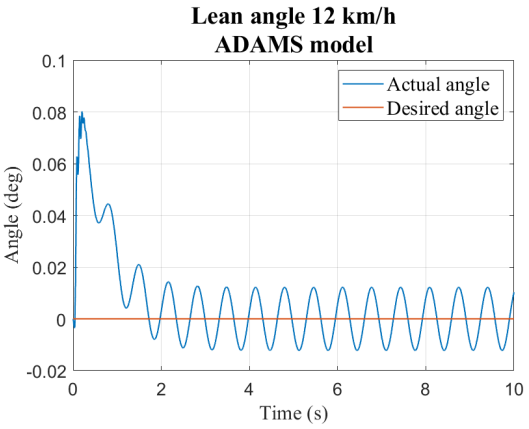


Figure 45: The result of the lean angle of the bicycle where it is moving with a speed of 12 km/h. The blue line is the ADAMS model output lean angle and the red line desired leaning angle.

12.4.2 Whipple model

The simulation result of the ADRC acting on the mathematical Whipple model in continuous time, can be seen in Figure 46 and 47. The experimental result is recorded when the bicycle is moving with a speed of 12 km/h and with a small lean torque disturbance applied. Observe that the disturbance is needed to force the model out of its equilibrium point.

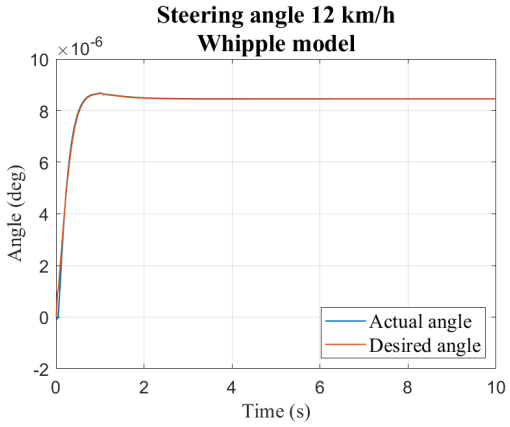


Figure 46: The result of the steering angle of the bicycle where it is moving with a speed of 12 km/h. The blue line is the Whipple model output steer angle and the red line desired steering angle estimated through the stabilising controller.

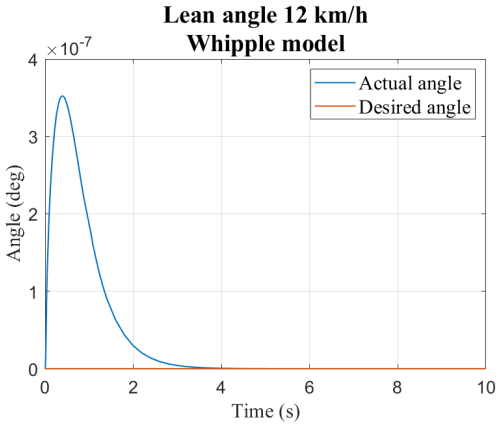


Figure 47: The result of the lean angle of the bicycle where it is moving with a speed of 12 km/h. The blue line is the Whipple model output lean angle and the red line desired leaning angle.

12.4.3 Discussion

As the results indicate both the Whipple and the ADAMS model are stable using the ADRC control structure. The ADAMS model has asymmetric mass distribution, which is visible in Figure 44, where the steering angle needs to compensate for the weight by steering more to the other side. Another noteworthy behaviour is that the bicycle is not completely stable since the system is oscillating, which the controller acting on the Whipple does not do. However, the result of

ADRC effecting the Whipple model shows really small changes, so small that the changes could be interpreted as noise generated by the IMU, and can not really be applied to reality. The result graphs for 8 km/h and 15 km/h can be seen in Appendix B.

12.5 Result reinforcement learning controller

Author: Therése Eriksson

The reinforcement learning controller was tested on the Whipple bicycle dynamic model. These tests were performed in 8 km/h, 12 km/h, and 15 km/h. During the tests the desired lean angle is set to 0 degrees. The results from one of the 12 km/h tests can be seen in Figure 48, the figures for 8 km/h and 15 km/h can be seen in Appendix C.

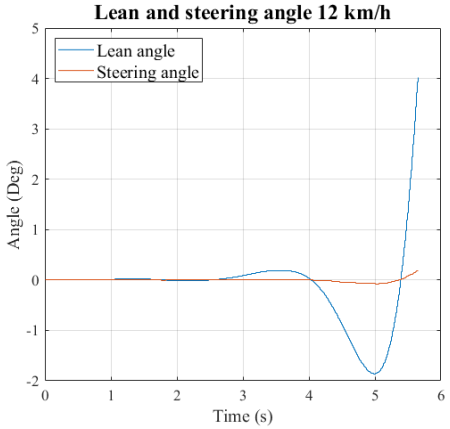


Figure 48: The result of the bicycle lean angle and steering angle where the bicycle is moving with a speed of 12 km/h for the Whipple bicycle model.

12.5.1 Discussion

As seen in Figure 48 the Whipple model is not stable using the Q-learning controller. This could be due to a variety of reasons such as the state setup where having more conditions related to the steering angle, the lean velocity, and the steering velocity could be helpful. Another reason could be the reward system for the algorithm, where consequential beneficial actions in terms of balancing capability need to be emphasised.

12.6 Result bicycle experiments

Author: Gustav Carlstedt

None of the conducted experiments resulted in a self-balancing bicycle using the ADRC. The roller tests showed a promising outcome, the ADRC did by the looks of it compensate for the change in lean angle. But, unfortunately, no data were collected during that test run. When moving on to testing on a flat surface the bicycle fell, twice, and unfortunately damaged the connector to the rear wheel motor and the experiment had to be cancelled. Also, the data logged during the experiment run on the flat surface was insufficient to be able to show any behavioural results.

12.6.1 Discussion

Even though none of the experiments proved to be successful it is believed that the balancing algorithm in itself is not to blame. Running a bicycle experiment on a roller is not an optimal way to test the system; this is due to the loss of centripetal forces that would be useful for this kind of application. But aside of that, when testing bicycle on the roller it has been easier to keep the bicycle steady using constant forward speed compared to a flat surface experiment. With the

roller, the bicycle actually managed to, with some help, balance for short periods of time compared to the experiments performed on a flat surface. When testing the bicycle on a flat surface it was a challenge to accelerate and release the bicycle in a fair way. This is something that is needed to be figured out for future experiments.

13 Conclusion

Author: Therése Eriksson

Presented in this paper is the detailing of the design and developing process of an automated bicycle platform at Mälardalen University named 'Project AutoBike'. A study of related research projects in the area of self-stabilising bicycles and bicycle dynamics was conducted to lay the groundwork for the project. Two linearised dynamic models have been studied more in-depth; the Whipple bicycle model and the point mass model. AutoBike is built upon a rear-wheel driven electric bicycle, and to this foundation, additional sensors and an accompanying electrical system have been added in order to achieve the goal of self-stabilising behaviour. Included in this are functional subsystems for steering control, braking, speed control, lean angle measurement, and remote control to meet the required specifications set by EuroNCAP and Volvo Cars. An NI roboRIO has replaced the Raspberry Pi used in the previous iteration as the main computer, thus LabVIEW has been utilised to control the separate subsystems and read the necessary sensor data to gain access to correct pose information of the bicycle. A model of the bicycle was created using SolidWorks CAD software, and two separate controllers have been created and tested in simulations of the balancing of the modelled bicycle using ADAMS software. These were an ADRC as well as a reinforcement learning controller using a Q-learning algorithm. Given more time additional controllers could have been designed and tested in simulation for comparison. These were tested with two different bicycle plants; the Whipple model plant and an ADAMS plant. Out of these two controllers, the ADRC performed better in simulation and was thus chosen for implementation onto the constructed bicycle platform. Experiments were performed in two different settings, with the bicycle unable to stabilise itself fast enough to maintain balance. Conclusions can be however be drawn from the acquired results, such as the need for some alterations to the IMU and the communication protocol, and possible improvements can be implemented in future iterations of the bicycle. Many upgrades in comparison to the previous iteration of the bicycle was implemented, such as a faster and more reliable processing unit, and as such not a lot of work is required of the current hardware or mechanical structures. A good basis is in place in regards to the dynamic model of the bicycle, which due to using a rear-wheel driven bicycle model is now more grounded in related works in the associated field.

13.1 Project goal reflections

One goal of project AutoBike was to successfully balance in simulation, and this target was reached for both of the investigated dynamic models. Another goal was to be able to maintain a speed of 15 km/h, this target has been reached as well. The goal of achieving balance on the bicycle platform was not reached, but with all of the subsystems in place and a good knowledge foundation to build upon the target is not far away.

13.2 Future work

In this final section, an overview of the suggested improvements and ideas for further implementations to the bicycle system is presented.

The first main improvement suggested is that the IMU needs to be relocated since its current placement is not optimal and needs to be located much further down. An ideal position would be beneath the wheelhouse of the bicycle or close by it. Furthermore, the component mounting needs to be altered so that the cross section of the bicycle will look more like a normal bicycle from the side since this is required by the vehicles used in the Volvo validation tests. An alternative placement is inside a bicycle basket in the front of the bicycle, since this is a common accessory and does not alter the silhouette of the bicycle. A possible downside to this could be that the electronics are less accessible, however. Additionally, the test system for the IMU needs to be

improved greatly as it is unreliable at the moment. As suggested a better servo motor to ensure correct readings. The IMU itself needs to be switched out for a better one since it currently is the bottleneck in the system in terms of processing speed.

From a software standpoint, the SPI communication needs to be relocated from the real-time target on the roboRIO to the FPGA target for enhanced processing capabilities. Distribution of the computation and filtering is needed since it is a heavy processing task which slows down the roboRIO. Implementing this along with improvements to the IMU would mean more accurate lean angle estimations and better control over the bicycle. One important problem area to implement that was out of scope for this iteration is trajectory tracking so that the bicycle can know where it is in the environment and to follow a set target path. Turning the bicycle needs to be researched and implemented for more complex bicycle movements, but should be a focus point only after adequate balance of the bicycle in a straight path is achieved.

From a mechanics standpoint, the dummy needs to be integrated into the bicycle platform. This means that some of the mounting needs to be evaluated, such as the safety handlebars and the aforementioned IMU mounting. The dummy will probably alter the dynamic model of the bicycle so this needs to be considered. Finally, a hinged support wheel system would be preferential so that it can be used only when necessary in lower speeds and be inactive at higher speeds.

References

[1] D. G. Wilson, *A short history of bicycling.* MITP, 2004. [Online]. Available: https://ieeexplore-ieee-org.ep.bib.mdh.se/document/6300282

[2] J. Kooijman, J. Meijaard, J. Papadopoulos, A. Ruina, and A. Schwab, "A bicycle can be self-stable without gyroscopic or caster effects," *Science (New York, N.Y.)*, vol. 332, pp. 339–42, 04 2011.

[3] E. Carvallo, *Théorie du mouvement du monocy-cle et de la bicyclette.* Gauthier-Villars, Paris, France, 1899.

[4] F. Whipple, *Stability of the Motion of a Bicycle.* Quarterly Journal of Pure and Applied Mathematics 30, 1899.

[5] N. H. Getz and J. E. Marsden, "Control for an autonomous bicycle," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2, May 1995, pp. 1397–1402 vol.2.

[6] H. Yetkin and U. Ozguner, "Stabilizing control of an autonomous bicycle," in *2013 9th Asian Control Conference (ASCC)*, June 2013, pp. 1–6.

[7] J. Meijaard, J. Papadopoulos, A. Ruina, and A. Schwab, "Historical review of thoughts on bicycle self-stability," *Cornell E-commons*, 04 2011.

[8] K. J. Astrom, R. E. Klein, and A. Lennartsson, "Bicycle dynamics and control: adapted bicycles for education and research," *IEEE Control Systems Magazine*, vol. 25, no. 4, pp. 26–47, Aug 2005.

[9] R. S. Hand, "Comparisons and stability analysis of linearized equations of motion for a basic bicycle model," Ph.D. dissertation, Cornell University, 1988.

[10] M. D. E. Roland, R. D, "A digital computer simulation of bicycle dynamics," *Cornell Aero. Lab. Report no. YA-3063-K-1.*, 1971.

[11] R. Roland, "Computer simulation of bicycle dynamics," *Mechanics and Sport*, vol. 4, pp. 35–83, 1973.

[12] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*, 1st ed. Springer Publishing Company, Incorporated, 2013.

[13] J. P. Meijaard, J. M. Papadopoulos, A. Ruina, and A. L. Schwab, "Linearized dynamics equations for the balance and steer of a bicycle: A benchmark and review," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 463, no. 2084, pp. 1955–1982, 2007.

[14] M. Ekström and A. Forsberg, "Student project - autobike 2017," Mälardalen University, Tech. Rep., January 2018. [Online]. Available: http://www.es.mdh.se/publications/5229-

[15] S. Tamayo-León, S. Pulido-Guerrero, and H. Coral-Enriquez, "Self-stabilization of a riderless bicycle with a control moment gyroscope via model-based active disturbance rejection control," in *2017 IEEE 3rd Colombian Conference on Automatic Control (CCAC)*, Oct 2017, pp. 1–6.

[16] Y. Huang, Q. Liao, L. Guo, and S. Wei, "Simple realization of balanced motions under different speeds for a mechanical regulator-free bicycle robot," *Robotica*, vol. 33, no. 9, p. 1958–1972, 2015.

[17] M. Hsieh, Y. Chen, C. Chi, and J. Chou, "Fuzzy sliding mode control of a riderless bicycle with a gyroscopic balancer," in *2014 IEEE International Symposium on Robotic and Sensors Environments (ROSE) Proceedings*, Oct 2014, pp. 13–18.

[18] A. Utano and M. Yamakita, "Automatic control of bicycles with a balancer," in *Proceedings, 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics.*, July 2005, pp. 1245–1250.

[19] M. Yamakita, A. Utano, and K. Sekiguchi, "Experimental study of automatic control of bicycle with balancer," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2006, pp. 5606–5611.

[20] L. Keo, K. Yoshino, M. Kawaguchi, and M. Yamakita, "Experimental results for stabilizing of a bicycle with a flywheel balancer," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 6150–6155.

[21] C. Hwang, H. Wu, and C. Shih, "Fuzzy sliding-mode underactuated control for autonomous dynamic balance of an electrical bicycle," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 3, pp. 658–670, May 2009.

[22] H. Jin, D. Yang, Z. Liu, X. Zang, G. Li, and Y. Zhu, "A gyroscope-based inverted pendulum with application to posture stabilization of bicycle vehicle," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec 2015, pp. 2103–2108.

[23] N. Aphiratsakun and K. Techakittiroj, "Autonomous au bicycle: Self-balancing and tracking control (ausb lt;sup gt;2 lt;/sup gt;)," in *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec 2013, pp. 480–485.

[24] A. Suebsomran, "Balancing control of bicycle robot," in *2012 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, May 2012, pp. 69–73.

[25] M. Baquero-Suárez, J. Cortés-Romero, J. Arcos-Legarda, and H. Coral-Enriquez, "A robust two-stage active disturbance rejection control for the stabilization of a riderless bicycle," *Multibody System Dynamics*, no. 45, pp. 1–29, 2018.

[26] J. He, M. Zhao, and S. Stasinopoulos, "Constant-velocity steering control design for unmanned bicycles," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec 2015, pp. 428–433.

[27] Y. Huang, Q. Liao, S. Wei, and L. Guo, "Stable-balancing motion analysis of a bicycle robot with front-wheel drive based on moment balance," in *2010 International Conference on Intelligent Computation Technology and Automation*, vol. 3, May 2010, pp. 367–371.

[28] The MathWorks, Inc, "MATLAB 2018B," [Online]. Available: https://se.mathworks.com/products/matlab.html Accessed: 2019-01-31.

[29] M. A. Anjumol and V. R. Jisha, "Optimal stabilization and straight line tracking of an electric bicycle," in *2014 International Conference on Power Signals Control and Computations (EPSCICON)*, Jan 2014, pp. 1–6.

[30] P. Wang, J. Yi, T. Liu, and Y. Zhang, "Trajectory tracking and balance control of an autonomous bikebot," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 2414–2419.

[31] R. Hand, "Comparisons and stability analysis of linearized equations of motion for a basic bicycle model," 1988.

[32] J. P. Meijaard, J. M. Papadopoulos, A. Ruina, and A. L. Schwab, "Linearized dynamics equations for the balance and steer of a bicycle: A benchmark and review," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 463, no. 2084, pp. 1955–1982, 2007.

[33] N. H. Getz and J. E. Marsden, "Control for an autonomous bicycle," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2, May 1995, pp. 1397–1402 vol.2.

[34] D. J. N. Limebeer and R. S. Sharp, "Bicycles, motorcycles, and models," *IEEE Control Systems Magazine*, vol. 26, no. 5, pp. 34–61, Oct 2006.

[35] Y. M. Z. A. M. Sharma, S. Wang and A. Ruina, "Towards a maximally-robust self-balancing bicycle without reaction-moment gyroscopes or reaction wheels," *Bicycle and Motorcycle Dynamics 2016*, 09 2016.

[36] Apache Software Foundation, "Gazebo." [Online]. Available: http://gazebosim.org/

[37] Coppelia Robotics, "V-rep." [Online]. Available: http://www.coppeliarobotics.com/

[38] MSC Software, "Adams." [Online]. Available: http://www.mscsoftware.com/product/adams

[39] S. Ivaldi, V. Padois, and F. Nori, "Tools for dynamics simulation of robots: a survey based on user feedback," *CoRR*, vol. abs/1402.7050, 2014. [Online]. Available: http://arxiv.org/abs/1402.7050

[40] L. S.-C. Nogueira, "Comparative analysis between gazebo and v-rep robotic simulators," School of Electrical and Computer Engineering Universidade de Campinas, Tech. Rep., 2014.

[41] C. ELis, "Electrical bicycle," [Online]. Available: https://www.crescent.se/elton-10-vxl.html Accessed: 2019-01-31.

[42] C. ELton, "Electrical bicycle," [Online]. Available: https://www.crescent.se/elis-24-vxl-1225.html Accessed: 2019-01-31.

[43] G. P. Da Silva, B. Ferreira, G. C. Da, S. Onety, E. D. Verri, S. Siéssere, M. Semprini, V. R. Nepomuceno, S. Fabrin, and S. C. H. Regalo, "Comparative Analysis of Angles and Movements Associated with Sporting Gestures in Road Cyclists," *The Open Sports Medicine Journal*, vol. 8, pp. 23–27, 2014.

[44] National Instruments, "NI roboRIO," [Online]. Available: http://www.ni.com/pdf/manuals/374474a.pdf Accessed: 2019-01-31.

[45] R. Pi, "Raspberry pi 3," [Online]. Available: https://www.dustinhome.se/product/5010909893/3-model-b-12ghz-64-bit-arm-1gb-ram-wifibt?ssel=false&_ga=2.184521955.473873732.1549106988-1861302013.1549106988&_gac=1.39936662.1549106988.EAIaIQobChMIpJ3zovmc4AIVR-aaCh2uqQ9NEAAYASAAEgIat_D_BwE Accessed: 2019-02-01.

[46] National Instruments, "NI LabVIEW FPGA," [Online]. Available: http://sweden.ni.com/fpga?fbclid=IwAR0AqhSt4q5ZWQ8mr_gba3gcILjpSUlm5CC2MJMcql6zdFnqBkhoVTcPi8I Accessed: 2019-02-01.

[47] ——, "NI LabVIEW FPGA," [Online]. Available: http://www.ni.com/sv-se/shop/select/labview-real-time-module Accessed: 2019-02-01.

[48] M. Motor, "Dc motor," [Online]. Available: https://www.maxonmotor.com/maxon/view/product/motor/dcmotor/DCX/DCX32/DCX32L01GBKL470 Accessed: 2019-02-01.

[49] ——, "Gear," [Online]. Available: https://www.maxonmotor.com/maxon/view/product/gear/planetary/GPX/GPX32/GPX32-1-Stufig-A/GPX32AAKLSL03D9CPLW?etcc_cu=onsite&etcc_med=Header%20Suche&etcc_cmp=mit%20Ergebnis&etcc_ctv=Layer&query=Planetary%20gearhead%20GPX%2032 Accessed: 2019-02-01.

[50] ——, "Encoder," [Online]. Available: https://www.maxonmotor.com/maxon/view/product/sensor/encoder/Optische-Encoder/ENCODERHEDS5540/110517 Accessed: 2019-02-01.

[51] J. Kempkes and P. K. Sattler, "Comparison of true running at different concepts of controlling brushless dc-motors," in *1993 Fifth European Conference on Power Electronics and Applications*, Sep. 1993, pp. 15–20 vol.5.

[52] Copley Controls, "CME^2 for JUNUS," [Online]. Available: https://www.copleycontrols.com Accessed: 2019-01-31.

[53] KVH, "Guide to Comparing Gyro and IMU Technologies – Micro-Electro-Mechanical Systems and Fiber Optic Gyros."

[54] T. I. sense MPU-6000, "Imu sensor," [Online]. Available: https://store.invensense.com/Products/Detail/MPU6000-TDK-InvenSense/420595/ Accessed: 2019-02-01.

[55] S. I. B. MPU-9250, "Imu sensor," [Online]. Available: https://www.sparkfun.com/products/13762 Accessed: 2019-01-31.

[56] S. . 000-112, "Fog," [Online]. Available: https://saab.com/globalassets/commercial/land/weapon-systems/gyro-products/fog-gyro/8088000-112.pdf Accessed: 2019-02-01.

[57] D. PmodISNS20, "Current sensor," [Online]. Available: https://reference.digilentinc.com/_media/pmod:pmod:pmodisns20_rm.pdf Accessed: 2019-01-31.

[58] H. 103SR13A-9, "Npn hall effect sensor," [Online]. Available: https://docs-apac.rs-online.com/webdocs/13e7/0900766b813e78ef.pdf Accessed: 2019-01-31.

[59] D.-E. S36SE05003NRFB, "Dc/dc converter 36v to 5v„" [Online]. Available: https://www.elfa.se/Web/Downloads/_t/ds/DE_S36SE3R305NMFB_eng_tds.pdf Accessed: 2019-01-31.

[60] M. P. S. UWE-15/5-Q48N-C, "Dc/dc converter 36v to 15v," [Online]. Available: https://www.elfa.se/Web/Downloads/95/63/uwe_eng_tds.pdf Accessed: 2019-01-31.

[61] T.-L. I6A4W010A033V-001-R, "Dc/dc converter 36v to 24v," [Online]. Available: https://www.mouser.se/datasheet/2/400/i6a4w-1079455.pdf Accessed: 2019-01-31.

[62] P. edge Hv 60, "Electronic speed controller," [Online]. Available: http://www.castlecreations.com/phoenix-edge-hv-60-esc-010-0106-00 Accessed: 2019-01-31.

[63] Seeed studio works - L298, "Duall H-bridge ," [Online]. Available: https://www.electrokit.com/produkt/motordrivare-l298-dubbel-h-brygga/ Accessed: 2019-02-01.

[64] National Instruments, "Web-Based Installer LabVIEW roboRIO Software Bundle 2016," [Online]. Available: http://www.ni.com/download/ni-roborio-software-2016/6205/en/ Accessed: 2019-01-31.

[65] ——, "LabVIEW Training Courses," [Online]. Available: http://sine.ni.com/tacs/app/fp/p/ap/ov/lang/sv/pg/1/sn/n8:28/ Accessed: 2019-02-01.

[66] Moo127, "Modification of FPGA personailty," [Online]. Available: https://forums.ni.com/t5/Academic-Hardware-Products-myDAQ/modification-of-FPGA-personailty/td-p/3666161 Accessed: 2019-02-01.

[67] National Instruments, "Read/Write Control Function," [Online]. Available: http://zone.ni.com/reference/en-XX/help/371599L-01/lvfpgahost/readwrite_control/ Accessed: 2019-01-31.

[68] Engineers Garage, "COMPARISON BETWEEN SERIAL COMMUNICATION PROTOCOLS," [Online]. Available: https://www.engineersgarage.com/blogs/comparison-between-serial-communication-protocols-spi-i2c-uartusrt-0 Accessed: 2019-02-06.

[69] T. TGY-i6S, "Remote controller," [Online]. Available: https://hobbyking.com/en_us/i6s-afhds-2a-white-mode2-6ch-radio-with-colour-box.html?____store=en_us Accessed: 2019-01-31.

[70] T. TGY-iA6C, "Telemetry receive," [Online]. Available: https://hobbyking.com/en_us/turnigy-ia6c-ppm-sbus-receiver.html Accessed: 2019-01-31.

[71] The MathWorks, Inc, "Simulink 2018," [Online]. Available: https://se.mathworks.com/products/simulink.html Accessed: 2019-01-31.

[72] C. Watkins, "Learning from delayed rewards," 01 1989.

[73] X.-R. Cao, "Stochastic learning and optimization—a sensitivity-based approach," 03 2009.

[74] A. D. Tijsma, M. M. Drugan, and M. A. Wiering, "Comparing exploration strategies for q-learning in random stochastic mazes," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec 2016, pp. 1–8.

[75] Cykelkraft, "Roller Elite Arion Mag," [Online]. Available: https://www.cykelkraft.se/trainer-elite-arion-mag-rullar?fbclid=IwAR0vV2w1C2o50TIAC-E_Yx0_dywBCWUyBwt6PPYQaE5YWAhZNAxfap2OtF8 Accessed: 2019-02-01.

Appendices

A PCB designs

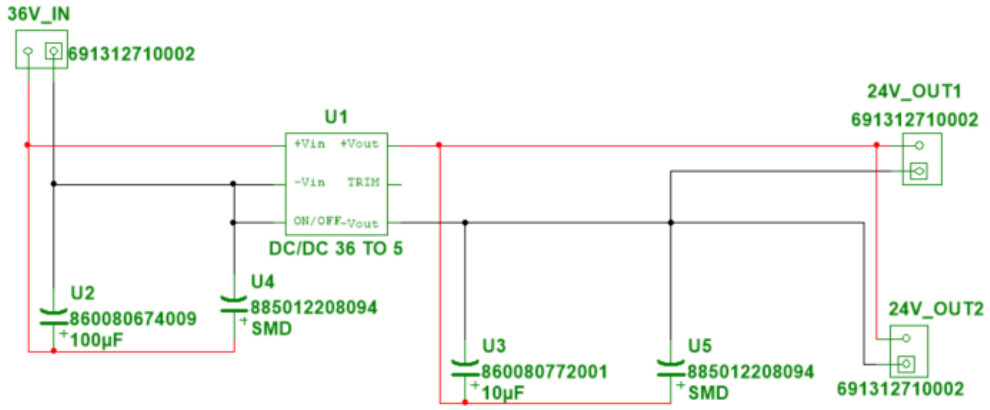


Figure 49: Electrical schematic for the 36 V to 5 V conversion, the component list is shown below in Table 7.

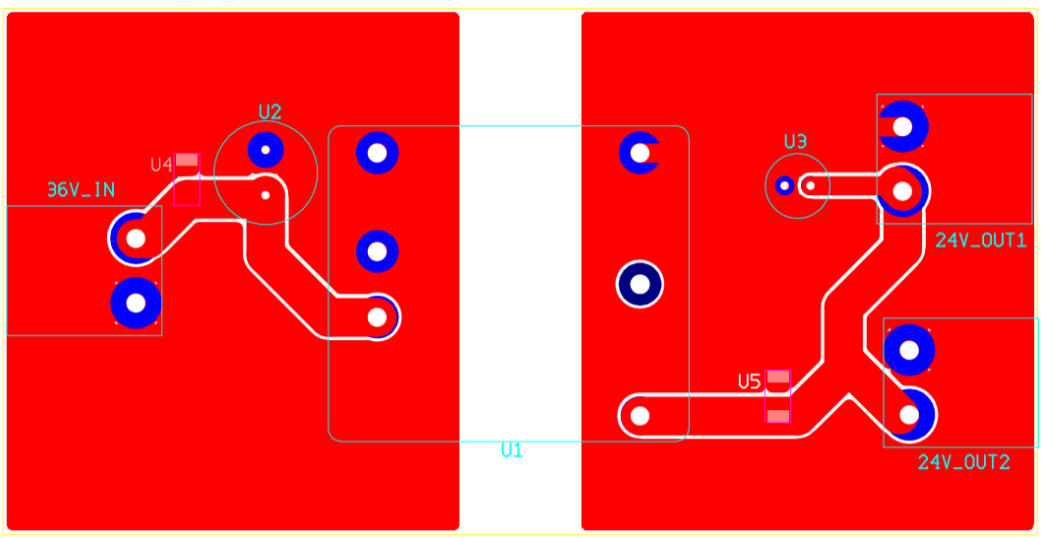


Figure 50: Ultiboard component layout for the 36 V to 5 V conversion PCB.

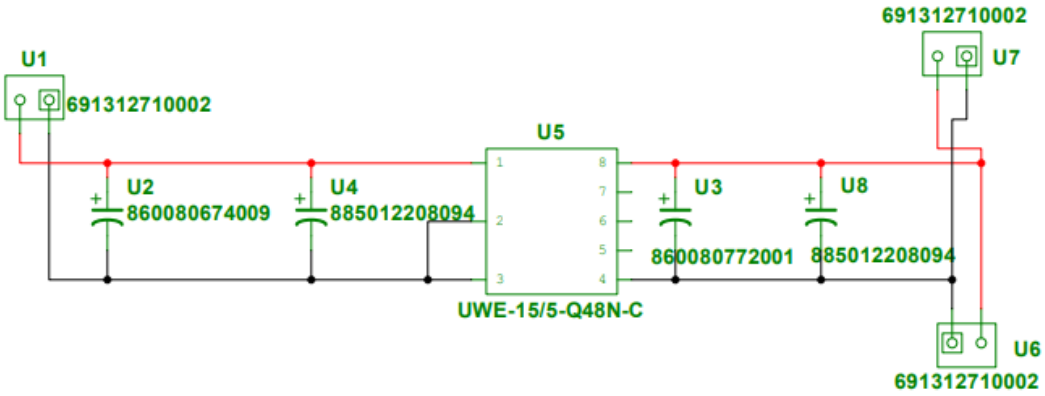


Figure 51: Electrical schematic for the 36 V to 15 V conversion, the individual components can be view in Table 8

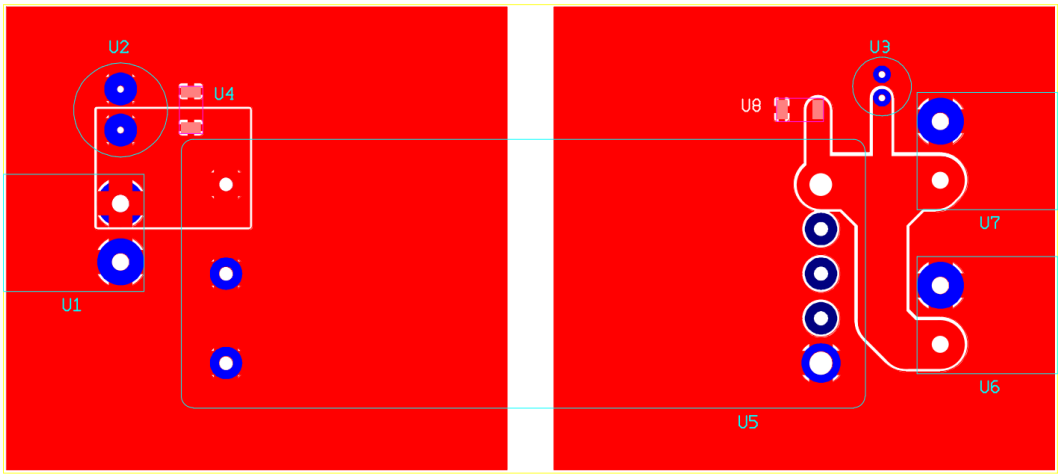


Figure 52: The PCB layout of the 15 V supply board.

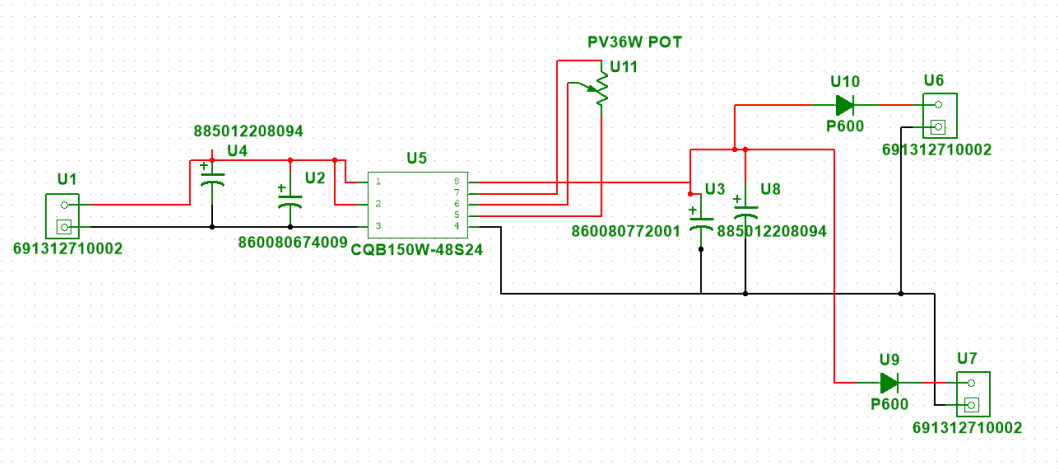


Figure 53: Electrical schematic for the 36 V to 24 V conversion, the individual components can be view in Table 9

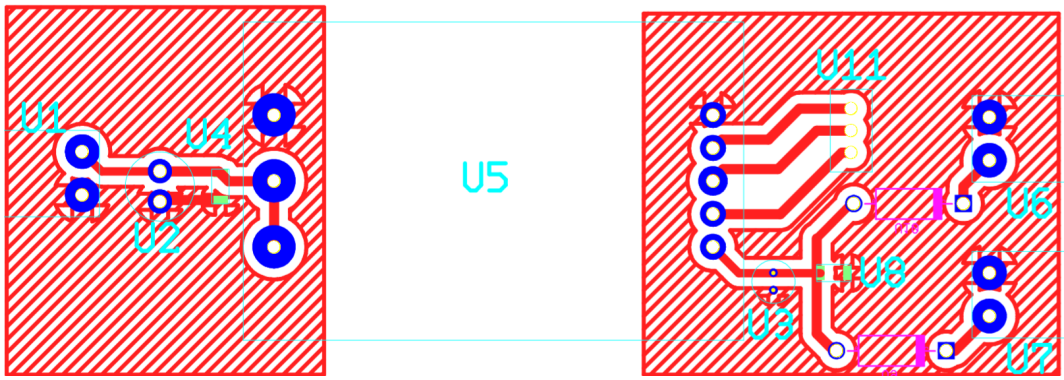


Figure 54: The PCB layout of the 24 V supply board.

Table of Materials

Supply Board 5 Volt

Schematics Comp.	Description	Part Number
36V_IN, 24V_OUT1, 24V_OUT2	WR-TBL Series 3127 5.00 mm Open Horizontal PCB Header	691312710002
U2	WCAP-ATLI Aluminum Electrolytic Capacitors 100μF	860080674009
U3	WCAP-ATLI Aluminum Electrolytic Capacitors 10μF	860080772001
U4, U5	WCAP-CSGP Ceramic Capacitors 1206	885012208094
U1	DC/DC Power Modules: 18 - 75V in 3.3V/5A out	S36SE05003NRFB

Table 7: A bill over the different parts used in the 5V supply board PCB design.

Supply Board 15 Volt

Schematics Comp.	Description	Part Number
U1, U6 , U7	WR-TBL Series 3127 5.00 mm Open Horizontal PCB Header	691312710002
U2	WCAP-ATLI Aluminum Electrolytic Capacitors 100μF	860080674009
U3	WCAP-ATLI Aluminum Electrolytic Capacitors 10μF	860080772001
U4, U8	WCAP-CSGP Ceramic Capacitors 1206	885012208094
U5	15V, 5A, Single Output Isolated 1/8 Brick DC/DC Converter, 18 - 75V Input Range	UWE-15/5-Q48N-C

Table 8: A bill over the different parts used in the 15V supply board PCB design.

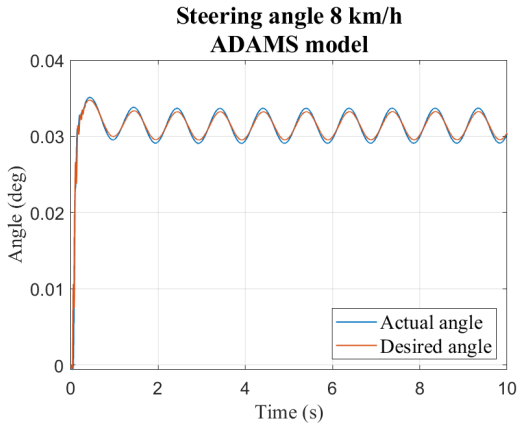
Supply Board 24 Volt

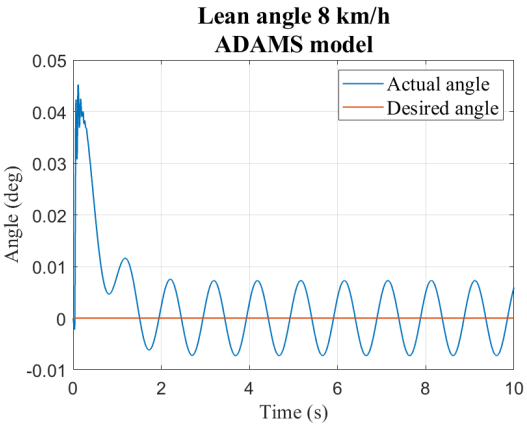
Schematics Comp.	Description	Part Number
U1, U6 , U7	WR-TBL Series 3127 5.00 mm Open Horizontal PCB Header	691312710002
U2	WCAP-ATLI Aluminum Electrolytic Capacitors 100μF	860080674009
U3	WCAP-ATLI Aluminum Electrolytic Capacitors 10μF	860080772001
U4, U8	WCAP-CSGP Ceramic Capacitors 1206	885012208094
U5	24V, 6.4A, Single Output Isolated 1/8 Brick DC/DC Converter, 18 - 75V Input Range	CQB150W-48S24

Table 9: A bill over the different parts used in the 24V supply board PCB design.

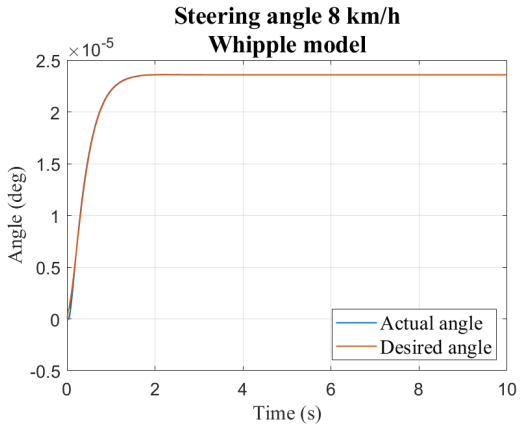
B Result closed loop controller

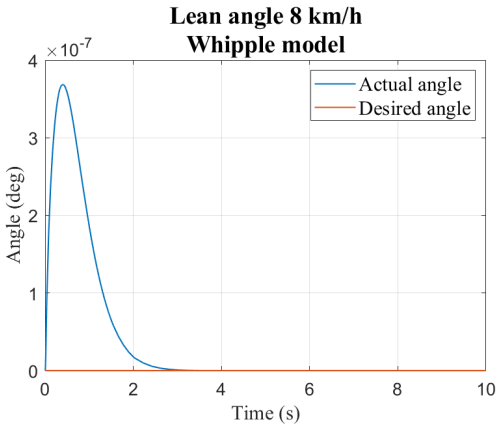
B.1 ADAMS 8 km/h



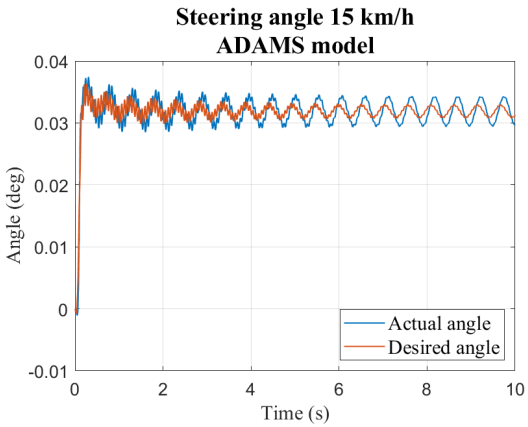


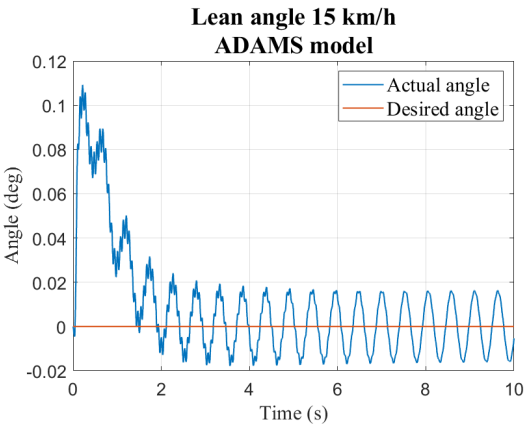
B.2 Whipple 8 km/h



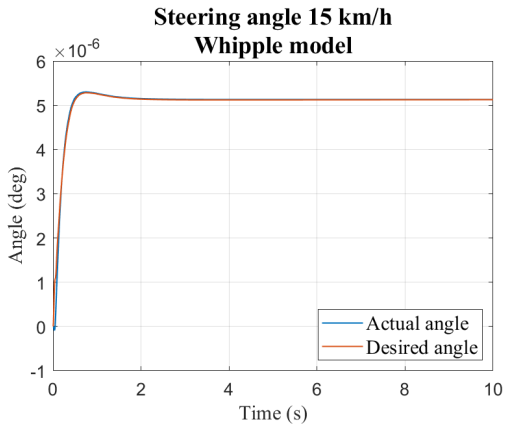


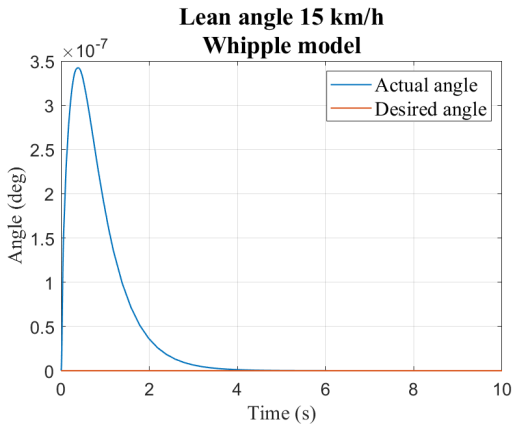
B.3 ADAMS 15 km/h





B.4 Whipple 15 km/h





C Result reinforcement learning controller