Linköping University | Department of Management and Engineering
Master's thesis, 30 ECTS | Mechanical Engineering - Mechatronics
2021 | LIU-IEI-TEK-A--21/04015–SE

Controlling a Hydraulic System using Reinforcement Learning

– Implementation and validation of a DQN-agent on a hydraulic Multi-Chamber cylinder system

David Berglund
Niklas Larsson

Supervisor : Henrique Raduenz
Examiner : Liselott Ericson

External supervisor : Kim Heybroek (Volvo CE)

Linköpings universitet
SE–581 83 Linköping
+46 13 28 10 00 , www.liu.se

Abstract

One of the largest energy losses in an excavator is the compensation loss. In a hydraulic load sensing system where one pump supplies multiple actuators, these compensation losses are inevitable. To minimize the compensation losses the use of a multi chamber cylinder can be used, which can control the load pressure by activate its chambers in different combinations and in turn minimize the compensation losses.

For this proposed architecture, the control of the multi chamber cylinder systems is not trivial. The possible states of the system, due to the number of combinations, makes conventional control, like a rule based strategy, unfeasible. Therefore, is the reinforcement learning a promising approach to find an optimal control.

A hydraulic system was modeled and validated against a physical one, as a base for the reinforcement learning to learn in simulation environment. A satisfactory model was achieved, accurately modeled the static behavior of the system but lacks some dynamics.

A Deep Q-Network agent was used which successfully managed to select optimal combinations for given loads when implemented in the physical test rig, even though the simulation model was not perfect.

Acknowledgments

First we would like to thank Martin Hochwallner for all the explanations and time he put down to realise the real time implementation, and Samuel Kärnell for sharing his hydraulic knowledge in the lab.

Our fellow student colleagues Chester, Andy and Keyvan, for the cooperation in the early stages in the work, and the Mathworks personell Chris, Juan & Gaspar for their assistance with matlab struggels and technical inputs.

We would also thank our examiner and supervisors. Liselott Ericson for quick response for administrative quest, good tips and guidance for hardware. Kim Heybroek from Volvo CE for good inputs and insights. Finally a special thanks to our supervisor, Henrique Raduenz, for all the long conversations, inputs and technical support making this project possible.

Contents

List of Figures

List of Tables

List of Abbreviations

DQN	Deep Q-Network
DV	Digital Valve
HMI	Human Machine Interface
LVDT	Linear Variable Differential Transformer
MC	Multi Chamber Cylinder
MC$_{A-D}$	Multi Chamber Cylinder chambers A to D
MPC	Model Predictive Control
NN	Neural Networks
PCV	Pressure Compensating Valve
PPO	Proximal Policy Optimization
PRV	Pressure Relief Valve
PV	Proportional Valve
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
SC	Single Chamber Cylinder

List of Symbols

Quantity	**Description**	**Unit**
β	Angle	rad
δ	Damping	$-$
γ	Discount factor	$-$
ω_a	Angular acceleration of body a	m/s^2
ω_{PV}	Resonance Frequency	rad/s
Φ	Angle	rad
ψ	Angle	rad
ρ	Density	kg/m^3
τ	Time delay	s
θ	Reinforcement Learning Policy parameters	$-$
ε	Reinforcement Learning Exploration Rate	$-$
φ	Angle	rad
A	Area	m^2
a	Reinforcement Learning Action	$-$
C_q	Flow coefficient	$-$
F	Force	N
g	Gravitational acceleration	m/s^2
$gear$	Combination of active chambers	$-$
I_a	Moment of inertia	kgm^2
L	Circumference	m
M	Torque	Nm
m	Mass	kg
P	Power	W
p	Pressures	Pa
Q	Reinforcement Learning Value Function	$-$
q	Flow	m^3/s
r	Reinforcement Learning Reward	$-$
r_{AB}	Distance between points A and B	m
$rate$	Rate limit	$-$
s	Reinforcement Learning State	$-$
T_s	Sample time	s
V	Volume	m^3
v	Velocity	m/s
x	Position	m

1 Introduction

Advancements in hydraulics and machine learning opens up for opportunities to develop optimized and sophisticated control strategies for complex problems. This could help increase the energy efficiency and performance of construction equipment which are known for low efficiency. Because of global warming and increasing oil prices, new technology for excavators are required to improve fuel economy and reduce emission.

1.1 Background

Load sensing systems are a widely used system in modern construction machines. Their ability to adjust the system pressure enables energy savings and the proportional valves (PV) allows for smooth control at low velocities. To maintain a constant pressure drop over the proportional valve a pressure compensating valve (PCV) is used, which gives same velocity for the same valve displacement despite the external load. Using multiple actuators and a single pump will cause different pressure drops between the pump and actuators. In combination with pressure compensation valves, this pressure drops creates compensation losses when actuated. These losses are known to be one of the largest hydraulic losses in such hydraulic architectures.

A multi chamber cylinder (MC) system can change the resulting load pressure for a given load through the combination of different areas. This system can be seen as a *hydraulic force gearbox*, where each chamber combination corresponds to a specific gear. However, both position- and velocity control of this type of system are difficult to design. Especially at low loads and low velocity.

If a MC is used in a load sensing architecture it opens up the opportunity to adjust the pressure drop between the pump- and the load pressure and thereby minimize the compensation losses. In cooperation with Volvo Construction Equipment (Volvo CE), a hydraulic system combining these architecture is designed. The idea is to control the velocity by a PV and the pressure drop adjusted by a MC. The problem with this system is to select the optimal gear for each given state during load cycles, like dig & dump or trenching. In this thesis the usage of machine learning to control such system is explored. More specifically, reinforcement learning (RL) will be used to develop an optimised gear selection for the *hydraulic force gearbox*.

1.2 Aim

The aim of this thesis is to develop an optimization-based controller for a *hydraulic force gearbox* using reinforcement learning. This will be developed in a simulation environment and implemented on a test rig.

1.3 Research Questions

1. How can reinforcement learning be used for gear selection to improve the energy efficiency while maintaining the performance of a hydraulic system with a hydraulic force gearbox?
2. How shall the training process of a reinforcement learning model be performed for a hydraulic force gearbox system?
3. What changes are needed for the control of the proportional valve to maintain the same system performance?

1.4 Delimitations

Delimitation of this thesis is presented in the list below:

- Focus will be on development of controller for finding optimal gear, not an optimal switch sequence between gears.
- The validation of the system will not be performed in a real application environment.
- No component selection is carried out, all of the components are selected in advance.
- Different reinforcement learning approaches will not be tested. One will be chosen and carried out. The alternatives to chose from are the ones included in the Reinforcement Learning Toolbox™ from *Mathworks*.

2 Method

To find out however or not the reinforcement learning (RL) is a suitable option for the *hydraulic force gearbox*, the work was divided into two major parts: validation of the simulation environment and implementation of the RL controller. The validation is an important part as a model representing the real world is crucial for the learning algorithm to actually learn how to behave in the final application. Along with the validation of the model, a literature study of RL was carried out. This provided theoretical knowledge for the selection of a suitable algorithm, setting up the training and testing procedures, and finally for the deployment on the real platform.

2.1 Validation of System

The used simulation tools are *Hopsan*, for the modelling of the physical system, and *MATLAB/Simulink* for control development. The physical models is first validated on a component level followed by system level.

The components for validation are the proportional valve (PV) and digital valves (DV). This was carried out by isolating the components as much as possible and measure representative quantities. The main component, the MC, was validation at a system level due to the need of the PV and DVs for control. A more detailed explanation of the procedure is found in Chapter 5.

2.2 Select, Train and Implement Reinforcement Learning

The literature study about RL gave a wide perspective of possible algorithms and techniques suitable for this thesis. For the development of the RL model a reward function was designed, including both how and when a reward is generated. The training of the model was carried out using the validated system model with representative loads. Once the RL model was trained to satisfactory performance in the simulation environment it was implemented in the test rig for validation.

The development of the RL controller was an iterative process. First a basic environment, reward and policy was used and a simple task. Once the RL model learned to handle the task, the complexity of the task and environment increased. This cycle was then repeated until the final RL model was trained. See figure 2.1.

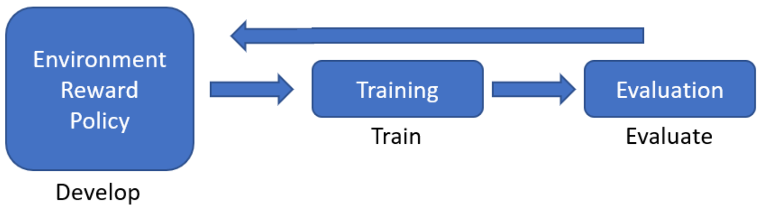


Figure 2.1: Reinforcement Learning controller development process.

2.3 Simulations

The simulation environment, controller and RL-training was built in *Simulink*. The physical system (excavator arm and hydraulics) was modeled in *Hopsan*, a simulation software developed at *Linköping University*. This model was then exported to *Simulink* where the RL controller was designed.

3 System Description

The system consists of an excavator boom and arm, hydraulics with a multi chamber cylinder (MC) and electronics.

3.1 Excavator Arm

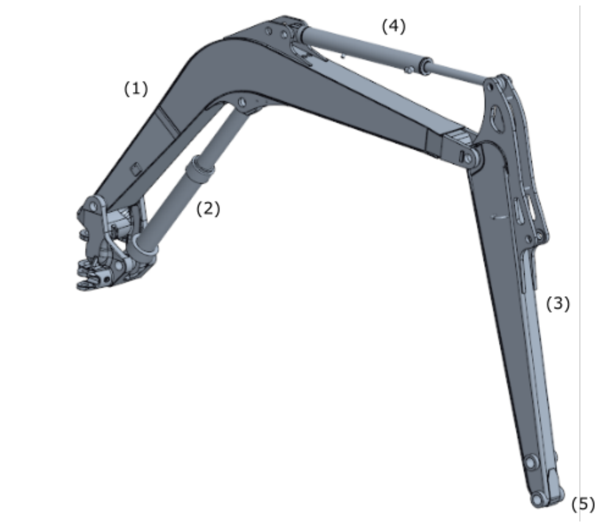
The test rig is an excavator arm from Volvo CE. It consists of a boom, arm, MC cylinder and a conventional single chamber cylinder (SC). A CAD-model of the excavator arm is seen in figure 3.1a, the physical rig and its surrounding structure is seen in figure 3.1b. Due to the area ratios of the MC, it can handle high loads for extension movements but is weak in retraction, described more in Section 3.2.1. Because of this, it is more suitable as the boom cylinder.

3.2 Hydraulics

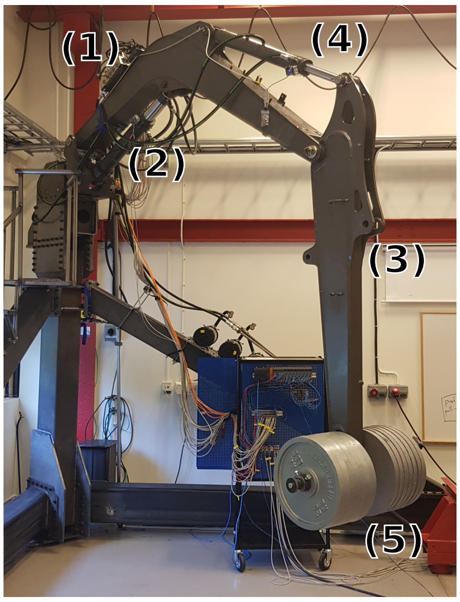
The MC system used in this thesis is described by Raduenz et.al in [1]. The physical system hydraulic schematic diagram is seen in figure 3.3. A pump is connected to two 4/3 load sensing proportional valves (PV) for controlling the SC and MC, which are connected to the arm and boom, respectively, of the excavator.

To control the MC, a block containing a set of 2/2 on-off digital valves (DV) is controlling the flow to each chamber, represented by the valves in the dashed box in figure 3.3. There are three inputs to this block: high and low pressure port from the PV and a port directly connected to tank. Each of these are connected to four digital valves, one for each of the MC chambers (MC$_{A-D}$). If the DV connected to PV high pressure and the DV valve to MC_A chamber is opened, high pressure is supplied to the MC_A, see figure 3.3 for a detailed view. The opening areas of the DVs connected to MC_A, MC_B and MC_C have twice the area of the ones connected to MC_D, due to different flow rate requirements. A more elaborate description of the DV block, MC and the connection between is found in [2]. The other components are discussed in Chapter 5.

The idea of the concept is to use the DV block to control the MC by activate or deactivate chambers (i.e. pressurize), effecting the resulting load pressure and thereby minimize the pressure drop. The flow rate and direction is controlled by the PV. In the case of a load sensing



(a) (1) Boom, (2) Boom cylinder (MC), (3) Arm, (4) Arm cylinder (SC), (5) Position for external load.



(b) The physical test rig.

Figure 3.1: The excavator arm.

system where the SC is setting the system pressure, the MC and DV block can adjust the load pressure to minimize the pressure drop and thereby minimize the losses. An example of a pq-diagram is shown in figure 3.2.

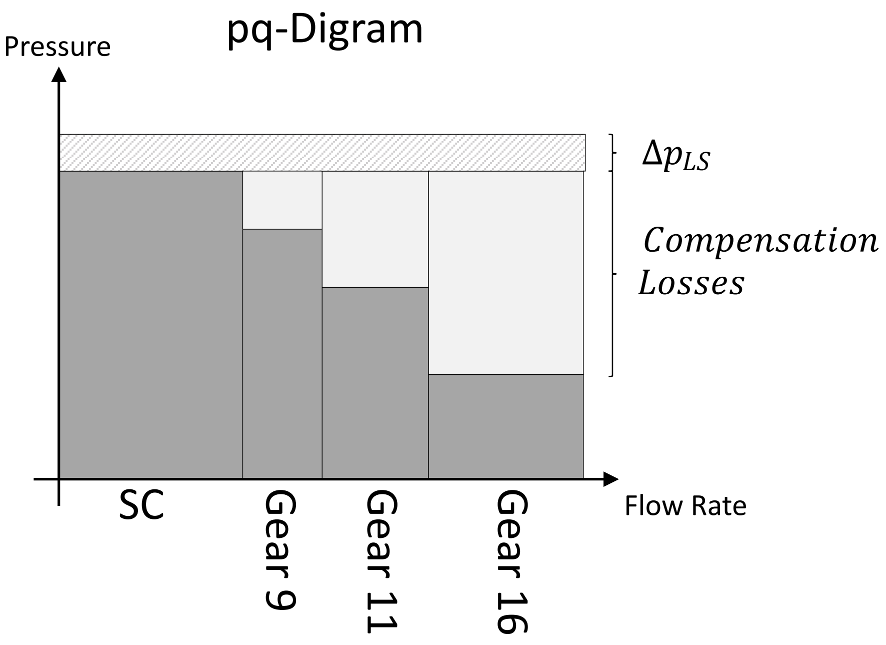


Figure 3.2: A pq-diagram showing losses for different gears while SC is controlling the system pressure.

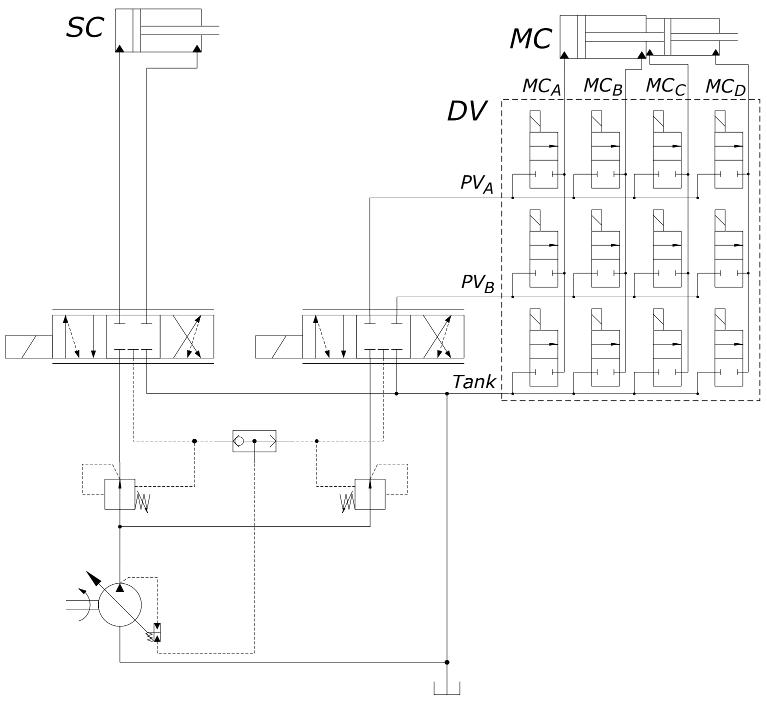


Figure 3.3: The hydraulic system which will be used in this thesis. Digital valve block is marked with the dashed box. Credit [1].

Note: This is a simplified view of the DV block used for a simplified simulation. In the physical rig there is a total of 27 valves, four for each connection to chamber MC_A, two for MC_B and MC_C each, and one for MC_D, i.e. $3 * (4 + 2 + 2 + 1) = 27$.

3.2.1 Multi Chamber Cylinder

A cross section view of the MC is seen in figure 3.4. Pressurizing chambers MC_A or MC_C results in an extension movement of the piston and the opposite are true for the MC_B and MC_D chamber. The area ratio difference between the chambers are significant, MC_A being the largest and MC_D the smallest. Since the MC is used for the boom this is not an issue due to the load acts mostly in the same direction as the gravity. See table 3.1 for the areas, area ratios and movement direction.

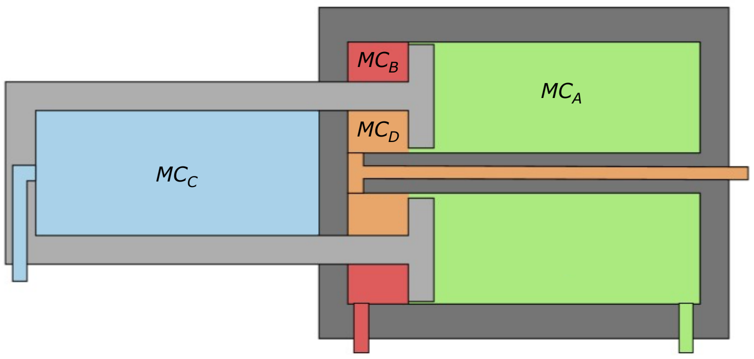


Figure 3.4: Cross section area of the MC. Green is MC_A, red is MC_B, blue is MC_C and orange is MC_D. Credit [2].

Table 3.1: Areas of the chambers in the Multi Chamber Cylinder.

Chamber	Area	Ratio	Movement
MC_A	0.0049	27	Extending
MC_B	0.0006	3	Retracting
MC_C	0.0017	9	Extending
MC_D	0.0002	1	Retracting

The gears are created by setting high pressure to different combination of the chambers. The effective area of each gear is calculated by

$$A_{effective} = A_{MC_A} - A_{MC_B} + A_{MC_C} - A_{MC_D} \tag{3.1}$$

where A is area and the signs are decided by the movement direction of the chambers relative an outward stroke.

Only considering one of pressure ports of the proportional valve, PV_A, there are a total of 16 discrete combinations. Since the retract movement is controlled by the PV, the combinations only containing MC_B and MC_D are removed. To succeed a retracting movement, the PV position is reversed (i.e. pressurize PV_B), resulting in neither MC_A or MC_C can be connected to this port. For those cases either chamber MC_A or MC_C is connected directly to tank. Connecting chambers to tank, instead of PV_B, also reduces the risk for cavitation within the MC because the flow is less limited from tank. The final, possible gears for this thesis are presented in table 3.2. Considering the area sizes, direction of the areas, system pressure and no losses, maximum loads can be calculated. The maximum load, for a full stroke of the cylinder, for each possible gear using a supply pressure of 100 bar is presented in figure 3.5. Further elaboration of possible gear is explained in [1].

Table 3.2: Different gears of the MC, sorted in ascending resulting force, system pressure at 100 bar. Starting at gear 4 by convention from previous project, where gears 1-3 generate negative forces.

Gear	PV_A	PV_B	Tank	Maximum Load [kN]
4	-	-	-	0
5	B C D	-	A	9.0
6	B C	D	A	11.1
7	C D	B	A	14.5
8	C	B D	A	16.6
9	A B D	-	C	41.5
10	A B	D	C	43.6
11	A D	B	C	47.0
12	A	B D	C	49.1
13	A B C D	-	-	58.1
14	A B C	D	-	60.2
15	A C D	B	-	63.6
16	A C	B D	-	65.7

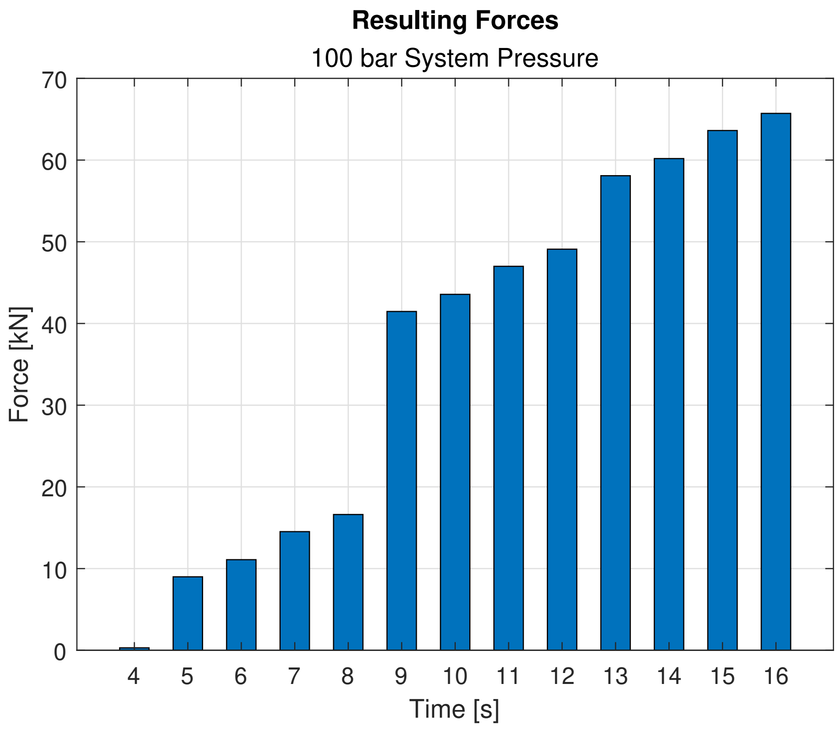


Figure 3.5: Possible forces at system pressure 100 bar.

3.3 Connection

The information flow for controlling the system is shown in figure 3.6. The Human Machine Interface (HMI), used for controlling the system, is located in a *MATLAB/Simulink* environment. To convert these commands to the hardware, the program *B&R's Automation Studio* is used. This program transfers the commands to electrical signals via *B&R's* PLC (x20-series). These signals controls the valves, which in turn controls the flow in the hydraulic system. The measurements from the sensors follows the same chain of communication but in opposite direction.

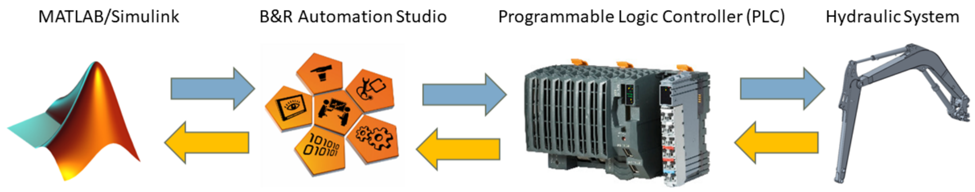


Figure 3.6: The control cycle of the physical system where *MATLAB's Simulink* is used as HMI.

Measurements of the rig is made by pressure sensors, linear transducers and a linear variable differential transformer (LVDT). The pressures are measured in all chambers for both cylinders as well as the system pressure. The position of both cylinders are measured by linear transducers and the spool position of the PV is measured by the LVDT. The sensors used in the rig are presented in table 3.3.

Table 3.3: Sensors used in the system

Sensor	Placement	Description
p_{MC_A}	DV block at chamber A	Pressure in MC chamber A
p_{MC_B}	DV block at chamber B	Pressure in MC chamber B
p_{MC_C}	DV block at chamber C	Pressure in MC chamber C
p_{MC_D}	DV block at chamber D	Pressure in MC chamber D
p_{SC_A}	Port A at PV for SC	Pressure in SC chamber A
p_{SC_B}	Port B at PV for SC	Pressure in SC chamber B
p_{sys}	Between pump and PV	System pressure
x_{MC}	Multi-Chamber Cylinder	Position of MCs stroke
x_{SC}	Single-Chamber Cylinder	Position of SCs stroke
$LVDT$	PV controlling the SC	Position of PVs spool

3.4 Control Flow

The gear selection controller (agent) is set as an inner loop that automatically finds the best gear for a given state. The operator is part of an open loop, requesting a velocity and adjust by hand according to the visual feedback. When the operator request a certain velocity the PV will open and supply flow for the DV block. The agent will read these signals along with pressures and choose an appropriate gear for the current load and velocity request. The control flow of the final system is illustrated in figure 3.7.

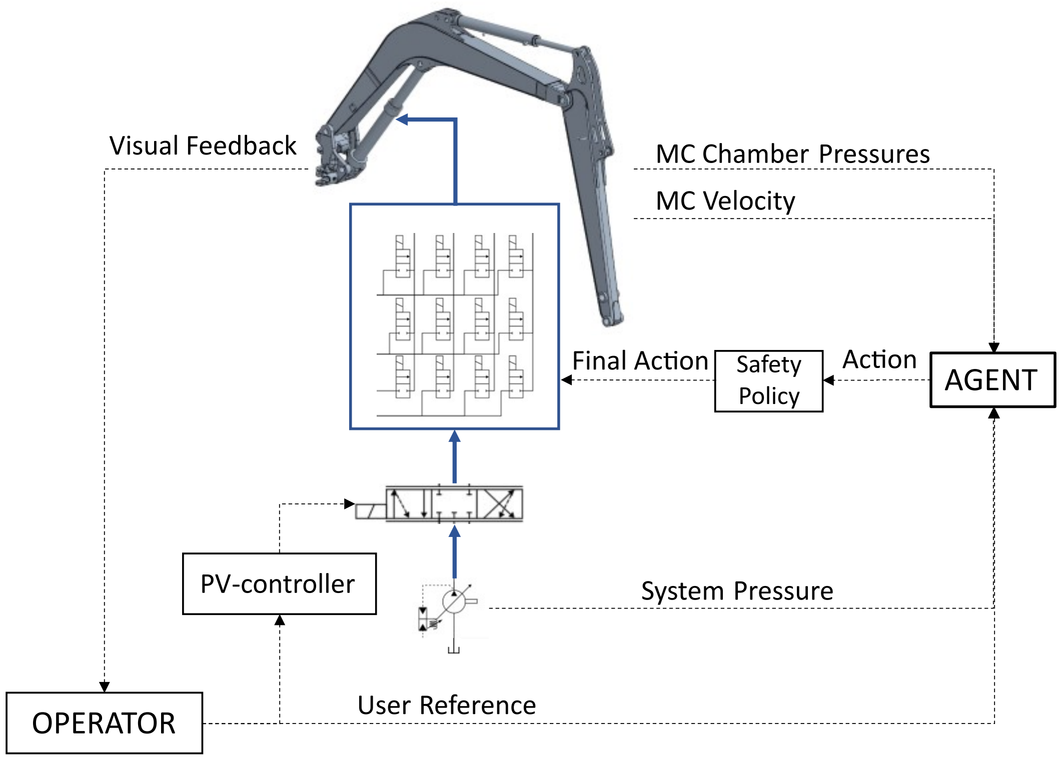


Figure 3.7: Control flow of the system.

3.5 Derivations of Calculated Signals

Not all the signals needed by the agent can be measured. Some needs to be calculated. Considering the signals from the sensors, table 3.3, and the system constants, the velocity, flow, pressure drop and power loss can be calculated.

To calculate the velocity, six time steps of the position is sampled and the derivative numerically calculated, presented in [3].

$$v = \frac{5x_t + 3x_{t-1} + x_{t-2} - x_{t-3} - 3x_{t-4} - 5x_{t-5}}{35T_s} \tag{3.2}$$

where x is the measured position, indexing the number of time steps ago the position was measured, and T_s is the sample time.

To calculate the flow through the PV, the calculated velocity, known chamber areas and active gear is used to approximate the value.

$$q_{PV} = A_{MC}v_{MC} = \begin{bmatrix} A_{MC_A} & -A_{MC_B} & A_{MC_C} & -A_{MC_D} \end{bmatrix} * gear^T * v_{MC} \tag{3.3}$$

Where q_{PV} is the flow, A is areas for the chambers, $gear$ is a [1x4] -vector of the combination of chambers according to table 3.2 and v_{MC} the velocity.

To calculate the pressure drop over the PV, the pressure directly after it is needed. Since this is not measured, it is approximated to be the same as the highest pressure of the active chambers, assuming no pressure losses between PV and DV. The system pressure is measured, and the pressure drop over the PV is calculated by equation (3.4).

$$\Delta p_{PV} = p_{sys} - max([p_A, p_B, p_C, p_D]. * gear)) \tag{3.4}$$

where " .* " indicates element-wise multiplication.

The power loss over the PV is calulated by equation (3.5), only the hydraulic losses are included.

$$P_{PV} = \Delta p_{PV} q_{PV} \tag{3.5}$$

This power loss considers both the pressure compensating valve and the PV itself.

4 Related Research

Related research regards previous work of the multi chamber cylinder (MC) and how this can reduce energy consumption, reinforcement learning (RL) theory and RL applied to hydraulic applications.

4.1 Pressure Compensation

Losses related to the hydraulics are a significant part of the total losses in an excavator. Of all the energy used by an excavator, 13% are hydraulic losses and 12% are hydraulic power supply losses [4]. Hydraulic losses are divided into compensation losses, control losses, actuator losses and, for this system, switching losses when changing active chambers, see figure 4.1.

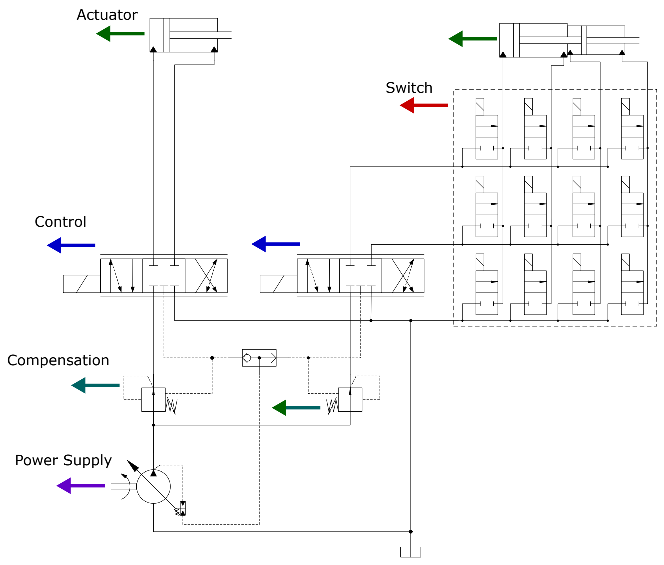


Figure 4.1: Location of hydraulic losses in a system using load sensing and MC.

The compensation losses occurs in systems where one pump is delivering flow to multiple actuators [5]. In a load sensing system, the highest load pressure decides the supply pressure for the entire system. For actuators with lower load pressure, this creates a pressure drop over the components between pump and actuator i.e. the PV in this case. The hydraulic power losses, P_{loss}, is calculated by

$$P_{loss} = \Delta p q \tag{4.1}$$

where Δp is the pressure drop over the PV and q is the flow though it. A third of the total hydraulic energy consumed by an excavator in a digging cycle are related to these losses [6], which gives reason for improvement.

4.2 Digital Hydraulics

Digital hydraulics have gained attention latest decade and its systems architecture differs from conventional hydraulic systems for construction equipment (e.g. excavators). The main benefits compared to traditional systems are the use of simple and reliable components, potential for improved performance due to the fast dynamics of on/off valves and the flexibility of the system. The control strategy defines the system characteristics instead of using complex components for specific tasks [5].

Using a MC, discrete force levels can be achieved from the combination of pressure sources and cylinder chambers [7]. In [2], three supply pressures are used with a MC creating 81 possible forces, linearly distributed. This was to be used for secondary force control of the cylinder. This architecture can be seen as a force gearbox, where each combination corresponds to one gear. This approach have been seen to reduce the energy consumption by 60% compared to the convectional load sensing system [8], which shows the potential of this type of cylinder.

One of the main drawbacks of using digital hydraulics is the increased complexity of the controller. Partly because of possible gears and lower controllability when switching gear, as oscillations occur when pressurized and non-pressurized chambers are connected. Also velocity control is hard to achieve with force control, especially for lower velocities [1]. Introducing a PV, as described in Chapter 3, velocity control can be improved while keeping the reduced energy consumption.

Finding the optimal gear to switch to from a certain gear while aiming to increase the overall efficiency, keep the controllability and follow the reference set by the operator is a complex control task. Combined with the complexity of digital controls the gear selection gets even more difficult. This raise enough reason to try out machine learning and deep learning to develop a controller for the MC.

4.3 Reinforcement Learning

There are different kinds of numerical optimal control. In [9] different options are analysed for real time control. A solution, not applicable as a real time controller, is the Dynamic Programming (DP) approach. DP will generate a global optimum within its discretization range and the given load cycle. This can be considered the possible optimum, used as reference while developing other controllers. For real time implementation, alternatives are Equivalent Consumption Minimization Strategy (ECMS) or a rule-based strategy. The ECMS only works at a given load cycle, due to the equivalence factor. The final output can differ a lot even at small deviations of this factor, making it load dependent. A rule-based strategy requires unreasonably amounts of rules to always do the optimal control in all possible situations.

To solve this issue, having a long term, optimal controller that is working for more than one cycle, Reinforcement Leaning (RL) is applicable. This approach almost reaches the same optimality as DP, since both are using the same concept for calculating the optimality [9]. The main difference is that RL isn't as computational heavy, and is therefore an alternative as a real-time controller.

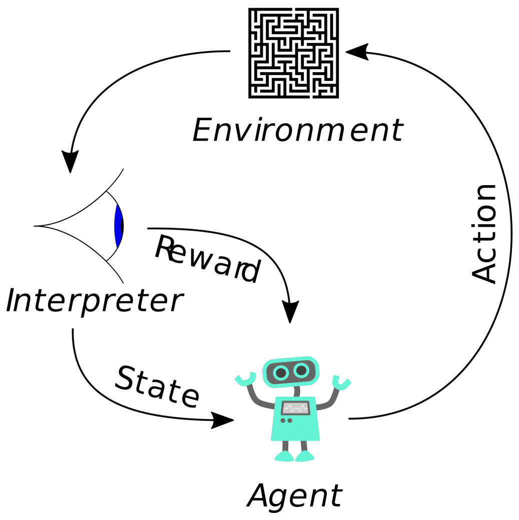


Figure 4.2: The interaction between the components in a RL system [11].

RL is a machine learning class, designed to learn from a "Trial and Error" - approach [10]. The RL model, called the *agent*, is not given what to do, or how to do it, but will figure this out by it self. This is done by trying out different *actions*, or sequence of actions, and afterwards receive a *reward*, or punishment, depending on the outcome. The agent is always striving to receive as high reward as possible, performing the actions it considers best to achieve this. After a number of actions the agent will have learned which actions are good and what to avoid. This is different compared to the other machine learning methods, i.e. *supervised* and *unsupervised learning*. Supervised learning is taught what the correct action is by the use of labeled training data, which is not the case in RL. Unsupervised learning is used for recognizing patters in a given data series, where as the RL is doing an action and learns from previous experiences.

The information flow for a RL system is seen in figure 4.2. It consists of two main parts, the agent and the *environment*. The agent is the decision maker, where the decision making part is called the *policy*. Everything outside is the environment, i.e. the surroundings what the agent is interacting with. The agent affects the environment through its *actions*, the output from the agent. The signals that the agent sees is the *observations*, which is used of the agent to interpret the environment, for example positions, pressures or previous action. The *states* are the values of the observations at a given time, which are used by the agents policy to determine the next action. The reward is the feedback the agent receives from the environment, giving it the information of how well it is performing the task. Based in this feedback, the agent can update its decision process to maximize the reward value.

There are two different components when constructing an agent: actor and critic. An actor is doing the action that is considered the best for the moment, focusing on the short term reward. A critic analyses the long term gains for the agent, i.e. what actions will receive most reward in the long run. An agent can be either an actor, a critic or an actor-critic. In the actor-critic case, the actor is defining what action to do, which is then analysed by the critic to update the agents parameters depending on the reward.

To explain how much the agent will investigate new actions the terms *exploitation* and *exploration* is used. Exploitation is using the best known solution to the problem, the agent exploits what it already found. Exploration is trying out new actions, with the goal of finding a better solution. The agent is exploring the action space. Finding a balance between these are important for both the learning time and the final performance.

One technique to learn the agent complex tasks is to divide the tasks into multiple smaller steps, this is called *Graded Learning* [12]. Graded learning is a simplified method of *Curricu-*

lum Learning which normally requires design of algorithms and implementation of complex framework. Graded learning can be implemented just by simplifying the task or environment and let the agent train for a set number of episodes or until convergence, then complexity is added to the task or environment. The trained weights and biases are then transferred to the new agent and the process repeats. Transferring weights and biases from a previously trained agent is called transfer learning [12].

4.3.1 Neural Networks

Due to the curse of dimensions a Neural Network (NN) is used, to make it possible to map observations to actions. A NN works as a universal function approximator, giving the probabilities of doing an action depending on the observed states. Due to the flexible structure and size of the NN, it is possible to map large and complex structures. In the RL application, this is used to calculate the next action, whether it is the next action directly, the probability for an action or expected reward for the actions [10].

The structure and computations within a neural network is simple, layers that are made up of neurons (nodes) that are connected, and information is sent between the layers. The information is multiplied by weights and a bias is added, the values of these weights and biases are what the learning algorithms tries to optimize. Each layer also have a so called *activation function* which helps to capture non-linearity's of the data [10]. One such activation function is the rectified linear unit (ReLU) that returns zero for negative values and the input value for all positive, i.e. $ReLU(x) = max(x, 0)$.

4.3.2 Agents

The action of the agent, in this work, is an integer value, representing a selected gear. Each value represents a unique set of which DVs to open. All the observations are continuous measurements from the system or reference signals to the system, made in real time. Because of this the agent needs to be designed to deliver actions in discrete space and observe in continuous space. The agents delivered by *Mathworks* with the action and observation space are presented in table 4.1. There are two alternatives, Deep Q-Network (DQN) and Proximal Policy Optimization (PPO) [13]. DQN is an agent consisting of a critic, while the PPO is an actor-critic. Because DQN is a simpler agent, this was selected.

Table 4.1: Mathworks agents and recommended using.

Agent	Action	Observation
Q-learning	Discrete	Discrete
Deep Q-Network	Discrete	Continuous
SARSA	Discrete	Discrete
Proximal Policy Optimization	Discrete	Continuous
Deep Deterministic Policy Gradient	Continuous	Continuous
Twin-Delayed Deep Deterministic Policy Gradient	Continuous	Continuous
Soft Actor-Critic	Continuous	Continuous

Deep Q-Network

A DQN agent consists of a critic value function, Q-function, trying to estimate future returns for given actions [14]. The return is the sum of all future discounted rewards. The discount factor, γ in equation (4.3), makes more distant rewards less valuable. During training, the agent gathers the current state s_t, the action taken a_t, the reward r_t it received and the state it came to s_{t+1}, creating a quadruple of saved values for each update (s_t, a_t, r_t, s_{t+1}) [15]. These values are saved for all the time steps in the agents *experience buffer* are and used for updating the policy. The policy is a NN, where the policy values are weight and biases.

To define the balance between exploration and exploitation the DQN-agent uses an ε-greedy function [16]. A greedy action is the action that maximizes the reward, the agents exploits the environment. The ε-value is a probability that the agent is forced it to do a non-greedy move and thereby exploring the environment. Otherwise, with a probability of $(1-\varepsilon)$, a greedy action is made. In the beginning of training, it is preferable to have a higher ε-value to explore more of all the options. As the learning progresses, the need to explore is reduced and it is preferable to have an agent that performs the best actions. The parameters ε_{decay} and ε_{min} are explaining at what rate the ε-value is decreasing and what the minimum value should be.

If a non-greedy action will be performed, a random action of the ones available is selected. If a greedy action will be performed, equation (4.2) is used for selection [16]. When the action is performed and the next state and reward is observed, the Q-function is updated, equation (4.3) [14]. See figure 4.3 for the agents interaction with the environment.

$$a_t = \arg\max_{a_t} Q(s_t, a_t|\theta_t) \tag{4.2}$$

$$Q_{t+1}(s_t, a_t) \leftarrow Q_t(s_t, a_t) + \alpha_t \left[r_{t+1} + \gamma \max_{a_{t+1}} Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t) \right] \tag{4.3}$$

Where Q is the value function, s the state, a the action taken, θ is the policy settings and r the reward. The t-subscript is the time of observation, γ is a discount factor (to make short term rewards more profitable) and α is the learning rate (adjusting how much the value function changes from the last states).

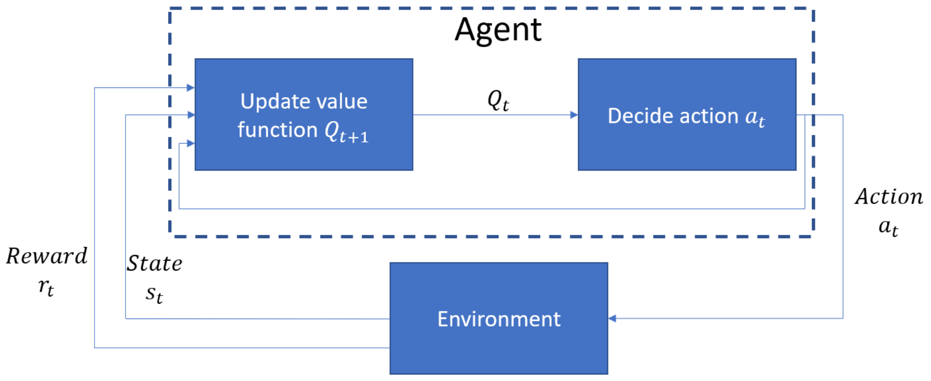


Figure 4.3: DQN information flow.

To decouple the simulation steps, the results (s_t, a_t, r_t, s_{t+1}) are saved in an experience buffer. Each time the Q-function is updated, equation (4.3), a mini-batch of random values are used for calculations from this buffer.

The hyperparameters, the parameters defining the agent, for a DQN are the presented in table 4.2.

Table 4.2: The agents hyperparameters.

Parameter	Description
ε	Exploration rate
ε_{decay}	Exploration decay rate per episode
ε_{min}	Minimum exploration rate
TargetSmoothFactor	Learning rate
TargetUpdateFreq.	Period between each update of critics parameters
ExperienceBufferLength	Experience buffer size
MiniBatchSize	Size of random experience mini-batch
NumStepsToLookAhead	Number of future rewards to calculate for the action
DiscoutFactor	Future rewards discount factor i.e. importance of future rewards
SampleTime	Agents sample time i.e. agent execution per sim. sample time

There is no universal strategy to set these parameters. Instead they have to be iterated for new problems. I this work, parameters are set from system limitations and using trial and error for satisfactory learning. When tuning the parameters during trail and error, the theoretical background of the parameters guided the direction of the tuning. The main focus of the development was set on the reward function to deliver an agent completing the task.

4.4 Reinforcement Learning used with Hydraulics

Applying RL to teach hydraulic construction systems to perform different tasks have previously been done with success. The use of RL over conventional control theory to develop controllers in construction equipment is mostly motivated by the trouble to realize the control rules and they are usually not based in an optimization. Standard PID-controllers can't handle the unknown terrain and more sophisticated controllers, like model predictive control (MPC), would be complex and require advanced and tedious tuning and modeling to succeed [17].

One specific field where RL is used for hydraulic control is bucket loading of wheel loaders [17]. Here a RL agent is trained for an autonomous control. Bucket loading is a demanding task including many parameters to consider: safe operation, wheel slip, bucket fill factor, fuel efficiency etc [18]. There is also requirements to handle unknown material (earth, gravel, rocks) and different particle sizes in the pile [17]. The combination of these difficulties makes use of a RL controller the most reasonable approach for a generalized autonomous controller. In [19] such a controller is implemented, managing to load 75% of maximum bucket load on average for the filling manouvers. This shows promising result for using reinforcement learning in construction equipment.

Excavators are machines with multiple tasks: dig & dump, trenching and grading, each requiring a different level of accuracy, force and movement [4]. This is another field where fully automation can be realised using RL. In [20] an agent learning a dig & dump cycle is presented. The agent successfully moved 67% of the buckets maximum load from a specified loading spot to a hopper. This agent is not optimal for a real system, due to the angle of attack of the bucket when it enters the ground damages the system. Another research [21] trained agents to do grading operations. The performance of the agent were acceptable, but more development is needed before it can be deployed to a real system, mainly due to oscillations.

RL is also used for minimizing energy consumption. For Hybrid Electric Vehicles it is a solution to optimize the battery usage to minimize the fuel consumption. In [22], RL successfully reduced the fuel consumption. In [9], same kind of energy optimization is used but for an excavator.

5 Model Validation

The components in the simulation models were validated with recorded measurements from tests performed on the test rig. The tests were performed in such a way to try to isolate the tested component as much as possible, without disassemble the system more than necessary. A load equation was created to give an indication of the load feedback for different positions of the cylinders.

5.1 Models

In this section the models, tests, tuning and validation are explained.

5.1.1 Proportional Valve

The position of the spool was measured to validate the model of the proportional valve (PV), a linear variable differential transformer (LVDT) was used to measure the spool position. The sensor had a span of +/- 4 mm, giving it 8 mm total stroke, the same length as the valves spool displacement in one direction. The pressure is measured at three different locations, the PV_A and PV_B ports as well as the system pressure (before the PV).

To keep the system pressure constant, and to reduce the influence of the pump dynamics, a pressure relief valve (PRV) was used to control the supply pressure to the PV and therefore a constant pressure source could be used in the validation model seen in figure 5.1. The two volumes represents the hoses of the real system and two 2/2 directional valves to stop the flow (always closed while validating).

The dynamics of the spool is different depending if a positive or negative stroke is made while the spool is centered or not (in position 0). At a positive stroke, the spool dynamics depends on the force produced by the solenoid and on a counter acting force from a spring, when changing direction while the spring is contracted the spring force will act in the same direction as the reference and solenoid, giving it a different behaviour. A third case happens when the spool is positioned off centre and the reference is set to zero, at this moment the spring force is the main contributor to the dynamic which can be seen as a linear motion in figure 5.6. Because of this, different spool dynamics needed to be modeled and validated. By using two second order transfer functions as the input to the proportional valve component in *Hopsan*. The dynamics of the valve itself is therefore set at high frequency to let the spool position only depend on the external transfer functions. A first order high pass filter is used to calculate and hold the sign of the input derivative. To handle the third case when there are no noticeable dynamics from the solenoids, a rate limiter, $rate_{PV}$, was used combined with logics to activate it only when the reference signals is close to 0 (to avoid numerical issues). For results see table 5.2.

To tune and validate the parameters of the proportional valve two different reference signal were used, step and sine wave.

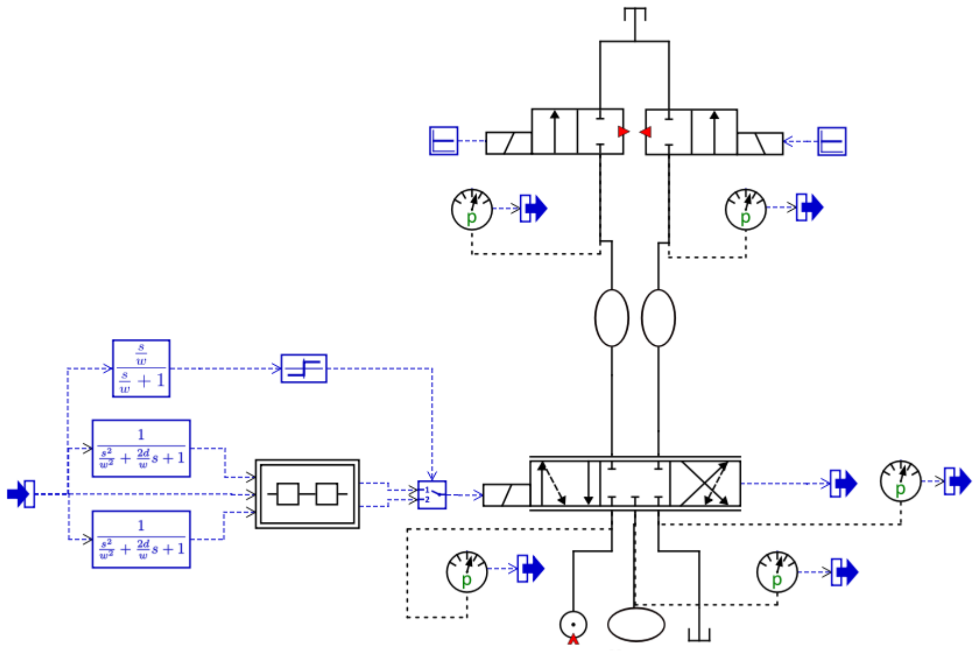

$\frac{\frac{s}{w}}{\frac{s}{w}+1}$
$\frac{1}{\frac{s^2}{w^2}+\frac{2d}{w}s+1}$
$\frac{1}{\frac{s^2}{w^2}+\frac{2d}{w}s+1}$
p
p
p
p
p

Figure 5.1: The *Hopsan* model of the system used for validation of the Proportional Valve, using two 2/2 on/off valve to restrict the flow at each port.

5.1.2 Pressure Dynamics of Digital Valve Block

To validate the response time of the DVs, the DV blocks output ports were plugged to minimize the volume after the valves and to isolate the valves as much as possible.

First the pressure in the selected chamber was set to tank pressure by opening and closing the chambers respective tank-valve, then the high pressure valve was opened while the PV was kept fully open. The depressurization was also tested by first pressurize the chamber, close the high pressure valve and then open the tank-valve.

To minimize the pump dynamics the supply pressure was limited by a PRV. This test was performed twice, once for single digital valve (connected to MC_D) and once for a double valve-chamber (MC_A, MC_B and MC_C have this setting).

The model used for validating the DVs consist of a fully open PV followed by 12 DVs, representing the DV block. The hoses between the PV_{MC_A} and PV_{MC_B} and the DV block are modeled as volumes, see figure 5.2. The approximated values of these and the volume of the chambers inside the DV block are seen in table 5.3. The four 2/2 valves connected after the DV blocks volumes are used to stop the flow and are always closed, representing the plugged ends in the test rig.

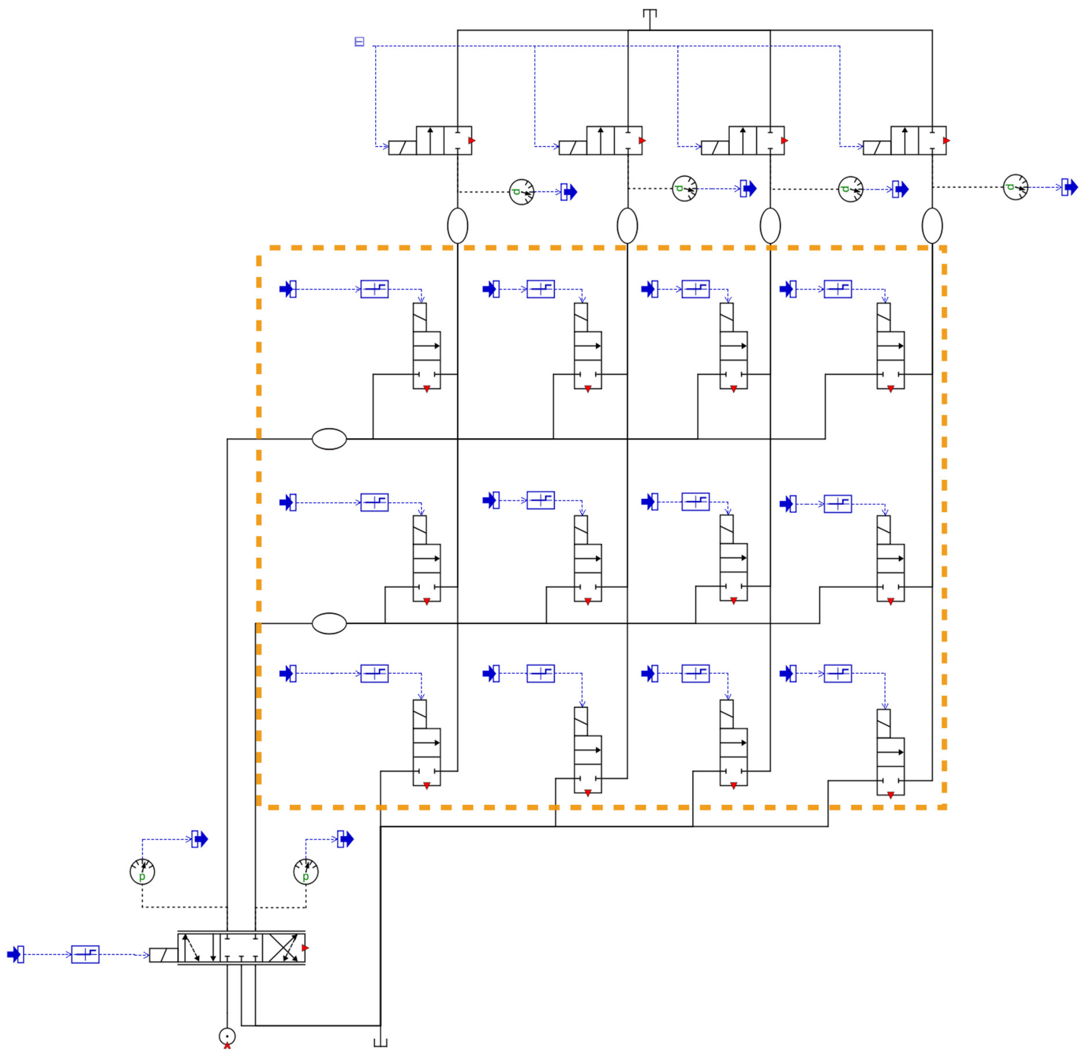


Figure 5.2: The *Hopsan* model of the system used for validation of the digital valves in the digital block. The digital valve block is marked within the orange dashed box.

During the tests the pressures inside the block was found to decrease rapidly when trying to perform the depressurization test. This is caused by a small leakage and explains why the pressure is not constant at system pressure, seen in figure 5.8 and 5.9. The reason behind this is due to the valves not being completely leak free and is noticeable as the pressurized volume is small, see table 5.3. Therefore a test was performed to confirm this, by keeping the DV blocks output ports blocked and pressurize the chambers by open and close the valves, the leakage is clearly shown as a steady depressurizing, see figure 5.3. The initial peak of each pressurizing happens when the valve opens, it is then kept open for a few second to let it stabilize (about 5 bar under system pressure). Then the valve is closed and the pressure is "trapped" within the block, this is when the pressure starts to rapidly decrease, the final drop to 0 bar marks the end of the test. The leakage was not modeled as the effect is negligible once the MC cylinder is connected and while the system is running, for future work where more detail is needed this could be introduced in the model.

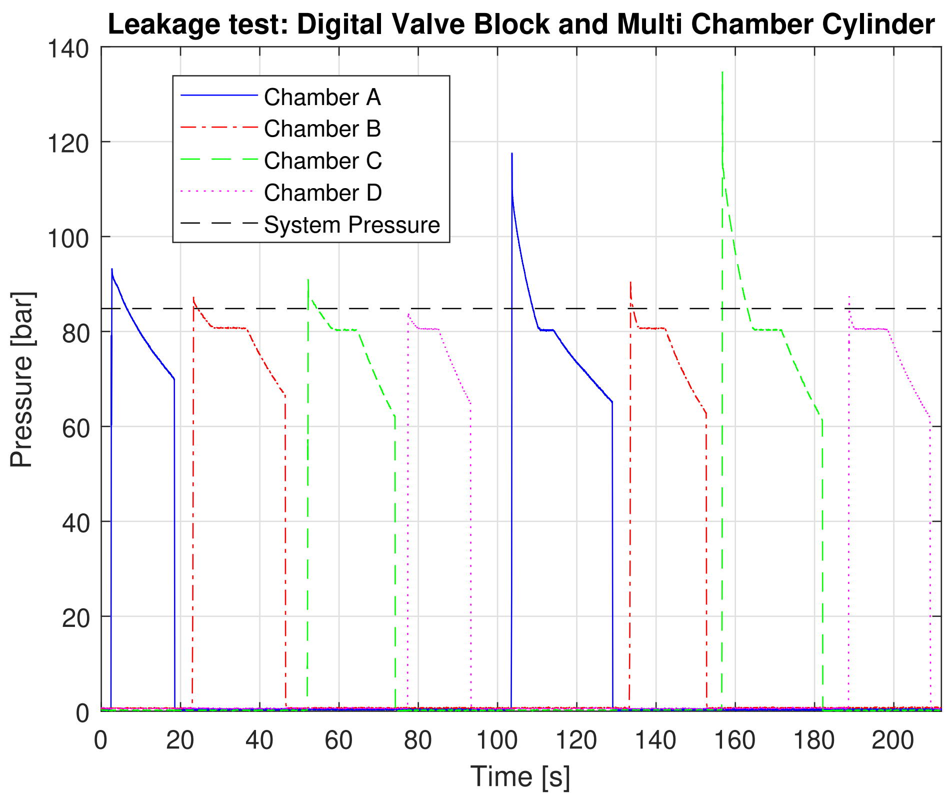


Figure 5.3: Small leakage from the digital valves is clearly shown due to a small pressurized volume. The four first test are the chambers connected to PV_A and the last four is PV_B.

5.1.3 Digital Valves and Multi Chamber Cylinder

The DVs and MC were validated and tested simultaneously since the MC cannot be used without the DV block. The tests were performed by recording data from steps with DVs (keep PV fully open) and step with PV (keeping DVs open) for two different gears. A full test run was starting at a low position and let the cylinder extend at full speed to about $\frac{2}{3}$ of the full stroke then hold for a few seconds to stabilise, from there a step was made back to start position. For the test with steps by the DV, the gear combination was reversed, see table 5.1. For PV-step the spool positions was as follow, in mm:

$$0 \rightarrow 8 \rightarrow 0 \rightarrow \text{-}8 \rightarrow 0$$

To minimize the effect of the pump dynamics, the pump pressure was set to 140 bar and the PRV to 100 bar, resulting in a constant supply pressure.

Table 5.1: Two different gears selected for validation.
Convention: port A /port B / Tank

Gear	Extension	Retraction
12	A/BD/C	BD/A/C
16	AC/BD/-	BD/AC/-

A sine wave reference with amplitude of 6 mm and a frequency of $\frac{\pi}{3}$ was also conducted for validation.

5.1.4 Single Chamber Cylinder

The single chamber cylinder (SC) was tested and validated in the same way as the MC, but only using the proportional valve.

5.1.5 Load Function

A load function was derived to give an approximation of the force acting on the cylinder and to account for some dynamics. Validation tests was performed by extending and retracting the MC and SC in a predefined pattern and then multiply the chamber pressures and areas, giving the resulting force acting on the cylinder, see equation 5.1.

$$F_{MC} = \begin{bmatrix} A_{MC_A} & -A_{MC_B} & A_{MC_C} & -A_{MC_D} \end{bmatrix} \begin{bmatrix} p_{MC_A} \\ p_{MC_B} \\ p_{MC_C} \\ p_{MC_D} \end{bmatrix} \tag{5.1}$$

All distances and angles in figure 5.4 are known except β, φ and Φ which were calculated by measuring the cylinders position. The derivative of these signals will be used for velocity and acceleration as well. Setting the equation (5.2) to zero and solve for F_{cyl} gives the force acting on the MC, see equation (5.3).

$$M_O = F_{cyl} r_\perp - M_{stat} - M_{dyn} \tag{5.2}$$

where

$$r_\perp = r_{OP} sin(\varphi_r)$$

$$M_{stat} = m_b g r_{G_bO} cos(\varphi + \psi) - m_a g(r_{JO} cos(\varphi) + r_{G_aJ} sin(\beta)) - m_l g(r_{JO} cos(\varphi) + r_{JL} sin(\Phi))$$

$$M_{dyn} = (I_b + c_{b1})\psi_b + c_{b2}\omega_b + (I_a + m_a r_{JO}^2)\psi_a c_{a1} - c_{a2}\omega_a^2 m_a r_{GaJ} r_{JO} cos(\varphi) +$$

$$+(I_l + m_l r_{JO}^2)\psi_a - w_a^2 m_l r_{JL} r_{JO} cos(\varphi)$$

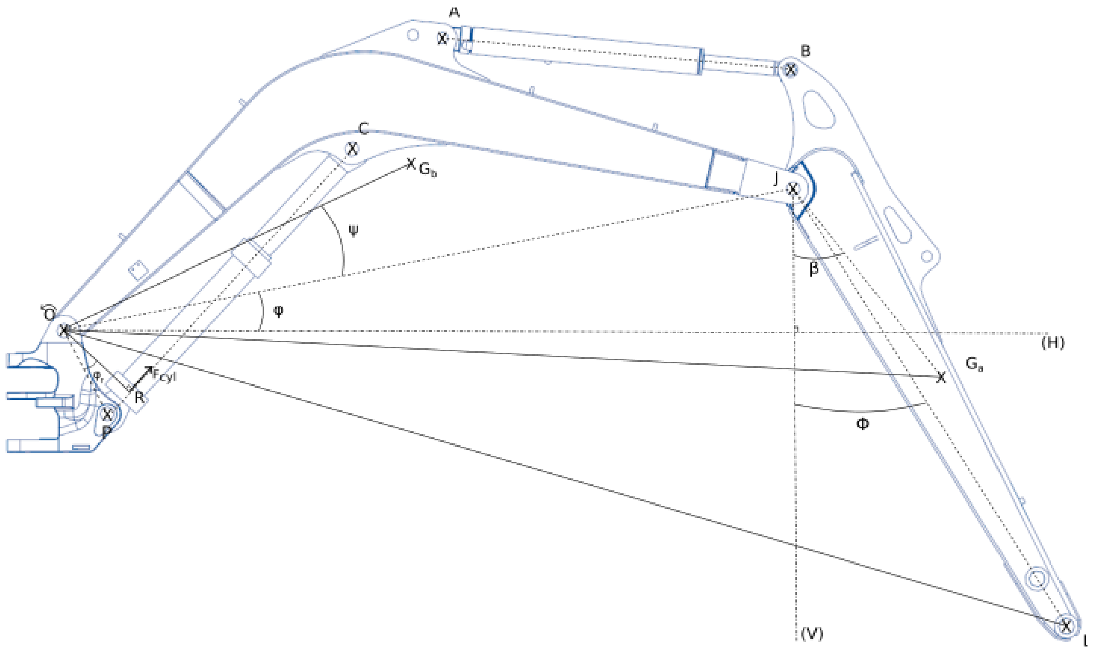


Figure 5.4: Geometry of the excavator boom and arm.

5.2 Validation Results

In this section the results for the validation of the model are presented.

5.2.1 Proportional Valve

Before tuning the parameters of the model, multiple steps of 8 mm were made with different system pressures to see the pump pressure dependency. As seen in figure 5.5, the difference was negligible.

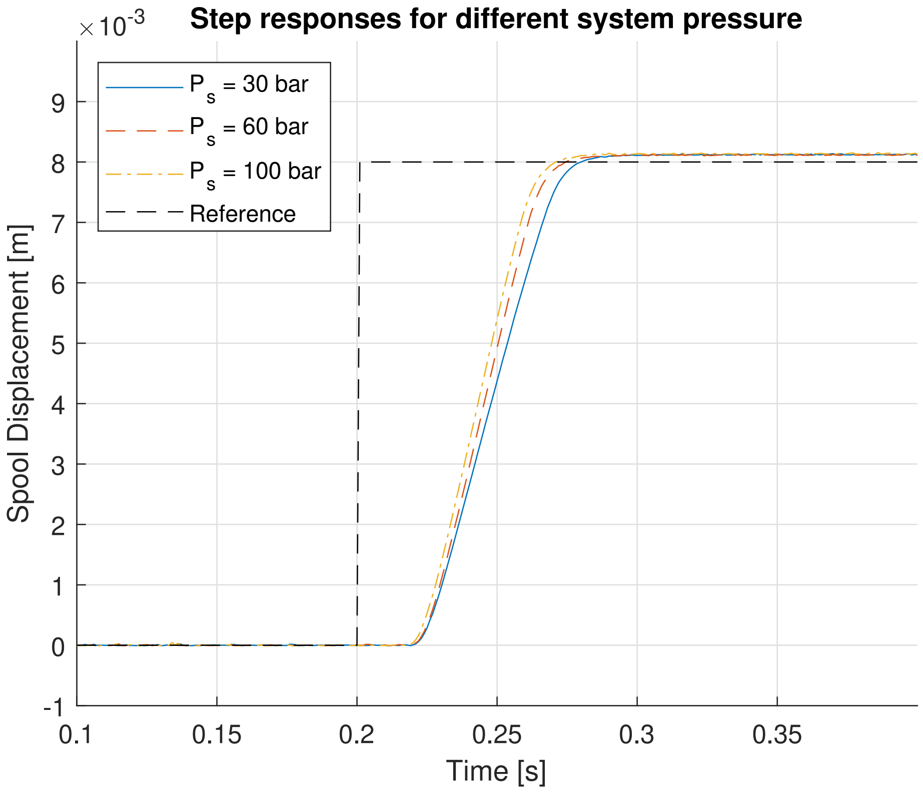


Figure 5.5: Step responses for different system pressures

The tuned PV parameters are the resonance frequency, ω_{PV}, the damping, δ_{PV}, and the delay τ_{PV}, the volume in the hoses between the PV and the closed valves where approximated, see table 5.2.

Quantity	Opening	Closing	Unit
ω_{PV}	120	150	rad/s
δ_{PV}	0.74	1	–
τ_{PV}	0.02	0.009	s
$rate_{PV}$	∞	0.04	-

Quantity	Value	Unit
V_{Hose}	$1.2e^{-4}$	m^3

Table 5.2: Tuned parameters for the proportional valve.

The tuning was an iterative process using test data from a 4 mm step, see figure 5.6 for results. Only half of the available spool displacement was used for validation since full steps are rarely executed in comparison to smaller steps. The supply pressure was set to 100 bar.

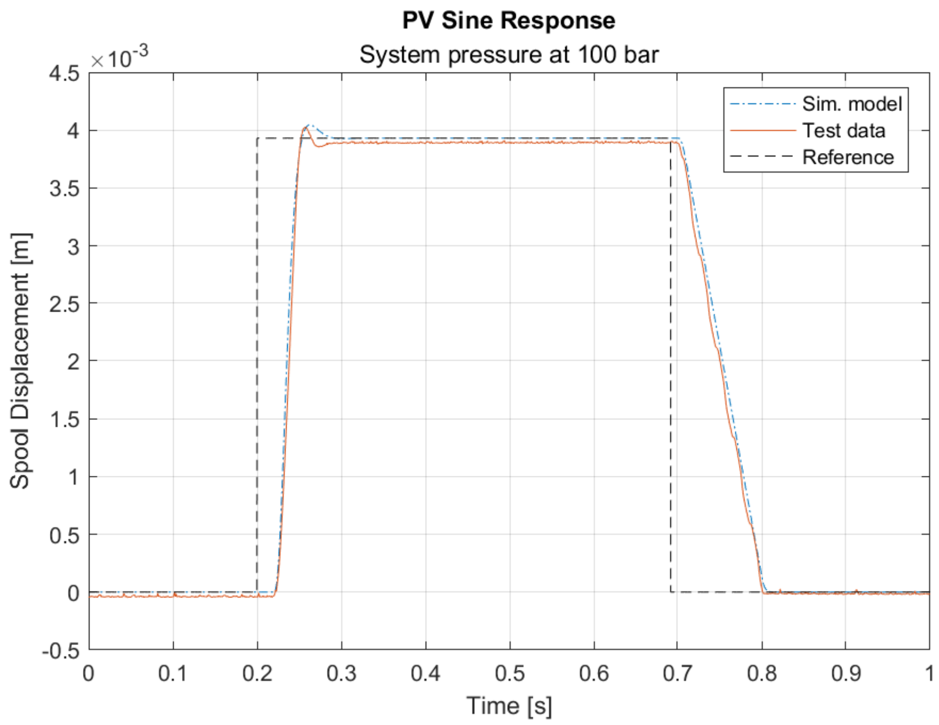


Figure 5.6: Step response for test data and tuned simulation model of proportional valve.

To validate continuous movement a sine wave was used as a reference. The results from this is shown in figure 5.7 and is the result from tuning the parameters only from the step data. The simulated models movement follows the measured signal satisfactory but note the discrete step the test data shows between 0-2 mm. This is a built in behaviour of the valve due to a 2 mm overlap, which was easily modeled by giving the valve component in *Hopsan* the same overlap and will therefore not result in any flow until the spool is positioned >2 mm off centre.

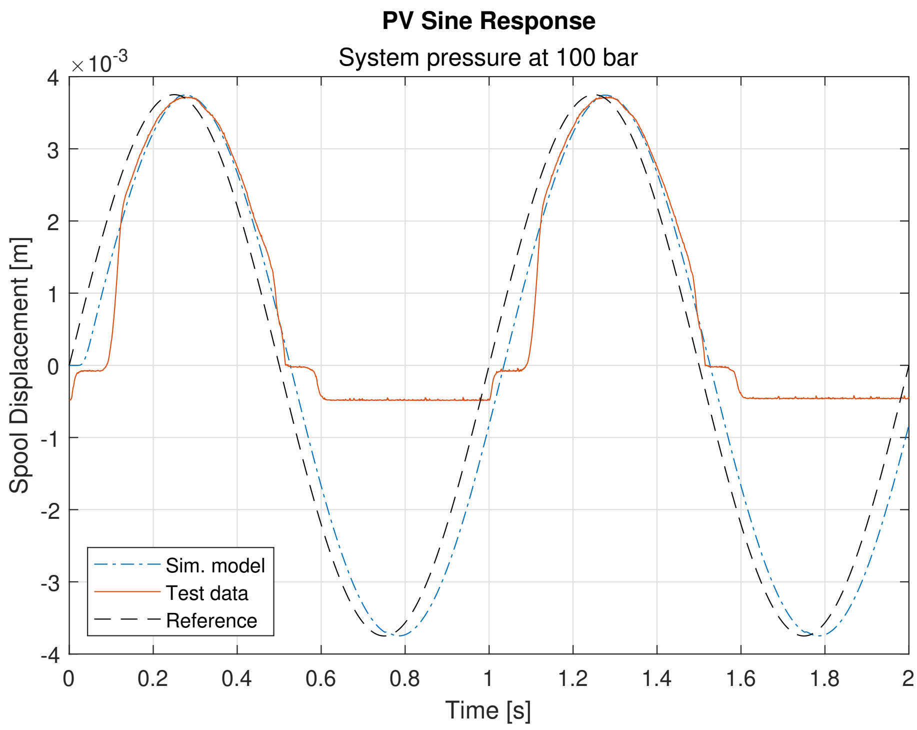


Figure 5.7: Sine wave response for test data and tuned simulation model, showing only the positive spool displacement due to limitations of measuring devices.

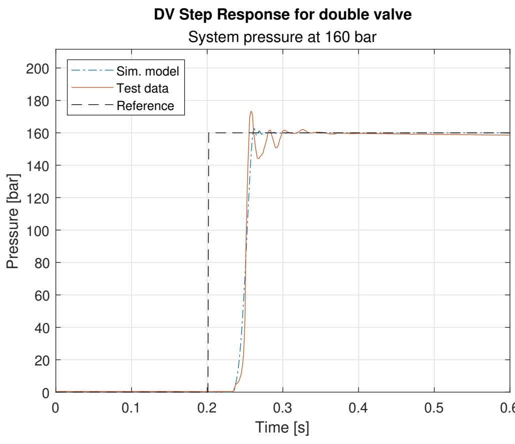
5.2.2 Pressure Dynamics of Digital Valve Block

This test validated the time delay, τ_{DV}, representing the time taken from the signal being sent until the pressure starts to change, see table 5.3. This parameter is set as a time delay before the control signal for each valve. The larger time delay of the double valve chambers is assumed to be due of them being connected in series, introducing more resistance in the circuit. The step responses from test data and model is seen in figure 5.8 and 5.9. The reference is a boolean value, zero for closed valve and open valve otherwise. The systems pressure drop is seen in the depressurization tests as the pressure is <160 bar before the step is taken, see figure 5.8a and 5.9b.

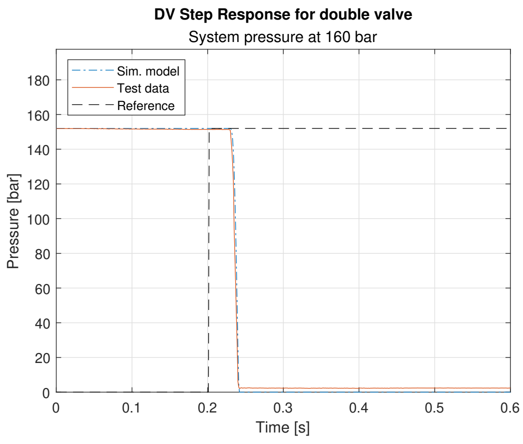
Quantity	Double	Single	Unit
ω_{DV_A}	125	125	rad/s
ω_{DV_T}	125	125	rad/s
δ_{DV_A}	0.8	0.8	–
δ_{DV_T}	0.8	0.8	–
τ_{DV}	0.027	0.011	s

Quantity	Value	Unit
V_A	$5.4e^{-3}$	m^3
V_B	$1.6e^{-3}$	m^3
V_{Block}	$3.5e^{-5}$	m^3

Table 5.3: Digital valve simulation parameters in left table, volumes are seen in the right table

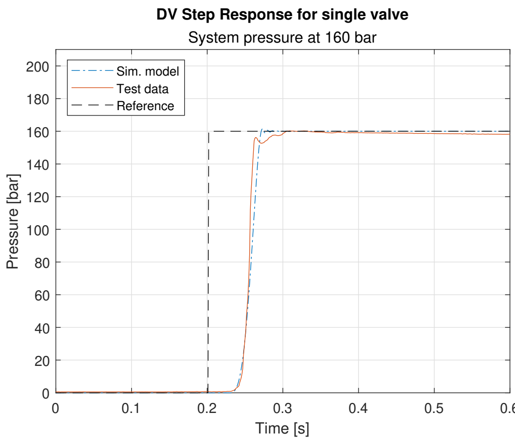


(a) Results from pressurization.

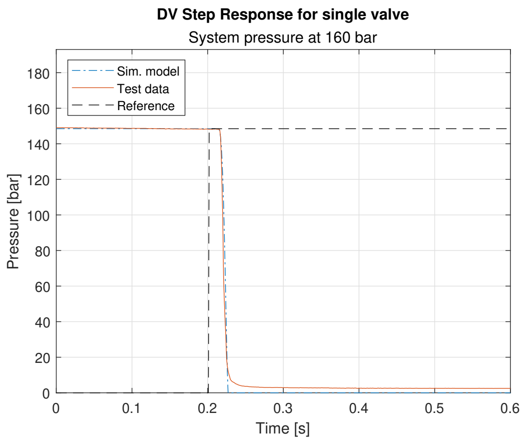


(b) Results from depressurization.

Figure 5.8: Validation results from testing and simulation of double connected digital valves.



(a) Results from pressurization.



(b) Results from depressurization.

Figure 5.9: Validation results from testing and simulation of a single digital valve.

5.2.3 Digital Valves and Multi Chamber Cylinder

The model was tuned by finding a good parameter fit for the pressure compensating valve (PCV), the MC itself and flow rate between ports of the PV. The model was mainly tuned from the PV step test, see figure 5.10 and 5.11. The velocity of the MCs extending and retracting movements deviates some which is why the model reaches few centimeters above the test data. The pressure levels between MC_A and MC_C in figure 5.11 differ from the test data but either MC_A is above test data and MC_C is under or vice versa. The sum of the forces ($F = p * A$) are approximately the same as the test data. This is most likely an effect of a not perfect load function. MC_B and MC_D reaches system pressure for the second retraction movement due to PVs change of direction.

Once tuned for the step test, the model was slightly adjusted after validated against the sine wave test. The position follows satisfactory even though the model is somewhat slower during the retraction movement. This is a trade off between step and sine wave test, see figure 5.12. The pressure levels for the models chambers are rising about 0.5 s earlier compared to the test data for the retraction movements, explained by lack of dynamics from the real system. Pressures in MC_B and MC_D does not fully reach the test data pressure levels during retraction, see 5.13. The final model was considered sufficient, see table 5.4 for the final parameters.

Table 5.4: Tuned parameters for MC, PV and PCV

Parameter	Value	Unit
PCV Open Pressure	5e5	Pa
PCV Flow	0.0025	m^3/s
PCV Spring	1e-6	$(m^3/s)/Pa$
MC Dead Volume A	2.5e-4	m^3
MC Dead Volume B	4e-4	m^3
MC Dead Volume C	2e-4	m^3
MC Dead Volume D	2e-4	m^3
MC Leakage AB	1e-11	$(m^3/s)/Pa$
MC Leakage CD	1e-11	$(m^3/s)/Pa$
MC Leakage AD	0	$(m^3/s)/Pa$
MC Viscous friction	1500	Ns/m
PV Spool diameter	0.01	m
PV Spool flow fraction PA	0.0834	-
PV Spool flow fraction PB	0.0934	-
PV Spool flow fraction BT	0.1284	-
PV Spool flow fraction AT	0.0299	-

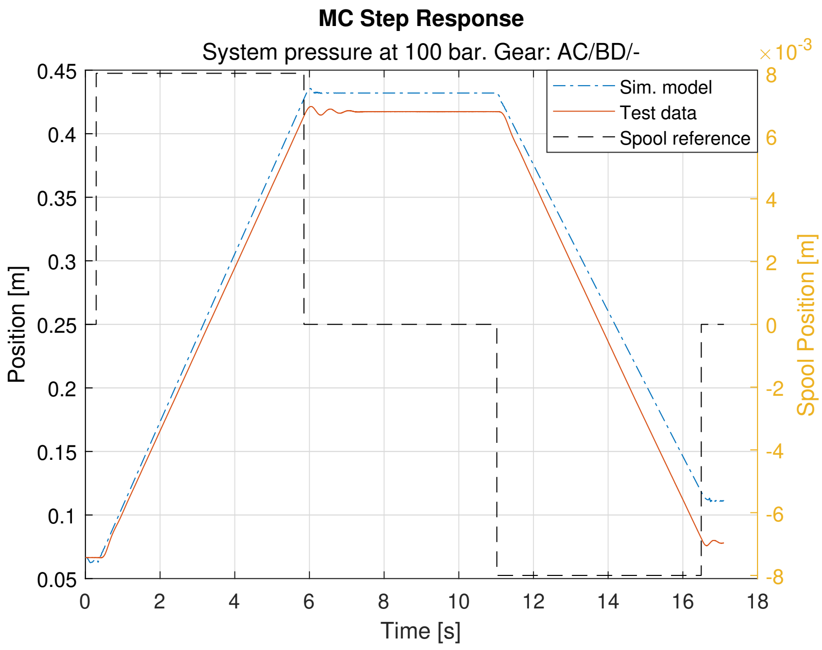


Figure 5.10: Position response for MC, step with PV.

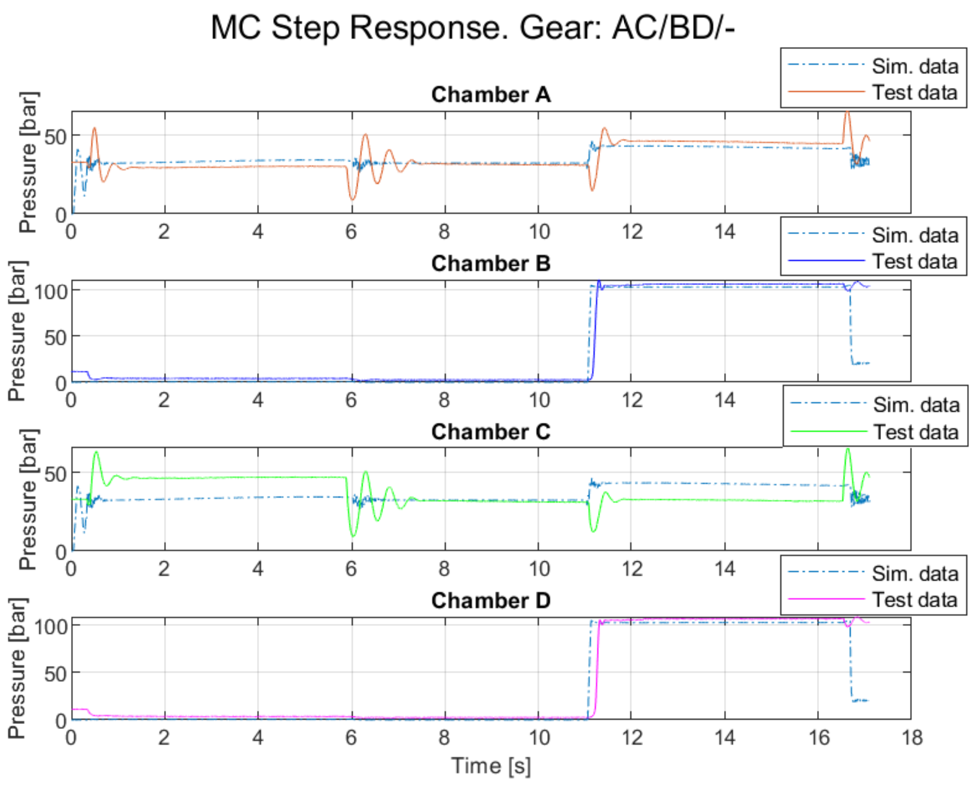


Figure 5.11: Pressure response for MC, step with PV.

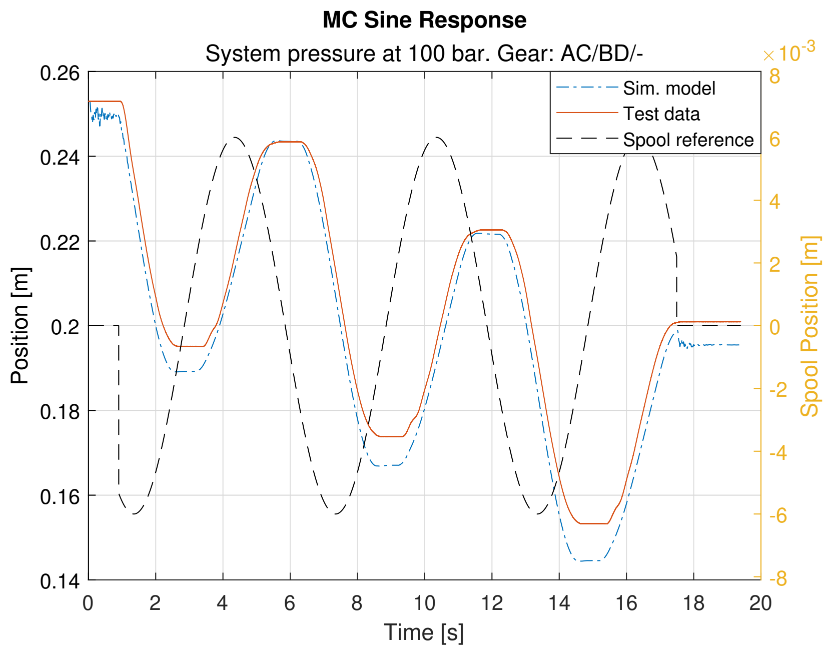


Figure 5.12: Position response for MC, sine wave reference.

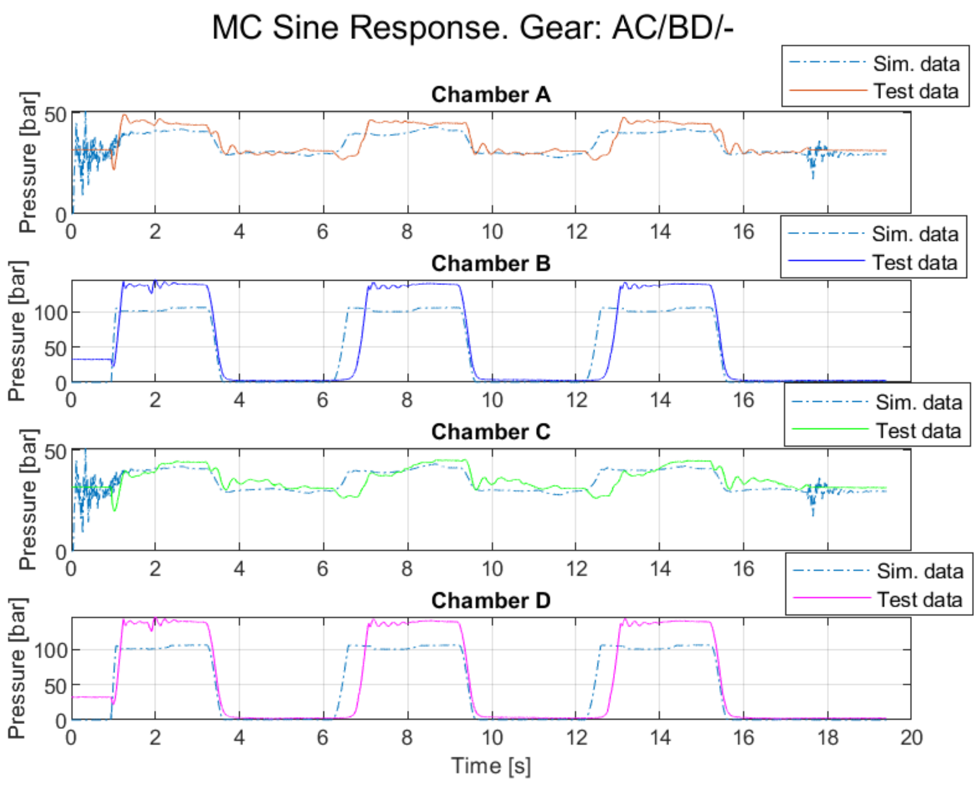


Figure 5.13: Pressure response for MC, sine wave reference.

5.2.4 Single Chamber Cylinder

The model was tuned and validated in the same manner as MC. See figure 5.14 and 5.15 for step responses. It's clearly seen that there are room for improvement regarding the SC model but some of its parameters (e.g. flow rate) are closely connected to the MC-model. However, the sine wave response turned out better, see figure 5.16 and 5.15. This is further discussed in Chapter 8. But since the main focus for this thesis is regarding the MC, the SC was kept at the same position for all further tests. Therefore was these results sufficient and no further tuning was conducted. See table 5.5 for final parameters.

Table 5.5: Tuned parameters for SC, PV and PCV

Parameter	Value	Unit
PCV Open Pressure	5e5	Pa
PCV Flow	0.0025	m^3/s
PCV Spring	1e-6	$(m^3/s)/Pa$
SC Area A	5.9e-3	m^2
SC Area B	3.5e-3	m^2
SC Dead Volume A	1e-3	m^3
SC Dead Volume B	1e-5	m^3
SC Leakage	0	$(m^3/s)/Pa$
SC Viscous friction	1500	Ns/m
PV Spool diameter	0.01	m
PV Spool flow fraction PA	0.0834	-
PV Spool flow fraction PB	0.0934	-
PV Spool flow fraction BT	0.1284	-
PV Spool flow fraction AT	0.0299	-

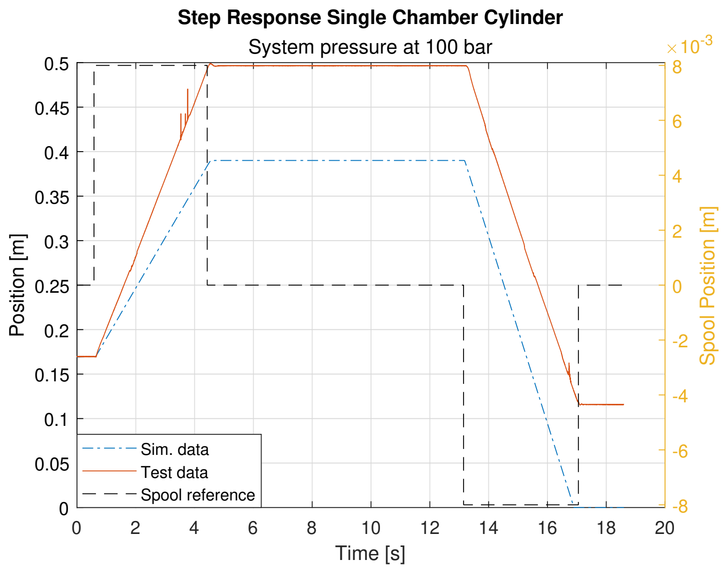


Figure 5.14: Position response for SC, step with PV.

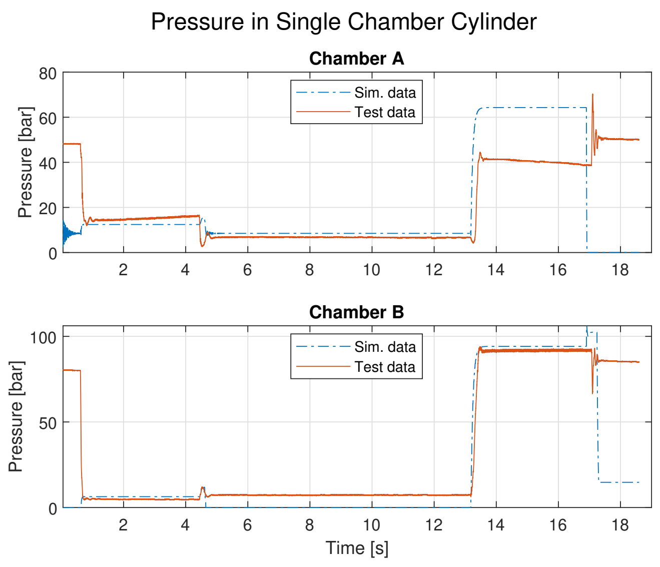


Figure 5.15: Pressure response for SC, step with PV.

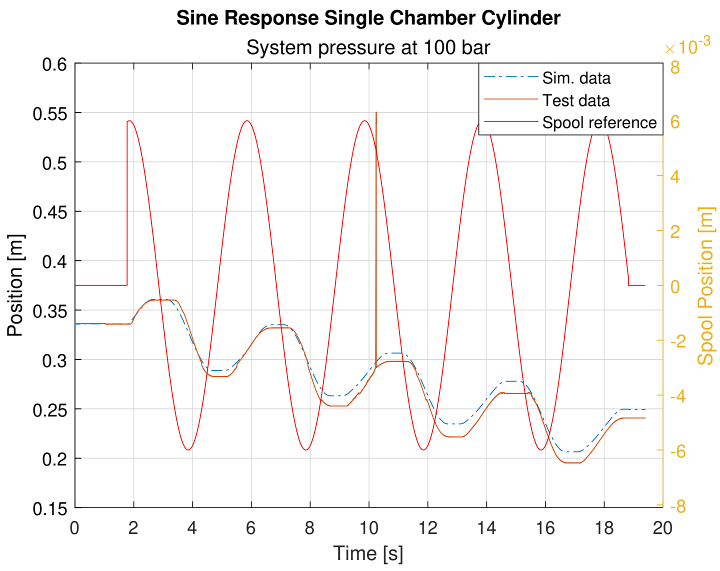


Figure 5.16: Position response for SC, sine wave reference. The spike at 10 sec. is due to faulty sensor.

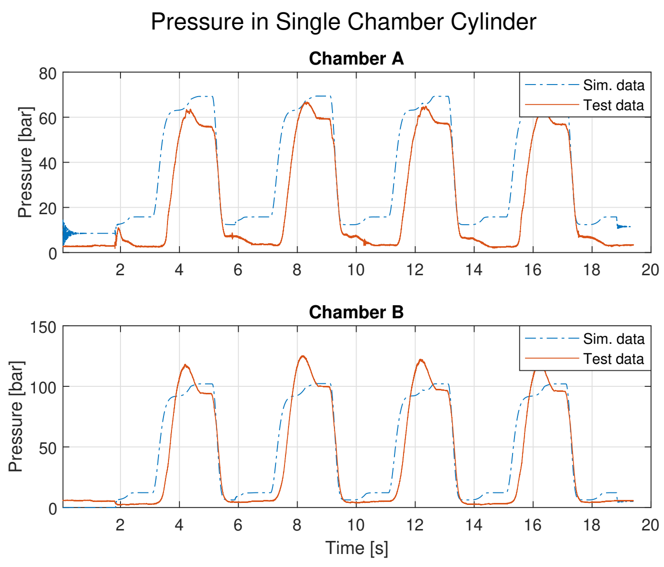


Figure 5.17: Pressure response for SC, sine wave reference.

5.2.5 Load Function

The resulting function when solving equation (5.2) for F_{cyl} is presented in equation (5.3) and is plotted as a function of position (with constant velocity) in figure 5.18. The coefficients, $c_{b1,2}$ & $c_{a1,2}$ were added to tune for friction and other external factors of the real system. The function gives a satisfactory approximation compared to the measured force as seen in figures 5.19 and 5.20.

$$F_{cyl}(x_{MC}, \dot{x}_{MC}, \ddot{x}_{MC}, x_{SC}, \dot{x}_{SC}, \ddot{x}_{SC}, m_l) = \frac{M_{stat}(x_{MC}, x_{SC}, m_l) + M_{dyn}(\dot{x}_{MC}, \ddot{x}_{MC}, \dot{x}_{SC}, \ddot{x}_{SC}, m_l)}{r_{\perp}(x_{MC})} \tag{5.3}$$

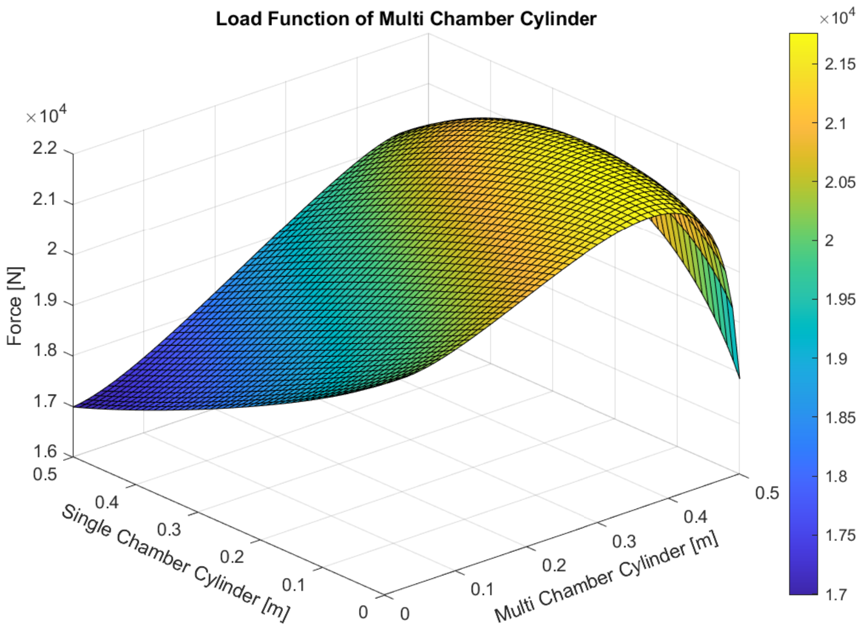


Figure 5.18: Force acting on multi chamber cylinder as a function of the two cylinders position, with 3kg external load, constant velocity at 0.03 m/s and no acceleration.

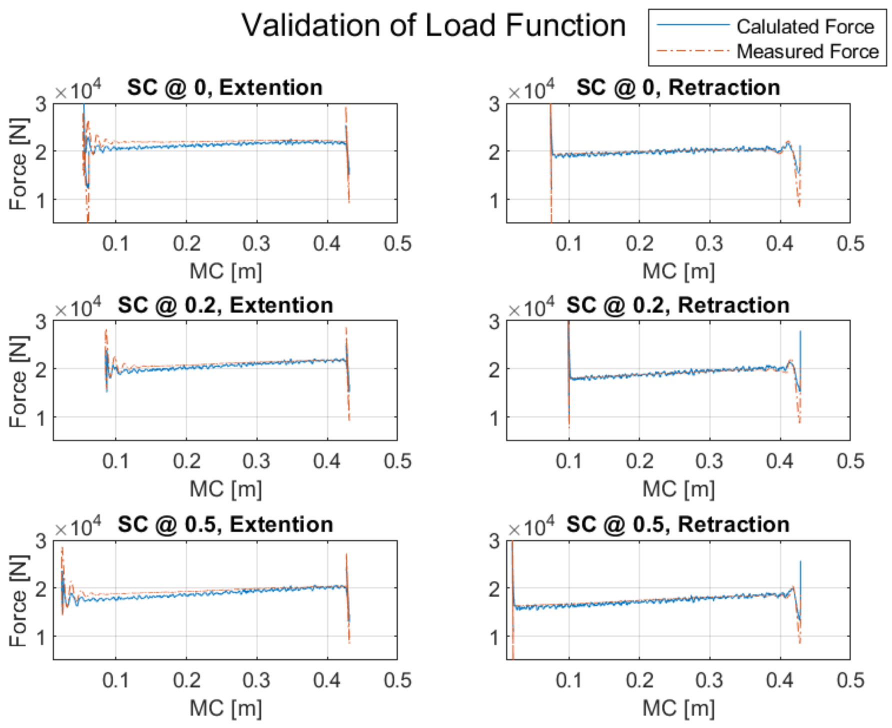


Figure 5.19: Test data and load function compared. The SC is set at a fixed position and MC extends or retracts.

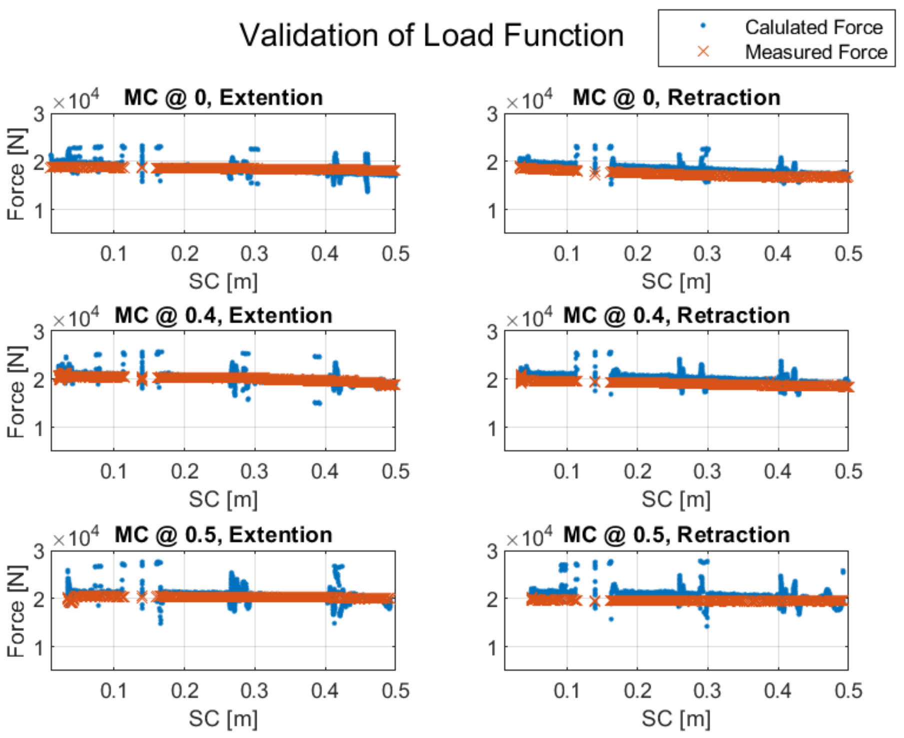


Figure 5.20: Test data and load function compared. The MC is set at a fixed position and SC extends or retracts. The gap around 0.15 m is due to faulty sensor.

6 Development of Reinforcement Learning Controller

The procedure of setting up training environment, validation and deployment are described in this chapter.

6.1 Position Control

To prove the concept an agent was trained for position control, deployed and tested. In this case the agent is the controller where the operator only sets reference positions. The agent interprets this and selects a gear. The information flow for this set up is shown in figure 6.1, where the observations works as the feedback signal in a conventional controller.

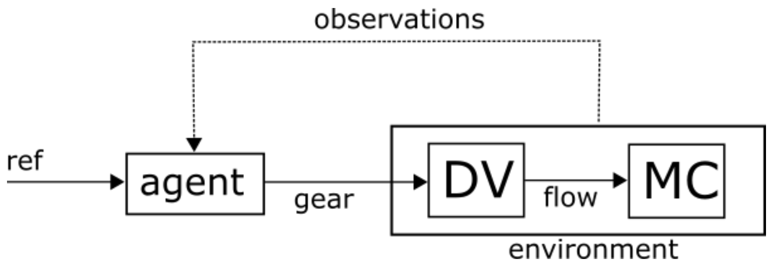


Figure 6.1: Information flow of the agent as a controller.

6.1.1 Training Setup

When trained for position control the agent needs a minimum of three actions (i.e. gears) that are shown in table 6.1. Therefore are only three gears chosen as it is enough to achieve position control and makes the learning process simpler. This gives the possibility to reach and and hold any position within the cylinders stroke range. The chosen gears are selected for being the strongest and the slowest. No chamber is connected to tank due to safety reason (if a gear switch takes place there will be a brief moment where the pressured side is connected to tank and results in a position drop and uncontrolled movement). These gears are also the inverse of each other which means there will not be any position drops when switching, since there are no cases where the pressure among the active chambers have to equalize. Also this is one of the gears the system model was validated for. The gears resulting direction is only valid while the PV is set to pressurize port A, therefore the PV is set at constantly fully open for each episode.

Table 6.1: Gear selection for position control.
Chamber convention: PV port A / PV port B / Tank

Action	Gear	Chambers	Direction
1	1	BD/AC/-	Retraction
2	4	-/-/-	No movement
3	16	AC/BD/-	Extension

6.1.2 Observations

The agents observations are shown in table 6.2. *Sensor*-types are measured values and *Calculated*-types are calculated using the measurements and input references to the system. The measured position was saved for 0.4 seconds. The error was calculated by $err = x_{ref} - x_{MC}$.

Table 6.2: The agents observations of the environment.

Observation	Description	Type
$x_{MC,t}$	Position of the MC	Sensor
$x_{MC,t-0.1}$	Position of the MC	Sensor
$x_{MC,t-0.2}$	Position of the MC	Sensor
$x_{MC,t-0.3}$	Position of the MC	Sensor
$x_{MC,t-0.4}$	Position of the MC	Sensor
err	Position error	Calculated
$\int err$	Integral of the error	Calculated

During training the integrated error was automatically reset as each episode restarts the simulation. Once the first part of training was over a reset function was implemented to reset the integrator when the error had been within a given tolerance for a short period, this is described further in Section 6.1.6. Noise was added as a uniform random distribution, $n \in [-3, 3] * 10^{-5}$, for the position observation to match the sensors measured noise while in steady state.

6.1.3 Reward Function

The reward function used for training was equation (6.1). The term R_{Error} is the difference between reference and current position and penalized if outside of a tolerance (0.01 m), otherwise rewarded. The $R_{Direction}$ term is penalizing the agent if it is not moving in the correct direction. Multiplying the velocity, v_{MC}, and the error gives a positive value if the velocity is in the right direction. Also, this punishment is only active while outside of the tolerance. The R_{Done} term worked as a bonus if the main goal was reached; to stay within the tolerance of 1 cm of the reference for 1 second. When the time condition was fulfilled, $t_{|err| \leqslant 0.01}$, the *isDone* flag switch to *true* and the simulation stops.

$$R_{Total} = R_{Error} + R_{Direction} + R_{Done} \tag{6.1}$$

where

$$R_{Error} = 10(|err| \leqslant 0.01) - e^{5|err|}(|err| > 0.01)$$

$$R_{Direction} = -5(v_{MC} * err < 0)(|err| > 0.01)$$

$$R_{Done} = 2000(isDone)(|err| \leqslant 0.01)(t_{|err| \leqslant 0.01} > 1)$$

6.1.4 Environment

Between each training episode the reference and start position of the MC cylinder was randomized within the MCs stroke length of 0.6 m and always kept at a minimum starting distance of 0.15 m, to ensure the episode wouldn't finish too early (due to the *isDone* condition). A special randomized case was introduced after 1000 episodes for 10% of the episodes: the MC start position was set to zero and the reference was > 0.4 m. The reason for this is described in Section 6.1.6

6.1.5 Hyperparameters

The agents hyperparameters are shown table 6.3. The opening time of a digital valve (DV) for the physical systems is 0.05 s and therefore was the agents sample time set to this. There are no reason to control the DVs faster for this case. *TargetUpdateFrequency* and *NumStepsToLookAhead* are dependent on the agents *sampleTime* and were set accordingly. The rest of the parameters were set either from default or by trial and error.

Table 6.3: The agents hyperparameters.

Parameter	Value
ε	0.99
ε_{decay}	0.004
ε_{min}	0.01
TargetSmoothFactor	0.001
TargetUpdateFreq.	7
ExperienceBufferLength	1e4
MiniBatchSize	128
NumStepsToLookAhead	25
DiscoutFactor	0.99
SampleTime	0.05

The agents network architecture, also known as the critic (neural network), is presented in table 6.4. The feature input layer corresponds to the observations seen in table 6.2. For each set of observation given the network will produce three values at the output layer, the largest of these values are chosen and they corresponds to the actions seen in table 6.1.

Table 6.4: The agents critic (neural network).

Layer	Details
Feature Input	7 features
Fully Connected	256 neurons
Activation Function	ReLU
Fully Connected	256 neurons
Activation Function	ReLU
Fully Connection	3 neurons (output)

6.1.6 Training

Each episode lasted maximum 10 s of simulation time. The training was conducted in two sessions, first iteration for 1000 episodes and the second for 400 episodes. See figure 6.2 for training progress. The blue line is the accumulated reward for each episode, red is the rewards moving average for 20 episodes and yellow is the agents estimated accumulated reward. The agent started to *"grasp"* the task around 15 episodes and *"perfected"* the task around $200 - 300$ episodes, which can be seen at the amount of reward received. There are still some outliers along the training, these are most likely the product of the exploration parameter ϵ, forcing the agent to act on random in certain situation and this can lead to poor performance which results in lower accumulated reward. The outliers during the retraining phase (1000+ episodes) are larger due to the extra added case where the start position is set at zero and a minimum stroke of 0.4 m. A longer stroke results in a longer episodes and if the exploration introduce random behavior it can result in poor performance. This however is a key stone in the learning process for the agent, to explore and learn.

When tested in simulation it encountered problems while running for longer than 80 seconds, due to the integrated error observation. It was not resetting at this stage, creating a large observation error which is an observation the agent never experienced during training. It also lacked a policy to leave the zero position. This was the reason for the retraining at 1000+ episodes. During the retraining the environment was set to expose the agent to the zero position more often and an integrator reset was implemented. The final agents results are presented in Chapter 7.

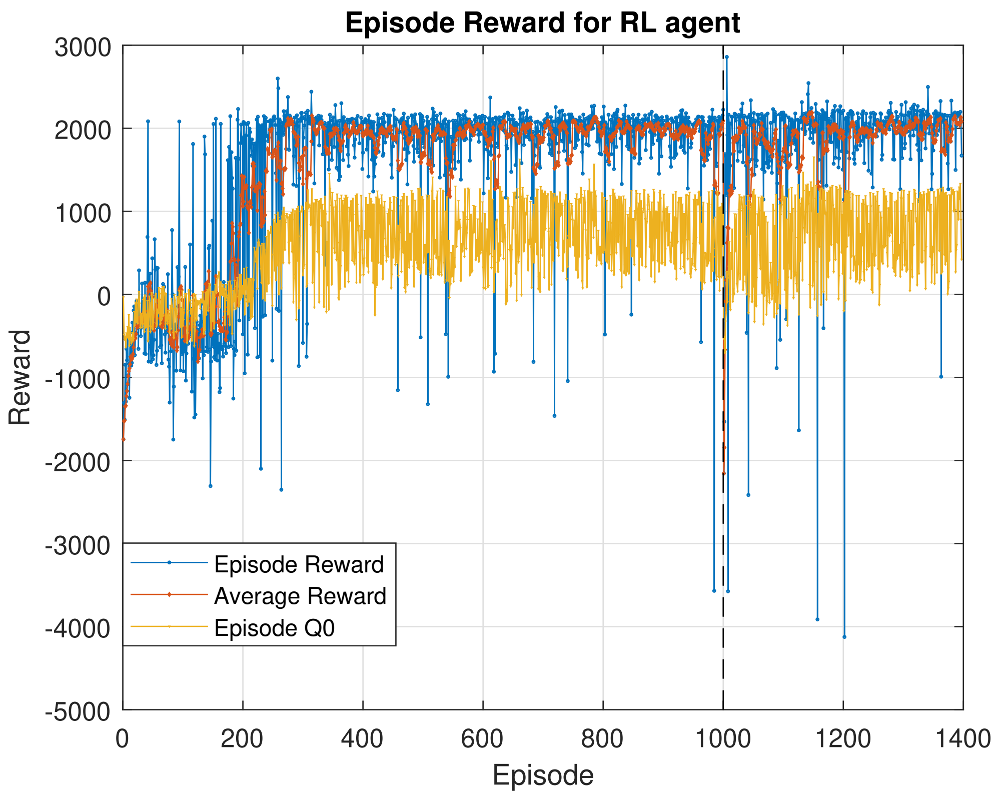


Figure 6.2: Reward the RL agent received during training for position control. The black dashed vertical line separates first and second training session.

6.1.7 Deployment

Once the agent was validated in simulation environment it was deployed in the physical system. The current process of deploying a MATLAB agent involves using third party files from Intel for compilations but this raised a security blockade by the university's IT-department. To speed up the process and since the trained network was small and simple, as seen in table 6.4. It was manually implemented as a MATLAB-function by extracting the weights and biases from the trained agent and do the matrix multiplication *"by hand"* (this solution is not as optimized regarding computational time though). See code below:

```
function y = policy(u, w1, b1, w2, b2, w3, b3, a)
% w1,2,3 & b1,2,3 are the weights and biases'
% matrices corresponding to respective layer

    lay1 = w1*u + b1; % Input layer
    lay1(lay1 < 0) = 0; % ReLU

    lay2 = w2*lay1 + b2; % Hidden layer
    lay2(lay2 < 0) = 0; % ReLU

    lay3 = w3*lay2 + b3; % Output layer

    [~,idx] = max(lay3); % Argmax Q
    y = a(idx); % Output action
end
```

6.2 Enabling Control

The next step was to implement a controller that selects the correct gear depending on the acting force on the MC. In this setup the position is controlled by the PV, P-controlled to represent an operator and the agent selects gears to enable the flow to the MC, see figure 6.3. Loads where selected to be optimal for a set of gears to be able to validate the results. Performing a real world load cycle is not possible with the rig used for this thesis, only lifting and lowering motions are allowed but not digging, for an example. This was the closest case to the final goal that could be performed in the current test rig which still includes gear switching while extending or retracting the MC.

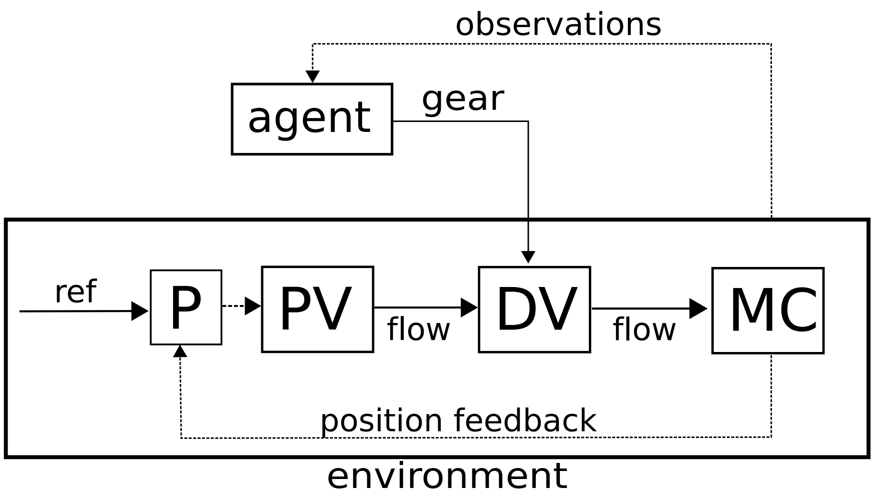


Figure 6.3: Information flow of the agent as an enabler.

6.2.1 Training Setup

Instead of using the agent to control the position, the agent acted as an enabler. The agent only opens valves in such a way that movement is enabled. The position and velocity are controlled by the PV. In normal excavators there is no feedback signal as the operator closes the loop, but for the simulation a P-controller was used to simulate an operator. To simplify the learning process, only a few amount of gears were selected and specific load cases for each gear used. The load cases was chosen to have an optimal gear, if a gear is too weak the cylinder would not be able to extend the cylinder fully but if too strong gear the power loss will increase. During training the external load was randomized from these load cases. The gears and loads used are presented in table 6.5.

Table 6.5: Gears and loads used for gear selection training

Gear	Chambers	Load [kg]
9	ABD / - / C	110
11	AD / B / C	170
14	ABC / D / -	230
16	AC / BC / -	260

6.2.2 Observations

The observations are presented in table 6.6. *Signal* types are control signals of the system, *Measured* are measured signals from sensors and *Calculated* are signals calculated from signals and measurements.

Table 6.6: Observations used for gear selection.

Observation	Description	Type
$gear$	The active gear	Signal
$x_{ref,PVMC}$	Reference signal to the PV	Signal
$p_{MC,A}$	Pressure in chamber A	Measured
$p_{MC,B}$	Pressure in chamber B	Measured
$p_{MC,C}$	Pressure in chamber C	Measured
$p_{MC,D}$	Pressure in chamber D	Measured
p_{sys}	System pressure	Measured
v_{MC}	Velocity of the MC	Calculated

All observations are normalized, by division of respectively maximum value or scaled to be close to the interval of $[-1, 1]$. The reason behind the velocity and the PV reference signal where to give the agent information of what the operator asks for and how it performs. A larger magnitude of reference signal means higher asked velocity. The pressures are meant to select an appropriate gear and the action mean to simplify the learning process by understanding what each gear is representing and how it change the other observations. All observations are part of the reward function.

6.2.3 Reward Function

For this case the reward function is designed to only give negative rewards, to encourage the agent to finish each episode as quick as possible, see equation (6.2). The idea for R_{Power}, equation (6.4), is to use as weak gear as possible since it is known to be more efficient. A weaker gear results in a smaller volume, resulting in less flow for the same velocity. Since the P-controller will be fully open for the most part of the episode this made it preferable to finish the simulation as quick as possible. A downside of this is when there is no flow, then there are no losses. This term was calculated using equation (3.3), (3.4) and scaled by the maximum flow and system pressures as well as weighted by a scalar.

Using a too weak gear the MC will not reach the reference and make it come to a halt. The term $R_{Velocity}$ was introduced to penalize if the MC was not keeping a set minimum velocity. In equation (6.3) the respective coefficient and constant $k_{ref,vel}$ and $m_{ref,vel}$ is transforming the PVs reference position to a minimum accepted velocity of the MC. The minimum accepted velocity is refereed to the slowest gear (16). The velocity, v_{MC}, is calculated by equation (3.2).

The combination of R_{Power} and $R_{Velocity}$ will force the agent to find gears where the losses are as low as possible and at the same time keep the minimum velocity.

A counter, $R_{SwitchCounter}$, was used to penalize unnecessary switching. A negative reward is received, depending on the number of time steps the new gear was different compared to the previous ones, see equation (6.5).

$$R_{Total} = R_{Velocity} + R_{Power} + R_{SwitchCounter} \tag{6.2}$$

where

$$R_{Velocity} = \begin{cases} -8, & if\, v_{MC} < (k_{ref,vel}x_{ref,PVMC} + m_{ref,vel} * sign(v_{MC}))\ \&v_{MC} > 0 \\ -8, & if\, v_{MC} > (k_{ref,vel}x_{ref,PVMC} + m_{ref,vel} * sign(v_{MC}))\ \&v_{MC} < 0 \end{cases} \tag{6.3}$$

$$R_{Power} = -100\frac{|q_{PV}\Delta p_{PV}|}{q_{Pump,max}p_{Sys,max}} \tag{6.4}$$

$$R_{SwitchCounter} = \sum_{t=1}^{T} -1(Action_t \neq Action_{t-1}) \tag{6.5}$$

6.2.4 Environment

The same model was used for this case as in the position control case. Every episode starts with a random selection of an external load which corresponds to one of the four gears. A 20% bias towards choosing the lowest load was implemented to let the agent experience more of those cases. The reason is that as the training runs the agents experience buffer gets filled with states and actions which is used for updating the network policy. Lower loads and weaker gears results in higher velocity, for a successful episode, and will therefore come to an end faster. This results in an experience buffer biased towards the longer simulations, i.e. high loads that require stronger gears. For the first part of the training the start and reference position was always constant, going from zero to full stroke. The complexity of the environment increased as the training progressed, this is described in more detail in Section 6.2.6.

6.2.5 Hyperparameters

The agents hyperparameters and network are shown table 6.7 and 6.8. When switching gears the system oscillated more or less depending on the number of chamber changing pressure. To handle that the agents sample time is set to 0.4s to give time for the oscillations to settle. This also decreased the computational time significantly. The *NumStepsToLookAhead* and *TargetUpdateFreq* parameters are affected by this and are set accordingly. The rest of the parameters are set either from default or by trial and error.

Table 6.7: The agents hyperparameters.

Parameter	Value
ε	0.99
ε_{decay}	0.003
ε_{min}	0.01
TargetSmoothFactor	0.002
TargetUpdateFreq.	1
ExperienceBufferLength	1e4
MiniBatchSize	128
NumStepsToLookAhead	10
DiscoutFactor	0.99
SampleTime	0.4

Table 6.8: The agents critic (neural network).

Layer	Details
Feature Input	8 features
Fully Connected	256 neurons
Activation Function	ReLU
Fully Connected	128 neurons
Activation Function	ReLU
Fully Connection	4 neurons (output)

6.2.6 Training

The training was performed with the *Graded Learning* technique over three iterations, each adding more complexity.

First Iteration

During the first iteration only three gears and loads were used, the position was included in the observation space. The PV was set to constant fully open and each episode will end by either reaching maximum time or when the position reference was reached. If the reference was reached the *isDone* criteria was fulfilled and the agent was rewarded with a positive one time bonus, see figure 6.4.

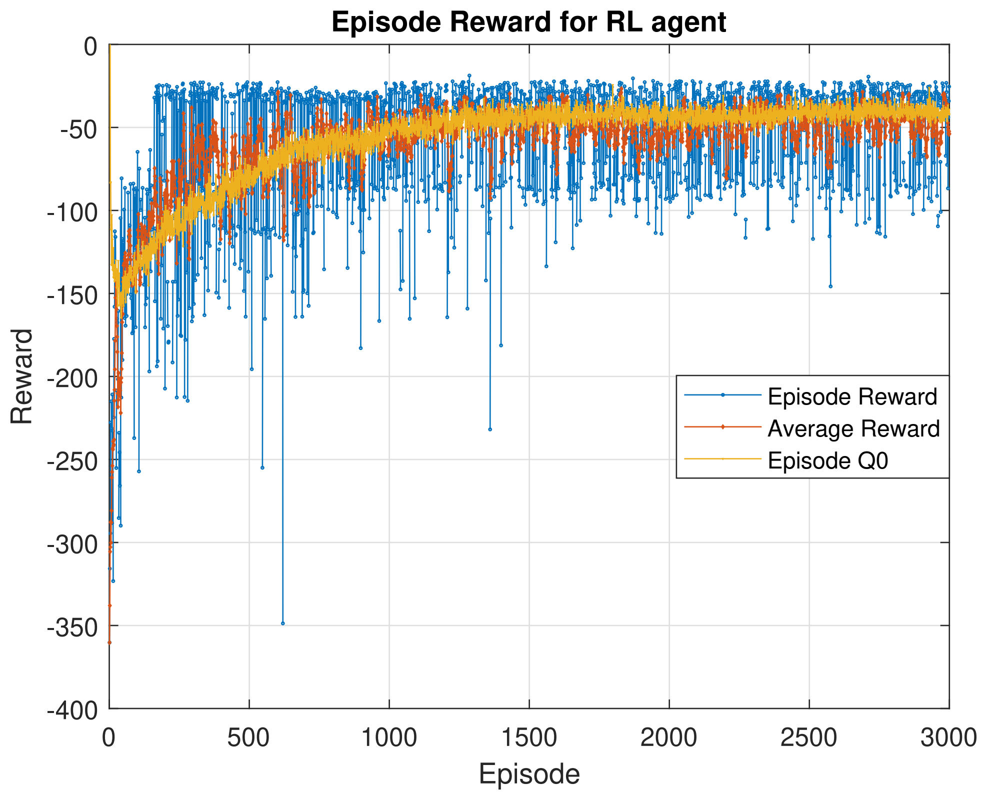


Figure 6.4: Training progress from first iteration.

Second Iteration

The position was removed from the observation space, a fourth gear was added (14) as well as an associated load. This changed the number of inputs and outputs, therefor didn't the network from first iteration fit for second iteration. But since the majority of the network was identical, weights and biases was extracted and implemented as initial values. For the output layer the weights and biases for the new action was randomly initiated within the interval $[-1, 1]$. This can be seen to as a type of *Transfer Learning* or a good initial guess.

Each load was modified by randomly set it to $\pm 5\%$ of the original value seen in table 6.5 to add generalization. See figure 6.5 for training progress.

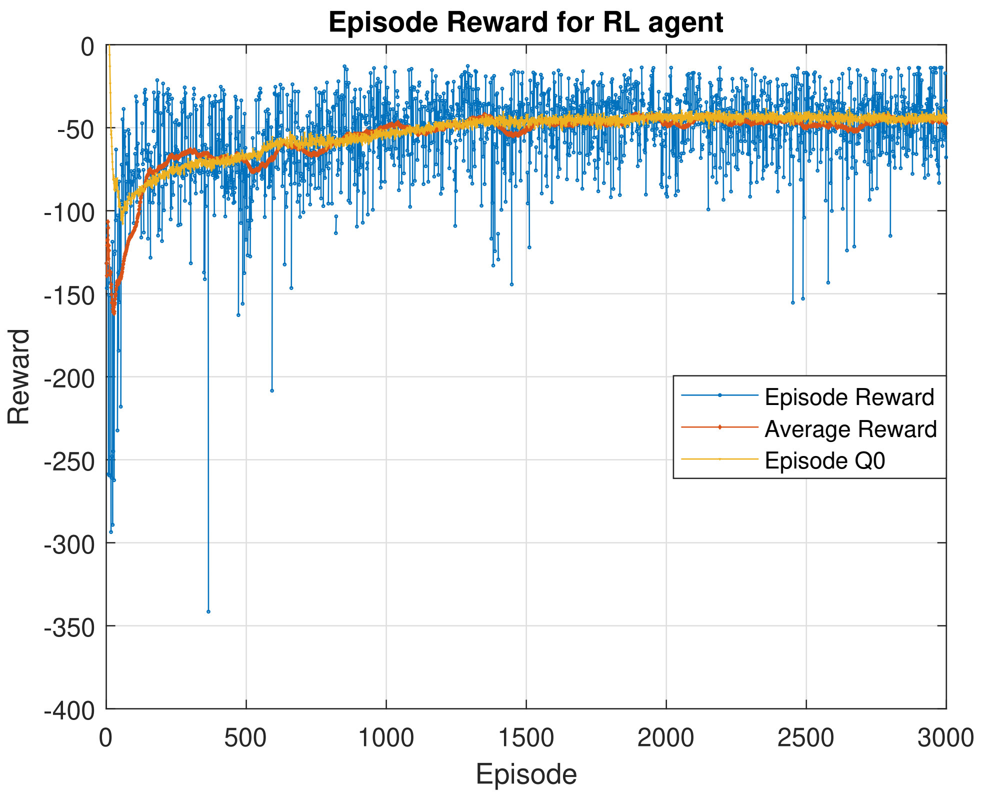


Figure 6.5: Training progress from second iteration.

Third Iteration

For the third iteration $R_{SwitchCounter}$ was added to the reward and the positive bones was removed. The "bonus" for this case was to just finish early, i.e. not accumulate more penalty. A P-controller was added to simulate an operator together with a function that overrode the agents action and closes all valve (gear 4) if the PV-reference is close to where the overlap starts (2 mm i.e. no flow). This means that the cylinder has reached or come close to the reference. A *isDone* criteria was added to terminate the episode if gear 4 had been active for $>$ 0.5 s. During the initializing of each episode the start position and reference was randomized to either start from zero and reach full stroke or vice versa. The downward cases was only chosen for 20% of the episodes to just give the agent some experience for this. See figure 6.6 for training progress.

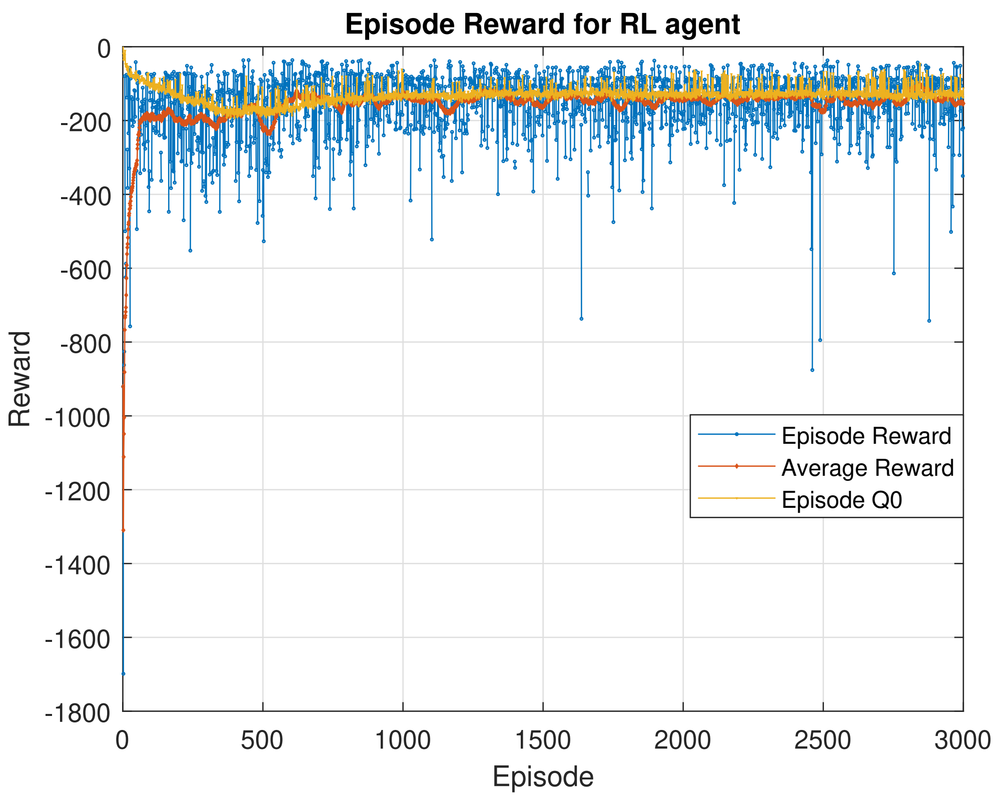


Figure 6.6: Training progress from third iteration.

6.2.7 Deployment

The deployment of the gear selection agent was done in the same way as position control, see Section 6.1.7.

6.3 Area Compensation

Since different areas will require different flow to maintain the same speed, a compensation is needed to maintain the same *drivers feeling* for the different gears. In this study, *drivers feeling* is defined as good if the same velocity of the cylinder is the same for same reference signal regardless of which gear is active.

Assuming no leakage, the flow to move MC is the same as the flow through the PV,

$$q_{MC} = q_{PV}. \tag{6.6}$$

The flow to the MC is calculated by

$$q_{MC} = v_{MC} A_{MC,gear} \tag{6.7}$$

where v_{MC} is the velocity, and $A_{MC,gear}$ is the area depending on the active gear. The flow through the proportional valve is calculated using the orifice equation,

$$q_{PV} = C_q L x \sqrt{\frac{2}{\rho}\Delta p} \tag{6.8}$$

where C_q is the flow coefficient, L is the spool circumference, x is the spool displacement, ρ is the density of the fluid and Δp is the pressure drop over the valve. Assuming the fluid parameters, C_q and ρ, the spool geometry, L, and the pressure drop Δp (using a pressure compensation valve) being constant, a PV constant, K_{PV}, is introduced:

$$K_{PV} = C_q L \sqrt{\frac{2}{\rho}\Delta p} \tag{6.9}$$

Introducing a constant K_A, the velocity of the cylinder is calculated by

$$v_{MC} = \frac{x K_{PV}}{A_{MC,gear}} K_A \tag{6.10}$$

where

$$K_A = A_{MC}/A_{MC,ref}. \tag{6.11}$$

The parameter K_A is dimension less constant to compensate for the different areas and $A_{MC,ref}$ is the area of the gear used as a reference. This was applied to the control signal to the PV, to have the same reference signal result in the same velocity, independent of the gear.

In this study, gear 9 is used for reference, $A_{MC,ref}$. This is the weakest and fastest of the selected gears used in the *Enabling Control*-agent. The normalized effective areas, K_A, PV spool displacement and the PV spool effective open length are shown in table 6.9.

Table 6.9: PV spool displacement for different gears with the area compensator.

Gear	Chambers	Effective Area	K_A	PV reference max	PV effective open
9	ABD	23.0	1.0	8.0	6.0
11	AD	26.0	1.1	7.3	5.3
14	ABC	33.0	1.4	6.2	4.2
16	AC	36.0	1.6	5.8	3.8

Different PV reference positions results in an adjusted opening positions of the PV for different gears, a set of these are presented in table 6.10. The 2 mm overlap and the 8 mm maximum stroke of the PV is included as saturation. For gear 16 the saturation happens at a 70% of the operator max signal, while gear 9 saturates at 100% operator max signal.

Table 6.10: PV reference position depending on operator signal and each gear.

PV ref. position	Gear 9	Gear 11	Gear 14	Gear 16
2.0	2.0	2.0	2.0	2.0
2.6	2.6	2.7	2.9	2.9
3.2	3.2	3.4	3.7	3.9
3.8	3.8	4.0	4.6	4.8
4.4	4.4	4.7	5.4	5.8
5.0	5.0	5.4	6.3	6.7
5.6	5.6	6.1	7.2	7.6
6.2	6.2	6.7	8.0	8.0
6.8	6.8	7.4	8.0	8.0
7.4	7.4	8.0	8.0	8.0
8.0	8.0	8.0	8.0	8.0

7 Results

The results from position control, enabling control and area compensation are presented in this chapter.

7.1 Position Control

Two tests were performed, one with the PV fully open, figure 7.1, and one with the PV 50% open, figure 7.2. The second test was conducted to see how well the agent performed to changes of the environment that it had never experienced previously, this could been tested more thoroughly but the simple approach was chosen due to safety and time constraints.

The top plot in figures 7.1 and 7.2 shows the selected action, both from simulation and physical tests. Gear 1 results in retraction of the cylinder, gear 4 stand still and gear 16 extension. The lower plots show the positions, including the reference signals. The fully open PV managed to control the position accurately while the 50% open-case had a small position error. This is discussed further in Chapter 8.

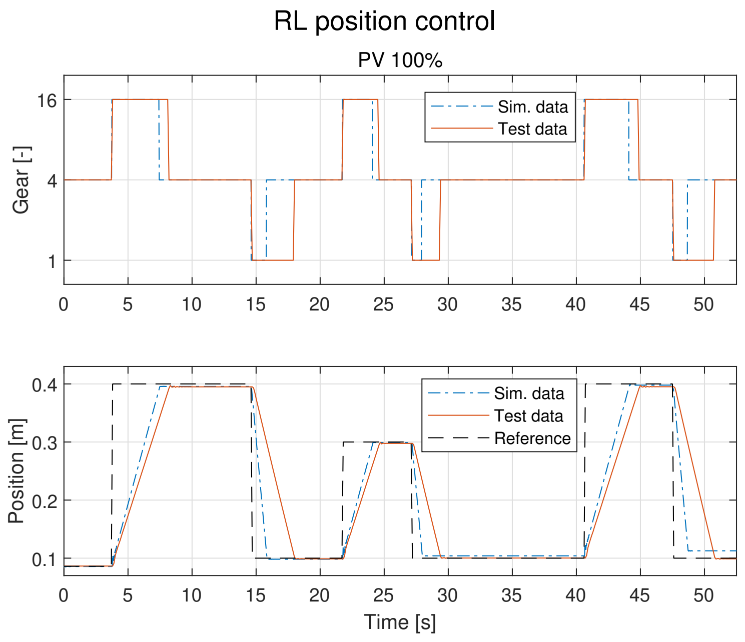


Figure 7.1: Test data from physical rig and from simulation, using 100% open PV.

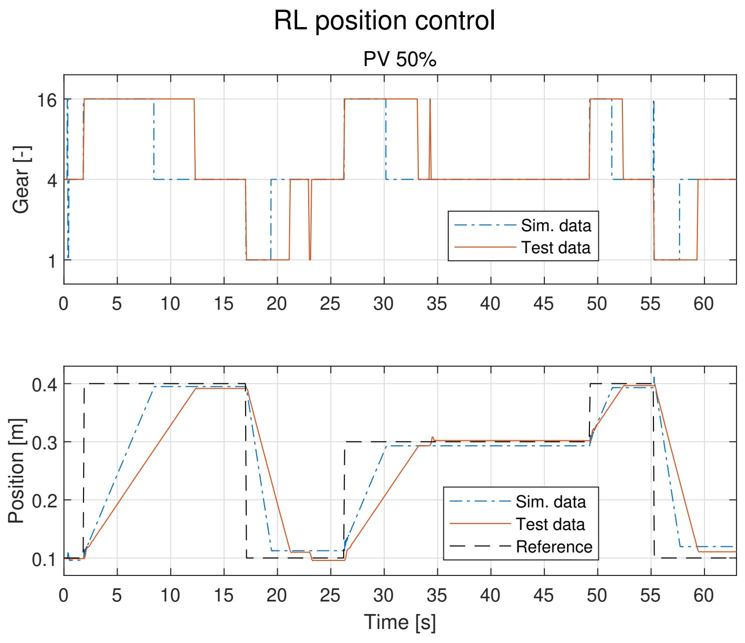


Figure 7.2: Test data from physical rig and from simulation, using 50% open PV.

In the simulation environment the direction of the PV was reversed to test the generalization of the agent further. This reverses the resulting direction of the gears, turning everything "up side down" in comparison to what the agent have been trained for. It can handle this, even thought far from perfect, see figure 7.3. This test was only carried out in simulation environment due to safety reasons.

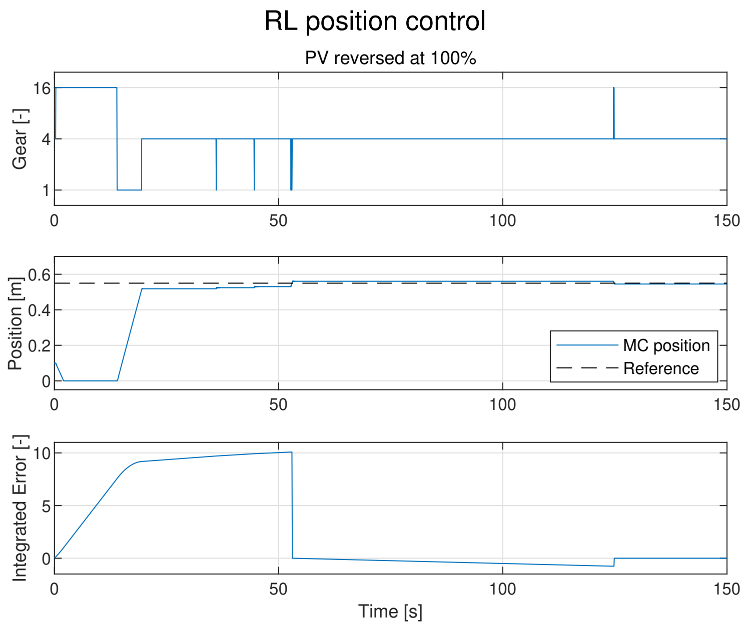


Figure 7.3: Simulation results when using reversed direction of PV. Integrated error is shown to explain the time for the final adjustment.

A sine wave reference test was performed in simulation environment to validate if the agent learnt the task, i.e. to follow a reference. There are a significant amount of gear switches but that was expected since the controller is discrete and the reference continuously changing, see figure 7.4. The agent was only trained on constant references and had never experienced such case before.

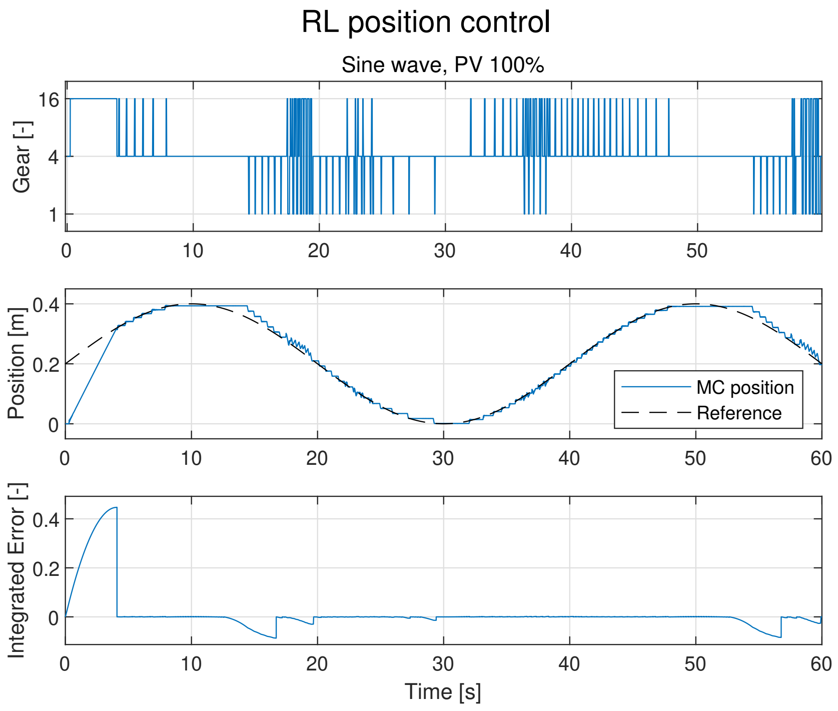


Figure 7.4: Simulation results when following a sine wave reference.

7.2 Enabling Control

Two different load cases were tested, 80 kg and 200 kg. The reason these loads where used instead of the loads the agent had been trained for is because of the load function and the model are not a perfect fit of the rig, see table 7.1. For 80 kg and a system pressure at 100 bar the test rig could lift the boom for a full stroke using the weakest gear (9) out of the four selected gears. This weight was chosen just because the agent should not select any other gear for this case. For 200 kg, gear 14 is needed to reach the full stroke, but a weaker gear should be used during the stroke. That would minimize the losses due to the load on the cylinder increases as the cylinder extends. This load was selected to verify that the agent did not choose the strongest gear but instead the most efficient one(s).

Table 7.1: Gears and loads used during training and test. The Load Rig column shows the physical systems equivalent loads to the simulation.

Gear	Chambers	Load Case [kg]	Load Rig [kg]
9	ABD / - / C	110	80
11	AD / B / C	170	160
14	ABC / D / -	230	200
16	AC / BC / -	260	280

7.2.1 Load Case 80 kg

The test was successful regarding what gears was chosen by the agent. A simulation was replicated by using the same reference and is used as a base line. For 80 kg only gear 9 is supposed to be used, as seen in figure 7.5 the agent choose gear 9 in most cases and few times gear 11. Gear 14 and 16 was used for short periods which could be explained by the resulting load on the cylinder was larger at some periods, due to oscillations. This is an effect of the MCs chamber pressures, which are observations for the agent. A comparison between the agents observations during simulation and test is seen in figure 7.6. Most observation from the simulation data is similar to the test data which is the results of a fairly good model. The largest difference is seen between pressures in chambers MC_B and MC_D, this is due to the oscillations of the load acting on the cylinder, seen in figure 7.5.

A sine wave reference test was performed successfully, see performance and observations in figure 7.7 and 7.8.

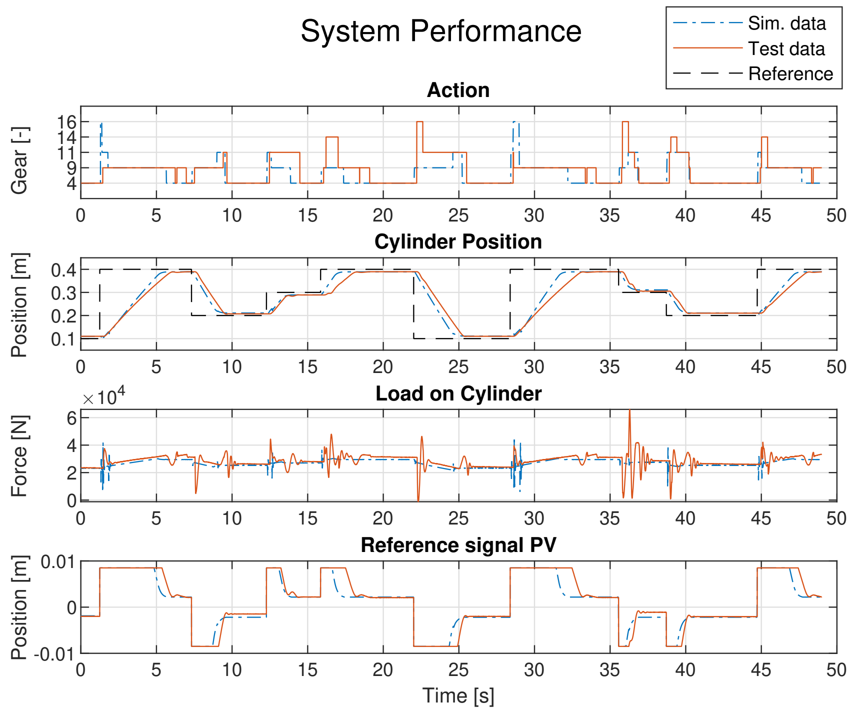


Figure 7.5: Performance for load case 80 kg.

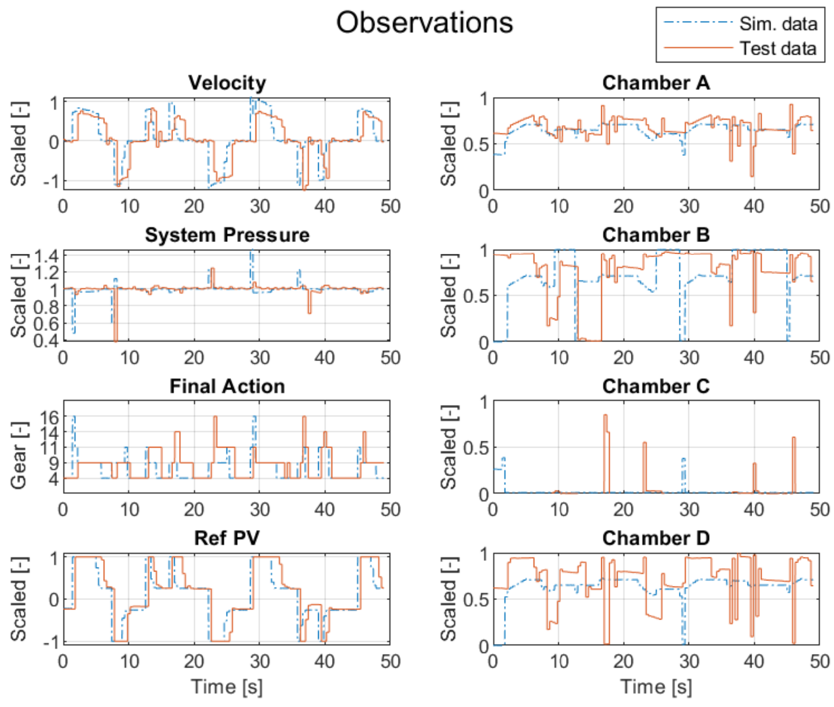


Figure 7.6: Normalized observations for load case 80 kg.

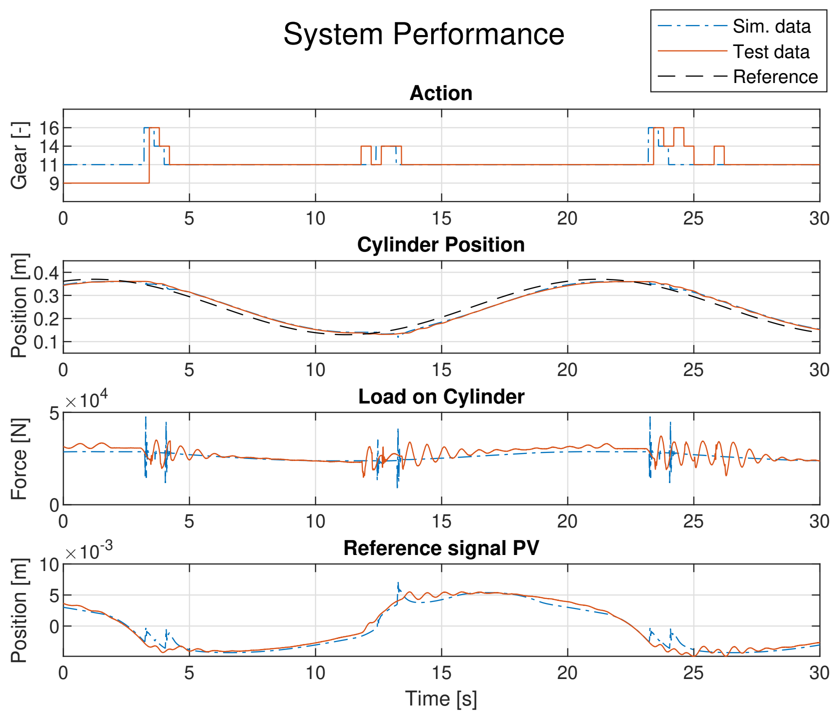


Figure 7.7: Performance for load case 80 kg, sine wave reference.

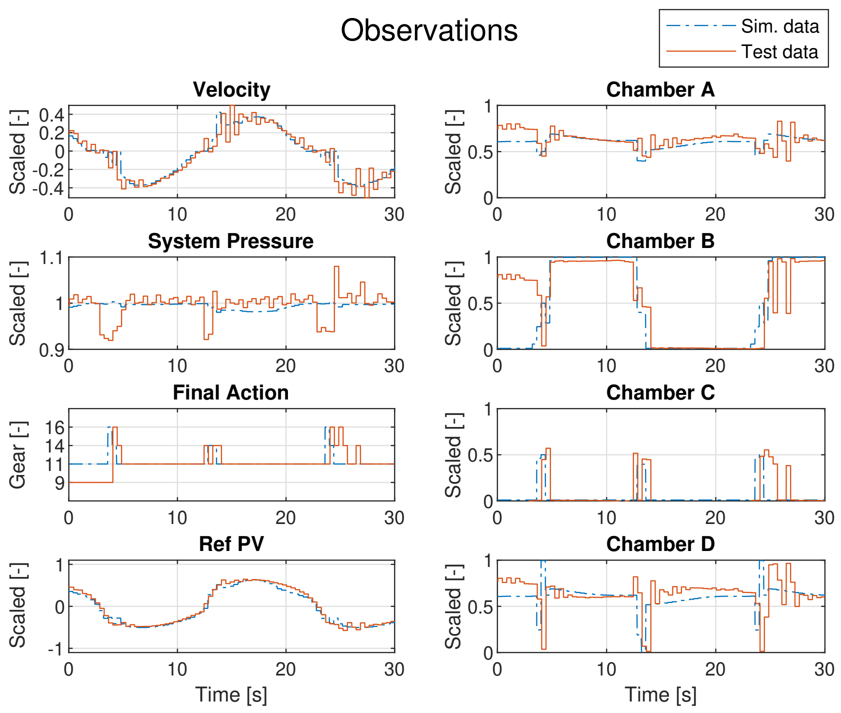


Figure 7.8: Observations for load case 80 kg, sine wave reference.

7.2.2 Load Case 200 kg

For the 200 kg load case using multiple gears during one full stroke was found to be optimal. As seen in figure 7.9 the agent chooses a weaker gear during the first part of the stroke and later switch to a stronger once it starts to slow down, see figure 7.10 for observations. The sine wave test was less successful when trying to control a heavier load, the dynamics of the load have a great influence on the system (e.g. P-controller) which affects the agents decisions. On the other hand had the agent never experienced a moving reference during training so the result is somewhat satisfying, see figure 7.11 and 7.12.

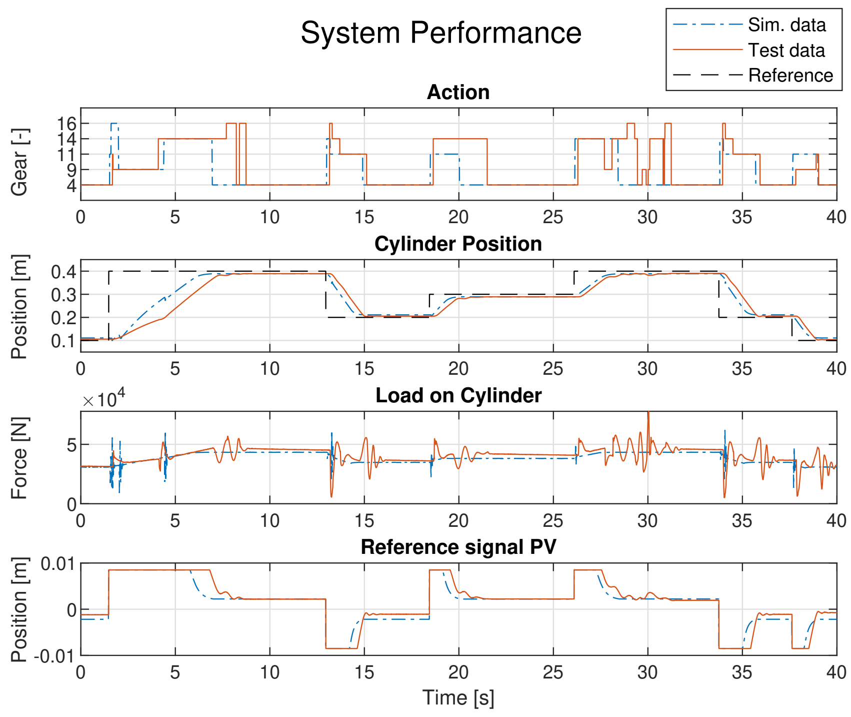


Figure 7.9: Performance for load case 200 kg.

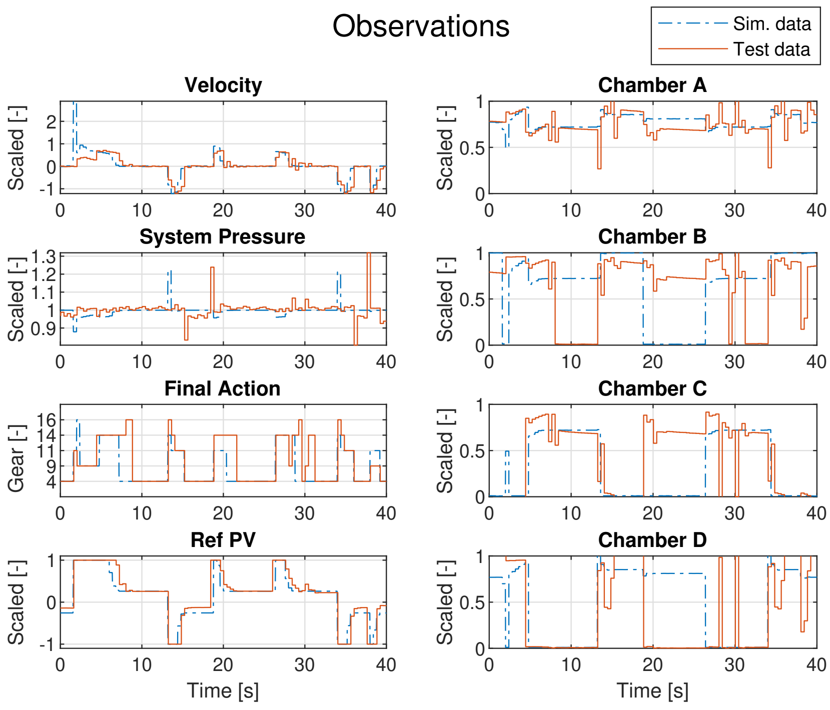


Figure 7.10: Observations for load case 200 kg.

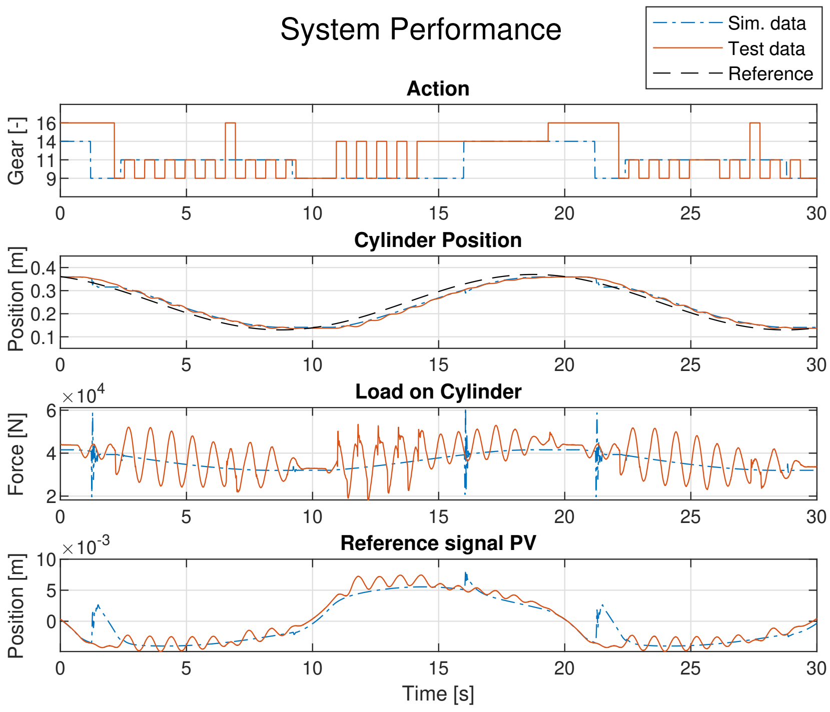


Figure 7.11: Performance for load case 200 kg, sine wave reference.

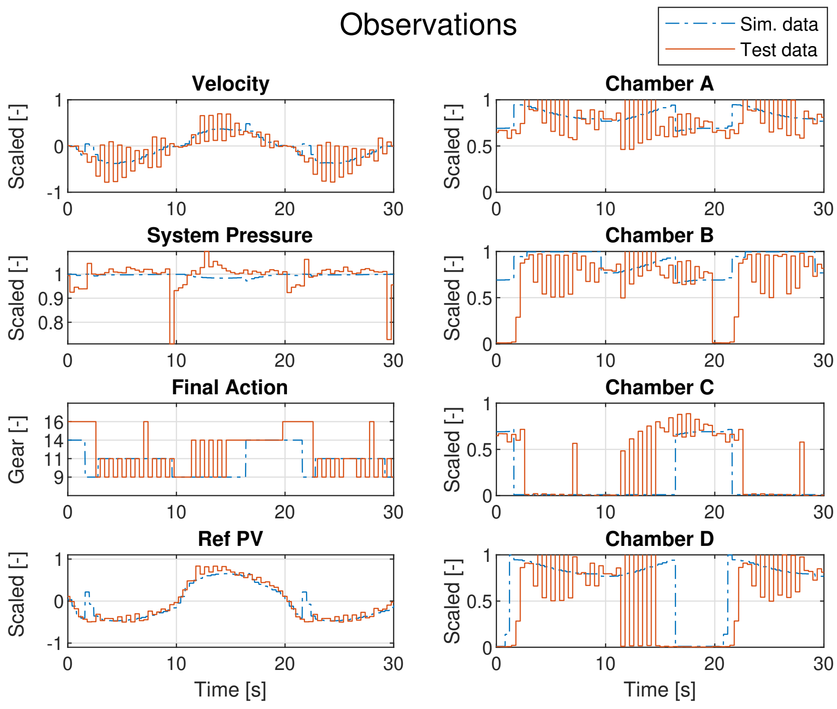


Figure 7.12: Observations for load case 200 kg, sine wave reference.

7.3 Energy Consumption

A comparison between using a single gear of the MC and the agent was performed to compare the energy consumption, but only conducted in simulation. If the MC was to be replaced by a SC, it had still been required to handle the highest load. Therefore was the strongest gear, 16, used as reference. This test was made with two different loads, 260 kg and 110 kg. See table 7.2 for the results. The energy consumption was calculated by integrate $P_{loss} = \Delta p_{PV} q_{PV}$, equation (3.5), by using the trapezoidal rule. The energy reduction is the energy saved using the agent gear selection policy instead of keeping gear 16 at all time, named "Manual" in the figures and the table.

In figure 7.13 the comparison between gear 16 and the agent is shown using a load of 260 kg. This is the most representative case since the load was optimized for gear 16 and the velocity is almost the same for both the manual and agent case. The agent choose to use the weakest gear possible and switch when the MC slows down and only uses the strongest gear at the end of the stroke. The main part of the agents lower energy consumption comes from the initial part of the stroke.

The 110 kg load case is shown in figure 7.14. The agent uses the weakest gear, as expected and reach full stroke faster. The lower energy consumption is a result from both a faster stroke and less power loss due to the smaller pressure drop. This velocity difference results in worse controllability for the operator and is the reason for the area compensation.

Table 7.2: Comparison of compensation energy losses over the PV for the cases.

Mode	Load Case	Value	Unit
Manual	260 kg	6161	J
Agent	260 kg	4397	J
Energy reduction	260 kg	28.5	%
Manual	110 kg	12450	J
Agent	110 kg	3496	J
Energy reduction	110 kg	72	%

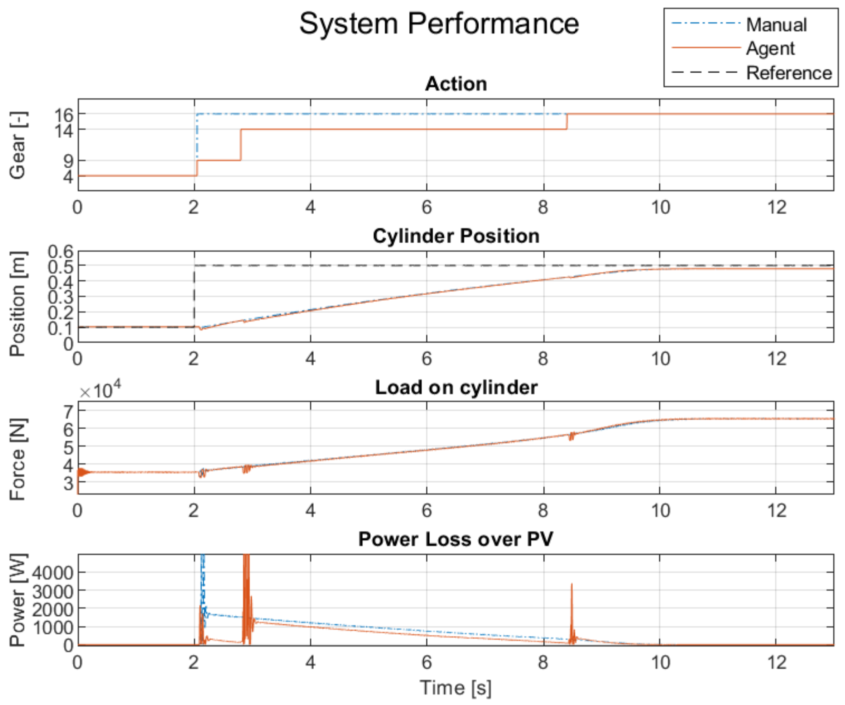


Figure 7.13: Energy and performance comparison between using gear 16 and the agents policy at load case 260 kg.

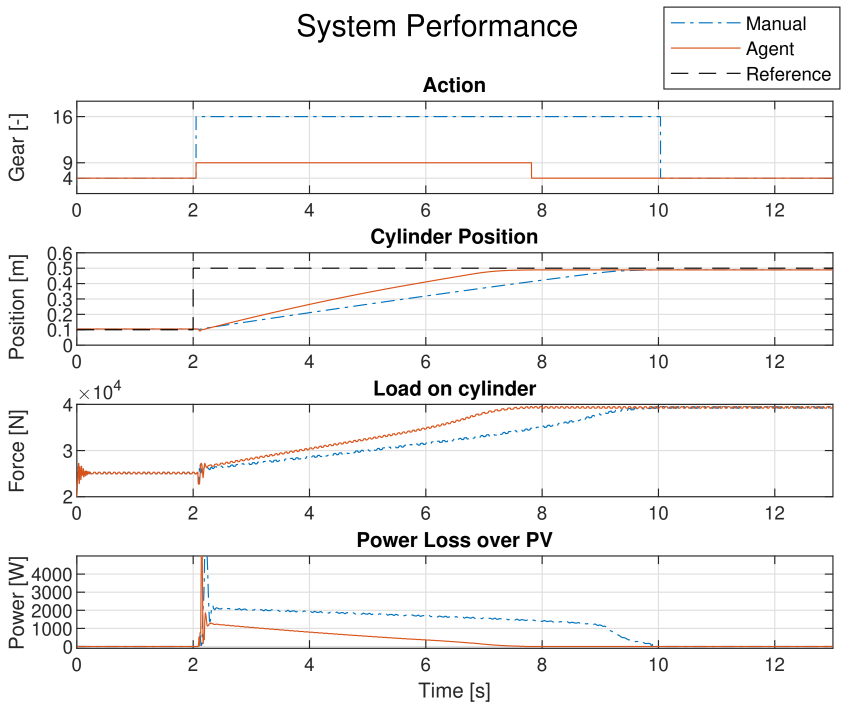


Figure 7.14: Energy and performance comparison between using gear 16 and the agents policy at load case 110 kg.

7.4 Area Compensation

The effects of using a scaling constant K_A for adjustment of the PV-controller is seen in figure 7.15. The reference test was a full stroke of the MC. The upper two plots are without K_A and the lower uses the compensation K_A. For comparable results the reference signal to the PV was limited to a maximum of 5.6 mm, to avoid saturation of the PV spool displacement to affect the results. This is the highest reference signal where no gear saturates the PV, see table 6.10. As seen on the right hand side plots, without K_A the velocities vary from 0.03 to 0.05 m/s, while all being close to 0.05 m/s when this is included.

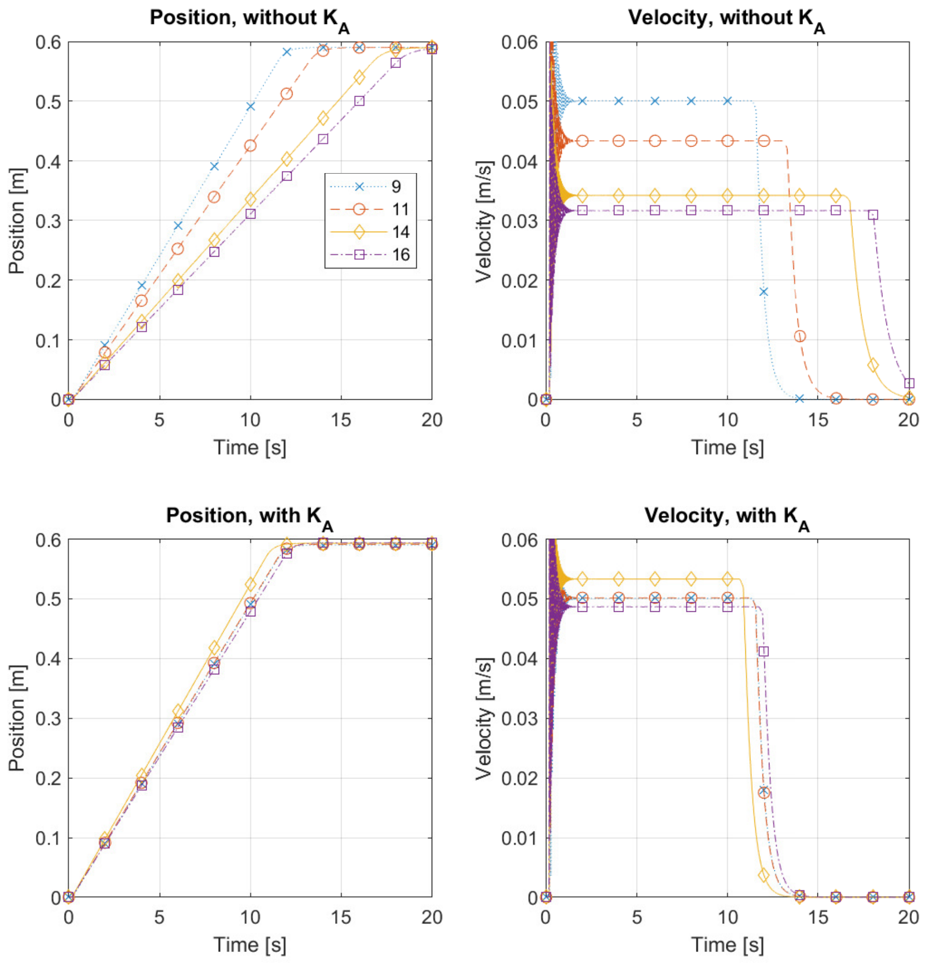


Figure 7.15: Position and velocity of the gears 9, 11, 14 and 16, with and without the area compensation parameter K_A.

8 Discussion

Discussion of the validation of system model, the Reinforcement Learning method, results and the PV controller adjustments is treated in this chapter.

8.1 Validation

The validations gave a model that follows the physical system good. The deviation between the simulation and the measurement remained small for all test cases except from SC step test.

The PV was validated with closed ends and therefore the flow forces was easily neglected. This showed to be a problem when validating the cylinders as they are dependant of the flow from the PV and the flow did differ between small and large steps with the spool. Especially when testing against a sine wave where the spool moves all the time. In hindsight it would been better to test steps/sines of the actuators and at the same time record the spool movement. That might be harder to validate as there are more parameters to tune but with the knowledge of them separately validated (which we now have) it should be possible. Mainly the flow rate from the different ports should be tuned. Leakage between the ports within the PV have been seen to have an impact in the test data which was only accounted for between port A and B of the PV. There are still the P-T, A-T, B-T, P-A and P-B ports to consider.

SC was closely connected to MCs parameters since they were sharing the same spool fraction parameters. These could been separated but was not for two reasons; The PV for SC and the PV for MC are identical and the dynamics of the SC model did only have a fraction of impact for this thesis and was therefore not prioritized. As mentioned above, if a second round of model validation should be conducted where the flow over the PV is accounted for this could greatly increase its precision.

8.2 Position Control

In both figure 7.1 and 7.2 the simulation model and the real system doesn't corresponds that good. For an extension movement is it quite close, but not for retraction. This is because the models parameters are tuned to match a step of the PV (i.e. using the same gear but changing direction of the PV). The PV have different flow rates for different ports, causing this difference in movement depending on PV position. On the other side, this means that the agent could still learn even though the model isn't perfect and therefore it has managed to generalize, which can be seen in both real test cases. The agent performs rather good and reaches the reference in each case. For the case when the PV is held 50% open, it closes the valves too early in some cases but adjusts by open and close quickly a brief moment later, seen in 7.2.

Setting the PV at 50% open, resulted in some fast spike gear selection for the physical system, which fine-tuned the position to be closer to the reference signal. This did not happen for the simulated version. Probably because of the integrated error: since the real system moved slower the integrated error increased faster which indicated that it needed to move further. Running the simulation longer would give time build up this value and eventually do the adjustment.

The position control when the PV is set at opposite direction reverses the direction for the gears. From the simulation, figure 7.3, the agent still manage to follow the reference signal. The expected results was to select action 3 at all times. This was true initially, but after a few seconds the agent reverses the actions and followed the reference. One possible explanation to this is the integrated error. When the agent observes a positive integrated error, the best action is to select action 1, and a negative observation select action 3. This value is to be compared to the position error. Initially the error was large, resulting in some time before the integrated error was large enough to overrule this decision. The tuning spikes later in the simulation did not have to wind up for too long since the position error was small (the drop of the integrated value to zero is because of the included reset function).

The sine wave simulation test behaved as one could expect. The agent works as a discrete controller, three discrete actions, it would be hard to follow a continuously changing reference. Although it switched extremely often it shows that it "understands" what it is supposed to do even thought it has never experienced a moving reference, not even a simple step.

Reward function

During the development of the reward function for the position control, the $R_{Direction}$ function, equation (6.1), contributed the most for an efficient training. This condition penalizes the agent while it is not actively moving towards the reference by comparing the sign of the error and velocity. This gave the agent an instant feedback and it quickly learned whether it was choosing the right action. Once this condition was implemented the agent started to grasp the task after 15 instead of $150 - 200$ episodes witch were the case before, the resulting agent also performed noticeably better.

The $R_{Direction}$ in position control was a good example of how direct feedback makes it easier for agents to learn. Using some direct feedback to learn the basics and then delayed rewards for more sophisticate tasks.

8.3 Enabling Control

During the 80 kg tests the agent choose more or less the same action in simulation environment and test. Comparing the observations from tests and simulations one can see that they are much more similar than for the heavier load case. This might be the load function not accounting for heavier loads good enough. The pressures are similar but the largest difference is seen when changing the flow rate from the PV, which affects the velocity and acceleration which in turn is the input for the load function. This does of course affect the system as a whole too, which makes it hard to define the root of the problem.

The reason why the agent takes different actions compared to the simulation test is that the decision taken depends on the observation at that time step, and there are differences between the simulation environment and the test rig. A safety function is used where all DVs are closed once the reference is reached. This will result in a trapped pressure inside the previously active chambers of the MC. Since the pressures are observed, this trapped pressure will affect the next decision. This could explain why different gears are chosen. For an example could there be 70 bar pressure in chamber B in simulation while there is no pressure during the test. This difference in observation is large, which could result in different actions.

Dynamics from the load plays an important role as it affects the behaviour of the system significantly. It induces more oscillations which changes the pressures in active chambers as well as changes the P-controller to oscillate if the position is close to reference. The load dynamics difference is clearly shown in figure 7.5 and 7.9.

In figure 7.9 and 7.10 there are some unnecessary/uncontrolled switching taking place at around 30 seconds. It is hard to say exactly what trigger those decision since the pressures are dependant on the active gear and the oscillations that started due to the gear switch. If the cylinder position changes, the P-controller will try to compensate for that, which in turn changes the observations as well.

The agent perform well on the sine wave test for the 80 kg load case, almost identical to the simulation, see figure 7.7 and 7.8. For the 200 kg load case it didn't perform as well, it seems to be a combination of a heavier load which gives higher inertia along with a poorly tuned P-controller. These two in combination seems to start the oscillations. If a PI-controller would be implemented the issue might been avoided but it wasn't prioritized due to the simulation tests gave good results when tested. The over-all performance is good, considering that the agent had never experienced a moving references during a training episodes.

An agent was tested to be trained on sine waves but with worse result than the agent used for these tests. Most likely was it hard to grasp how to cooperate with the P-controller. The reward function as well as the P-controller could need some development for such case. Alternative for a more successful training could be to only observe the P-controllers signal before is compensated (due to velocity and gears). Or use a PI-controller which would soften the movement.

Reward Function

The velocity penalty is a discrete function which activates if the cylinder is moving too slow depending on how much the PV is open. This function and the power loss function formed a good combination where the agent had to move fast enough but at the same time use the lowest possible gear. The velocity function was only tuned by measuring the velocity of the slowest gear (16), for simplicity.

The power loss function only accounts for the pressure drop and flow over the PV. However, when using gears that have a direct connection to tank there are losses due to suction of oil from tank. For a more realistic power loss function these needs to be included.

The switch counter penalty are accumulating for each gear switch that takes place, telling the agent to switch as few times as possible during training. The back side of this approach is that lots of switching in the beginning of an episode will result in a higher accumulated penalty than for lots of switches towards the end of the episode. An alternative is to give a final reward at the end of the simulation based on the total number of switches. The backside of this could result in a training where the agent doesn't want finalize the episode, if it only consider ending the simulation with a penalty instead of associating it with the actual reason; the amount of switches. Another approach would be to use the switch counter as it is but implement a decay rate that reduced the penalty gradually after a set time, this would even out the penalty and it might be easier to grasp for the agent.

Implementing our knowledge can guide the agent to learn quicker. For instance, knowing that a lower gear is better and faster. There is no point in letting the agent figure that out. An important difference is to tell the agent *what* to do and not *how* to do it. Already known facts could most likely be implement manually as a rule based controller.

The reward function is how you as a developer communicates with the agent and to do that you have to understand how the algorithms works, how good it is to exploit poorly defined reward function etc. At the same time the reward function should not be too detailed, as said before, how to solve the task is what the agent should learn, in that way we can learn from it. More difficult problems of gear switching are when the system (environment) have multiple actuators and controllers, varying system pressure which is not included in this work. It seems to work well by using *Graded Learning* so this technique would be recommended when trying to tackle a more complex environment and task. Also how the actual switching sequence between two gears could be optimized to minimize the drop that occurs due to gravity and connecting depressurized chambers to pressurized ones.

8.4 Agent Settings

There are many parameters to consider for tuning: all hyperparameters in the DQN-agent, construction and size of the neural network and reward function. Since the goal was to develop an agent that worked, details in these parts was not analyzed, instead parts that worked were built upon. A structured analysis was not carried out, but could be a way of optimizing the agent. The size of the network (the agents critic) is also an important aspect for the real time implementation, where a smaller one requires less computational power.

We used a rather large network for the position control which worked good (7x256x256x3). This network was later diminished for the gear switch case to 8x256x128x4 and could most likely be reduced further but due to lack of experience, lots of different settings to try out and time it wasn't researched further.

8.5 Energy Consumption Comparison

As mentioned earlier, the most representative case is the heavier case, figure 7.13, where the cylinders velocity is almost the same during the full stroke. Most of the performance is the same but with some smaller position drops due to gear switching. The result of almost 30% efficiency is promising result for the MC and digital hydraulics-systems, however the test case was conduced as a constant pressure system and not a load sensing system. For a system with one actuator, a load sensing architecture is enough for a high efficiency. Used in a system with multiple actuators, where the MC doesn't affect the system pressure, is where the large energy savings can be made.

The smaller load case, figure 7.14, has about 70% better efficiency which is impressive. It is partly due to the fact that it finishes the stroke earlier, but even if the two cases would reach the reference at the same time the agents case would be more efficient. The power loss over the PV for the agents case is always about half of the manual case. It might not be a fair comparison since it's obvious that a smaller area for a small gear will result in a better efficiency compared to an oversized one. But if a system would been designed one would need to design it to handle the heaviest case, which justifies these results some.

This comparison does only account for the pressure drop and flow through the PV connected to the DV block. Other losses for a DV-block/MC like switching and leakages are not accounted for. But since the comparison is made for the same component it should be seen as representative since the efficiency is notable!

8.6 Area Compensation

The parameter K_A to adjust the proportional gain works good. Without this the velocities differs from 0.04 m/s for the weakest gear to 0.025 m/s for the strongest. The compensation made the velocities differs between 0.05 m/s and 0.03 m/s. For the same input reference the output speed was the same. This parameter is a possible change to the PV controller to keep the same driving feeling, despite the occurrence of gear switching.

In a case where all possible gears, table 3.2, are included in the gear selection, a weaker gear have to be used as the reference area to adjust the controller gain. A large K_A will saturate the PV at a lower reference value than what a lower would, which affects the control range of the stronger gears. In table 8.1 this is shown using the weakest and strongest gears. Including the overlap in the PV, the valve is saturated already at the effective reference 1 mm, resulting in a very limited range to control the velocity. A reference above 1 for gear 16 will not increase the velocity, creating a feeling of play in the joysticks outer values.

Table 8.1: Effects of K_A when using gear 5 as the reference gear.

Gear	Chambers	Effective area	K_A	PV reference max	Effective PV reference max
5	BC	6	1	8	6
16	AC	36	6	3	1

8.7 Gear Selection

There are two parts of the reward function: one for making the cylinder move (favoring stronger gears) and one for maximizing the efficiency (favoring weaker gears). These are the two extreme points of the learning: performance (moving) and efficiency (energy consumption). Finding a balance between these were a challenge. Favoring the performance too much results in an agent that in most cases selects the strongest gear, since this is the extreme point of the performance. Favoring efficiency too much results in an agent that always selects the weakest gear, either it will move and complete the task or it will stand still and have no losses at all.

The performance extreme point is a result of that the strongest gear will always complete the task, no matter the load for our cases. The weaker gears requires low enough load to complete the stroke. After many iterations this results in an agent that always uses the strongest gear no matter the load, despite a reward function that punishes the power losses. This is probably a result of the agent only experience that the stronger gear always complete the task, and realize it is the best action(s) to do in the long run. Even though it receives a large punishment, it considers the largest gear the safest to select.

For more efficient learning, the difference for rewards between actions needs to be large. If the most favorable action gives a reward with a significant higher value compared to the next best, there is a higher probability that this will be fond during training. To accomplish this in our study, the load for each gear was selected in such way that all the lower gear(s) couldn't handle it. Each load case was also designed such that the resulting load pressure where close to the system pressure, creating large losses for using stronger gears.

The amount of gears available to the agent used as gear selector controller is limited, using a set of 4 of the 16 possible gears from table 3.2. The more loads used, the harder it is to find these sweet-spots where both movement takes place and a not too strong gear is selected. Furthermore are loads in between required to be used during training to make an agent to select gears in all possible situations. Also when a full cycle such as dig & dump is used in training, future rewards will play an important role to find the optimal gear selection considering the entire cycle. These extreme points needs to be handled carefully to not always end up in any of the extreme cases.

Over all the concept is proven to work for gear selection and the agent found an optimal behaviour to change gear during the stroke as the load increases by the extension of the cylinder.

8.8 Unsuccessful Tries

Originally the position controller was intended to include multiple extension gears and trained using different loads, to learn it to select an optimal gear for the given load. Due to unsuccessful tries in simulation this idea were abandoned, in favor for the work of using the agent as the enabler together with an P-controller.

A problem we faced were that the agent always favored the stringest gear, no matter the external load. By averaging the reward and learning it most likely realized that no matter the case, the strongest gear would complete the task. When the power loss reward function was successfully implemented this was finally solved.

Power losses during gear switch is a known problem for MCs. The gear switching procedure is not in the scope of this thesis, but different reward functions have been used to minimize the number of switching, i.e. reduce unnecessary switching. One way is to use a time constraint, the agent have to keep the same gear for some time before it receives rewards for a positive actions.

8.9 Multi-Agent Application

As mentioned earlier, the gear switching was not investigated in this study. As this is a difficult task, where the number of possible ways of changing gear are large as well as include different opening timings between valves to keep the switching losses to a minimum. This might be a task for another agent, where the agent in this work controls which valves to open and close in between a gear switch.

In [20] an agent was trained to do the full movement of an excavator. Using this in combination with this agent could result in even better efficiency. This would mean that a predefined path is calculated, and possibility to optimize the gear selection depending on that (instead of just consider the current load). This combination gives the possibility to plan the gear selection, both what gear and when to do the switch.

9 Conclusion

Reinforcement learning is easy to play around with and is a completely different approach to develop control systems compared to conventional methods. But time and experience are needed to create well defined reward functions, choose correct observations and setting up the simulation for each episode. Knowledge of how the algorithm works such as experience buffer, back propagation for the neural network and much more is needed for more sophisticated problems. This technique offers another take on control problems that are hard to define or solve, and could be used as a compliment in a larger control system where the agent controls specific hard parts while known control theory is used for their specific problems. An agents decision should be constrained by some safety measurements since it's hard to predict what the decision will be for all cases.

9.1 Research Questions

- **How can Reinforcement Learning be used for gear selection to improve the energy efficiency while maintain the performance of a hydraulic system with a *hydraulic force gearbox*?**
  By training a DQN Agent in simulation environment, it managed to learn which gear to use depending on the load. Selection of a gear that is strong enough keeps the system performance while allowing weaker gears reduces the losses. Due to the DQN Agents generalization it was possible to implement this in the real world test rig with similar performance as the simulation.

- **How shall the training process of a Reinforcement Learning model be performed for a *hydraulic force gearbox* system?**
  The Agent was successfully trained by using *Graded Learning*, where the complexity of the environment and the task was increased step wise. This also included the reward function which is an important aspect to find the balance between performance and efficiency. The reward function and task should explain what to do, not how to do it.

- **What changes are needed for the control of the proportional valve to maintain the same system performance?**
  The control of the proportional valve needs some adjustment to maintain a good drivers

feeling. As the different gears have different areas which will result in different velocities for the same flow. Therefore are different flow rates needed to keep the same velocity. Introducing a scaling factor, K_A, can make the the same reference signal result in the same velocities for the different gears.

9.2 Future Work

This thesis have showed that the concept works: a Reinforcement Learning agent can learn to select the right gear for a given load and be implemented in a real system. The number of gears used have been limited to four. For a wider operation range, more of the 16 gears needs to be included in the training process. Also continue the training with more advanced load cycles, based on real world operations, needs to be made. This will make the decision making even closer to the global optimum. This can also be validated using a Dynamic Programming simulation.

A thorough sensitivity analysis of the agents observations should be conducted to gain more knowledge of how they affect its decisions. Some observations might be unnecessary and therefor could either be removed or replaced. This would also be a good approach to validate the agents robustness.

The switch between two gears could need some improvement since there will be position drop when switching, especially when activate or deactivate chamber M_A or M_C. Using a rule based controller or train an agent to find a suitable gear switch sequence could improve the performance and controllability.

The agent have not been introduced to a load sensing system, different constant pressures have been used during training. However, the agent have not experienced situations where the system pressure depends on the selected gear. The agent will in this case experience two different scenarios: constant system pressure when the single chamber cylinder sets the system pressure, or the system pressure depends in the gear when the multi chamber cylinder sets the system pressure.

The adjustable gain for the control of the proportional valve also needs to be validated using as real system. In simulations the velocities are kept the same for the same reference, but if this will achieve a good drivers feeling for the physical system is not validated. Using a real joystick and an experienced driver are needed for validation as drivers feeling is hard to measure.

Bibliography

[1] Henrique Raduenz, Liselott Ericson, Kim Heybroek, Victor J. De Negri, and Petter Krus. “Improving the efficiency of valve-controlled systems by using multi-chamber actuators”. In: *The 17th Scandinavian International Conference on Fluid Power* (2021). Sweden, Linköping.

[2] Alessandro Dell’Amico, Marcus Carlsson, Erik Norlin, and Magnus Sethson. “Investigation of a Digital Hydraulic Actuation System on an Excavator Arm”. In: *The 13th Scandinavian International Conference on Fluid Power* (2013). Sweden, Linköping.

[3] A. Harrison and D. Stoten. “Generalized finite difference methods for optimal estimation of derivatives in real-time control problems”. In: *Journal of systems and control* (1995).

[4] M. Vukovic, R. Leifeld, and H. Murrenhoff. “Reducing Fuel Consumption in Hydraulic Excavators - A Comprehensive Analysis”. In: *Energies 2017, 10, 687* (2017).

[5] Damiano Padovani, Massimo Rundo, and Gabriele Altare. “The Working Hydraulics of Valve-Controlled Mobile Machines: Classification and Review”. In: *Journal of Dynamic Systems, Measurement, and Control. July Vol. 142* (2020).

[6] Zimmerman J. D., M. Pelosi, C. A. Williamson, and S. Anwar. “Energy Consumption of an LS Excavator Hydraulic System”. In: *ASME Paper No. IMECE2007-42267* (2007).

[7] V.H. Donkov, T.O. Andersen, M Linjama, and M.K. Ebbesen. “Digital Hydraulic Technology for Linear Actuation: A State of the Art Review”. In: *International Journal of Fluid Power, Vol 21_2* (2020).

[8] Linjama M, Vihtanen H-P, Sipola A, and Vilenius M. “Secondary Controlled Multi-Chamber Hydraulic Cylinder”. In: *The 11th Scandinavian International Conference on Fluid Power* (June 2009). Sweden, Linköping.

[9] Qian Zhu and Qing-feng Wang. “Real-time energy management controller design for a hybrid excavator using reinforcement learning”. In: *Journal of Zhejiang University-SCIENCE A (Applied Physics & Engineering)* (2017).

[10] Richard.S. Sutton and Andrew.G. Barto. *Reinforcement Learning, An Introduction - second edition*. The MIT Press, 2018.

[11] Wikimedia Commons. *Reinforcement Learning Diagram*. 2021-03-24. URL: `https://en.wikipedia.org/wiki/Reinforcement%5C_learning`.

[12] Rajesh Siraskar. *Reinforcement Learning for Control of Valves*. 2021.

[13] MathWorks. *Reinforcement Learning Agents*. URL: `https://se.mathworks.com/help/reinforcement-learning/ug/create-agents-for-reinforcement-learning.html`. (accessed: 03.19.2021).

[14] Melrose Roderick, James MacGlashan, and Stefanie Tellex. "Implementing the Deep Q-Network". In: *30th Conference on Neural Information Processing Systems* (2016). Spain, Barcelone.

[15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. "Playing Atari with Deep Reinforcement Learning". In: *DeepMind Technologies* (2013).

[16] MathWorks. *Deep Q-Network Agents*. URL: `https://se.mathworks.com/help/reinforcement-learning/ug/dqn-agents.html`. (accessed: 03.22.2021).

[17] S. Dadhich, U. Bodin, F. Sandin, and U. Andersson. "Machine Learning approach to Automatic Bucket Loading". In: *24th Mediterranean Conference on Control and Automation (MED)* (2016).

[18] S. Dadhich, U. Bodin, and U. Andersson. "Key challenges in automation of earth-moving machines". In: *Automation on Construction 68* (2016).

[19] S. Backman, D. Lindmark, K. Bodin, M. Servin, J. Mörk, and H. Löfgren. "Continuous control of an underground loader using deep reinforcement learning". In: (2021).

[20] H. Kurinov, G. Orzechowski, P. Hämäläinen, and A. Mikkola. "Automated Excavtor Based on Reinforcement Learning and Multibody System Dynamics". In: *IEEE Access 2020 Vol 8* (2020).

[21] B.J. Hodel. "Learning to Operate an Excavator via Policy Optimization". In: *Complex Adaptive Systems Conference with Theme: Cyber Physical Systems and Deep Learning* (2018).

[22] T. Li, Y. Zou, D. Liu, and F. Sun. "Reinforcement Learning of Adaptive Energy Management With Transition Probability for a Hybrid Electric Tracked Vehicle". In: *Transactions on Industrial Electronics* (2015).