university of groningen

Stabilization and Trajectory Tracking of a Segway

Jochem van der Veen
1st supervisor: dr. L.P. Borja Rosales, prof. dr. ir. J.M.A. Scherpen
2nd supervisor: dr. ir. G. Jonker

University of Groningen
Faculty of Science and Engineering

Bachelor Integration Project IE&M

June 2018

Borglaan 2
9301 ZE Roden
(050) 5073250
(06) 29485775
j.l.van.der.veen@student.rug.nl
Student Number: 2975971

Abstract

The project considers the stabilization of a Segway (two-wheeled inverted pendulum). Throughout the project, a linearized model of a Segway system is used and assumed to be a true representation of the Segway. The first part of the project consists in the design of a Linear-Quadratic Regulator (LQR), which stabilizes the Segway in its upward configuration. In the continuation of the project, a so called 'automatic pilot' is implemented into the systems controller, stabilizing the system at a desired velocity moving in a straight line. In the project, the ability of the Segway to turn is not considered. Two controllers are proposed for this trajectory tracking. All three controllers are extensively evaluated on robustness to disturbances. In the final part of the project, the theoretical results are validated through simulations that represents different realistic scenarios. Furthermore, the performance of each control approach is evaluated.

Table of Contents

1. Introduction 1
   1.1. Motivation 1
   1.2. Preliminary problem statement 1
   1.3. Research questions 2
   1.4. Preview 2
2. Background 3
3. System and main assumptions 4
   3.1. Sensors and actuators 4
   3.2. Assumptions 4
4. System modeling 5
   4.1. State-space model 5
   4.2. Stability in the open-loop 6
   4.3. Open-loop Simulink model 7
5. Selection of control strategy 8
6. LQR control 9
   6.1. LQR design 9
      6.1.1. Cost matrix and gain vector design 9
   6.2. LQR closed-loop stability 13
   6.3. LQR implementation 14
7. Trajectory Tracking 17
   7.1. Reference position 17
   7.2. Steady-state error 18
8. Augmented system allowing integral action 19
9. LQR control with integral action 20
   9.1. LQR with integral action design 20
   9.2. LQR with integral action closed-loop stability 20
   9.3. LQR with integral action implementation 21
10. PID+LQR Control 22
   10.1. PID design 22
      10.1.1. Heuristic methods 22
      10.1.2. Simulink PID Tuner App 25
      10.1.3. Manual tuning 27
   10.2. PID+LQR closed-loop stability 28
   10.3. PID+LQR implementation 28
11. Disturbances and signal noise 30

11.1. Noise in sensor signals 30
11.1.1. Size of sensor noise 30
11.1.2. LQR controller 30
11.1.3. LQR controller with integral action 31
11.1.4. PID+LQR controller 32
11.2. Disturbances resulting from differences in users 33
11.2.1. Size of user disturbances 33
11.2.2. LQR controller 34
11.2.3. LQR controller with integral action 36
11.2.4. PID+LQR controller 37
11.3. Disturbances in error angle 39
11.3.1. Size of error angle disturbance 39
11.3.2. LQR controller 39
11.3.3. LQR controller with integral action 40
11.3.4. PID+LQR controller 41
11.4. Disturbances in the trajectory 42
11.4.1. Size of slope disturbance 43
11.4.2. LQR controller 45
11.4.3. LQR controller with integral action 46
11.4.4. PID+LQR controller 47
11.5. Testing the controllers under multiple combined disturbances 49
11.5.1. Test situation 49
11.5.2. LQR controller with integral action 49
11.5.3. PID+LQR controller 51
12. Discussion 53
13. Conclusions 54
Bibliography 55
Appendix A: Matlab script used for simulations 58

1. Introduction

1.1. Motivation

The Segway is a vehicle for short distance personal transportation. Moving the Segway that is currently available on the market is done by the user moving his balance forward and backward. The first patent for the Segway was filled in 1994 and granted in 1997 (USA Patentnr. 5,701,965, 1997). After this first patent, many patents followed (Segway Inc., 2016). The Segway was first introduced to the public in 2001, after which the first Segways were delivered to customers in 2002 (Segway Inc., 2006). Nowadays, these vehicles are widely used for various applications, for example in patrol purposes, in the medical community or in the tourism sector. When first brought to the market, the Segway inventor Dean Kamen expected to be selling 10,000 units a week by the end of 2002 (Golson, 2015). According to Forbes, however, Segway sold less than 30,000 units in over six years (Schroter, 2007). Thus, it appears that the sales numbers of the Segway were rather disappointing. According to literature (Golson, 2015), safety issues can mainly be blamed for the failure of the Segway as a product. The most notorious example of this is the death of James W. Heselden, the owner of the Segway corporation, after he steered his Segway of a cliff in 2010 (Williams, 2010). Another example of an accident which made the news was at the 2015 World Championships Athletics, were cameraman Song Tao of China's CCTV drove his Segway into Olympic athlete Usain Bolt (Rubinroit, 2016).

As mentioned previously, sales of the Segway have been far below expectation in recent years. The most likely cause of this, is the high amount of accidents involving Segways. Many websites claim that riding a Segway is intuitive and easy to learn (Segway tours Copenhagen, sd). However, when the Segway is investigated more closely, it seems that it is not that intuitive at all for new users.

1.2. Preliminary problem statement

The Segway system has been under production for years, and the aim of this work will not be to reproduce a controller comparable to the controller that is present in current Segways. Because of its disappointing sales, however, the Segway system seems to be in need for a change. In this project, the aim is to design a controller that ensures the safety of the users no matter their experience in interacting with the system.

Towards this end, an automatic pilot function to follow a simple trajectory, namely, a straight line is proposed in this project. Therefore, the control objective is twofold: first, if the automatic pilot function is activated, the Segway must move forwards at a constant velocity even in presence of slopes. Second, if the automatic pilot is not activated, the Segway must remain vertically stable even in presence of small disturbances. The presence of an automatic pilot will allow new users who are inexperienced with the Segway (for example tourists) to use the Segway safely, without the need for extensive practicing.

Thus, the preliminary problem statement is defined as: 'A controller has to be designed which stabilizes the Segway system while maintaining a constant desired velocity and meeting safety requirements, in order to improve the safety of the Segway'. The goal corresponding to this problem statement is defined as follows: 'To design a control law, consisting of a regulator with integral action, such that the Segway follows a desired trajectory rejecting disturbances'.

To meet this goal, the designed controller consists of two parts; one for vertical stabilization and one for maintaining constant velocity. The controller for vertical stabilization was specified to stabilize the Segway model from an initial error angle of $0.3\ rad$ within a time limit of five seconds. The controller for maintaining constant velocity was specified to at the very least stabilize the system at a velocity of $2\ m/s$, from the initial condition $z_i = [0 \quad 0 \quad 0 \quad 0]^T$. The allowable overshoot was specified to be

20%, the allowable rise time 1 second and the settling time 3.5 seconds. The steady state error should converge to zero. These controller requirements consider the velocity response of the system in absence of disturbances.

1.3. Research questions

In order to solve the problem in the problem definition and reach the goal of the project, the following research questions should be answered. For certain research questions, sub questions were defined in order to be able to answer these research questions.

How can the system of the Segway be modeled?

- Of what components does the Segway system exist (actuator/ sensors)?
- How can the system be linearized and modeled in state-space form?

How can this system be controlled in terms of upward configuration?

How can this systems velocity be regulated while maintaining upward stability using integral control?

- What kind of reference results in a stable system at the desired velocity?
- What type of disturbances can be expected?
- How to design a control law rejecting these disturbances?

1.4. Preview

This report sets off with the description of the problem modelling approach in the background section. Secondly, the system and main assumptions in this work will be treated. Next, the modelling of the system will be discussed. Then, stabilization of the system will be elaborated on, as well as on the implementation of the automatic pilot in the system (trajectory tracking). After that, the controller's rejection of disturbances will be considered. Finally, this report concludes by a brief conclusion and recommendations for future research.

2. Background

The Segway (or two-wheeled inverted pendulum) is a vehicle which is a popular system in control theory (Boubaker, 2012). This popularity arises from the fact that the Segway is unstable without control, but with (relatively simple) modelling and control it can be stabilized. The modelling of the Segway can be done as a simple inverted pendulum, making it a good exercise for control engineering students, for example in tutorials (University of Groningen, 2017). However, the Segway can also be modelled more extensively containing parameters of the Segway motor or the Segways environment (Younis & Abdelati, 2009) (Castro, Modeling and dynamic analysis of a two-wheeled inverted-pendulum (masters thesis), 2012). In this thesis, a linearized model of the Segway which takes the voltage applied to the DC-motor as an input will be used. The main objective is to design a controller to regulate the input voltage.

The controllers designed were tested on performance extensively by modelling in Matlab Simulink (version 2018a). The Simulink solver method used was fixed step, ode8 (Dormand & Prince, 1980). For the performance of the controllers, model responses were evaluated based on the transient response properties: rise time, overshoot, settling time, and steady-state error. Rise time refers to the time required for the signal to rise from $10\ \%$ of the desired signal gain to $90\ \%$ of the desired signal rise. Overshoot refers to the maximum percentage by which the signal exceeds its set target. Settling time refers to the time, starting from the moment the signal passes the mid reference level ($50\ \%$), required before the signal remains within a range of $2\ \%$ of the final value. The steady-state error is defined as “the difference between the desired final output and the actual one” (Lipták, 2003).

3. System and main assumptions

3.1. Sensors and actuators

The Segway in this project is assumed to contain two sensors. The first sensor is a gyroscope measuring the error angle (θ, figure 3.1) between the direction of the gravitational force and the axis perpendicular to the Mats of the Segway (the base on which the user is standing (Segway Inc. , 2014)). The gyroscope is a sensor which inevitably contains noise in its measured signal.

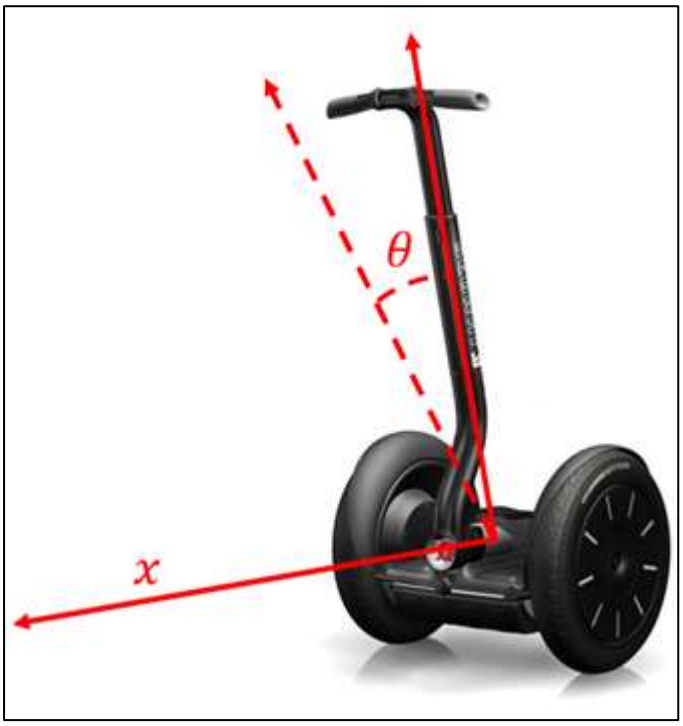

Figure 3.1: Solidworks model of a Segway with indicated distance (x) and error angle (θ)

The second sensor is a tachometer, measuring the velocity of the Segways wheels. From the velocity of the wheels, the velocity of the Segway ($\dot{x}$, derivative of x, see figure 3.1) is calculated. Because the trajectory in this work is a straight line, the velocity of both wheels is identical. Should the ability of turning be implemented into the system, the velocity would be measured by two tachometers. The velocity is then equal to the average of the two measured velocities. The tachometer is assumed to contain noise in its measurement signal as well.

The actuators considered are two DC-motors, and each motor drives its own wheel. As the option of turning was not considered, the force applied to the surface by the wheels was equal to double the torque generated by one DC-motor times the radius of the wheels. The maximum voltage allowed over the DC-motors was 48 volts.

3.2. Assumptions

The assumptions done prior to starting this project are listed below.

- The Segway is moving in a straight line; the possibility of turning is not considered.
- The sensors in the Segway measure the velocity and the error angle of the system.
- The DC-motors have a saturated voltage between -48 and $+48$ volts.
- The wheels of the Segway remain in contact with the ground.
- There is no slipping of the wheels.
- Segway users do not express unexpected behaviour and follow the user manual provided by Segway. For the model, this means the user does not move; he or she is modelled as a rigid mass.
- A typical initial error angle resulting from a user mounting the Segway is 10 degrees or less.

4. System modeling

4.1. State-space model

The model of the system is largely based on the Grasser model (Grasser, D'Arrigo, Colombi, & Rufer, 2002) and the Younis model (Younis & Abdelati, 2009). The model used is derived from the linearized equations of motion of a two-wheeled inverted pendulum and the linear model for a DC-motor (more on the derivation of the force generated by the DC-motors in section 11.4). The dynamics of these systems are modeled separately in the beginning but are brought together in two equations of motion, resulting in the model of the system. States of the model are the distance covered by the Segway (x), velocity of the Segway ($\dot{x}$), error angle of the Segway compared to the upright position (θ), and angular velocity of the error angle ($\dot{\theta}$). The input to the system is given by the voltage applied to the two DC-motors; each wheel is connected to its own DC-motor. The model of the system assumes the Segway to move in a straight line. Thus, there is no differential mode present that makes the Segway turn. Furthermore, the model assumes the Segway's wheels to stay in contact with the ground without slipping.

The linearized state-space equation for the modelled Segway system is a state-space model of the form;

$$\begin{aligned} \dot{z} &= Az + Bu \\ y &= Cz \end{aligned} \tag{4.1}$$

With;

$$z = \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{4.2}$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \dfrac{2k_m k_e (M_p l r - I_p - M_p l^2)}{R r^2 \alpha} & \dfrac{M_p^2 g l^2}{\alpha} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \dfrac{2k_m k_e (r\beta - M_p l)}{R r^2 \alpha} & \dfrac{M_p g l \beta}{\alpha} & 0 \end{bmatrix} \tag{4.3}$$

$$B = \begin{bmatrix} 0 \\ \dfrac{2k_m (I_p + M_p l^2 - M_p l r)}{R r \alpha} \\ 0 \\ \dfrac{2k_m (M_p l - r\beta)}{R r \alpha} \end{bmatrix} \tag{4.4}$$

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.5}$$

And;

$$\alpha = I_p \beta + 2M_p l^2 \left(M_w \frac{I_w}{r^2} \right) \tag{4.6}$$

$$\beta = 2M_w + \frac{2I_w}{r^2} + M_p \tag{4.7}$$

Table 4.1: Definition of system parameters (Younis & Abdelati, 2009)

Parameter	Definition	Value
$\boldsymbol{x}$	Distance	m
$\dot{\boldsymbol{x}}$	Velocity	m/s
$\ddot{\boldsymbol{x}}$	Acceleration	m/s^2
$\boldsymbol{\theta}$	Error angle	rad
$\dot{\boldsymbol{\theta}}$	Error angular velocity	rad/s
$\ddot{\boldsymbol{\theta}}$	Error angular acceleration	rad/s^2
$\boldsymbol{u}$	Voltage applied to the DC-motor	V
$\boldsymbol{k_m}$	Constant of the motor torque	$0.869\ Nm/A$
$\boldsymbol{k_e}$	Constant of the motor's back-EMF	$0.083\ Vs/rad$
$\boldsymbol{l}$	Length of the pendulum	$1.7\ m$
$\boldsymbol{r}$	Wheel radius	$0.2\ m$
$\boldsymbol{R}$	Resistance of the motor	$1\ \Omega$
$\boldsymbol{M_p}$	Mass of the pendulum	$85\ kg$
$\boldsymbol{I_p}$	Moment of inertia of the pendulum	$68.98\ kg \cdot m^2$
$\boldsymbol{M_w}$	Mass of the wheel	$3.5\ kg$
$\boldsymbol{I_w}$	Moment of inertia of the wheel	$0.07\ kg \cdot m^2$
$\boldsymbol{g}$	Acceleration of gravity	$9.81\ m/s^2$

The system states used in the model, along with the values of parameters which remain constant, are displayed in table 4.1. The system parameter values in table 4.1 were considered to be the same as in the article by Younis and Abdelati (Younis & Abdelati, 2009). When the parameter values from table 4.1 were plugged into equations (4.3)-(4.7), and equations (4.2)-(4.7) were plugged into equation (4.1) the following state-space model in equation (4.8) was obtained.

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.1074 & 21.3441 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -0.0471 & 14.1063 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0.2587 \\ 0 \\ 0.1136 \end{bmatrix} u$$
$$y = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{4.8}$$

4.2. Stability in the open-loop system

Following from theory, a state space system is asymptotically stable if and only if the eigenvalues of A all have a strictly negative real part, and is unstable if at least one eigenvalue of A has a strictly positive real part (de Persis, Control Engineering – Lecture 4, 2017).

$$eig(A) = \begin{bmatrix} 0 \\ -3.79 \\ -0.04 \\ 3.72 \end{bmatrix} \tag{4.9}$$

Using Matlab 2018a, the eigenvalues of the open loop system matrix A were calculated as in equation (4.9). As predicted, the open loop was not stable, as the fourth eigenvalue had a positive real part.

4.3. Open-loop Simulink model

The open-loop state-space matrices obtained (A, B, C, D) were inserted into the program Matlab 2018a. The open-loop model was then produced in Matlab Simulink (version 2018a), as illustrated in figure 4.1.

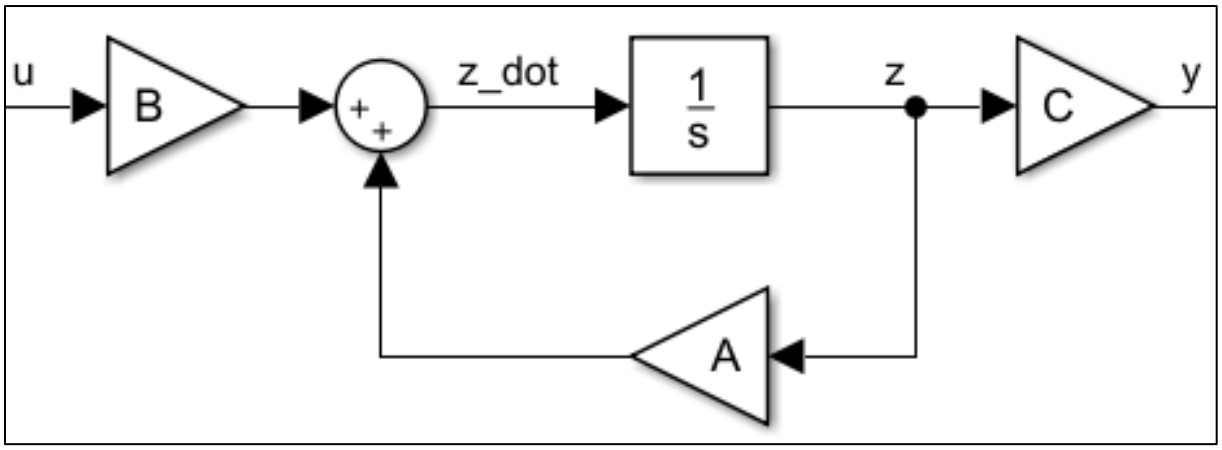

Figure 4.2: Open-loop state space system in Matlab Simulink

5. Selection of control strategy

Before designing a controller, the control strategy was decided on. First of all, the vertical stabilization was considered. It was decided that the upward stabilization of the Segway in the equilibrium point ($z_e = [0 \quad 0 \quad 0 \quad 0]^T$) should be optimal. In order to achieve this, a Linear Quadratic Regulator (LQR) was proposed, as described in the next section.

For trajectory tracking, the system is required to stabilize at a desired reference velocity ($z = [0 \quad 0 \quad 0 \quad 0]^T$). It was required that the controller would be able to reject constant disturbances. For rejecting disturbances while maintaining a constant velocity, integral action in the controller was required (Franklin, Powell, & Workman, 1990). In order to apply integral action in the controller, first, the method of integral control via linearization (Khalil, 1996) was considered. This approach uses LQR not only for the error of the states, but also for the integrated errors, by using an augmented system. This augmented system contains both the states, as well as the desired states' integrated error, resulting in a LQR controller with integral action. However, when trying this approach to control both the error angle and the velocity of the system, this method turned out to result in an uncontrollable model, implying that this approach would lead to instability of the closed-loop system. The only error which could be integrated resulting in a controllable system was that of the position. Although this type of control was not as optimal as when both the error velocity and the error angle could be integrated, this controller was still evaluated based on performance.

Because the integral control via linearization method was not as applicable as desired, other methods were evaluated. In the end, it was chosen to design the controller such that the upward stabilization of the system at velocity equal to zero was done by a LQR controller, while the system stabilization at a desired velocity was done by a PID regulator (over the position error of the Segway) combined with the LQR controller, like in the paper 'Optimal Control of Nonlinear Inverted Pendulum System Using PID Controller and LQR: Performance Analysis Without and With Disturbance Input' (Prasad, Tyagi, & Gupta, 2015). In this PID the integral of the position was computed by using the method of integral control, using an augmented matrix like in the controller discussed before. As though it would seem applying two controllers to a system at the same time would result in the controllers counteracting each other, by tuning of the PID satisfactory results were obtained from the linearized model.

The LQR controller for vertical stabilization, as well as the LQR controller with integral action and the PID+LQR controller for trajectory tracking were extensively tested on performance.

6. LQR control

6.1. LQR design

For the system to return to its equilibrium position ($z_e = [0 \quad 0 \quad 0 \quad 0]^T$) in an optimal way, a linear-quadratic regulator (LQR) was implemented into the system. The LQR controller consists of a feedback gain which returns the system to its equilibrium position while minimizing the 'cost'. Thus, the LQR is an optimal controller, as it minimizes cost. The 'cost' in this system is the error of actual states compared to the desired states. For example when we desire to stabilize the system at a velocity of $3\,m/s$ $(10.8\,km/h)$ and the current velocity is $2\,m/s$ $(7.2\,km/h)$, the velocity error is $1\,m/s$ $(3.6\,km/h)$. In the first part of the project we desire to stabilize the system around the equilibrium position. Thus, the error of any state is equal to the states value, i.e. $e_\theta = \theta$.

$$J = \int_0^\infty [z(\tau)^T Q z(\tau) + u(\tau)^T R u(\tau)] d\tau \tag{6.1}$$

In the quadratic cost function in equation (6.1), the cost assigned to each state is given by the Q matrix, while the value of R assigns a cost to the input signal.

$$\mathrm{A}^T\mathrm{P} + \mathrm{PA} - \mathrm{PBR}^{-1}\mathrm{B}^T\mathrm{P} + \mathrm{Q} = 0 \tag{6.2}$$

$$\mathrm{K} = \mathrm{R}^{-1}\mathrm{B}^T\mathrm{P} \tag{6.3}$$

The cost matrices (Q and R) are then plugged in to the Algebraic Riccati Equation (6.2), in order to find P, where $P = P^T > 0$. The found P matrix is then plugged in to equation (6.3), obtaining the LQR feedback gain K.

The LQR controller that was proposed is essentially a function of the cost matrices. Thus, the selection of values for the cost matrices had a large influence on the quality of the resulting LQR control law.

6.1.1. Cost matrix and gain vector design

The values of the Q and R matrices in equation (6.1) were computed using multiple methods (Murray, 2006). From these methods, the method resulting in the best system response to an initial error angle of $0.3\,rad$ ($z_i = [0 \quad 0 \quad 0.3 \quad 0]^T$) was selected based on rise time, overshoot and settling time. The initial angle of $0.3\,rad$ was chosen based on the assumption that the initial error angle would never be greater than this. This section discusses the design of the cost matrices Q and R.

Simple cost matrix design

For this method, The Q and R matrices where chosen as in equations (6.4) and (6.5).

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.4}$$

$$R = \rho[1] \tag{6.5}$$

In order to obtain a better response, ρ could be varied and was chosen as $\rho = 0.0001$. After plugging these cost matrices into equation (6.1), the cost function is obtained as in equation (6.6).

$$J = \int_0^\infty [x(\tau)^2 + \dot{x}(\tau)^2 + \theta(\tau)^2 + \dot{\theta}(\tau)^2 + \rho u(\tau)^2] d\tau \tag{6.6}$$

After plugging the cost matrices into equation (6.2), and plugging the obtained matrix P into equation (6.3), the LQR feedback gain vector was obtained as in equation (6.7). The models response to an initial error angle of $0.3\ rad$ is displayed in figure 6.1 below.

$$\mathrm{K} = [-100 \quad -198.4 \quad 1700 \quad 747.5] \tag{6.7}$$

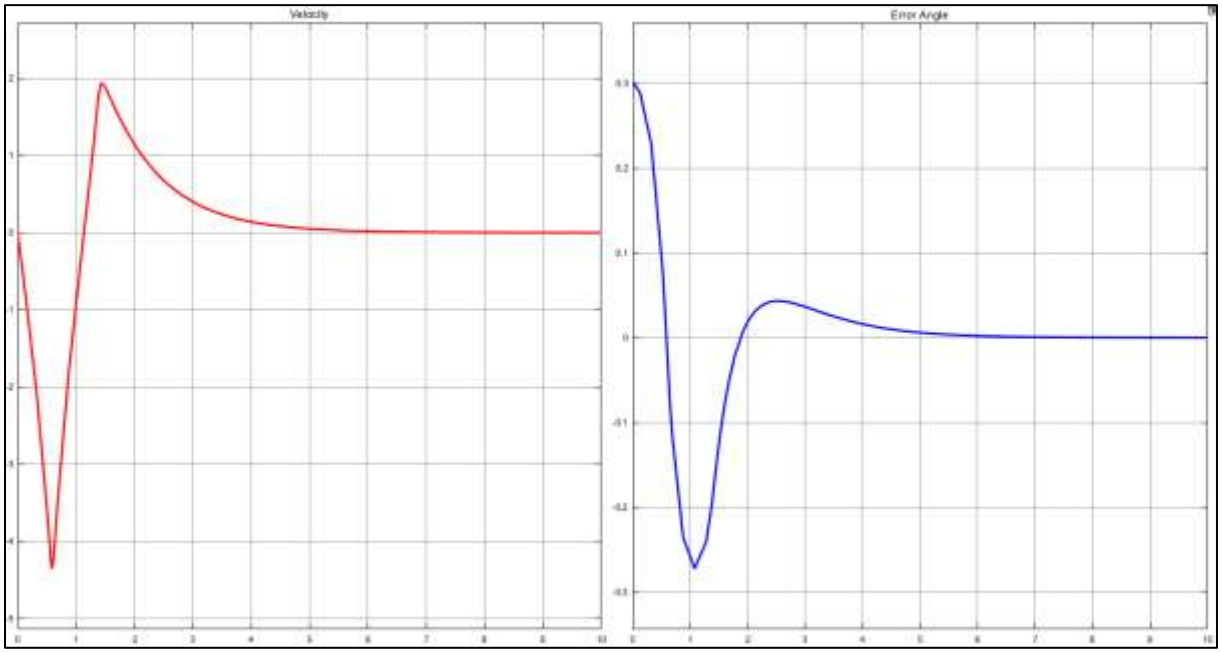

Figure 6.1: Systems response to an initial angle of $0.3\ rad$ with an LQR controller based on simple design

Diagonal weights

Using this method, the design of the cost matrices was based on the allowable error of all states ($e_{i,allowable}$) and the allowable value of the input ($V_{allowable}$). The allowable errors were based on the maximum initial conditions for which the linearized Segway system remained stable using an LQR controller based on values for the Q and R values found in literature (Vinodh Kumar & Jerome, 2013). These values where modified to be more realistic in a real Segway, as the linearized model is only valid for a tight area around the equilibrium point ($z_e = [0 \quad 0 \quad 0 \quad 0]^T$). The value for the allowable input arises from the fact that the motor has a saturation between -48 and $+48$ volts.

$$e_{x,allowable} = 1\ m \tag{6.8}$$

$$e_{\dot{x},allowable} = 1\ m/s \tag{6.9}$$

$$e_{\theta,allowable} = 0.3\ rad \tag{6.10}$$

$$e_{\dot{\theta},allowable} = 0.8\ rad/s \tag{6.11}$$

$$V_{allowable} = 48\ V \tag{6.12}$$

The allowable errors and the allowable input used to compute the Q and R matrices for the LQR controller are given in equations (6.8)-(6.12). How the Q and R matrices were designed using this method will be described below.

$$Q = \begin{bmatrix} q_x & 0 & 0 & 0 \\ 0 & q_{\dot{x}} & 0 & 0 \\ 0 & 0 & q_\theta & 0 \\ 0 & 0 & 0 & q_{\dot{\theta}} \end{bmatrix} \tag{6.13}$$

Q is a diagonal matrix with dimensions $4x4$ (equation (6.13)), with the weights to be computed on the diagonal. Meanwhile R is a scalar which was computed as well.

$$J = \int_0^\infty [q_x\, x(\tau)^2 + q_{\dot{x}}\, \dot{x}(\tau)^2 + q_\theta \theta(\tau)^2 + q_{\dot{\theta}}\, \dot{\theta}(\tau)^2 + Ru(\tau)^2] d\tau \tag{6.14}$$

When the matrix in equation (6.13) was inserted into the quadratic cost function (equation (6.1)), the resulting cost function was obtained in equation (6.14).

$$q_i {e_{i,allowable}}^2 = 1 \tag{6.15}$$

$$Ru^2 = 1 \tag{6.16}$$

Then, values of q_i and R were calculated for any state i such that equations (6.15) and (6.16) were solved.

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 11.1 & 0 \\ 0 & 0 & 0 & 1.6 \end{bmatrix} \tag{6.17}$$

$$R = 0.00043 \tag{6.18}$$

The resulting values in the Q and R matrices are in equations (6.17) and (6.18) above. The value of R could still be adjusted in order to tune the input/state balance, however, using trial and error it was found that changing the value of R did not result in a better controller performance.

After plugging the cost matrices into equation (6.2), and plugging the obtained matrix P into equation (6.3), the LQR feedback gain vector was obtained as in equation (6.19). The systems response to an initial error angle of $0.3\ rad$ is displayed in figure 6.2 below.

$$\mathrm{K} = [-48.2 \quad -99.6 \quad 965.5 \quad 401.7] \tag{6.19}$$

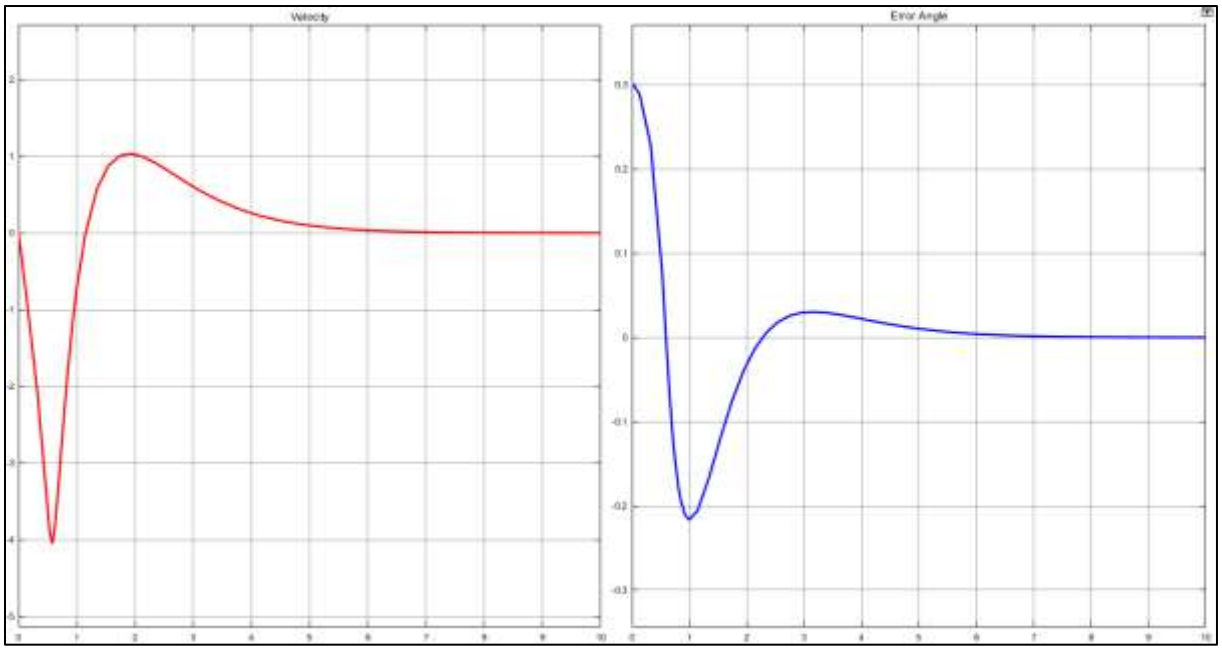

Figure 6.2: Systems response to an initial angle of $0.3\ rad$ *with an LQR controller based on diagonal weights*

Output weighting

In this method, $v = Hz$ is the output which is desired to remain small. Thus, the vector H had to be designed. For the values in H, the allowable state errors in equations (6.8)-(6.11) where used as in equation (6.20).

$$H = \begin{bmatrix} e_{x,allowable}{}^{-2} \\ e_{\dot{x},allowable}{}^{-2} \\ e_{\theta,allowable}{}^{-2} \\ e_{\dot{\theta},allowable}{}^{-2} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 11.1 \\ 1.6 \end{bmatrix} \tag{6.20}$$

$$Q = H^T H = \begin{bmatrix} 1 & 1 & 11.1 & 1.6 \\ 1 & 1 & 11.1 & 1.6 \\ 11.1 & 11.1 & 123.21 & 17.76 \\ 1.6 & 1.6 & 17.76 & 2.56 \end{bmatrix} \tag{6.21}$$

$$R = \rho = 0.00043 \tag{6.22}$$

Next, the Q and R matrixes were computed as in equations (6.21) and (6.22). The value of R can be chosen arbitrarily in this method, in order to obtain a good response. Thus, the value of R was taken from equation (6.18).

After plugging the cost matrices into equation (6.2), and plugging the obtained matrix P into equation (6.3), the LQR feedback gain vector was obtained as in equation (6.23). The systems response to an initial error angle of $0.3\ rad$ is displayed in figure 6.3 below.

$$\mathrm{K} = [-48.2 \quad -135.9 \quad 1382.2 \quad 548.2] \tag{6.23}$$

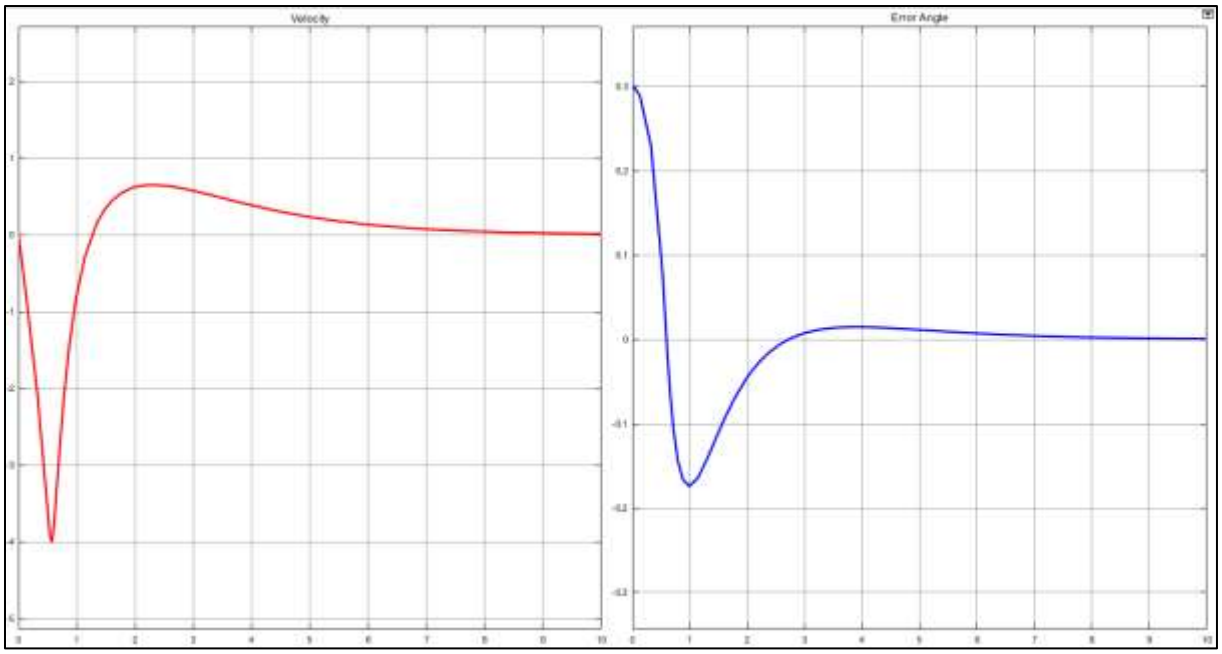

Figure 6.3: Systems response to an initial angle of $0.3\ rad$ *with an LQR controller based on diagonal weights*

Cost matrix and gain vector selection

From the three cost matrices and gain vectors computed previously, the best solution was selected. This selection was based on the systems response to an initial error angle of $0.3\ rad$ in terms of rise time, overshoot and settling time in the Segways velocity. The results are in table 6.1 below.

Table 6.1: Performance of different LQR designs

Method	LQR feedback gain vector	Rise time	Overshoot	Settling time
Simple design	$[-100 \quad -198.4 \quad 1700 \quad 747.5]$	$0.42\ s$	$44.85\ \%$	$3.48\ s$
Diagonal weights	$[-48.2 \quad -99.6 \quad 965.5 \quad 401.7]$	$0.32\ s$	$37.27\ \%$	$4.82\ s$
Output weighting	$[-48.2 \quad -135.9 \quad 1382.2 \quad 548.2]$	$0.48\ s$	$15.88\ \%$	$5.75\ s$

As can be seen, these controller's responses were quite different from each other. The controller that was selected for the continuation of the project was the LQR feedback gain vector based on the diagonal weights method. The resulting full-state feedback law obtained is in equation (6.24)

$$u = -Kz = 48.2x + 99.6\dot{x} - 965.5\theta - 401.7\dot{\theta} \tag{6.24}$$

6.2. LQR closed-loop stability

When the feedback law in equation (6.24) was substituted back into equation (4.1), the following closed loop system was obtained (equation (6.25)):

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = (A - BK)z = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 12.4771 & 25.6658 & -228.4663 & -103.9368 \\ 0 & 0 & 0 & 1 \\ 5.4759 & 11.2641 & -95.5295 & -45.6153 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{6.25}$$

Calculating the closed-loop systems eigenvalues in Matlab 2018a yielded the following results (equation (6.26)):

$$eig(A - BK) = \begin{bmatrix} -14.91 \\ -2.57 \\ -1.23 + 0.14i \\ -1.23 - 0.14i \end{bmatrix} \tag{6.26}$$

As can be seen in the above equation, the real parts of the eigenvalues of the closed-loop system ($A - BK$) are all strictly negative. Thus we can say that the closed loop system will be stable when using the LQR controller to maintain upward stability of the Segway.

6.3. LQR implementation

As the LQR controller was added to the open-loop system, the closed-loop Simulink model in figure 6.4 below was obtained. A saturation was added in the control signal because the Segway is actuated by two 48V DC-motors. Thus, the control signal was saturated between -48 and $+48$ volts. Note that there is no reference present, because the system is stabilized around the equilibrium point ($z_e = [0 \quad 0 \quad 0 \quad 0]$).

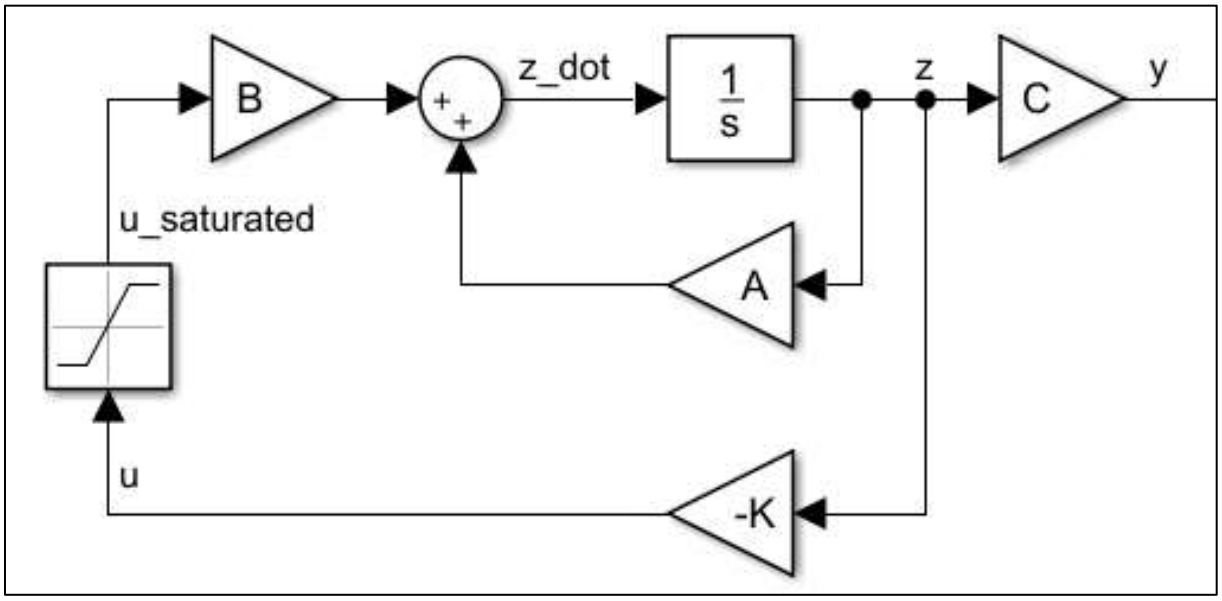

Figure 6.4: Closed-loop system with LQR control

The initial states of the Simulink model in figure 6.4 were entered in the integrator block. For the first part of the project, the state which was considered was the error angle. This is why it was investigated under what maximum initial error angle (with all other initial states at zero) the system would be able to maintain vertical stability. This was done by starting in the equilibrium point and gradually increasing the initial error angle.

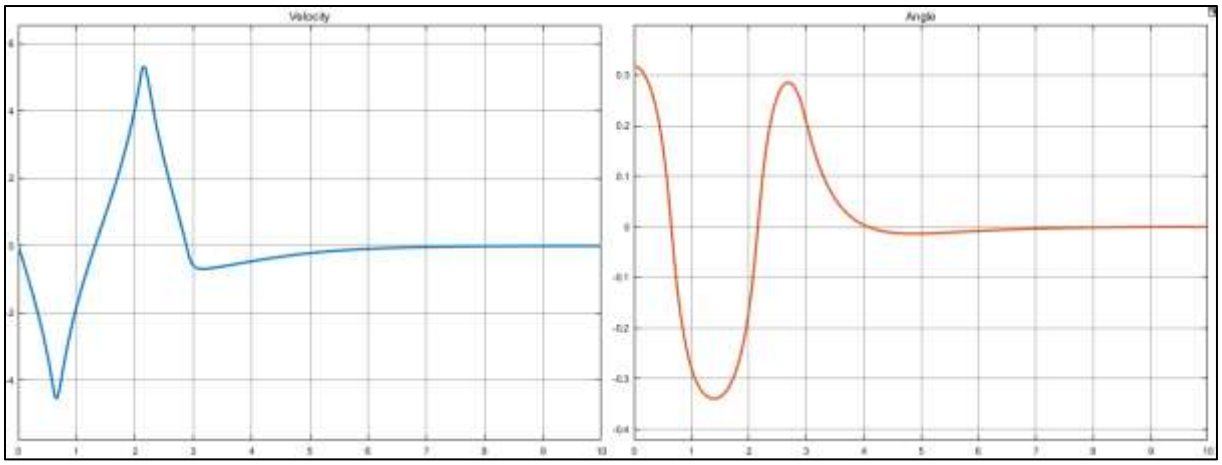

Figure 6.5: Systems response to an initial condition of $z_i = [0 \quad 0 \quad 0.317 \quad 0]$

The maximum error angle from which the system stabilized with the LQR controller was $0.317\ rad$. This is equivalent to roughly 18.1 degrees. Accordingly, the minimal initial error angle from which the system stabilized was $-0.317\ rad$. This is more than sufficient for the user to be able to mount the Segway safely. The systems response for an initial error angle of $0.317\ rad$ is displayed in figure 6.5.

The systems limiting factor in terms of stability is the saturation of the motor. Increasing the range of this saturation would allow for a larger range of initial error angles from which the system would be able to stabilize itself.

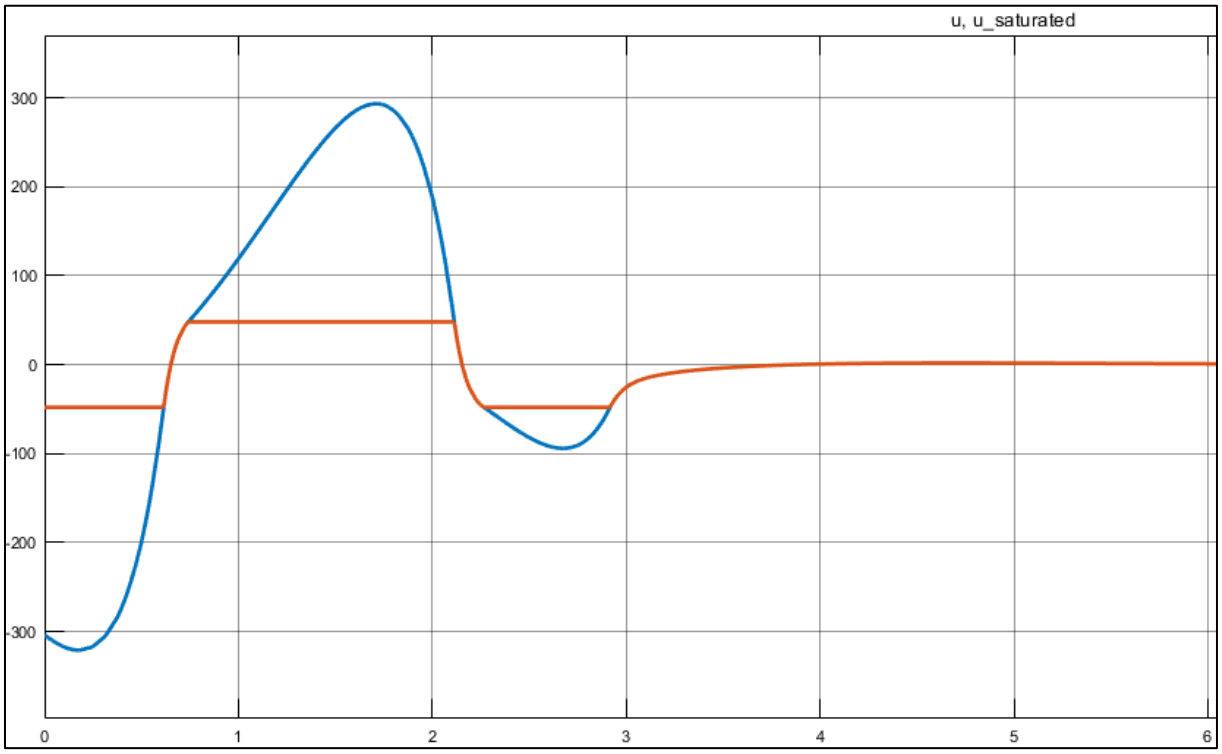

Figure 6.6: Control signal response to an initial condition of $z_i = [0 \quad 0 \quad 0.317 \quad 0]$

The control signal corresponding to the initial error angle of $0.317\ rad$ is displayed in figure 6.6. The blue line represents the input signal, while the orange line represents the saturated input signal. When the initial error angle is a little larger than $0.317\ rad$ (for example $0.3175\ rad$), the input allowed by

the saturation is not able stabilize the system. As a result, the control signal will then decrease infinitely;

$$\lim_{t\to\infty} u = -\infty \tag{6.27}$$

When the initial error angle is more than a little larger than $0.317\ rad$ (for example $0.32\ rad$ or more), the input allowed by the saturation is not powerful enough to stabilize the system and it becomes unstable even sooner. This results in the control signal increasing infinitely;

$$\lim_{t\to\infty} u = \infty \tag{6.28}$$

7. Trajectory Tracking

7.1. Reference position

For trajectory tracking of the Segway, it is desired that the system does not only stabilize in the vertical position ($\theta = \dot{\theta} = 0$), but also to be able to control the position and velocity of the system. In order to do so, a reference position (r) is compared to the actual output position. The resulting position error (e_x) is then added to the output of the LQR gain, resulting in the input signal. The resulting control feedback law is in equation (7.1):

$$u = -Kz + e_x = -[-48.2 \quad -135.9 \quad 1382.2 \quad 548.2]\begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + (x - r) \tag{7.1}$$

As before, this input signal is then saturated and put through to the open-loop state space system.

Because the automatic pilot requires the system to stabilize at a continuous velocity, the reference position was given by a ramp function. The slope of this ramp function is equal to the desired velocity ($\dot{x}_d$). Thus, the desired distance covered (x_d) is equal to the value of this ramp input.

$$C_1 = [0 \quad 1 \quad 0 \quad 0] \tag{7.2}$$

$$C_2 = [0 \quad 0 \quad 1 \quad 0] \tag{7.3}$$

$$C_3 = [0 \quad 0 \quad 1 \quad 0] \tag{7.4}$$

In order to enable the model to compare the position state to the reference position, the output matrix in equation (7.2) was designed in Matlab 2018a. Likewise, the output matrices in equations (7.3) and (7.4) were designed in Matlab 2018a to enable the system to display the states of velocity and error angle, using a scope block in Simulink.

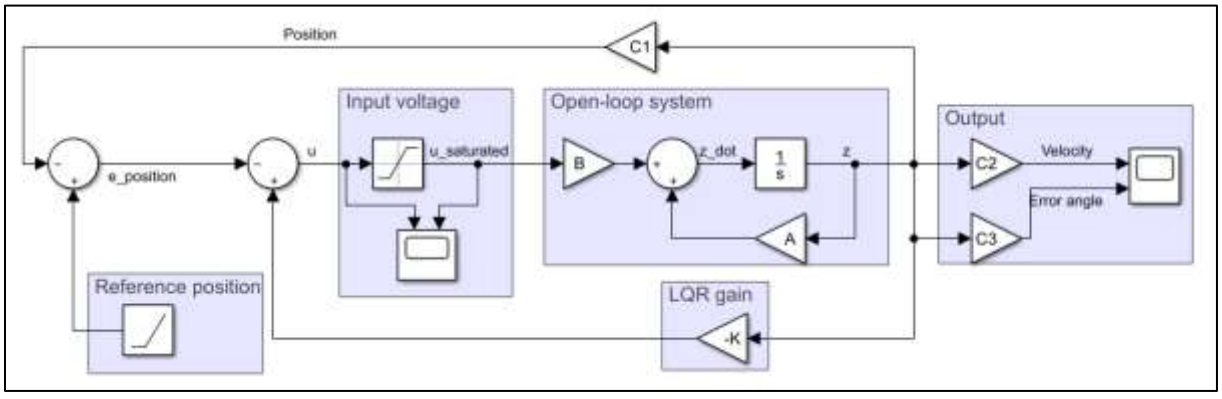

Figure 7.1: Closed-loop system with LQR control and reference position

In figure 7.1 above, the obtained Simulink model is displayed. In order to clarify the purpose of all parts of the model and decrease clutter, all parts of the Simulink model were placed in larger blocks containing headers which describe the purpose of each part.

Turning the automatic pilot off will result in the system stabilizing in the position $[0 \quad 0 \quad 0 \quad 0]$ by using the LQR feedback gain, while turning it on should result in the controller stabilizing the system in the position $z = [x_d \quad \dot{x}_d \quad 0 \quad 0]$, which is equal to $z = \left[r \quad \frac{r}{t} \quad 0 \quad 0\right]$ (with x_d the desired

distance, $\dot{x}_d$ the desired velocity, r the reference position and t the time since the automatic pilot was activated). The desired velocity is equal to the slope of the ramp input.

7.2. Steady-state error

Because of the set-up of the Simulink model, the system in figure 7.1 results in a steady-state error in the velocity of the Segway. This steady-state error is caused by the fact that at some point the control signal becomes constant, i.e. the derivative of the control law in equation (7.1) becomes equal to zero. This results in a velocity proportionally lower than the desired velocity.

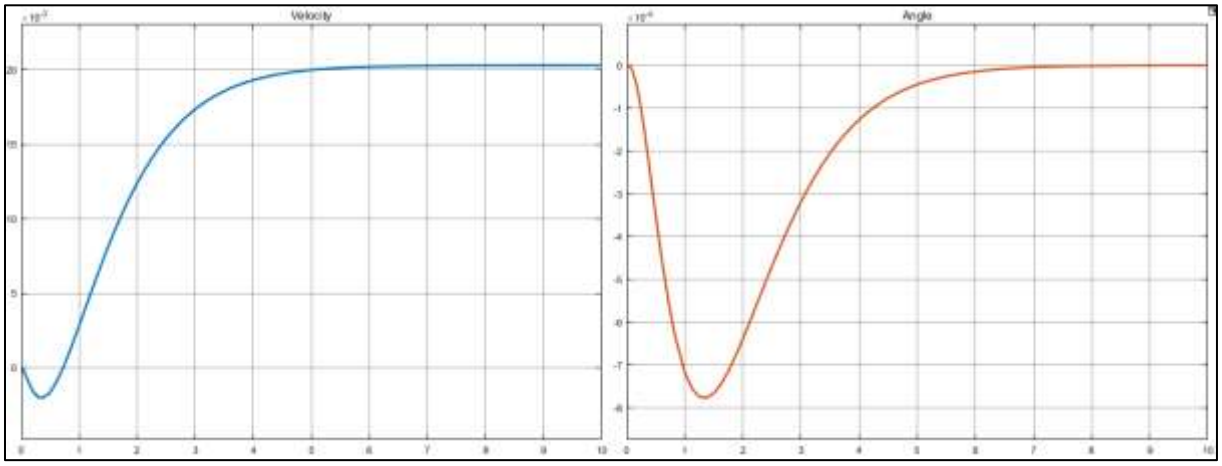

Figure 7.2: Systems response to a ramp reference signal with a slope of 1.0

For example a ramp reference with a slope of 1.0 results in the system response in figure 7.2. The velocity stabilizes at $0.02\ m/s$, while the velocity of the Segway should obviously rise to $1.0\ m/s$. This indicates that the resulting steady-state velocity of the system was almost 50 times smaller than it was desired to be.

In order to eliminate the steady-state error, a second controller had to be proposed. In the continuation of this report, the implementation of two controllers with integral action into the system will be discussed. The reason to choose for integral action was that it does not only eliminate the steady state error in the closed-loop system in the absence of disturbances, but rejects disturbances as well. Thus, the implemented controllers will play an important role in the finalized Segway system, as the performance of the system with both controllers under disturbances was extensively evaluated. Note that disturbances that can be expected will be matters related to the environment of the Segway, for example differences in users or slopes in the Segways' path.

Next, the controller based on integral action in the LQR using an augmented system will be discussed. After that, the controller based on the LQR computed before, combined with a PID controller on the Segways position will be elaborated on.

8. Augmented system allowing integral action

In order to be able to implement integral action into the Simulink model and the state-space system, the new state of the integrated position error (σ) had to be added to the system. Also the new system should be about the state errors instead of the states, because both the PID+LQR controller and the LQR controller with integral action use the state errors to compute an input signal. In order to do this, the augmented system in equation (8.1) was proposed (Khalil, 1996):

$$\begin{bmatrix} \dot{e} \\ \dot{\sigma} \end{bmatrix} = \begin{bmatrix} A & 0 \\ C_1 & 0 \end{bmatrix} \begin{bmatrix} e \\ \sigma \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u \tag{8.1}$$

$$\begin{bmatrix} e \\ \sigma \end{bmatrix} = \begin{bmatrix} e_x \\ e_{\dot{x}} \\ e_\theta \\ e_{\dot{\theta}} \\ \sigma \end{bmatrix} = \begin{bmatrix} x - r \\ \dot{x} - \dot{r} \\ \theta \\ \dot{\theta} \\ \int (x - r) \end{bmatrix} \tag{8.2}$$

In equation (8.2), the error state vector is defined. This vector allows the LQR controller (with or without integral action) to compute the voltage input from the error of the states (and the integrated position error). On the other hand, this vector allows PID action about the position by using e_x, σ and $e_{\dot{x}}$. Note that in a conventional PID controller, the derivative action would be a function of the derivative of the position error ($\frac{d}{dx} e_x = \dot{e}_x$). In the Segway system, however, the derivative of the position error is theoretically equal to the velocity error, i.e. $\dot{e}_x = e_{\dot{x}}$. Thus, the vector in equation (8.2) could be used to compute the input signal for the Segway.

9. LQR control with integral action

9.1. LQR with integral action design

For the LQR controller with integral action, an LQR controller like the LQR controller for upward stabilization was designed using the same method. The difference however, was that in this case the augmented system in equation (8.1) was used. For this augmented system, a different error cost matrix $Q_{augmented}$ had to be designed. R remained as in equation (6.18).

$$Q_{augmented} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 11.1 & 0 & 0 \\ 0 & 0 & 0 & 1.6 & 0 \\ 0 & 0 & 0 & 0 & 5 \end{bmatrix} \tag{9.1}$$

$$\mathrm{K}_{augmented} = [-224.9 \quad -224.3 \quad 1605.2 \quad 697.2 \quad -107.8] \tag{9.2}$$

The cost matrix used is in equation (9.1). This matrix was derived using the method of diagonal weights. The weights of the original LQR cost matrix were used, combined with a weight for the integrated position error. The cost for the integrated error was chosen based on trial and error. The resulting LQR feedback gain is in equation (9.2). In figure 9.1 below, the systems response to a desired velocity of $2\ m/s$ is imaged. The rise time was $0.60\ s$, the overshoot $44.20\ \%$ and the settling time $3.64\ s$ (in terms of velocity).

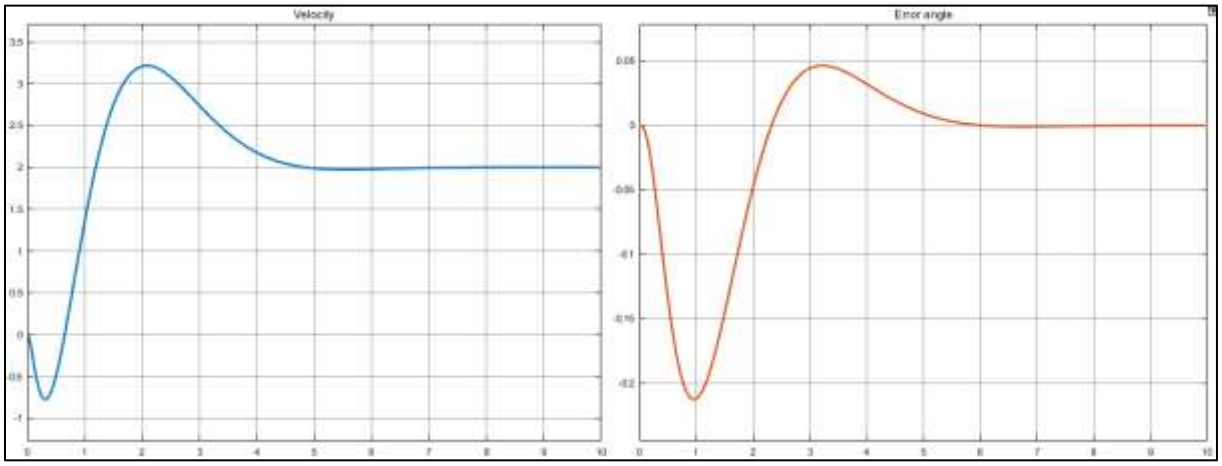

Figure 9.1: System response to a reference velocity of $2\ m/s$

9.2. LQR with integral action closed-loop stability

The input when using this controller was given by the closed-loop feedback law in equation (9.3)

$$u = -[K_{augmented}] \begin{bmatrix} e_x \\ e_{\dot{x}} \\ e_\theta \\ e_{\dot{\theta}} \\ \sigma \end{bmatrix} \tag{9.3}$$

$$\begin{bmatrix} \dot{e}_x \\ \dot{e}_{\dot{x}} \\ \dot{e}_\theta \\ \dot{e}_{\dot{\theta}} \\ \dot{\sigma} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 58.2008 & 57.9155 & -393.9799 & -180.3961 & 27.8997 \\ 0 & 0 & 0 & 1 & 0 \\ 25.5429 & 25.4177 & -168.1694 & -79.1715 & 12.2445 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} e_x \\ e_{\dot{x}} \\ e_\theta \\ e_{\dot{\theta}} \\ \sigma \end{bmatrix} \tag{9.4}$$

When equations (4.3), (4.4), (7.2), (9.2) and (9.3) are substituted back into equation (8.1), with values inserted and solved, the augmented closed-loop system was obtained in equation (9.4).

From this closed-loop system matrix $A_{augmented,cl}$, the closed-loop stability of the system was checked by evaluating the eigenvalues.

$$eig\left(A_{augmented,cl}\right) = \begin{bmatrix} -14.91 \\ -2.56 \\ -1.63 \\ -1.08 + 0.98i \\ -1.08 - 0.98i \end{bmatrix} \tag{9.5}$$

The eigenvalues of the augmented closed-loop system are in equation (9.5). As can be seen, all eigenvalues have a strictly negative real part, meaning that the closed loop system would be stable when the Segway is controlled by a LQR controller with integral action.

9.3. LQR with integral action implementation

In order to implement the controller, the model in figure 7.1 was adapted to obtain the Simulink model in figure 9.2 below.

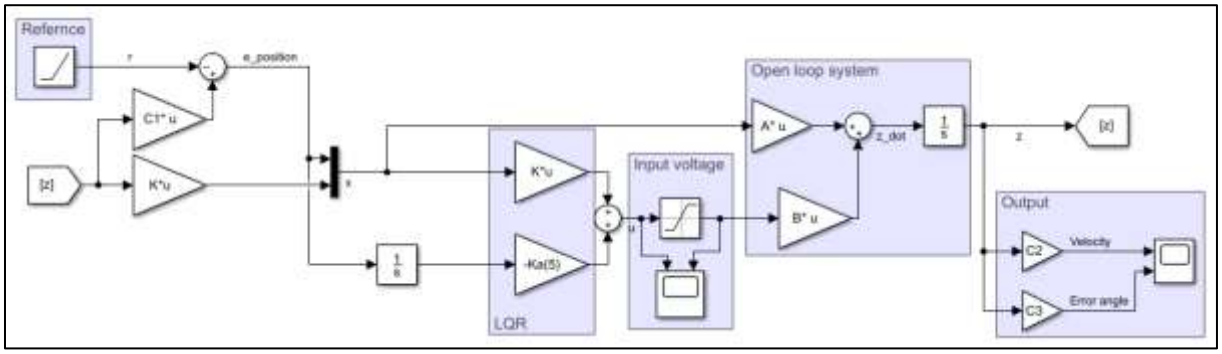

Figure 9.2: Simulink model of the LQR controller with integral action

For this system, it was investigated what was the maximum velocity the Segway would be able to reach using this control loop (with $z_i = [0 \quad 0 \quad 0 \quad 0]^T$). By gradually increasing the slope of the ramp signal, it was found that the maximum velocity the system could reach in this model, was $3.14\ m/s$, which is equal to $11.3\ km/h$. This velocity limit is probably high enough to be satisfactory for users. However, it would be useful if the velocity limit was a bit higher, to ensure safety when dealing with disturbances. This could be done by allowing a higher voltage input to the DC-motors, but then the DC-motors for the Segway would be required to be of a higher quality, resulting in a more expensive Segway.

10. PID+LQR Control

10.1. PID design

In order to reject the expected disturbances and maintain a desired velocity, a PI(D) controller was implemented into the model. The PI(D) controller was be placed over the error position (e_x) resulting from the difference between the reference position (r, given by a ramp signal) and the actual position (x).

A PID controller consists of a proportional (K_p), an integral (K_i) and a derivative (K_d) gain. In this way, the output of the PID controller will be a linear function of the controllers input, the integral of the controllers input and the derivative of the controllers input. When the PID was placed over the position error (e_x) (proportional), the derivative term became the velocity error ($e_{\dot{x}}$), and the integral term became the integrated position error (σ).

$$PID = K_p(e_x + \sigma K_i + e_{\dot{x}} K_d) \tag{10.1}$$

The equation for the output signal of the PID controller of the Segway is given in equation (10.1). The values of K_p, K_i and K_d should be tuned in order to obtain an optimal system response and disturbance rejection.

Multiple methods exist to tune a PID controller. In this project, three methods of PID tuning were used. First of all, multiple heuristic methods, among which the well-known Ziegler-Nichols method, were used to obtain values for the PID parameters. Secondly, the parameter values proposed by the PID Tuner App in Simulink were considered. Finally, the values found using the first two methods were adapted by manual tuning in order to see whether it was possible to improve the controllers proposed with the first two methods.

10.1.1. Heuristic methods

An easy method for PID tuning is using heuristics. Heuristic methods use empirical rules to tune the parameters of a PI(D) controller (de Persis, Control Engineering – Lecture 13, 2017). Both the Ziegler-Nichols heuristic method (Ziegler & Nichols, 1942) and the tuning rules found in the paper 'Rule-Based Autotuning Based on Frequency Domain Identification' (McCormack & Godfrey, 1998) were used to calculate gain values for the PID controller in equation (10.1). The PID parameters were found by applying the methods explained below.

The first step of the heuristic methods was to put all PID gains to zero.

$$K_{p,critical} = 146.3 \tag{10.2}$$

$$T_{critical} = 3.05 \tag{10.3}$$

Next, the proportional gain (K_p) was gradually increased until the ultimate gain ($K_{p,critical}$) was reached as in equation (10.2). At this value of the proportional gain the control signal (u) was stable with consistent oscillations, as shown in figure 10.1 below. The oscillation period corresponding to the ultimate gain ($T_{critical}$) is given in equation (10.2)

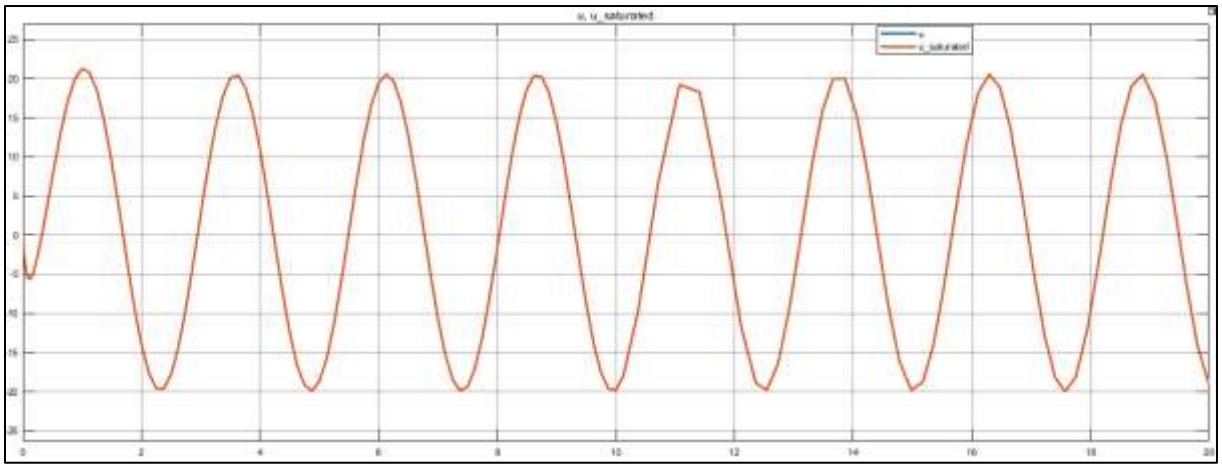

Figure 10.1: Control signal corresponding to a reference ramp signal with slope 1 and $K_p = 146.3$

The values of the ultimate gain $K_{p,critical}$ and the corresponding oscillation period $T_{critical}$ were used to obtain the values of the parameters of the controller. Because one of the main requirements of the controller was to reject disturbances, the integral term of the controller should be nonzero. This is why possible parameter values for a P or a PD controller were not investigated using heuristic methods.

Table 10.1: PID tuning rules

Control type	K_p	K_i	K_d
PI	$0.45K_{p,critical}$	$T_{critical}/1.2$	-
Classic PID	$0.6K_{p,critical}$	$T_{critical}/2$	$T_{critical}/8$
Pessen Integral Rule	$0.7K_{p,critical}$	$T_{critical}/2.5$	$3T_{critical}/20$
Some overshoot	$0.33K_{p,critical}$	$T_{critical}/2$	$T_{critical}/3$
Small overshoot	$0.2K_{p,critical}$	$T_{critical}/2$	$T_{critical}/3$

In table 10.1 the formulas to calculate the gain values for each type of controller are displayed. The first two methods were found in the article 'Optimum Settings for Automatic Controllers' (Ziegler & Nichols, 1942), while the final three methods are explained in the article 'Rule-Based Autotuning Based on Frequency Domain Identification' (McCormack & Godfrey, 1998), were the first method used the Pessen Integral Rule (Gopi & Suman, 2015). Plugging in the values of $K_{p,critical}$ and $T_{critical}$ into table 10.1 resulted in the parameter values in table 10.2 below.

Table 10.2: PID gain values resulting from tuning rules

Control type	K_p	K_i	K_d
PI	65.84	2.54	0
Classic PID	87.78	1.53	0.38
Pessen Integral Rule	102.41	1.22	0.46
Some overshoot	48.28	1.53	1.02
Small overshoot	29.26	1.53	1.02

Using the parameter values in table 10.2, all five controllers were tested with a reference velocity of $2\ m/s$. For the Ziegler Nichols method based PI controller and the Pessen Integral Rule based PID controller the reference signal resulted in an unstable system. In the figures 10.2-10.4 below the systems response for each of the other controllers is plotted.

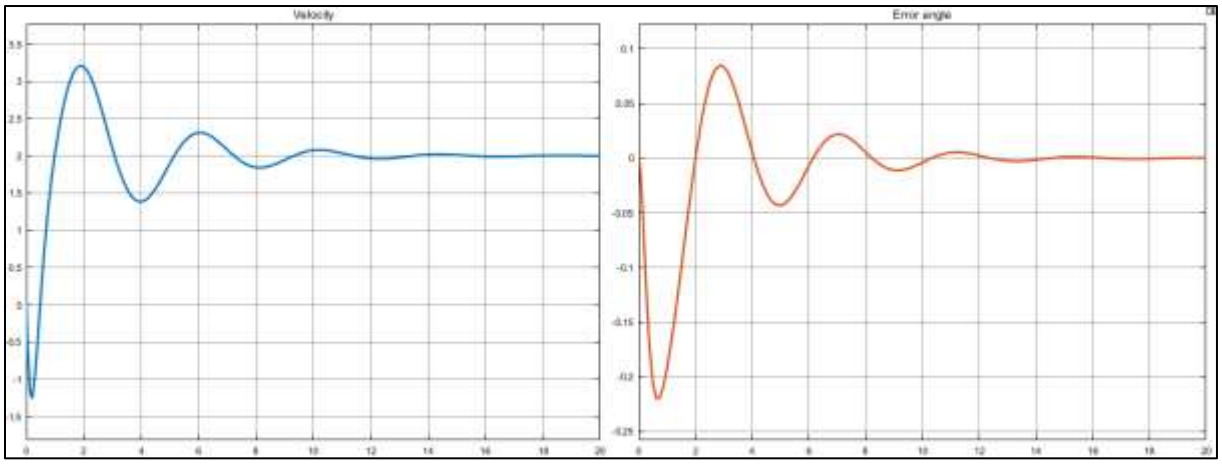

Figure 10.2: Closed-loop systems response to a reference ramp with slope 2 with classic PID controller (Ziegler & Nichols, 1942)

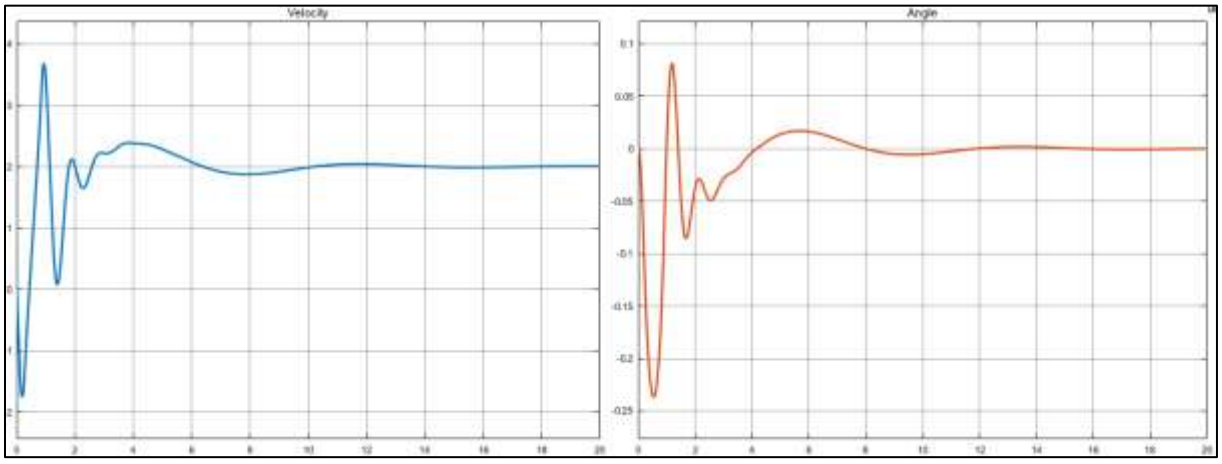

Figure 10.3: Closed-loop systems response to a reference ramp with slope 2 with a PID controller allowing some overshoot (McCormack & Godfrey, 1998)

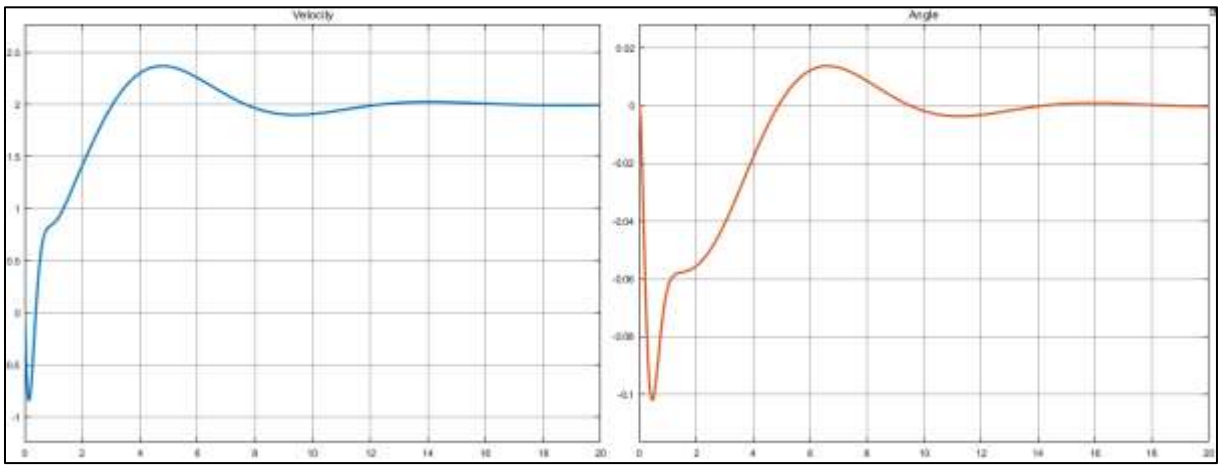

Figure 10.4: Closed-loop systems response to a reference ramp with slope 2 with a PID controller allowing small overshoot (McCormack & Godfrey, 1998)

Table 10.3: Characteristics of the controllers based on the Z-N method

Controller	Rise time	Overshoot	Settling time	Steady-state error
Classic PID	$0.47\ s$	$48.25\ \%$	$10.4\ s$	$0\ m/s$
Some overshoot	$0.24\ s$	$44.8\ \%$	$14.45\ s$	$0\ m/s$
Small overshoot	$2.23\ s$	$17.97\ \%$	$10\ s$	$0\ m/s$
Requirements	$1\ s$	$20\ \%$	$3.5\ s$	$0\ m/s$

The characteristics of the other three controllers are presented in table 10.3, so that they can easily be compared to the desired transient properties. As can be seen in the table, none of the three remaining controllers met all controller requirements. The required overshoot was only met by the 'small overshoot' controller, while the rise time requirement was met by the other two controllers. None of the proposed controllers met the desired settling time. This meant other controllers had to be proposed in order to find a controller which would meet all requirements.

10.1.2. Simulink PID Tuner App

A Simulink PID block was used to tune the PID controller applied to the error position (e_x). For PID tuning, the PID block in Simulink includes a transfer function based PID Tuner app. This PID Tuner App automatically computes a linear model of the complete plant. The App considers the plant to be the combination of all blocks between the PID controller output and input. For the linearized plant, the PID Tuner automatically computes an initial PID design with a balance between performance and robustness, based on the response to a unit step signal. The app computes the initial PID design using the proprietary tuning algorithm developed by MathWorks (MathWorks, 2018).

$$PID = K_p\left(1 + K_i\frac{1}{s} + K_d\frac{N}{1 + N\frac{1}{s}}\right) \tag{10.4}$$

Note that the standard equation for PID controllers of ideal form used by the PID block in Simulink is different from the PID controller in equation (10.1). The PID controller in the PID block is given by equation (10.4). Here, the N represents the filter coefficient of the derivative and multiplying or dividing by s represents the derivative or integral in the Laplace domain respectively.

In a conventional PID controller, the derivative gain requires filtering, because when a sensor measures the proportional signal, there is bound to be noise in the sensor signal. When this noise is differentiated, this results in large fluctuations in the derivative (Isaksson & Graebe, 2002).

In the Segway systems sensors, however, the velocity of the Segway is measured instead of the Segway's position. The position of the Segway is calculated by integrating the velocity. Thus, in the PID controller, there was no sensor signal that was differentiated and so there was no amplification of the sensor noise. This is why there was no need for a derivative filter in the PID controller and the value of N was left unused.

The PID tuner app tuned the values of K_p, K_i, K_d and N in order to obtain a stable controller with a good system response and disturbance rejection.

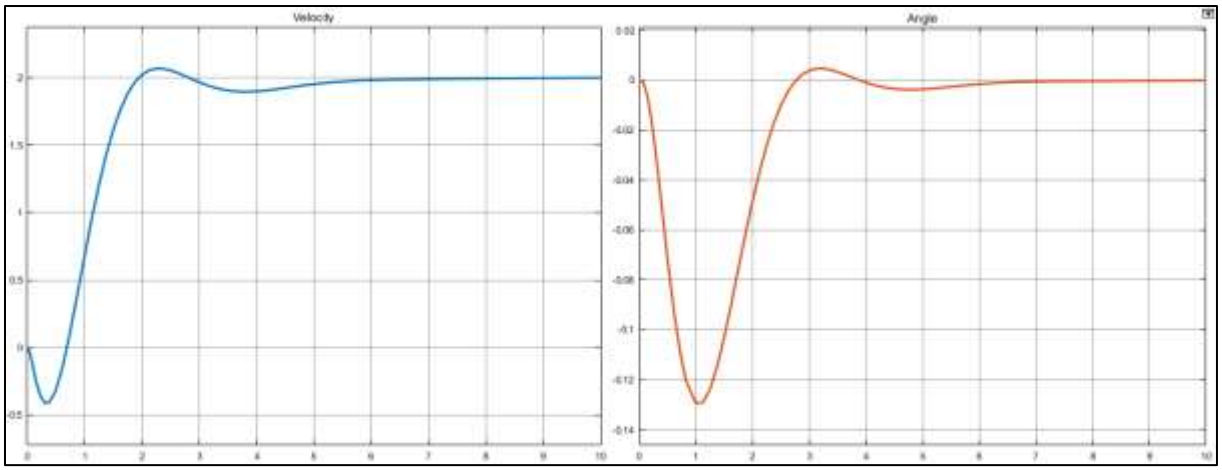

Figure 10.5: Closed-loop systems response to a reference ramp with slope 2 with the initial PID controller proposed by the PID Tuner App in Simulink

The initial PID design as proposed by the Tuner App was tested with a desired velocity of $2\,m/s$ (reference ramp with slope 2). The systems response is displayed in figure 10.5.

The PID Tuner App in Simulink allowed further tuning of the controller by increasing or decreasing the controller's response time and transient behavior. Because the initial PID design did not meet the desired controller characteristics, the controller was adapted in the PID tuner using trial and error.

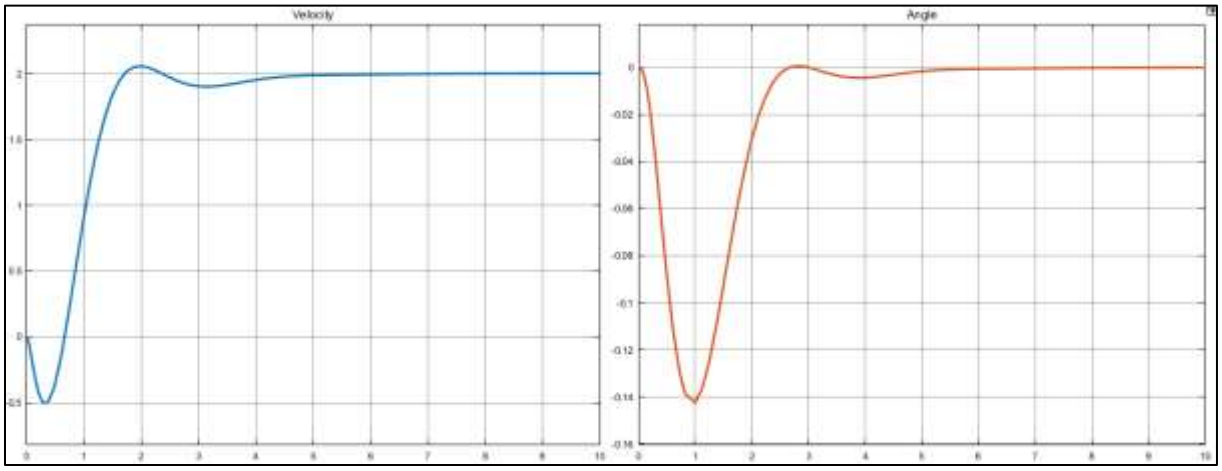

Figure 10.6: Closed-loop systems response to a reference ramp with slope 2 with a slightly lower response time compared to the initial PID design

By slightly decreasing the response time in the Tuner App from $2.245\,s$ to $2.105\,s$, the system response in figure 10.6 was obtained. This PID design was better compared to the initial design in the fact that its response and settling times were both lower than those of the initial PID design, as can be seen in table 10.4. The phase margin, however, was slightly larger in the initial PID design.

Table 10.4: Characteristics of the controllers based on the PID Tuner App in Simulink

Controller	K_p	K_i	K_d	Rise time	Overshoot	Settling time	Steady-state error
Initial PID design	57.31	0.61	0.21	1.27 s	1.61 %	4.02 s	0 m/s
Decreased response time	60.67	0.64	0.28	1.24 s	1.28 %	1.22 s	0 m/s
Requirements	-	-	-	1.5 s	20 %	3.5 s	0 m/s

In table 10.4 the response characteristics of the controllers based on the PID Tuner App in Simulink are compared to the required characteristics. As can be seen, the only controller which meets all requirements, is the Initial PID design with decreased response time. Thus, this controller was the starting point from which it was investigated whether or not this controller could be improved by manual tuning. From now on, this controller will be called controller 1.

10.1.3. Manual tuning

In manual tuning, controller 1 obtained in the previous section was adapted in order to improve its performance. Because it is not intuitively clear how the PID parameters influence the systems response, this influence was investigated. The found effects of increasing individual parameters are displayed in table 10.5 (Zhong, 2006) (Kiam Heong Ang, 2005).

Table 10.5: Effects of increasing PID gains individually

Parameter	Rise time	Overshoot	Settling time	Steady-state error	Stability
P	Decrease	Increase	Small change	Decrease	Degrade
I	Decrease	Increase	Increase	Eliminate	Degrade
D	Minor change	Decrease	Decrease	No effect in theory	Improve if D small

Because there is no parameter that definitely improves all aspects of the controllers performance, the K_p, K_i and K_d gains were simply rounded, either upward or downward.

Table 10.6: Controllers and controller characteristics obtained by rounding parameters of controller 1

Controller number	K_p	K_i	K_d	Rise time	Overshoot	Settling time	Steady-state error
1	60.67	0.64	0.28	1.24 s	1.28 %	1.22 s	0 m/s
2	61	0.7	0.3	1.12 s	3.29 %	2.49 s	0 m/s
3	61	0.7	0.2	1.08 s	9.77 %	3.97 s	0 m/s
4	61	0.6	0.3	1.25 s	0.41 %	3.02 s	0 m/s
5	61	0.6	0.2	1.14 s	3.85 %	4.06 s	0 m/s
6	60	0.7	0.3	1.22 s	2.87 %	2.52 s	0 m/s
7	60	0.7	0.2	1.12 s	8.52 %	4.39 s	0 m/s
8	60	0.6	0.3	1.30 s	0.42 %	3.09 s	0 m/s
9	60	0.6	0.2	1.19 s	2.69 %	4.34 s	0 m/s
Requirements	-	-	-	1.5 s	20 %	3.5 s	0 m/s

In table 10.6, the obtained controllers and the velocity response characteristics with a reference velocity of $2\ m/s$ are shown. It became clear that manually adjusting the PID parameters did only improve certain response characteristics if other characteristics deteriorated. Thus, controller 1 was chosen as the final PID controller for the Segway system.

10.2. PID+LQR closed-loop stability

The input voltage (u) to the DC-motors for this controller was given by the closed-loop feedback law in equation (10.5)

$$u = -[K \quad 0]\begin{bmatrix} e_x \\ e_{\dot{x}} \\ e_\theta \\ e_{\dot{\theta}} \\ \sigma \end{bmatrix} + [K_p \quad K_p K_d \quad 0 \quad 0 \quad K_p K_i]\begin{bmatrix} e_x \\ e_{\dot{x}} \\ e_\theta \\ e_{\dot{\theta}} \\ \sigma \end{bmatrix} \tag{10.5}$$

$$\begin{bmatrix} \dot{e}_x \\ \dot{e}_{\dot{x}} \\ \dot{e}_\theta \\ \dot{e}_{\dot{\theta}} \\ \dot{\sigma} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 28.1744 & 30.0610 & -228.4663 & -103.9368 & 10.0462 \\ 0 & 0 & 0 & 1 & 0 \\ 12.3650 & 13.1931 & -95.5295 & -45.6153 & 4.4090 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} e_x \\ e_{\dot{x}} \\ e_\theta \\ e_{\dot{\theta}} \\ \sigma \end{bmatrix} \tag{10.6}$$

When equations (4.3), (4.4), (6.19), (7.2) and (10.5) were substituted back into equation (8.1), with values inserted and solved, the augmented closed-loop system was obtained in equation (10.6).

From this closed-loop system matrix $A_{augmented}$, the closed-loop stability of the system was checked by evaluating the eigenvalues.

$$eig(A_{augmented}) = \begin{bmatrix} -10.03 \\ -2.53 \\ -1.16 + 1.19i \\ -1.16 - 1.19i \\ -0.68 \end{bmatrix} \tag{10.7}$$

The eigenvalues of the augmented closed-loop system are in equation (10.7). As can be seen, all eigenvalues have a strictly negative real part, meaning that the closed loop system would be stable when the Segway is controlled by a PID+LQR controller.

10.3. PID+LQR implementation

When the PID controller was applied to the system, the Simulink model in figure 10.9 was obtained by adapting the model in figure 7.1.

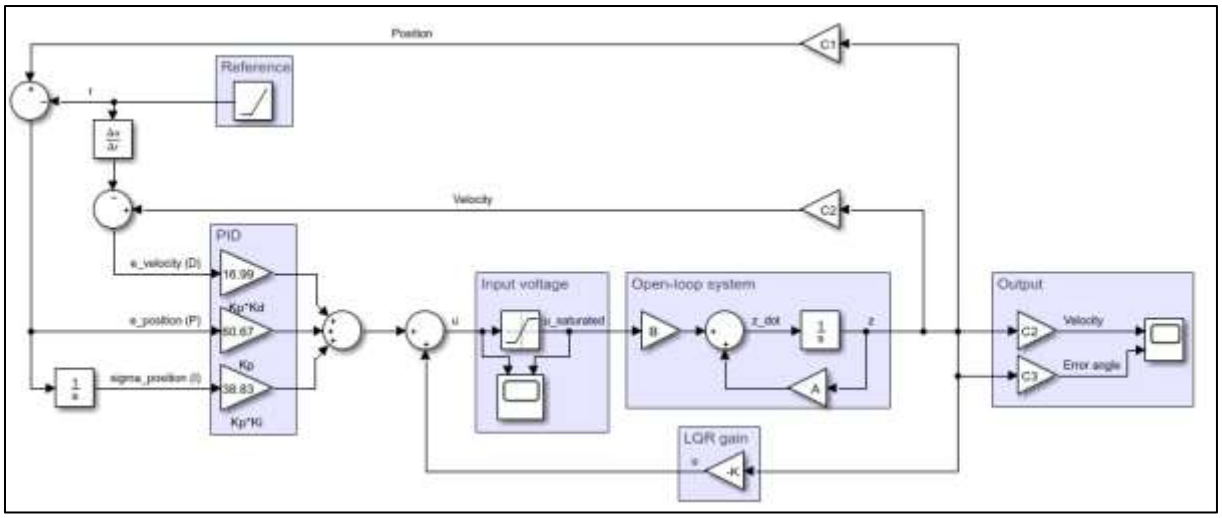

Figure 10.7: Closed-loop system with LQR and PID control

For this system, it was investigated what was the maximum velocity the Segway would be able to reach using this control loop. By gradually increasing the slope of the reference ramp signal, it was found that the maximum velocity the system could reach (from initial condition $z_i = [0 \quad 0 \quad 0 \quad 0]^T$) in this model, was $4.67\ m/s$, which is equal to $16.81\ km/h$. When this velocity is compared to the velocity of two-wheeled inverted pendulums in other research, this is more than enough to be sufficient (Searock, Browning, & Veloso, 2004) (Castro, Adams, & Singhose, Dynamic response characteristics of a two-wheeled inverted-pendulum transporter, 2013).

11. Disturbances and signal noise

For all three controllers, designed previously, the closed-loop models robustness to disturbances and noise was checked.

For signal noise, it was checked how the system would respond to noise in the velocity and in the error angle, as those are the states measured by the tachometer and the gyroscope, respectively. Next, the disturbances resulting from differences in human users were checked. Then, it was evaluated how the system would respond to the presence of positive or negative slopes. Finally, a combination of these disturbances was evaluated, simulating a real world situation.

11.1. Noise in sensor signals

The rigidity of the controllers was tested by applying white noise to the sensor signals in Simulink. This was done by modifying the Simulink models obtained earlier in order to insert noise into the signals.

11.1.1. Size of sensor noise

As the sensors in the Segway are a gyroscope and a tachometer, measuring the error angle and the velocity, these signals were disturbed with noise. In the white noise block representing the noise in the signal from the gyroscope, the noise power was set to $4.36e-07$, based on the article 'Gyro Modeling and Estimation of Its Random Noise Sources' (Lam, Stamatakos, Woodruff, & Ashton, 2003). Furthermore, the sample time was set to $0.01\ s$ and the seed was set to the initial setting of $[23341]$.

In the white noise block representing the noise in the signal from the tachometer, the noise power was set to $4.36e-06$, based on the fact that a small disturbance in the error angle signal has more drastic consequences than a small disturbance in the velocity. Thus, the tachometer signal was given a relatively powerful noise, in order to challenge the robustness of the controller. Furthermore, like in the gyroscope noise, the sample time was set to $0.01\ s$ and the seed was set to the initial setting of $[23341]$.

With this noise applied to the system, the LQR controller for vertical stabilization was tested, as well as the LQR with integral action and the PID+LQR controllers for stabilization at the desired velocity.

11.1.2. LQR controller

First, the stabilization of the Segway from an initial angle of $0.3\ rad$ was tested using the LQR controller. In order to do so, the Simulink model in figure 6.4 was adapted to obtain the model in figure 11.1.

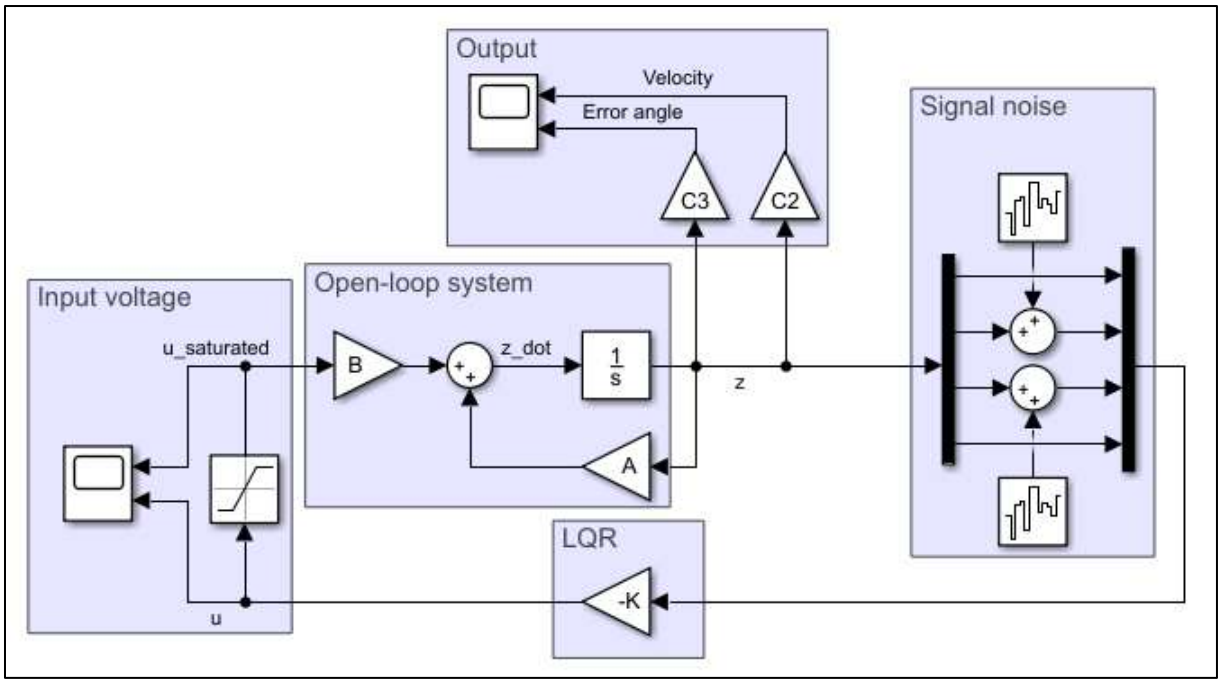

Figure 11.1: Closed-loop system with LQR control and sensor noise

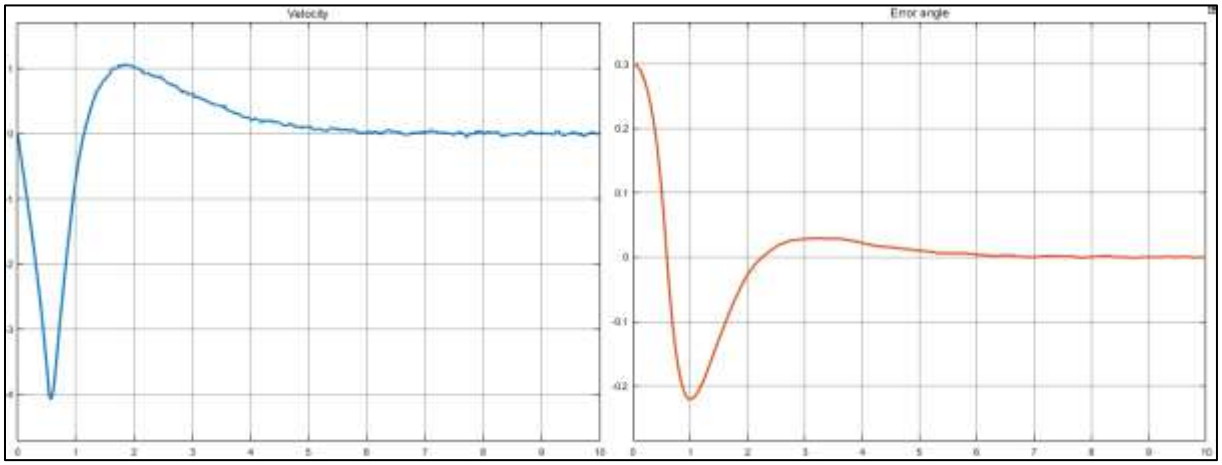

Figure 11.2: System response to an initial angle of $0.3\ rad$ *with white noise in the sensor signals*

In figure 11.2 the system response with white noise is imaged. As can be seen, the controller is easily robust enough to reject the noise; the response is comparable to the response when there is no noise, so there will barely be loss of the systems stability. When the noise is zoomed in to, it can be seen that the noise is carried on to the system response. However, the responses in terms of error angle or velocity of the system are so small that it is not likely that the user will notice the systems response to the noise.

11.1.3. LQR controller with integral action

Next, the stabilization of the Segway at the desired velocity of $3\ m/s$ using the LQR controller with integral action was tested. To be able to implement signal noise into the model, the Simulink model in figure 9.2 was adapted, resulting in the model in figure 11.3.

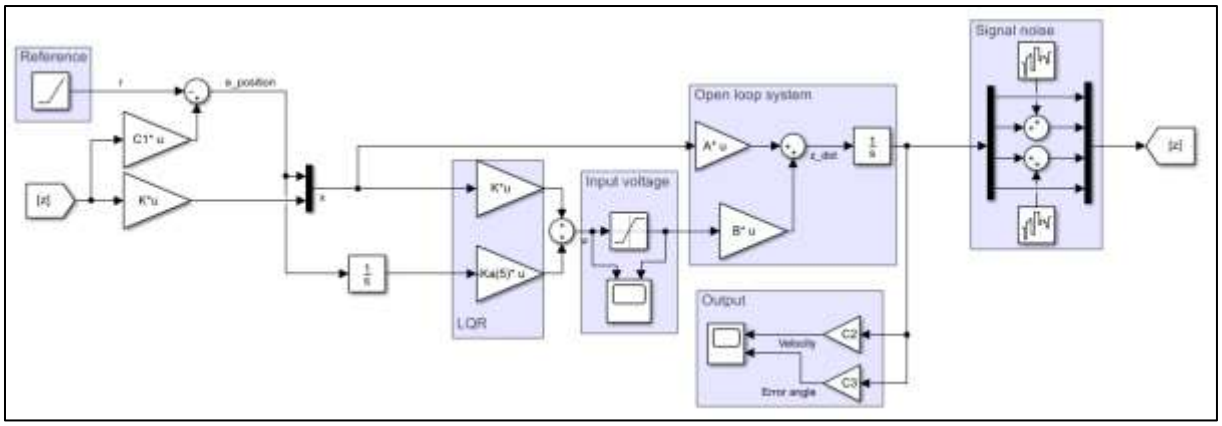

Figure 11.3: Closed-loop system with LQR control with integral action and sensor noise

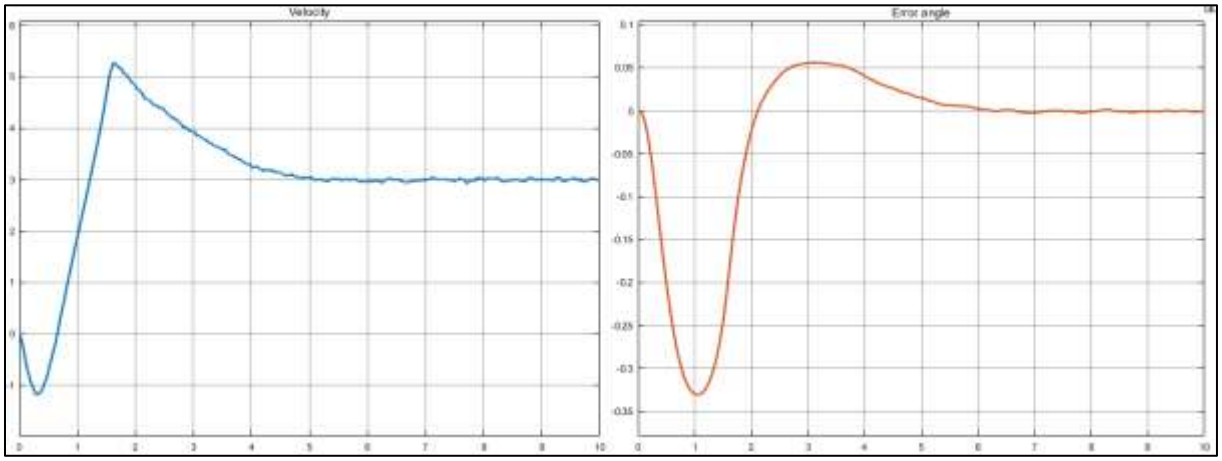

Figure 11.4: System response to a reference velocity of 3 m/s with LQR control with integral action and white noise in the sensor signals

In figure 11.4 the system response with white noise is illustrated. In the plot, the influence of the sensor noise is more clearly visible than with the LQR controller; the velocity curve has a sharp edge at $t = 1.6\ s$. This peak is curved and at a lower velocity level in the absence of noise. Still, the system stabilizes well and is robust enough to reject the sensor noise. Like with the LQR stabilization, the steady state responses in terms of error angle or velocity of the system are so small that it is not likely that the user will notice the Segways transient response resulting from signal noise.

11.1.4. PID+LQR controller

Finally, the stabilization of the Segway at the desired velocity of 3 m/s using the PID+LQR controller was tested. The noise was implemented into the model by changing the model in figure 10.4. The resulting Simulink model with noise is displayed in figure 11.5

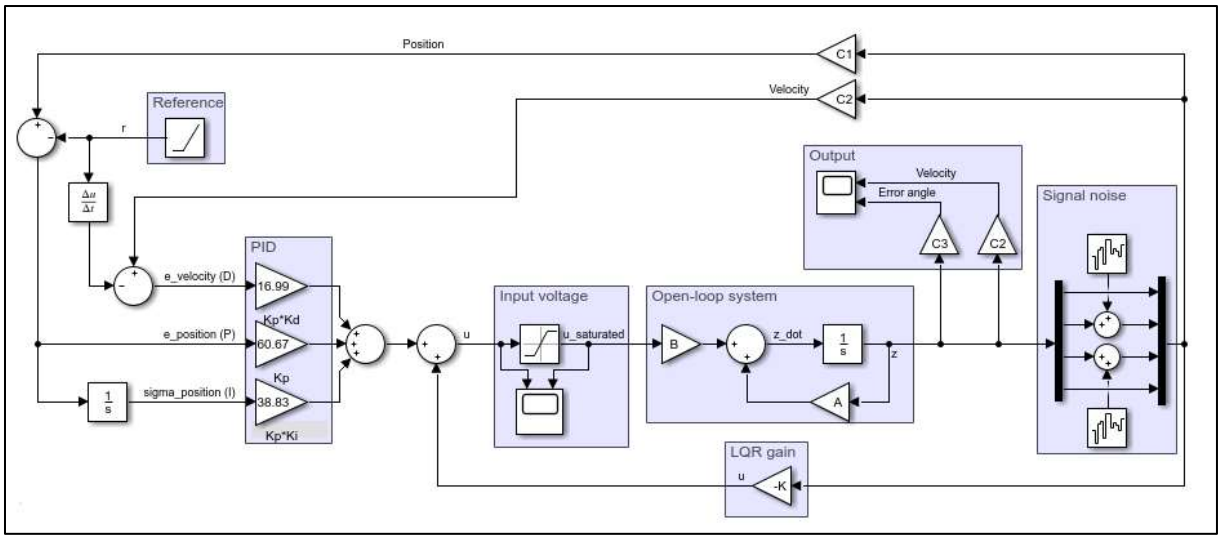

Figure 11.5: Closed-loop system with PID+LQR controller and sensor noise

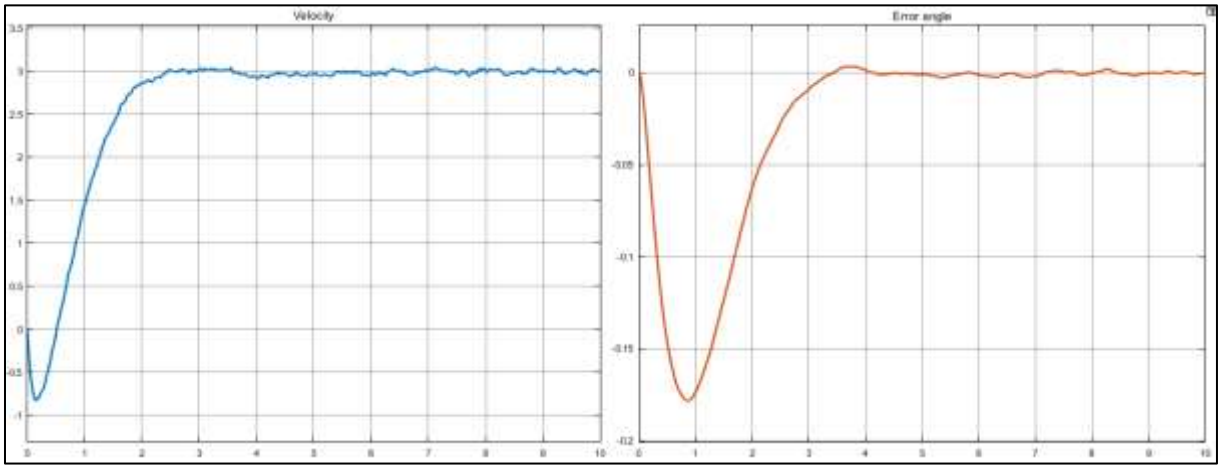

Figure 11.6: System response to a reference velocity of 3 with white noise in the sensor signals

In figure 11.6 the system response with white noise is illustrated. In the plot, the influence of the sensor noise is more noteworthy than with the LQR controllers discussed before. This most likely arises from the relatively aggressive gains of the closed-loop system with the PID+LQR controller. Still, the system stabilizes well and is robust enough to reject the sensor noise. Like with the LQR stabilization, the steady state responses in terms of error angle or velocity of the system are so small that it is not likely that the user will notice the Segways transient response resulting from signal noise.

11.2. Disturbances resulting from differences in users

11.2.1. Size of user disturbances

There can be large fluctuations in the length and weight of human users. Because the linearized system in equation (4.8) was computed for a user weighing $85\ kg$ and being $1.7\ m$ tall, with an inertia of $68.98\ kg \cdot m^2$, it needed to be checked how the system would respond to parameter uncertainties in terms of user parameters (M_p, l, I_p). In order to do so, the Segways response to the user of $85\ kg$ and $1.7\ m$ was compared to the systems responses to a small user of $60\ kg$ and $1.6\ m$ and a large user of $100\ kg$ and $2.0\ m$. First, the stabilization of the Segway using the LQR controller designed in section 6 from an initial error angle of 10 degrees ($0.17\ rad$) was checked for the three types of users. Secondly, the stabilization at a desired velocity using the LQR controller with integral action and the PID+LQR controller were checked for the three user types.

Table 11.1: Parameters and state-space matrices for different users

User mass (M_p)	User length (l)	User inertia (I_p)	System matrix (A)	System matrix (B)
$85\ kg$	$1.7\ m$	$68.98\ kg \cdot m^2$	$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.1074 & 21.3441 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -0.0471 & 14.1063 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0.2587 \\ 0 \\ 0.1136 \end{bmatrix}$
$60\ kg$	$1.6\ m$	$43.13\ kg \cdot m^2$	$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.1301 & 18.3673 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -0.0600 & 13.4885 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0.3134 \\ 0 \\ 0.1446 \end{bmatrix}$
$100\ kg$	$2.0\ m$	$112.32\ kg \cdot m^2$	$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.0944 & 21.3071 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -0.0348 & 11.772 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0.2276 \\ 0 \\ 0.0839 \end{bmatrix}$

To be able to check the system responses to the different users, the values ofM_p, l and I_p in table 4.1 had to be adapted.

$$I = \frac{1}{\rho} M l^2 \tag{11.1}$$

$$I_{p,new} = I_{p,old} \left(\frac{M_{p,new}}{M_{p,old}} \right) \left(\frac{l_{new}}{l_{old}} \right)^2 \tag{11.2}$$

The value for the user inertia was adopted by using the formula in equation (11.2). This equation was based on the standard formula for the of inertia for beams and rods (equation (11.1)) (Morin, 2010). Equations for the inertia of beams or rods all contain the mass times the length squared, divided by a real number ρ. Because the human body is not comparable to a beam or a rod, the value of ρ is not known. That is why equation (11.2) was used to transform the known inertia of the user of $1.7\ m$ and $85\ kg$ into good approximations of the moments of inertia of other users. Substituting these new parameter values into equations (4.3) and (4.4) resulted in the new system matrixes in table 11.1.

11.2.2. LQR controller

Using the system matrices in table 11.1 and an initial error angle of 10 degrees ($0.17\ rad$), the system responses in figures 11.7-11.9 were obtained. The initial error angle of 10 degrees was based on the assumption that this is a realistic error angle resulting from the user mounting the Segway.

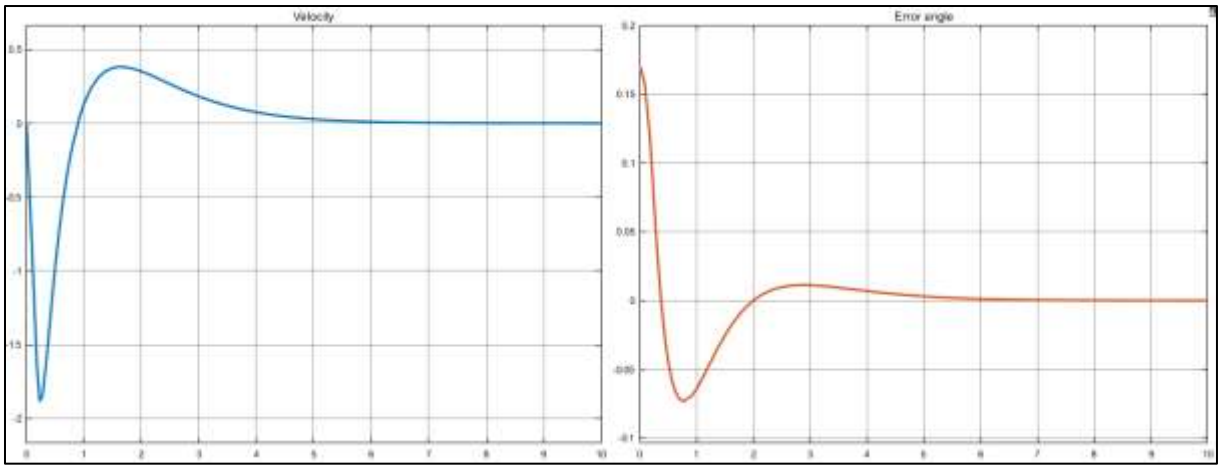

Figure 11.7: System response to an initial angle of $0.17\ rad$ *when carrying a user of* $85\ kg$ *and* $1.7\ m$

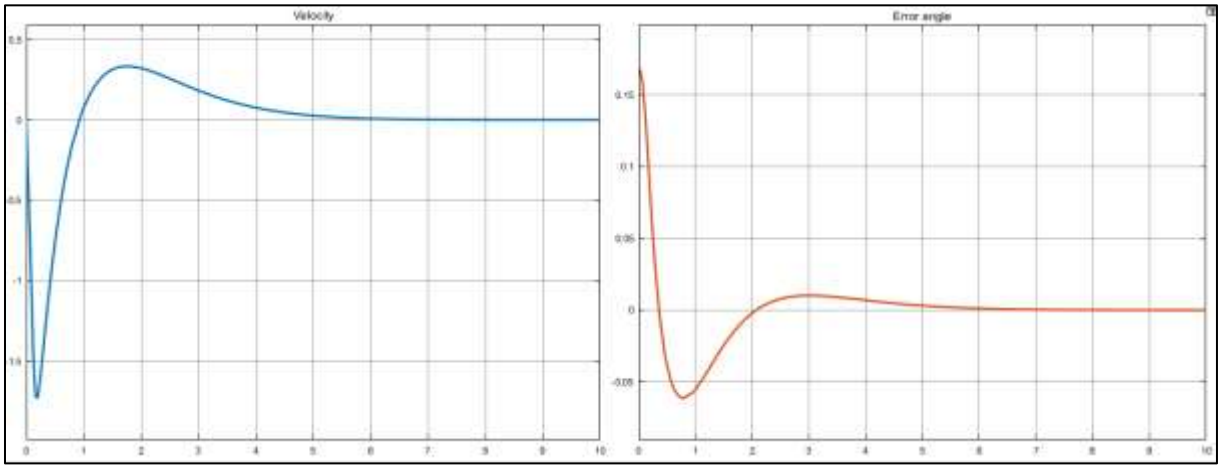

Figure 11.8: System response to an initial angle of $0.17\ rad$ *when carrying a user of* $60\ kg$ *and* $1.6\ m$

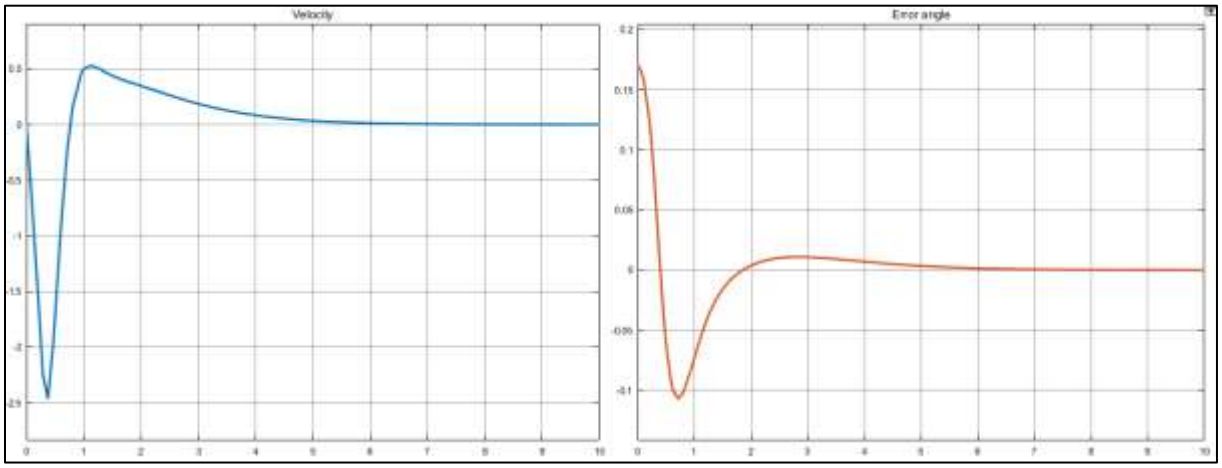

Figure 11.9: System response to an initial angle of $0.17\ rad$ *when carrying a user of* $100\ kg$ *and* $2.0\ m$

Table 11.2: System response properties with an initial angle of $0.17\ rad$ *when carrying different users*

User mass (M_p)	User length (l)	User inertia (I_p)	Rise time	Overshoot	Settling time	Steady-state error
$85\ kg$	$1.7\ m$	$68.98\ kg \cdot m^2$	$0.40\ s$	$25.00\ \%$	$4.11\ s$	$0\ m/s$
$60\ kg$	$1.6\ m$	$43.13\ kg \cdot m^2$	$0.42\ s$	$30.00\ \%$	$4.88\ s$	$0\ m/s$
$100\ kg$	$2.0\ m$	$112.32\ kg \cdot m^2$	$0.29\ s$	$21.34\ \%$	$3.90\ s$	$0\ m/s$

From the transient responses in figures 11.7-11.9, the response properties in table 11.2 were derived. From the properties it was concluded that the system experienced no issues in vertical stabilization when operating with smaller or larger users. The larger user's characteristics were even better than those of the user for which the model was derived. This was most likely caused by the more aggressive input signal to the DC-motors, resulting from the larger state errors for the large user. Downside to this increased performance was of course the larger state errors.

11.2.3. LQR controller with integral action

Using the LQR controller with integral action and the system matrices in table 11.1, with all initial conditions equal to zero ($z_i = [0 \quad 0 \quad 0 \quad 0]$) and a reference ramp with slope 3, the trajectory tracking of the Segway with small and big users was evaluated. Due to the saturation limits, the system became unstable when stabilizing the larger user at $3\ m/s$. Thus, for the large user, the velocity was stabilized at $2\ m/s$. The system responses are displayed in figures 11.10-11.12.

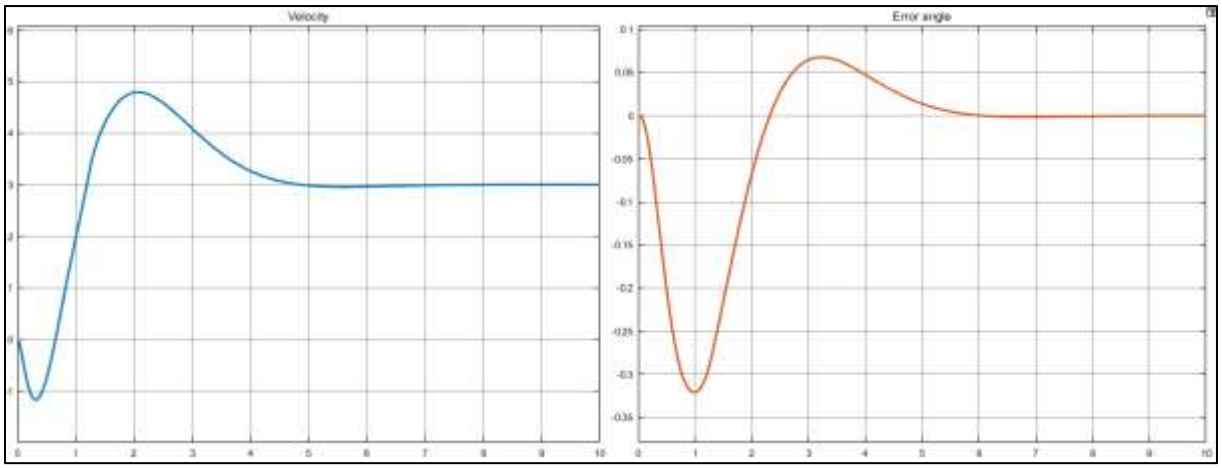

Figure 11.10: System response to a reference velocity of $3\ m/s$ *when carrying a user of* $85\ kg$ *and* $1.7\ m$ *(LQR with integral action)*

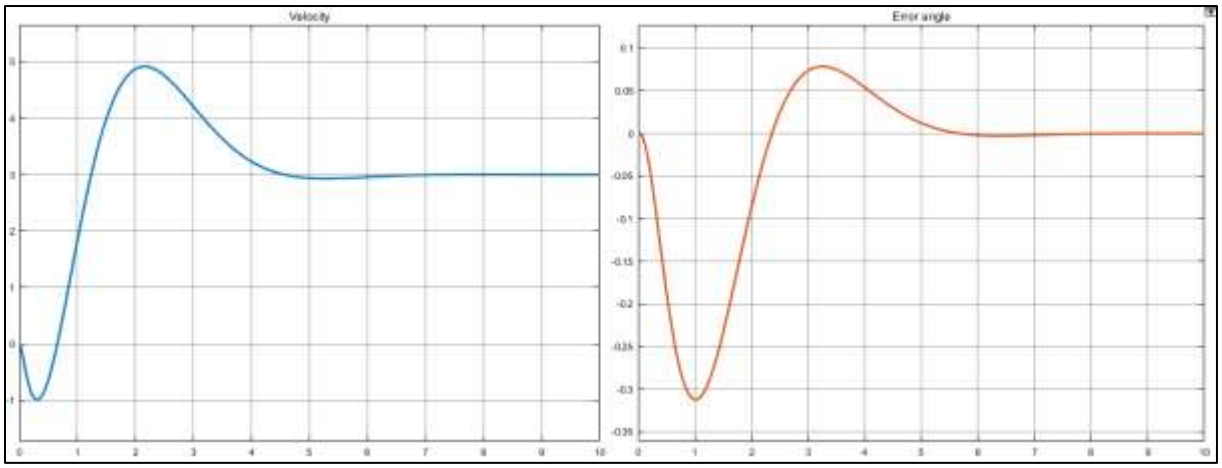

Figure 11.11: System response to a reference velocity of $3\ m/s$ *when carrying a user of* $60\ kg$ *and* $1.6\ m$ *(LQR with integral action)*

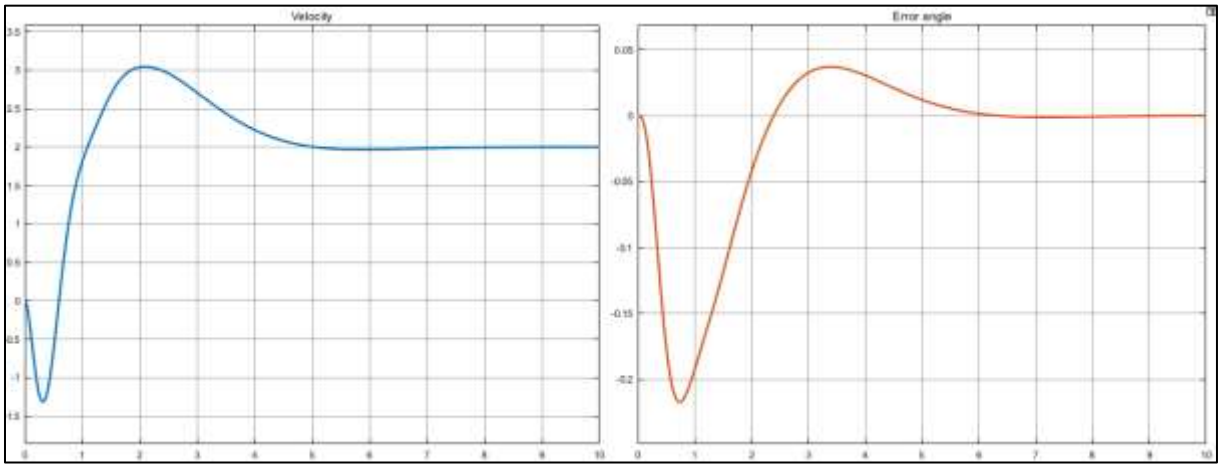

Figure 11.12: System response to a reference velocity of $2\ m/s$ *when carrying a user of* $100\ kg$ *and* $2.0\ m$ *(LQR with integral action)*

Table 11.3: System response properties with desired (reference) velocity of $3\ m/s$ *(*$2\ m/s$ *for the larger user) with LQR control with integral action when carrying different users*

User mass (M_p)	User length (l)	User inertia (I_p)	Rise time	Overshoot	Settling time	Steady-state error
$85\ kg$	$1.7\ m$	$68.98\ kg \cdot m^2$	$0.61\ s$	$44.20\ \%$	$3.74\ s$	$0\ m/s$
$60\ kg$	$1.6\ m$	$43.13\ kg \cdot m^2$	$0.63\ s$	$48.51\ \%$	$3.47\ s$	$0\ m/s$
$100\ kg$	$2.0\ m$	$112.32\ kg \cdot m^2$	$0.49\ s$	$32.67\ \%$	$4.07\ s$	$0\ m/s$

From the transient responses in figures 11.10-11.12, the response properties in table 11.3 were derived. From the properties it was concluded that the system experienced some issues when operating with a larger user. This is a result from the saturation of the DC-motor, which is not powerful enough to accelerate Segway as fast as would be desired with the larger user. Still, at a lower velocity of $2\ m/s$ this controller is also suited for heavier users.

11.2.4. PID+LQR controller

Using the PID+LQR controller and the system matrices in table 11.1, with all initial conditions equal to zero ($z_i = [0 \quad 0 \quad 0 \quad 0]$) and a reference ramp with slope 3, the trajectory tracking of the Segway with small and big users was evaluated. The system responses are displayed in figures 11.13-11.15.

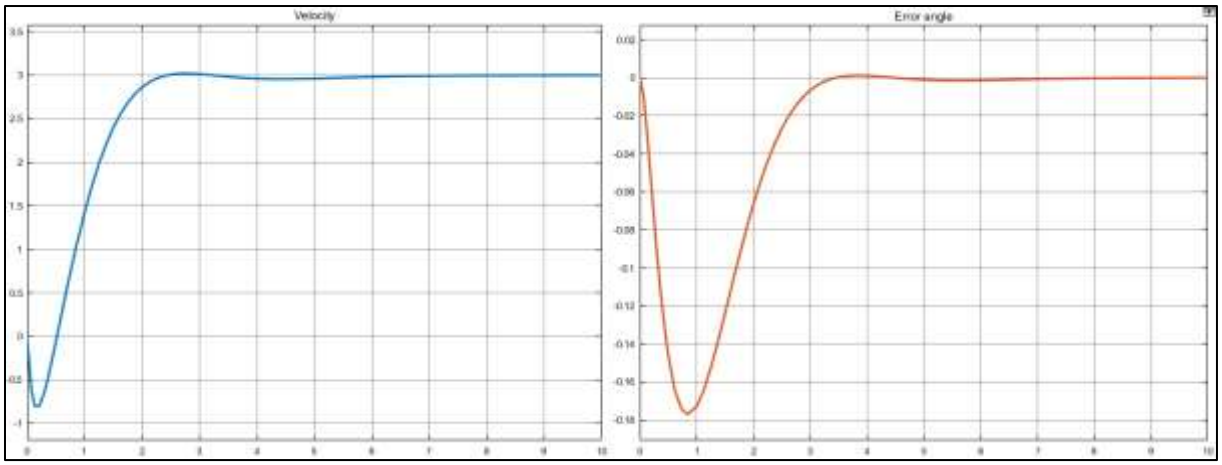

Figure 11.13: System response to a reference velocity of 3 when carrying a user of $85\ kg$ *and* $1.7\ m$ *(PID+LQR)*

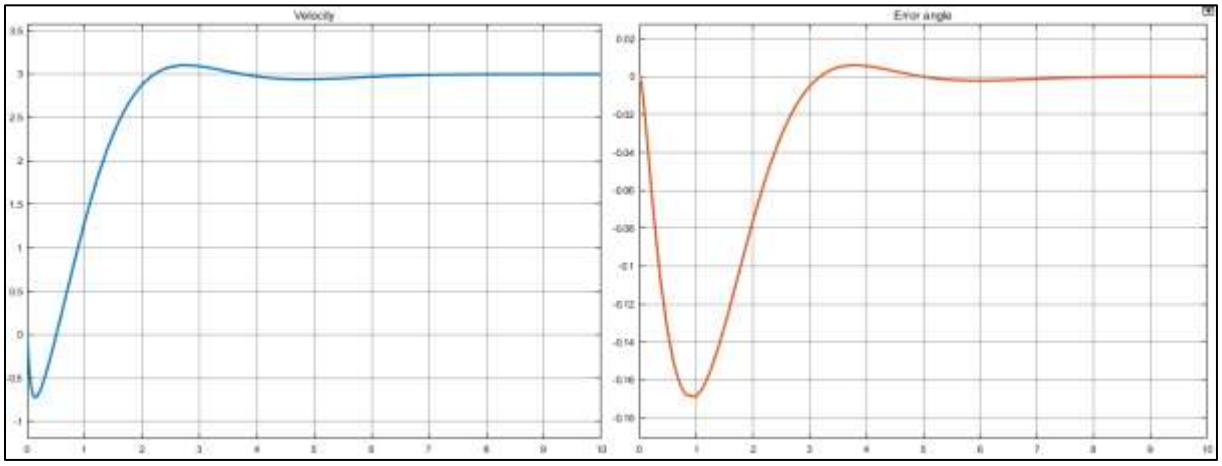

Figure 11.14: System response to a reference velocity of 3 when carrying a user of $60\ kg$ *and* $1.6\ m$

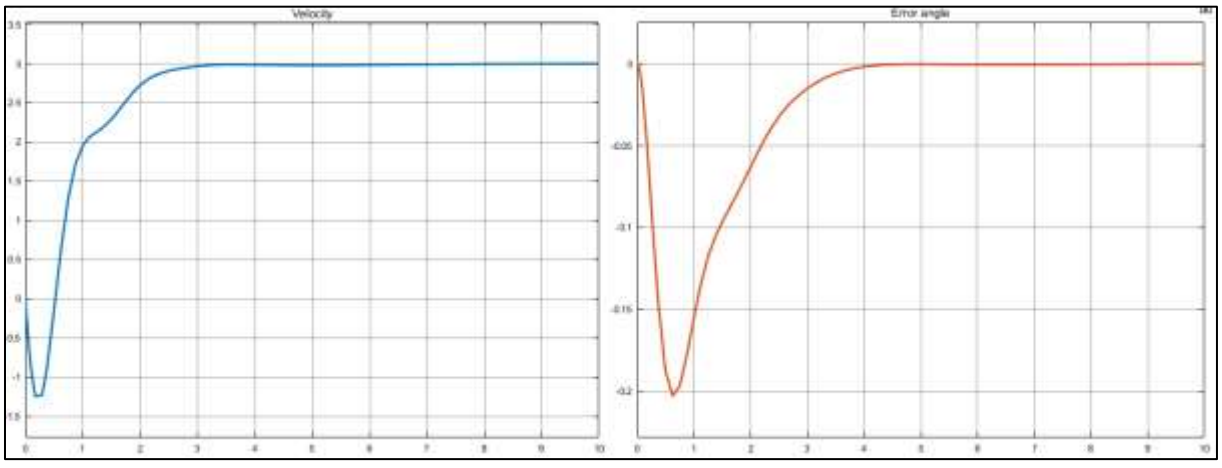

Figure 11.15: System response to a reference velocity of 3 when carrying a user of $100\ kg$ *and* $2.0\ m$

Table 11.4: System response properties with desired (reference) velocity of $3\ m/s$ with PID+LQR control when carrying different users

User mass (M_p)	User length (l)	User inertia (I_p)	Rise time	Overshoot	Settling time	Steady-state error
$85\ kg$	$1.7\ m$	$68.98\ kg \cdot m^2$	$1.17\ s$	$0.63\ \%$	$1.16\ s$	$0\ m/s$
$60\ kg$	$1.6\ m$	$43.13\ kg \cdot m^2$	$1.19\ s$	$3.17\ \%$	$4.16\ s$	$0\ m/s$
$100\ kg$	$2.0\ m$	$112.32\ kg \cdot m^2$	$1.37\ s$	$0.35\ \%$	$1.77\ s$	$0\ m/s$

From the transient responses in figures 11.13-11.15, the response properties in table 11.4 were derived. From the properties it was concluded that the system experienced no issues for stabilization at the desired velocity of $3\ m/s$ when operating with smaller or larger users. Some characteristics were even better for the other users, at the cost of other characteristics being slightly worse. Altogether the controller performed relatively well under user disturbances.

11.3. Disturbances in error angle

Another disturbance which was modelled was the disturbance to the error angle (upward configuration) of the Segway. Such a disturbance would be a result of the users lean forward or backward or the bar hitting an object.

11.3.1. Size of error angle disturbance

To evaluate the influence of disturbances in the error angle, this type of disturbance was implemented into the state-space model of the Segway. The disturbance was modelled as an impulse affecting the angle of the pendulum, with a duration of one second and an amplitude of 0.3.

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.1074 & 21.3441 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -0.0471 & 14.1063 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0.2587 \\ 0 \\ 0.1136 \end{bmatrix} u + \begin{bmatrix} 0 \\ 0 \\ 0.3 \\ 0 \end{bmatrix}$$
$$y = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{11.3}$$

The resulting state space model during the impulse disturbance is in equation (11.3). In the following sections, the responses of the three controllers are evaluated.

11.3.2. LQR controller

The Simulink model in figure 6.4 was adapted to obtain the model in figure 11.16. By doing this, the disturbance resulting from changes in the error angle was implemented into the model.

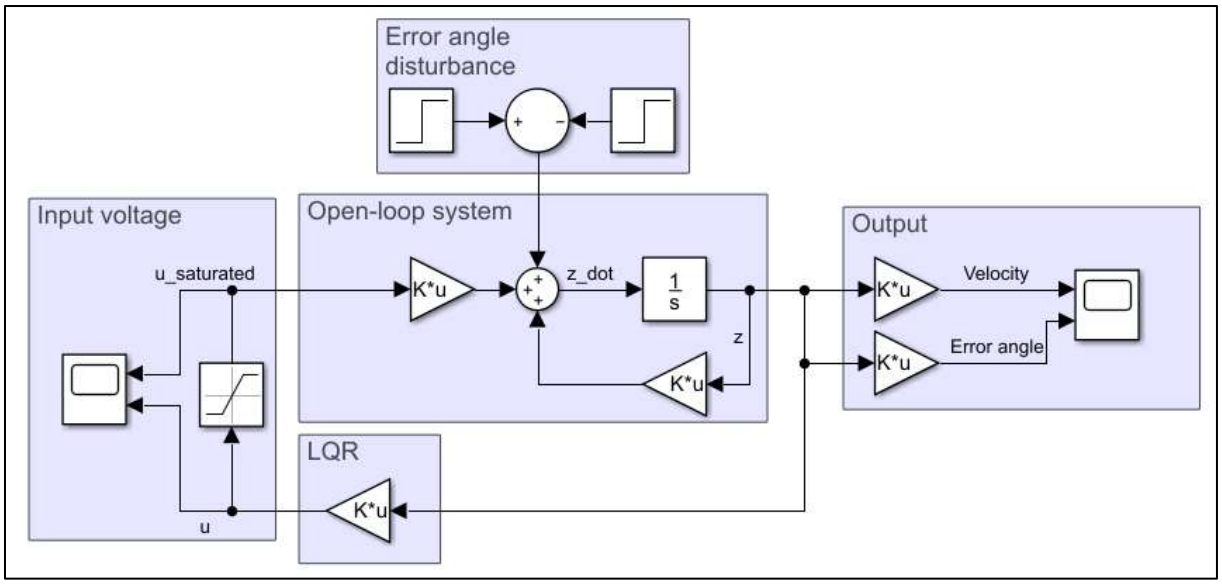

Figure 11.16: Closed-loop system with LQR control and error angle disturbance

The pulse disturbance was implemented into the model to start at $t = 6$ and end at $t = 7$. The response is imaged in figure 11.17.

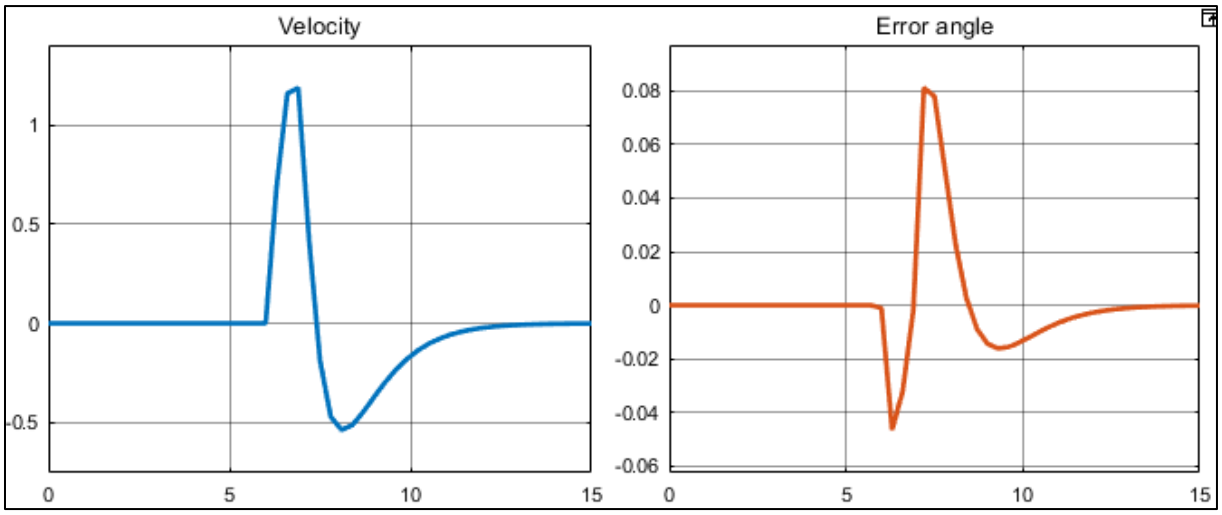

Figure 11.17: System response to an impulse error angle disturbance of size 0.3 (LQR)

As can be seen in the figure above, the disturbance was rejected by the controller rather quickly; within approximately five seconds, the system had returned to the equilibrium position. However, the systems response is quite aggressive; the actual error angle in the model the system responds to is smaller than $-0.05\ rad$. As a reaction, the velocity rises beyond $1\ m/s$ and the error angle rises to $0.08\ rad$. Thus, in terms of stabilization the controller performs well. Its aggressiveness should encourage the user to stand still and let the controller stabilize. If the user would move to much or to wildly, it might result in an unstable system.

11.3.3. LQR controller with integral action

Next, the stabilization of the Segway at the desired velocity of $3\ m/s$ using the LQR controller with integral action was tested. To be able to implement an error angle disturbance into the model, the Simulink model in figure 9.2 was adapted, resulting in the model in figure 11.18.

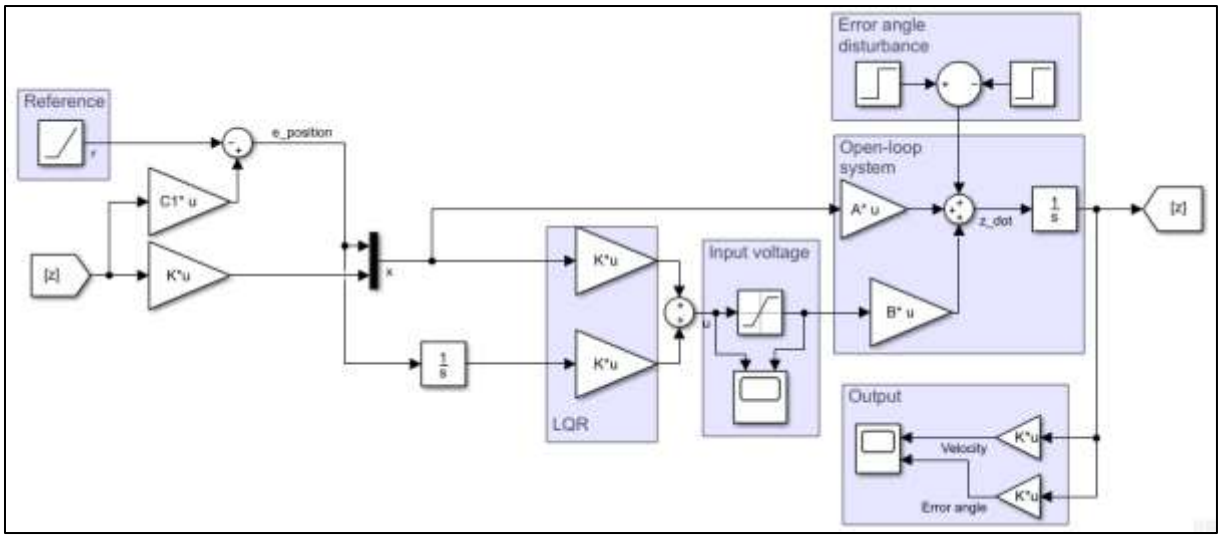

Figure 11.18: Closed-loop system with LQR control with integral action and error angle disturbance

Like with the LQR controller, the pulse was implemented to start at $t = 6$ and end at $t = 7$. The system response is in figure 11.19 below.

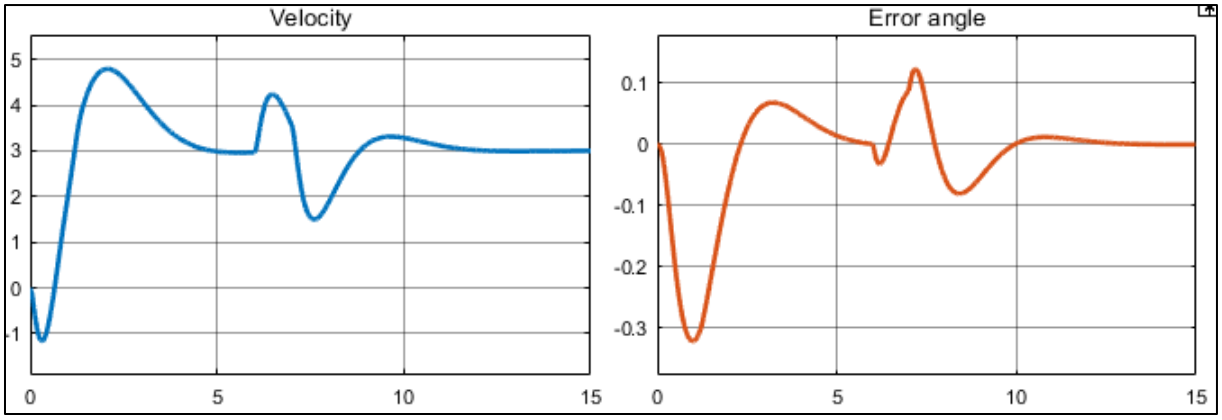

Figure 11.19: System response to an impulse error angle disturbance of size 0.3 (LQR with integral action)

The system response to the impulse disturbance was comparable to the response in figure 11.17, although the amplitudes in terms of velocity and error angle were slightly larger, due to the systems desired velocity and increased controller gains compared to the LQR controller. Altogether, the disturbance was rejected and the system stabilized within 5 seconds.

11.3.4. PID+LQR controller

Finally, the stabilization of the Segway at the desired velocity of $3\ m/s$ using the PID+LQR controller was tested. The error angle disturbance was implemented by adapting the model in figure 10.4, resulting in the Simulink model in figure 11.20.

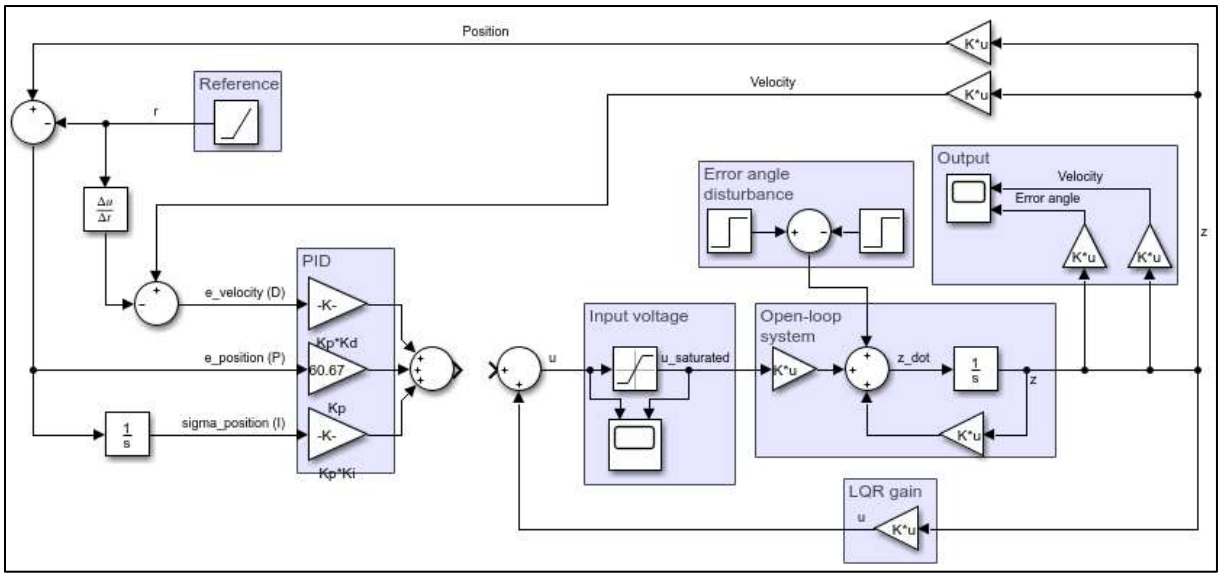

Figure 11.20: Closed-loop system with PID+LQR control and error angle disturbance.

Like in the two Simulink models discussed before, the pulse was implemented to start at $t = 6$ and end at $t = 7$. The system response is in figure 11.21 below.

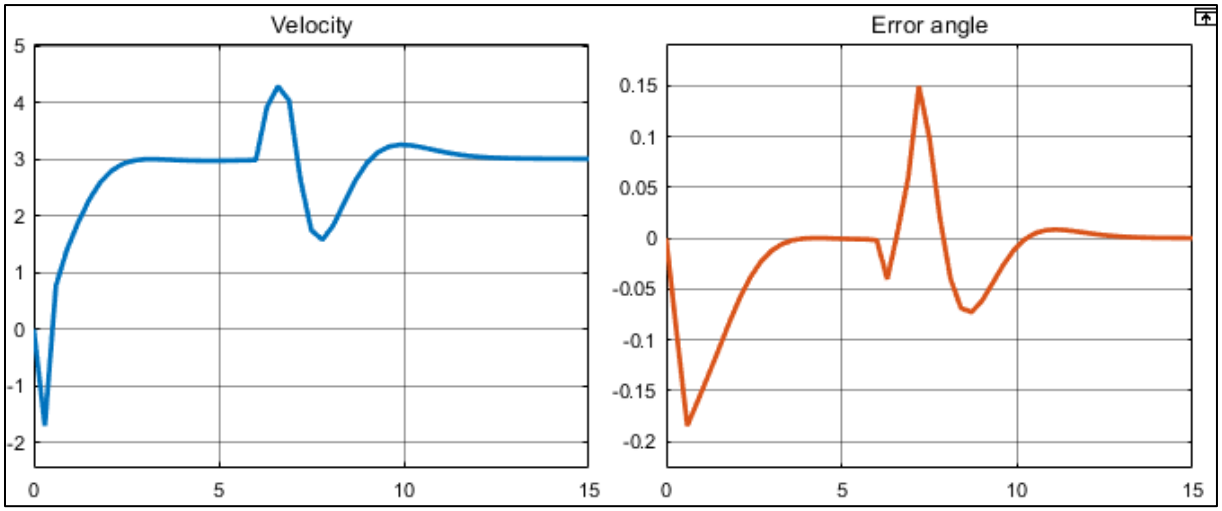

Figure 11.21: System response to an impulse error angle disturbance of size 0.3 (PID+LQR)

The system response to the impulse disturbance was comparable to the response in figure 11.19. However, the response in terms of the error angle for PID+LQR control was more aggressive than for LQR with integral action. Still, the response is largely the same as the responses when using the other controllers. Altogether, the disturbance was rejected and the system stabilized within 5 seconds.

11.4. Disturbances in the trajectory

For the trajectory, until this point it was assumed that the Segway was driving in a straight line on a perfectly flat surface. However, in a more realistic environment the presence of slopes is inevitable. When the slope would be known, the angle of the slope could be implemented into the dynamic equations of motion of the system resulting in a new (non)linear system (Castro, Modeling and dynamic analysis of a two-wheeled inverted-pendulum (masters thesis), 2012). If this model was linearized, a different model would have been obtained compared to the model in equation (4.8). In

this way, the slope would become part of the Segway model to what the LQR and PID controllers were tuned.

However, when the slope is not known, it cannot be implemented into the model. In the environment in which a Segway would act, this is probably the case; a Segway or a Segway user does not intuitively know what the slope is of the surface on which it is moving. Because the controllers cannot be tuned to the model containing the known slope, instead the Segways behavior will be investigated under various slope disturbances, using the controllers designed before.

11.4.1. Size of slope disturbance

In order to be able to implement a slope disturbance into the linearized model by Younis and Abdelati (Younis & Abdelati, 2009), the factor of the DC-motor force in the model had to be found and replaced with the DC-motor force plus the force on the Segway resulting from the slope. These forces represent the force acting on the base of the Segway.

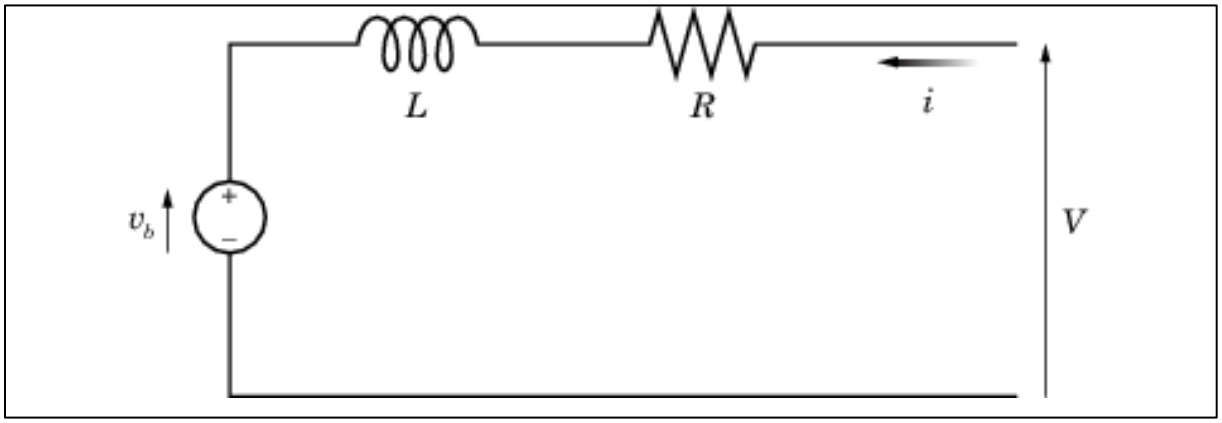

Figure 11.22: DC-motor equivalent circuit (MathWorks, 2018)

$$V = IR + L\frac{dI}{dt} + E \tag{11.4}$$

First, to obtain the DC-motor force, equation (11.4) was obtained from the DC-motor circuit in figure (11.22). I represents the current, R the resistance over the motor, L the motor inductance and E the motors back-EMF.

$$\frac{dI}{dt} = 0 \tag{11.5}$$

$$T = k_m I \tag{11.6}$$

$$V = \frac{T}{k_m}R + E \tag{11.7}$$

Under the assumption that the current through the circuit is constant (equation (11.5) and the simplification that the motor is moving at a constant velocity (equation (11.6)), equation (11.7) was obtained. Note that variable T represents the motor torque.

$$E = k_e\frac{\dot{x}}{r} \tag{11.8}$$

$$V = \frac{T}{k_m}R + k_e\frac{\dot{x}}{r} \tag{11.9}$$

Equation (11.8) was derived from the fact that the DC-motors back-EMF is equal to the back-EMF constant times the angular velocity of the motor. When this equation was substituted into equation (11.7), equation (11.9) was obtained.

$$T = V\frac{k_m}{R} - \dot{x}\frac{k_m k_e}{Rr} \tag{11.10}$$

$$F_{motor} = V\frac{k_m}{Rr} - \dot{x}\frac{k_m k_e}{Rr^2} \tag{11.11}$$

By rearranging equation (11.9) for the motor torque (T), equation (11.10) was obtained. By multiplying the torque in this equation by the radius of the Segways' wheels, the motor force in equation (11.11) was obtained. Note that this equation describes the force of one wheel to the surface; the total force resulting from the motor is twice as large, as there are two motors actuating two wheels.

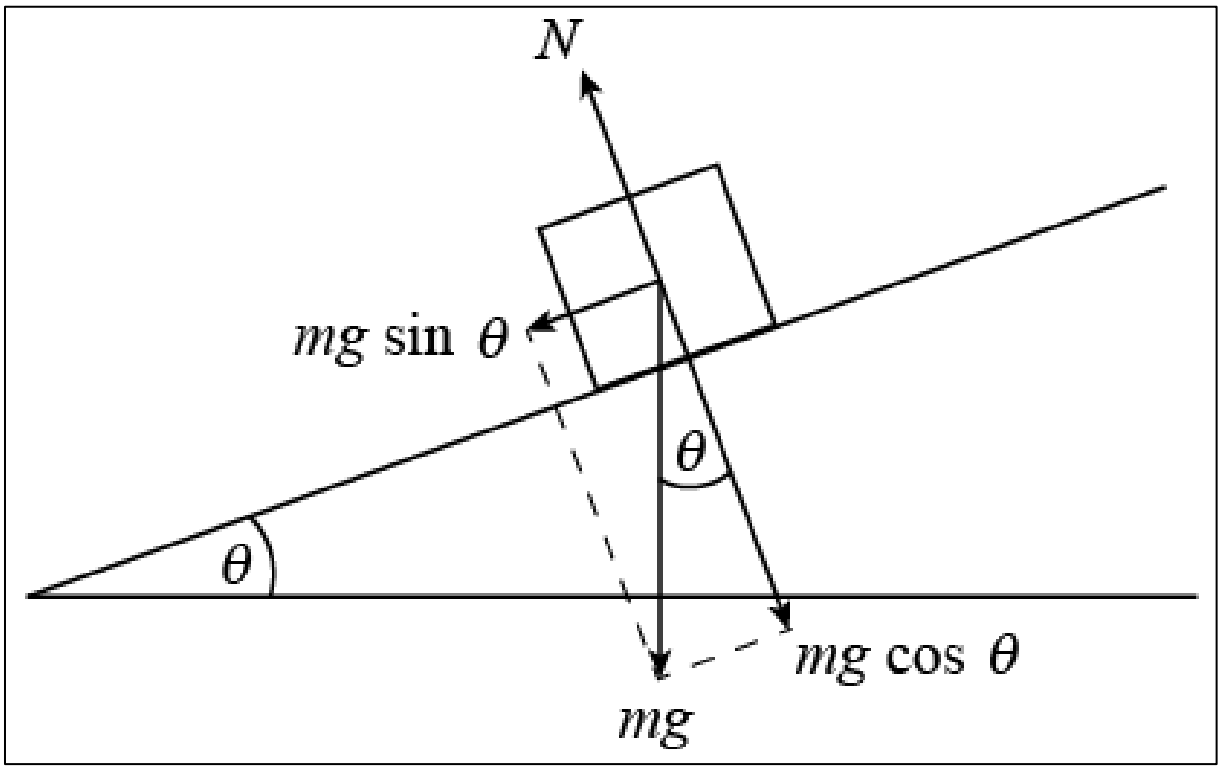

Figure 11.23: Gravitational forces on the Segway in an inclined surface (Advanced Instructional Systems, 2013)

Next, the Segway entering a slope (θ_s) was considered. Then, a new force comes into play; a component of the gravitational force now has the same direction as the force applied by the motor. This is the parallel component of the gravitational force ($F_{g,parallel}$).

$$F_{g,//} = \left(M_p + 2M_w\right) g \sin(\theta_s) \tag{11.12}$$

$$F_{total,base} = F_{motor} + F_{g,parallel} = V\frac{k_m}{Rr} - \dot{x}\frac{k_m k_e}{Rr^2} + \left(M_p + 2M_w\right) g \sin\theta_s \tag{11.13}$$

Following from the free body diagram in figure 11.23, the size of this parallel force was given by equation (11.12). This results in a new total force on the base of the Segway, shown in equation (11.13).

Then, to be able to investigate the influence of a slope on the behavior of the Segway, the motor force in the state space model in equation (4.8) was replaced by the total force acting on the base of the

Segway. The resulting state space disturbance (d_s) is in equation (11.14). Adding this disturbance to the state space system of Younis and Abdelati (Younis & Abdelati, 2009) and filling in parameter values (except for the slope) resulted in the state space system is in equation (11.15):

$$d_s = \begin{bmatrix} 0 \\ \frac{(M_p + 2M_w)g(I_p + M_p l^2 - M_p l r)}{\alpha} \\ 0 \\ \frac{(M_p + 2M_w)g(M_p l - r\beta)}{\alpha} \end{bmatrix} \sin\theta_s \tag{11.14}$$

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.1074 & 21.3441 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -0.0471 & 14.1063 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0.2587 \\ 0 \\ 0.1136 \end{bmatrix} u + \begin{bmatrix} 0 \\ 26.8711 \\ 0 \\ 11.7931 \end{bmatrix} \sin\theta_s$$
$$y = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{11.15}$$

Note that the physical meaning of a positive slope is the Segway moving downhill; there is a positive force acting on the Segways base. For a negative slope, likewise, there is a negative force acting on the base of the Segway.

Using this state-space system, the behavior of the Segway model under various slopes was checked. The behavior was checked using slopes ranging from $-0.40\ rad$ to $0.40\ rad$. The steepest road in the Netherlands is smaller than $0.2\ rad$ (klimtijd.nl, 2013). Thus, if the controller is able to maintain stability under a slope of $\pm 0.2\ rad$, the Segway would be able to operate on all Dutch roads.

The famous Baldwin Street, in Dunedin, New Zealand, is known as the world's steepest residential street (Rawlings-Way, Atkinson, & Hunt, 2009). Its official slope is $35\ \%$ which is equal to $0.34\ rad$. Thus, if the model maintains stability under a slope of $\pm 0.35\ rad$, the Segway would probably be able to operate on every road there is.

11.4.2. LQR controller

The Simulink model in figure 6.4 was adapted to obtain the model in figure 11.24. By doing this, the disturbance of an inclined surface was implemented into the model.

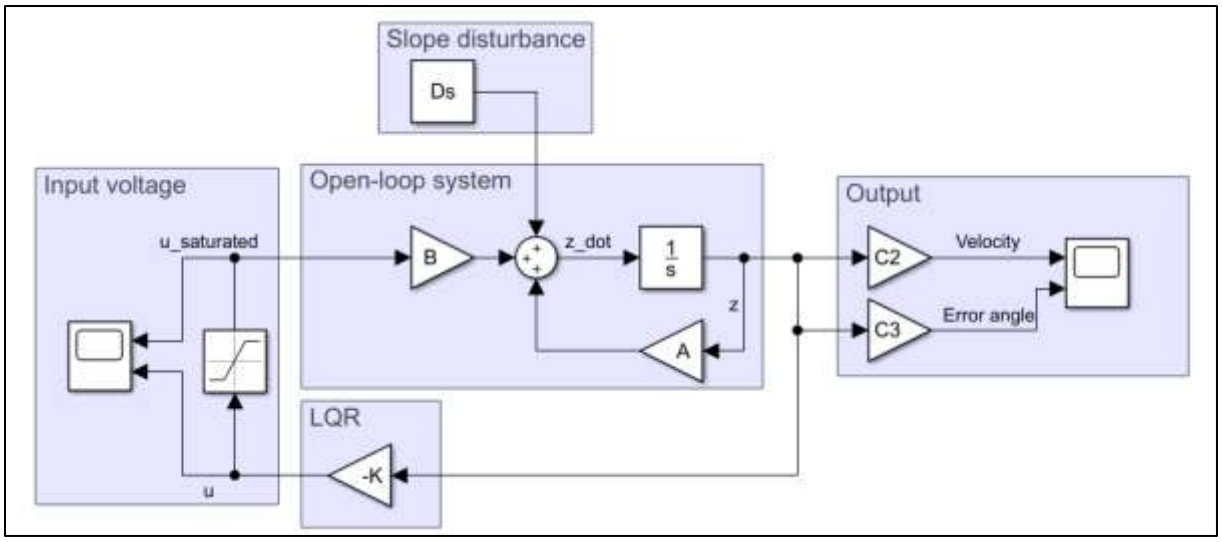

Figure 11.24: Closed-loop system with LQR control and slope disturbance

For the vertically stabilizing LQR it was investigated for which slopes the model would be able to maintain balance. Note that the gyroscope in the Segway measures the angle the system makes with the direction of the gravitational force. Thus, the system was still required to stabilize at a measured angle of zero.

In the linearized model, the maximum slope in which the Segway could stabilize was just below $0.40\ rad$. This was under an initial condition of $z_i = [0 \quad 0 \quad 0 \quad 0]^T$. When reversed, the model also stabilized from a minimum slope of a little more than $-0.40\ rad$. Thus, the real world Segway would presumably be able to stabilize in the vertical sense in any slope on a road.

11.4.3. LQR controller with integral action

By altering the Simulink model in figure 9.2, the disturbance resulting from an inclined surface was implemented into the model, resulting in the model in figure 11.25.

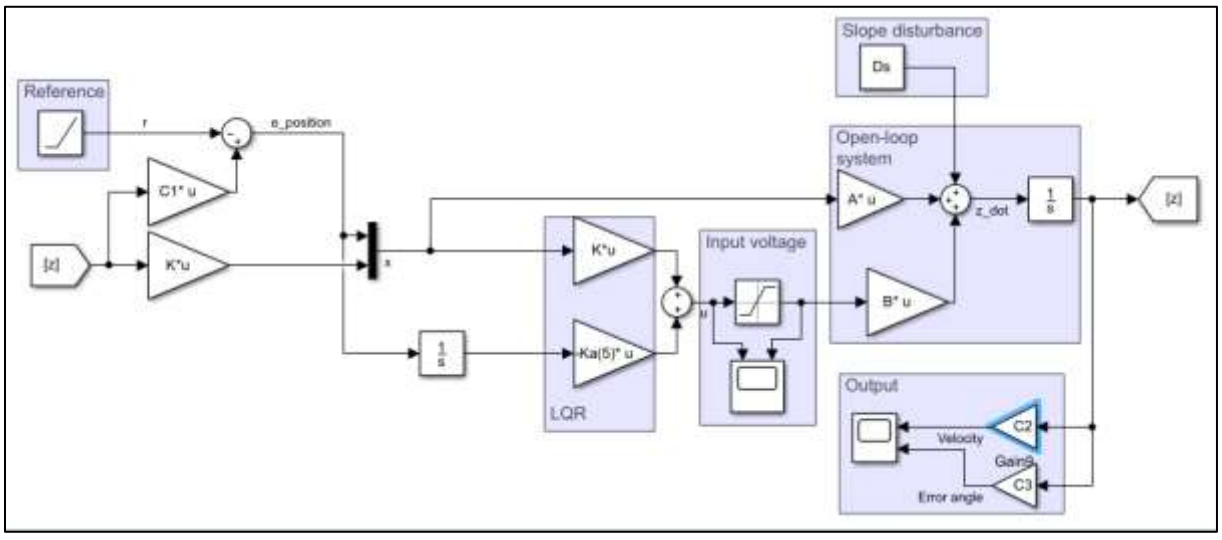

Figure 11.25: Closed-loop system with LQR control with integral action and slope disturbance

Using this model, the maximum velocity the model could attain while maintaining vertical balance under various slopes was investigated. The initial conditions considered for all the simulations were $z_i = [0 \quad 0 \quad 0 \quad 0]^T$. The results are displayed in table 11.5. The response characteristics consider the

velocity response. Note that these results correspond to the Segway regulating its velocity from zero to the desired velocity while operating in an inclined plane. This requires a larger motor force compared to the situation were the system is already at the desired velocity, and just needs to reject the disturbance resulting from the slope.

Table 11.5: Maximum velocity under various slopes and transient behavior characteristics (LQR with integral action)

Slope	Maximum reachable velocity	Rise time	Overshoot	Settling time	Steady-state error
$-0.4\ rad$	$0.15\ m/s$	$0.92\ s$	$63.12\ \%$	$4.74\ s$	$0\ m/s$
$-0.3\ rad$	$1.16\ m/s$	$0.91\ s$	$38.24\ \%$	$4.38\ s$	$0\ m/s$
$-0.2\ rad$	$1.83\ m/s$	$0.79\ s$	$24.27\ \%$	$4.25\ s$	$0\ m/s$
$-0.1\ rad$	$2.49\ m/s$	$0.75\ s$	$39.16\ \%$	$3.95\ s$	$0\ m/s$
$0\ rad$	$3.14\ m/s$	$0.71\ s$	$101.76\ \%$	$5.22\ s$	$0\ m/s$
$0.1\ rad$	$3.78\ m/s$	$0.65\ s$	$99.00\ \%$	$4.72\ s$	$0\ m/s$
$0.2\ rad$	$4.37\ m/s$	$0.64\ s$	$84.26\ \%$	$4.46\ s$	$0\ m/s$
$0.3\ rad$	$4.40\ m/s$	$0.55\ s$	$71.55\ \%$	$4.05\ s$	$0\ m/s$
$0.4\ rad$	$1.99\ m/s$	$0.20\ s$	$90.14\ \%$	$7.44\ s$	$0\ m/s$

As can be seen in the table, the rise time decreases as the slope of the inclined surface increases. This is logical; if the slope is positive, less force is required to move the Segway forward. In terms of overshoot, there were some fluctuations resulting from the differences in slope and maximum velocity. The settling time became smaller when the negative slope increased to zero. For the flat surface, however, the settling time rose to $5.22\ s$. This was caused by the absence of disturbance and the increased overshoot in the flat surface. When the slope increased to a positive angle, the settling time decreased until the slope of $0.3\ rad$.

Starting from a positive slope between 0.2 and $0.3\ rad$, a negative force is required from the DC-motor to slow the system down. This results in the fact that the model has a lower velocity limit for the steepest slopes, as a great motor force is required to counteract the parallel component of the gravitational force. The velocities given in table 11.5 are maximum velocities. Lower (positive) velocities would be attainable using the LQR controller with integral action.

Altogether, the results in table 11.5 imply that the LQR controller with integral action will be usable to stabilize the system at a velocity of $2\ m/s$ in environments containing slopes ranging from -0.1 to $0.3\ rad$. This means the controller is only suited for relatively flat areas.

11.4.4. PID+LQR controller

From the model derived earlier (figure 10.9) the new Simulink model in figure 11.26 was obtained, by inserting the slope disturbance.

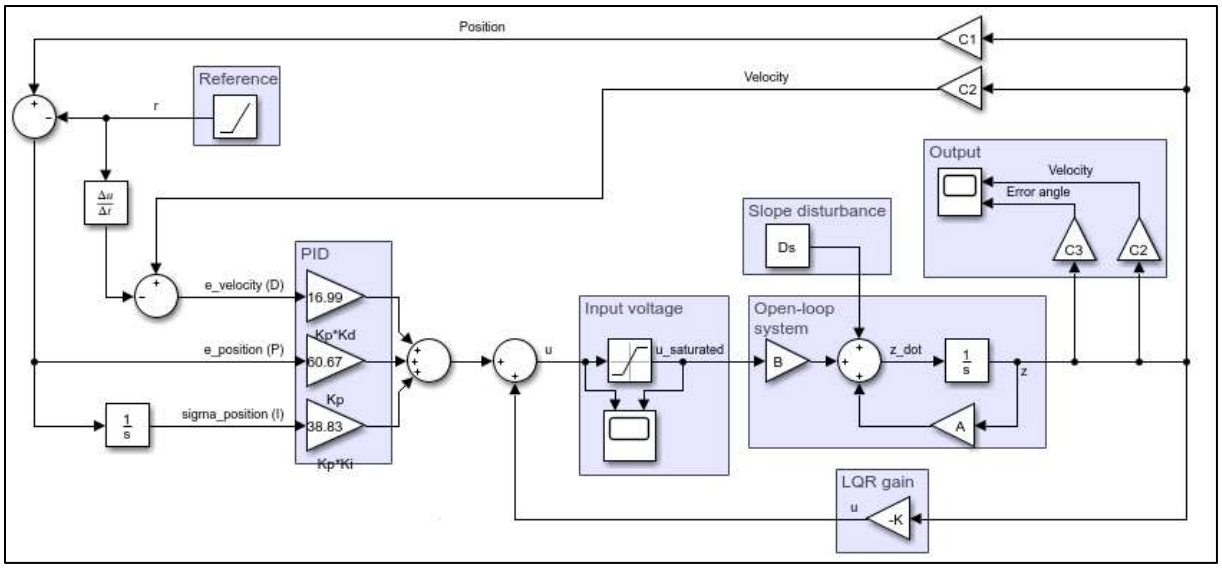

Figure 11.26: Closed-loop system with PID+LQR control and slope disturbance

Like with the LQR controller with integral action, the maximum velocity the model could attain using the PID+LQR controller under various slopes was investigated. Note that these results correspond to the Segway regulating its velocity from zero to the desired velocity while operating in an inclined plane. This requires a larger motor force compared to the situation were the system is already at the desired velocity, and just needs to reject the disturbance resulting from the slope.

Table 11.6: Maximum velocity under various slopes and transient behavior characteristics (LQR with integral action)

Slope	Maximum reachable velocity	Rise time	Overshoot	Settling time	Steady-state error
$-0.4\ rad$	Unstable for positive velocity	-	-	-	-
$-0.3\ rad$	$0.73\ m/s$	$0.30\ s$	$104.27\ \%$	$5.15\ s$	$0\ m/s$
$-0.2\ rad$	$2.71\ m/s$	$0.85\ s$	$9.80\ \%$	$3.29\ s$	$0\ m/s$
$-0.1\ rad$	$3.96\ m/s$	$0.90\ s$	$57.94\ \%$	$5.63\ s$	$0\ m/s$
$0\ rad$	$4.67\ m/s$	$0.91\ s$	$29.22\ \%$	$4.68\ s$	$0\ m/s$
$0.1\ rad$	$4.48\ m/s$	$0.65\ s$	$56.30\ \%$	$5.63\ s$	$0\ m/s$
$0.2\ rad$	$3.85\ m/s$	$1.59\ s$	$1.53\ \%$	$2.01\ s$	$0\ m/s$
$0.3\ rad$	$5.85\ m/s$	$0.52\ s$	$22.84\ \%$	$2.87\ s$	$0\ m/s$
$0.4\ rad$	$5.81\ m/s$	$0.57\ s$	$14.37\ \%$	$3.25\ s$	$0\ m/s$

As can be seen in table 11.6, the results when using the PID+LQR controller were not as linearly connected to the slope as with the LQR controller with integral action. Nonetheless, like with the other controller, for large slopes the maximum reachable velocity does not increase. Also, the Segway model was uncontrollable under a slope of $-0.4\ rad$. This is caused by the aggressiveness of the controller; it attempts to stabilize too fast, resulting in an error angle too large for the DC-motor to stabilize, because of the motors saturation. However, for angles larger than $-0.3\ rad$, the maximum reachable velocity is relatively high when compared to the results of the LQR controller with integral action. The velocities given in table 11.6 are maximum velocities. Lower (positive) velocities would be attainable

using the PID+LQR controller, except for the case when the slope is $0.4\ rad$. In this case, a positive velocity is required, as the parallel component of the gravitational force is too large to allow the system to stabilize at a low velocity (slower than $1\ m/s$).

Altogether, the results in table 11.6 imply that the PID+LQR controller will be usable to stabilize the system at a velocity of $2.5\ m/s$ in environments containing slopes ranging from -0.2 to $0.4\ rad$. This means the controller is able to reject most (normal) slopes. When the slopes get larger than $0.2\ rad$, the controller is not suited for control of the Segway system. The PID+LQR controller would thus suffice for most Segways acting as a transportation vehicle.

11.5. Testing the controllers under multiple combined disturbances

11.5.1. Test situation

As a final test, the disturbances discussed previously were combined in order to check the models behavior in a realistic situation for both controllers. The situation modelled consisted of the following steps:

- A user of $80\ kg$, $1.80\ m$ and $72.78\ kg \cdot m^2$ mounts the Segway at time $t = 0$ under an initial error angle of $-0.17\ rad$. Note that this represents the user leaning forward. Sensor noise is equal to the white noise in the section on noise rejection.
- The Segway is stabilized by the LQR controller.
- At time $t = 10$ the automatic pilot is started with a desired velocity as high as allowed by the saturation.
- The LQR controller with integral action or the PID+LQR controller stabilizes the Segway at the desired velocity.
- At time $t = 20$ the Segway enters an inclined surface with slope $-0.4\ rad$.
- At time $t = 30$ the surface plane becomes flat again.
- At time $t = 40$ the Segway enters a surface with slope $0.4\ rad$.
- At time $t = 50$ the surface becomes flat again.
- At time $t = 60$ the automatic pilot is turned off and the controller for trajectory tracking brings the velocity back to zero. This was done by the controllers for trajectory instead of the LQR controller for vertical stabilization, as this resulted in a better response in the linearized model.

The initial error angle of $-0.17\ rad$ was chosen because it is assumed to be a realistic error angle resulting from the user mounting the Segway. The velocity is regulated to its maximum in order to test the controller with noise. The changes in slope are relatively large; this was done because the system response to small slopes was barely visible once the Segway was at its desired velocity level. Finally, the automatic pilot was turned off to check the systems response to decreasing the desired velocity. Disturbances in the error angle were not considered in the test situation, because in the ideal situation, the user is not moving. This implies the test is actually a representation of a real situation were the user is rigid and does not move. If the model would maintain stability throughout the run, the quality of the controller's performance in the linearized model is high enough to be implemented and evaluated on a real life Segway.

11.5.2. LQR controller with integral action

In order to model the situation described above with the LQR controller with integral action for trajectory tracking, the models derived before were combined and adapted into the model that is in figure 11.27.

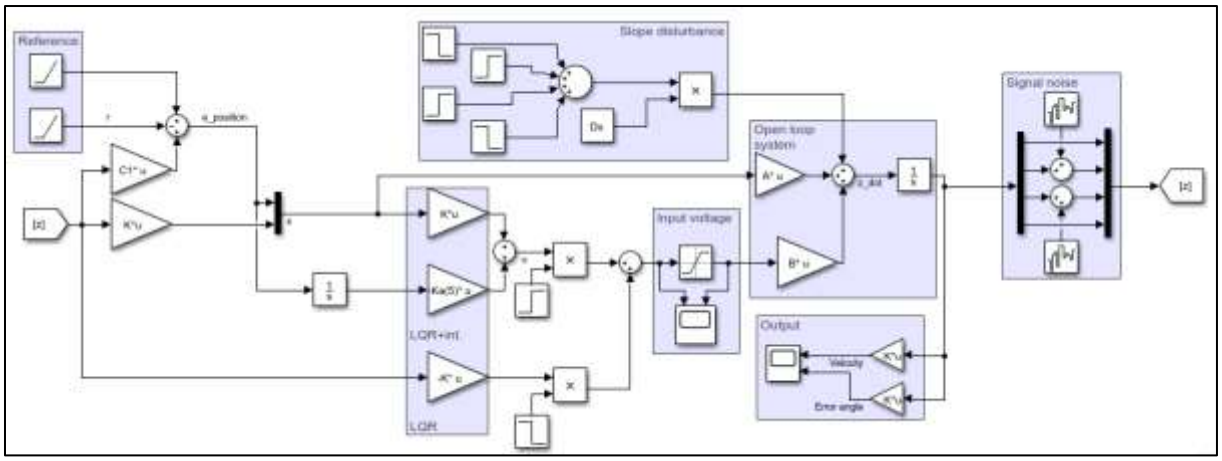

Figure 11.27: Closed-loop system with combined disturbances (LQR with integral action)

In the Simulink model in the figure above, the disturbances discussed before were combined. Simulink's step and product blocks were used to model transitions between controllers or slopes. The systems response in terms of velocity and error angle are in figure 11.28. The input signal (voltage) is displayed in figure 11.29.

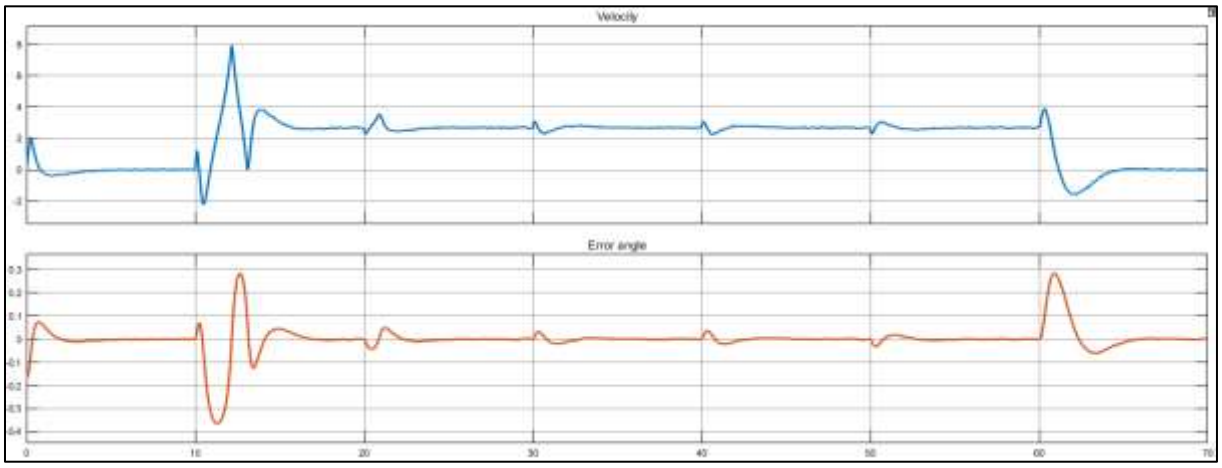

Figure 11.28: Systems response in terms of velocity and error angle

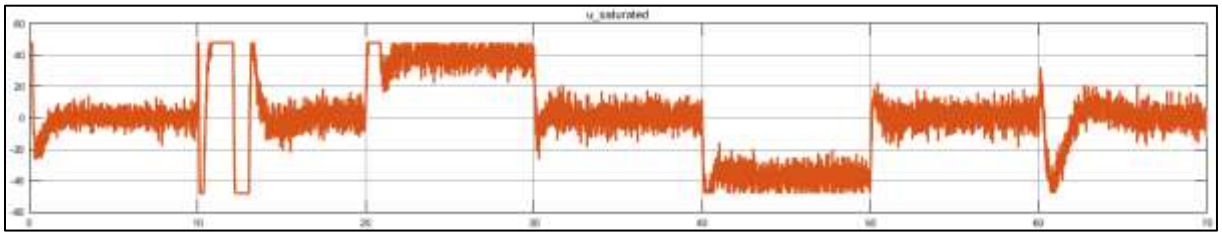

Figure 11.29: Systems response in terms of input voltage

The highest velocity which could be attained by the model 10 seconds after stabilizing from an error angle of $0.17\ rad$ was $2.68\ m/s$. As can be seen in the figures above, the influence of the noise carried on to the Segways states was negligible, while the noise in the input voltage was relatively large. Also the influence of the user being different from the user in the model was imperceptible. The largest oscillating behavior was resulting from the automatic pilot turning on, which is logical as the velocity is rising to its maximum. The influence of the slope disturbances was relatively small, even for the large slopes of $\pm 0.4\ rad$. From this fact it was concluded that once the Segway is at the desired velocity,

disturbances in the form of slopes pose no issue to the Segways stabilization. Noteworthy is the slowing down of the system, which expresses far less oscillating behavior than speeding up.

Altogether, the LQR controller with integral action is well suited to control the Segway with automatic pilot under the expected disturbances. To maintain stability in the real life Segway implementation, it is advised to set the value of the desired velocity when the automatic pilot is turned on to $2.5\ m/s$. This is slightly under the velocity used in the test above, to maintain some 'safety margin' to ensure safety for the Segway user.

11.5.3. PID+LQR controller

In order to model the situation described in the section on the test situation with the PID+LQR controller for trajectory tracking, the models derived before were combined and adapted into the model that is in figure 11.30.

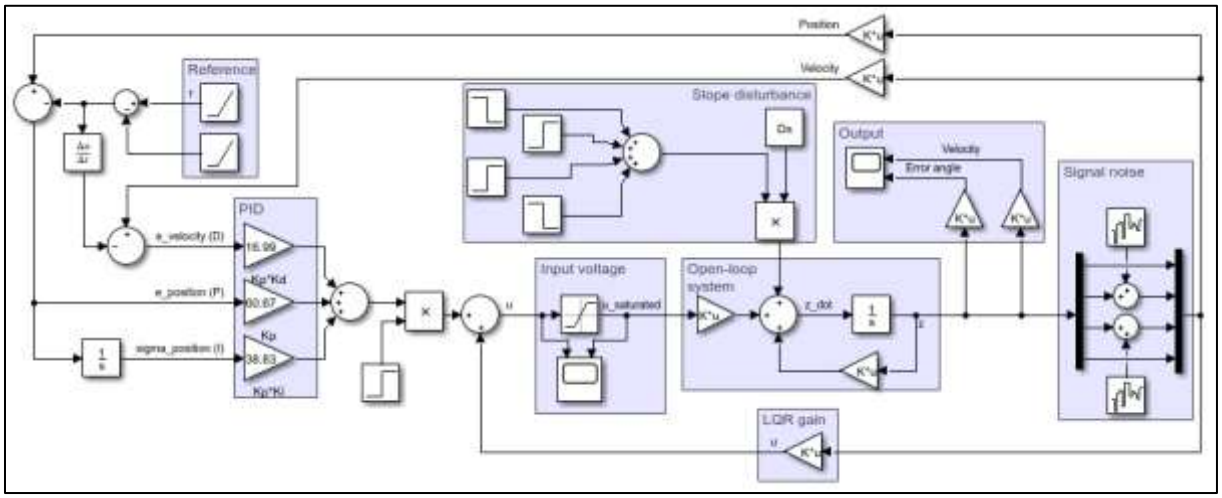

Figure 11.30: Closed-loop system with combined disturbances (PID+LQR)

Like in the model in figure 11.21, the disturbances discussed before were combined. Simulink's step and product blocks were used to model transitions between controllers or slopes. The systems response in terms of velocity and error angle are in figure 11.31. The input signal (voltage) is displayed in figure 11.32.

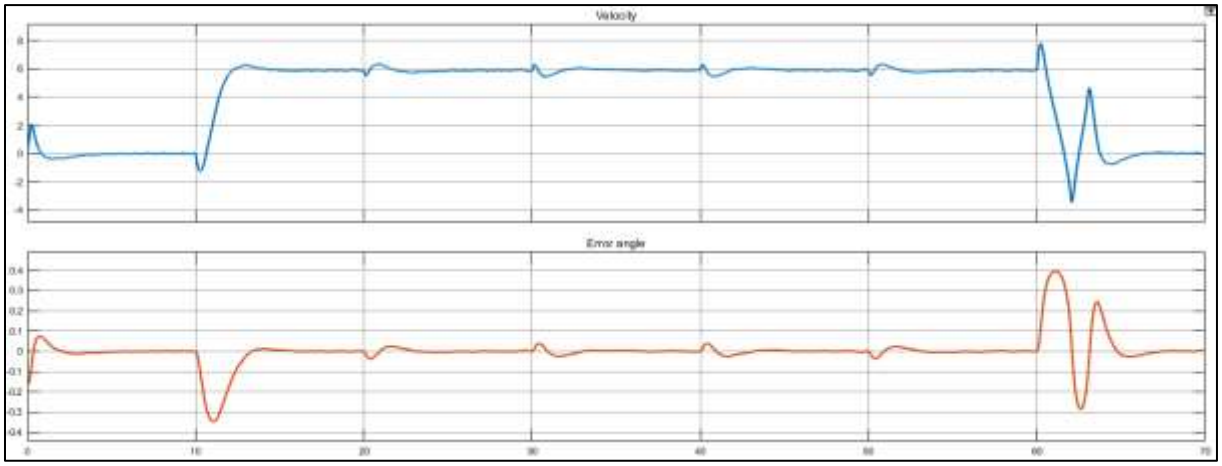

Figure 11.31: Systems response in terms of velocity and error angle

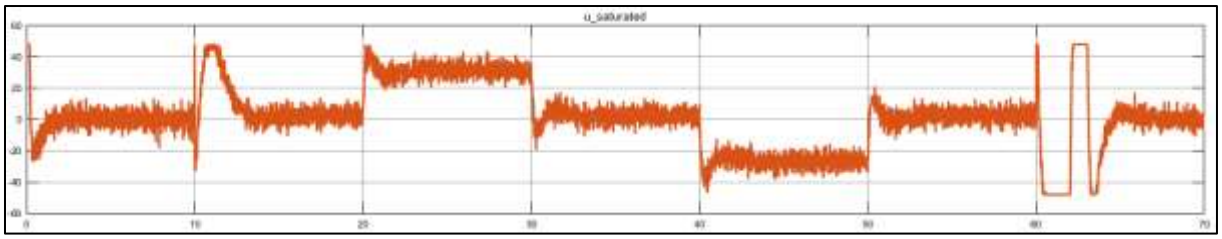

Figure 11.32: Systems response in terms of input voltage

The highest velocity which could be attained by the model 10 seconds after stabilizing from an error angle of $0.17\ rad$ was $5.9\ m/s$. As can be seen in the figures above, the influence of the noise carried on to the Segways states was negligible, while the noise in the input voltage was relatively large. Also the influence of the user being different from the user in the model was imperceptible, like was the case when using the LQR controller with integral action. The largest oscillating behavior was resulting from the velocity reducing back to zero. The speeding up of the system after turning the automatic pilot on resulted in way less oscillating behavior. This is exactly opposite to the behavior of the LQR controller with integral action discussed before.

The influence of the slope disturbances was relatively small, even for the large slopes of $\pm 0.4\ rad$. From this fact it was concluded that also for the PID+LQR controller, once the Segway is at the desired velocity, disturbances in the form of slopes pose no issue to the Segways stabilization.

Altogether, the PID+LQR controller is very well suited to control the Segway with automatic pilot under the expected disturbances. In terms of desired velocity for the automatic pilot, there is pretty much no boundary. As the test situation above could remain stable at a velocity of up to $5.9\ m/s$, any value between 0 and $5\ m/s$ should be applicable in a real situation. However, for most applications the desired velocity of the automatic pilot would most likely be below $5\ m/s$.

12. Discussion

The controllers proposed in this project were based on the linearized model of a Segway. In the section on disturbances and noise, the controller's robustness was tested. However, when a model is linearized, it is only valid for a tight set of values around the equilibrium point. Thus, the borders of stability found in this project may not comply with reality. Still, based on the findings in this project, the controllers will be expected to reject most disturbances, as long as they are not too large.

In order to state if the approaches hereby proposed are suitable or not for the design of the automatic pilot function, the controllers must be implemented in the nonlinear model of the system, which captures physical phenomena that are neglected by the linearized model. Accordingly, once the simulation results are satisfactory, it is possible to proceed with the experimental validation of this work.

Would the controllers be rendered unsuitable by implementation in the nonlinear model, multiple other approaches could be suggested. First of all, a nonlinear approach to the system would provide other and possibly better controllers. However, nonlinear control would result in a more complicated problem, which might be unnecessary in the relatively simple case of the Segway system. Another approach might be to tune the controller to the buyer; let the buyer specify his or her height and weight, so that the controller can be adapted to the user. A variant to this approach would be to implement a weight sensor into the Segway, so that the control signal can be scaled to the weight of the user. The same can be done to deal with slope disturbances; implement a sensor measuring the slope of the plane in which the Segway is moving and implement this slope into the controller. These approaches would lead to a better controller, but also to a more complicated and expensive system.

In the controllers proposed in this project, the desired velocity is increased from zero to the desired level instantly. Another adaption to the controllers in this project would be to gradually increase the velocity instead. This would result in a less aggressive stabilization, resulting in higher attainable velocities. The drawback to this approach is that when the user does want to reduce the velocity instantly, the Segway might become unstable.

Another alternative approach to trajectory tracking could be to regulate the system in a desired position. In this way, the reference is not the desired velocity of the system but the desired final position. However, this approach was not considered in this project, because a controller would try to move the system to the desired position as fast as possible. This approach would result in problems with the saturation of the DC-motors and variability in the velocity. Still, by tuning the controller parameters, these issues could be solved.

As a final remark, the model used in this system did not incorporate the option of turning. It was assumed that turning the Segway would have little influence on the stabilization of the vertical position of the Segway or its velocity. If the controllers in this project would be implemented into a Segway, the effect of steering should be investigated more deeply.

13. Conclusions

In the beginning of this project an LQR controller for vertical stabilization was proposed. This controller showcased a good balance between the aggressiveness of the stabilization and the required voltage over the DC-motors. The two controllers for trajectory tracking (LQR with integral action and LQR+PID) were designed and extensively tested. Altogether both controllers performed satisfactory, although the PID+LQR controller was considered more robust. Especially in the attainable velocity the PID+LQR controller performed better with an attainable velocity of $4.67\ m/s$. In terms of stabilization in a slope, the PID+LQR controller performed worse than the other two controllers. However, when the system was already at the desired velocity and the system entered a slope, the resulting disturbance was minimal for all controllers. When gaining velocity, the response of the PID+LQR controlled system was better, while decreasing in velocity resulted in a better response from the LQR with integral action controlled system.

In the end, both controllers would be well suited to control the system of the Segway with automatic pilot. In terms of robustness, the PID+LQR controller would be the best controller to use. When a larger robustness is required from the Segway, this might be solved by increasing the capacity of the actuating DC-motors.

Bibliography

Advanced Instructional Systems, I. a. (2013). *Acceleration of Gravity*. Retrieved from WebAssign: http://www.webassign.net/question_assets/tamucolphysmechl1/lab_2/manual.html

Boubaker, O. (2012). The Inverted Pendulum Benchmark in Nonlinear Control Theory: A Survey . *International Journal of Advanced Robotic Systems*, 1-9.

Castro, A. (2012). *Modeling and dynamic analysis of a two-wheeled inverted-pendulum (masters thesis).* Georgia: George W. Woodruff School of Mechanical Engineering.

Castro, A., Adams, C., & Singhose, W. (2013). Dynamic response characteristics of a two-wheeled inverted-pendulum transporter. (pp. 1532-1537). Florence, Italy: IEEE.

de Persis, C. (2017). *Control Engineering – Lecture 13.* Retrieved from nestor.rug.nl: https://nestor.rug.nl/bbcswebdav/pid-8678987-dt-content-rid-10255207_2/courses/TBKRT05E.2017-2018.1A/lecture13_2017-18.pdf

de Persis, C. (2017). *Control Engineering – Lecture 4*. Retrieved from nestor.rug.nl: https://nestor.rug.nl/bbcswebdav/pid-8645339-dt-content-rid-10101281_2/courses/TBKRT05E.2017-2018.1A/lecture_04_2017-18.pdf

Dormand, J. R., & Prince, P. J. (1980). A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, 19–26.

Franklin, G. F., Powell, J. D., & Workman, M. L. (1990). *Digital Control of Dynamical Systems.* Reading: Addison-Wesley Publishing Company.

Golson, J. (2015, January 16). *WELL, THAT DIDN'T WORK: THE SEGWAY IS A TECHNOLOGICAL MARVEL. TOO BAD IT DOESN'T MAKE ANY SENSE*. Retrieved from Wired: https://www.wired.com/2015/01/well-didnt-work-segway-technological-marvel-bad-doesnt-make-sense/

Gopi, P., & Suman, G. (2015). A new approach for Tuning of PID Load Frequency. *International Journal of Modern Trends in Engineering*, 18-28.

Grasser, F., D'Arrigo, A., Colombi, S., & Rufer, A. (2002). JOE: A Mobile, Inverted Pendulum. *IEEE Transactions on Industrial Electronics (Volume: 49, Issue: 1)*, 107-114.

Isaksson, A. J., & Graebe, S. F. (2002). Derivative filter is an integral part of PID design. *IEE Proceedings - Control Theory and Applications*, 41 – 45.

Kamen, D., Ambrogi, R., Duggan, R., Heinzmann, R., Key, B., Skoskiewicz, A., & Kristal, P. (1997). *USA Patent No. 5,701,965.*

Khalil, H. K. (1996). *Nonlinear systems (Chapter 12).* New Jersey: Prentice-Hall.

Kiam Heong Ang, G. C. (2005). PID control system analysis, design, and technology. *IEEE Transactions on Control Systems Technology*, 559-576.

klimtijd.nl. (2013). *Keutenberg*. Retrieved from klimtijd.nl: https://www.klimtijd.nl/beklimming/keutenberg

Lam, Q. M., Stamatakos, N., Woodruff, C., & Ashton, S. (2003). Gyro Modeling and Estimation of Its Random Noise Sources. *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 1-11.

Lipták, B. G. (2003). *Instrument Engineers' Handbook: Process control and optimization (p. 108).* CRC Press.

MathWorks. (2018). *DC Motor*. Retrieved from MathWorks: https://nl.mathworks.com/help/physmod/elec/ref/dcmotor.html

MathWorks. (2018). *Introduction to Model-Based PID Tuning in Simulink*. Retrieved from MathWorks: https://nl.mathworks.com/help/slcontrol/ug/introduction-to-automatic-pid-tuning.html

McCormack, A. S., & Godfrey, K. R. (1998). Rule-Based Autotuning Based on Frequency Domain Identification. *IEEE Transactions on Control Systems Technology*, 43-61.

Morin, D. (2010). *Introduction to Classical Mechanics: With Problems and Solutions.* Cambridge University Press.

Murray, R. M. (2006, January 11). *Lecture 2 – LQR Control.* Retrieved from Caltech: http://www.cds.caltech.edu/~murray/courses/cds110/wi06/lqr.pdf

Prasad, L. B., Tyagi, B., & Gupta, H. O. (2015). Optimal Control of Nonlinear Inverted Pendulum System Using PID Controller and LQR: Performance Analysis Without and With Disturbance Input. *International Journal of Automation and Computing*, 661–670.

Precision Microdrives Limited. (2018). *DC MOTOR SPEED: VOLTAGE AND TORQUE RELATIONSHIPS*. Retrieved from Precision Microdrives Limited.: https://www.precisionmicrodrives.com/tech-blog/2015/08/03/dc-motor-speed-voltage-and-torque-relationships

Rawlings-Way, C., Atkinson, B., & Hunt, E. (2009). *New Zealand's South Island.* Lonely Planet.

Rubinroit, S. (2016, July 20). *The true story behind Usain Bolt's Segway accident*. Retrieved from NBC Olympics: http://www.nbcolympics.com/news/true-story-behind-usain-bolts-segway-accident

Schroter, W. (2007, July 9). *When To Dump That Great Idea*. Retrieved from Forbes: http://web.archive.org/web/20080512091217/http://www.forbes.com/entrepreneurs/2007/07/06/apple-amazon-myspace-ent-manage-cx_ws_0709dump.html

Searock, J., Browning, B., & Veloso, M. (2004). Turning Segways into Soccer Robots. *Intelligent Robots and Systems*.

Segway Inc. . (2014). *User Manual Segway Personal Transporter (PT).* Retrieved from segway.com: http://www.segway.com/media/1195/24010-00001_aa_se_um_en_usb_user-manual.pdf

Segway Inc. (2006). *Segway Milestones.* Retrieved from segway.com: https://web.archive.org/web/20090318154531/http://www.segway.com/about-segway/segway-milestones.php

Segway Inc. (2016). *Segway Patent Information.* Retrieved from segway.com: http://www.segway.com/patents.pdf

Segway tours Copenhagen. (n.d.). *Segway tours Copenhagen*. Retrieved from Segway tours Copenhagen: https://segwaytourscopenhagen.com/

Sun, Y., & Yook, J. (1998, 12 6). *Function rscale: Finding the Scale Factor to Eliminate Steady-State Error.* Retrieved from Control Tutorials for Matlab and Simulink: http://ctms.engin.umich.edu/CTMS/index.php?aux=Extras_rscale

University of Groningen. (2017). *Mechatronics 2017-2018 Tutorial 3.* Retrieved from nestor.rug.nl: https://nestor.rug.nl/bbcswebdav/pid-8733198-dt-content-rid-10420546_2/courses/TBMETR-12.2017-2018.1B/tutorial-3-2.pdf

Vinodh Kumar, E., & Jerome, J. (2013). Robust LQR Controller Design for Stabilizing and Trajectory Tracking of Inverted Pendulum. *Procedia Engineering 64*, 169 – 178.

Williams, S. (2010, September 27). *Segway Owner Dies in Segway Crash*. Retrieved from The New York Times: https://wheels.blogs.nytimes.com/2010/09/27/segway-owner-dies-in-segway-crash/?_r=0

Younis, W., & Abdelati, M. (2009). Design and Implementation of an Experimental Segway Model. *Proceedings of the 2nd Mediterranean Conference on Intelligent Systems and Automation.* Zarzis, Tunisia.

Zhong, J. (2006). PID Controller Tuning: A Short Tutorial. Mechanical Engineering, Purdue University.

Ziegler, J., & Nichols, N. B. (1942). Optimum Settings for Automatic Controllers. *Transactions of the ASME*, 759–768.

Appendix A: Matlab script used for simulations

```
%% Parameters
Km = 0.869; % Constant of motor torque
Ke = 0.083; % Constant of the motor's back-EMF
L = 1.7; % Length of the pendulum
R = 0.2; % Wheel radius
Rm = 1; % Resistance of motor
Mp = 85; % Mass of the pendulum
Ip = 68.98; % Inertia of the pendulum
Mw = 3.5; % Mass of the wheel
Iw = 0.07; % Inertia of the wheel
g = 9.81; % Gravity
slope = 0.5*pi; % Surface slope

%% Matrixes
beta = 2*Mw + ((2*Iw) / (R*R)) + Mp;
alpha = Ip*beta + 2*Mp*L*L*((Mw*Iw) / (R*R));

A = [0, 1, 0, 0; 0, (2*Km*Ke*(Mp*L*R - Ip - Mp*L*L)) / (Rm*R*R*alpha),
(Mp*Mp*g*L*L) / alpha, 0; 0, 0, 0, 1; 0, (2*Km*Ke*(R*beta - Mp*L)) /
(Rm*R*R*alpha), (Mp*g*L*beta) / alpha, 0];

B = [0; (2*Km*(Ip + (Mp*L*L) - (Mp*L*R))) / (Rm*R*alpha); 0;
(2*Km*(Mp*L - R*beta)) / (Rm*R*alpha)];

Ds = [0; ((Mp+2*Mw)*g*sin(slope)*(Ip + (Mp*L*L) - (Mp*L*R))) /
(alpha); 0; ((Mp+2*Mw)*g*sin(slope)*(Mp*L - R*beta)) / (alpha)];

C = [0, 1, 0, 0; 0, 0, 1, 0];

C1 = [1, 0, 0 ,0];
C2 = [0, 1, 0, 0];
C3 = [0, 0, 1 ,0];
C4 = [0, 0, 0, 1];

D = 0;

%% Open-Loop system
states = {'x' 'x_dot' 'phi' 'phi_dot'};
inputs = {'u'};
outputs = {'x'; 'phi'};

sys_ol = ss(A, B, C, D, 'statename', states, 'inputname', inputs,
'outputname', outputs);

poles = eig(A);
co = ctrb(A, B);
controllability = rank(co);

%% LQR

Q = zeros(4, 4);
Q(1,1) = 1;
Q(2,2) = 1;
Q(3,3) = 11.1;
```

```
Q(4,4) = 1.6;

r = 0.00043;

K = lqr(A, B, Q, r);

Ac = [(A-B*K)];
Bc = [B];
Cc = [C];
Dc = [D];

sys_cl = ss(Ac, Bc, Cc, Dc, 'statename', states, 'inputname', inputs,
'outputname', outputs);

%% Augmented LQR trajectory tracking and PID
Kp = 60.67;
Ki = 60.67*0.64;
Kd = 60.67*0.28;

Aa = [A zeros(4, 1); C1 0];
Ba = [B; 0];
Ca = [C1 0];

sys_aug = ss(Aa, Ba, Ca, Dc);
augcontrollability = rank(ctrb(Aa, Ba));

Kpidlqr = [(K(1)-Kp) (K(2)-Kd) K(3) K(4) -Ki];
Qa = [Q zeros(4, 1); zeros(1, 4) 5];
Kalqr = lqr(Aa, Ba, Qa, r);
```