Computer-Based Environment for Lower Limb Neurorehabilitation and Assessment

Biomechatronic System for Human Machine Interconnection through Motion Platform, Virtual Reality and Motion Capture System

by

Jomås, David
Lien, Bård Kjetil

Supervisor:
Morten Ottestad, UiA

This Master's Thesis is carried out as a part of the education at the University of Agder and is therefore approved as a part of this education. However, this does not imply that the University answers for the methods that are used or the conclusions that are drawn.

University of Agder, 2020
Faculty of Technology and Science
Department of Engineering

Acknowledgments

We would like to thank our supervisor professor Morten Ottestad for outstanding support, guidance and advisory during this project.

Additionally, thanks to Jan Christian Strandene for excellent help during time of testing and for use of facilities.

Lastly, we would like to thank Sondre Tørdal for providing help during setup of the motion lab facilities.

Development of this system has been in cooperation with another Master's group developing the virtual world. We would like to thank this group for the cooperation and their VR contribution.

Abstract

As the need for neurorehabilitation increases, so does the demand for efficient rehabilitation facilities. Studies show that by gamifying the rehabilitation process, the patients are more motivated to continue the rehabilitating work compared to traditional therapy. Through this paper, the process of developing a prototype system for neurorehabilitation by using a six degree of freedom platform in conjunction with a virtual reality, will be described. This includes the production of a prototype system, development of a control system and testing. With the use of haptic technology, the platform is simulating the response of a board-like device in addition to simulating the motion in relation to the virtual world surrounding the user using VR-goggles.

The system is combined of several components, used for different purposes to create a complete system. A Stewart platform is used in conjunction with a treadmill placed on four load cells. The load cells are used to record the movements of the user, to further control the motion of the Stewart platform and the avatar in the virtual world. Additionally, a safety frame has been constructed in order for the user to be safely suspended in a safety harness to prevent injury in case of tripping or falling. Sensors and actuators, as well as the virtual reality software communicate via a constructed control program running on a myRIO device to both physically and visually emulate the feeling of moving within an artificial environment.

Through testing, the system worked as planned. The platform is successfully providing a force feedback to the user depending on the load the user is distributing on the treadmill. Furthermore, the combination of a moving platform and a functioning virtual reality visualized through VR-goggles is very immersive. As an initial testing platform for proof of concept, the system is promising. All the different components have been successfully combined in order to make the platform respond as intended with respect to the motion of the user and virtual topography. Even though some planned features were not implemented in the prototype and are instead recommended for future work, the prototype is considered a success in terms of proving the concept.

A short video about six minutes is made to give an impression of what the system looks like and how it behaves. Additionally, it shows some of the manufacturing process and initial test before final tests were carried out.

Click the "play-button" to play the video:

Nomenclature

Abbreviations

Abbreviation	**Word or phrase**
CAREN	Computer Assisted Rehabilitation Environment
CELNA	Computer-based Environment for Lower limb Neurorehabilitation and Assessment
DOF	Degrees Of Freedom
VR	Virtual Reality
AC	Alternating Current
A/D	Analog to Digital converter
D/A	Digital to Analog converter
3D	Three Dimensional
I/O	Input to Output relation
MIMO	Multiple Input Multiple Output
PS	PlayStation
NW	North West
NE	North East
SW	South West
SE	South East
fps	frames per second
FOV	Field Of View
QTM	Qualisys Track Manager
FPGA	Field Programmable Gate Array
UDP	User Datagram Protocol
IP	Internet Protocol
TCP	Transmission Control Protocol
HID	Human Interface Device
WLAN	Wireless Local Area Network
SI	Système international
USB	Universal Serial Bus
VI	Visual Interface
EMI	Electro-Magnetic Interference
CS	Coordinate System
CAD	Computer-Aided Design
aoa	available operating area
oa	operating area
CW	Clock-Wise
FBD	Free Body Diagram
KD	Kinetic Diagram
LHP	Left Hand Plane

Table 1: Abbreviations

Coordinate systems

Coordinate system	**Description (three dimensional unless noted otherwise)**
N	Neutral coordinate system of Stewart platform of which the transmitted signals are relative to
b	The Stewart platform's original control point
p	Point on the striding belt right beneath the person's feet, also known as desired original control point
NW	North West Load cell
NE	North East Load cell
SW	South West Load cell
SE	South East Load cell
aoa	Available operating area (two dimensional coordinate system formed by the load cells with origo right in the center. Ranges from -1 to 1)
oa	Operating area (two dimensional coordinate system formed by the virtual board with origo right in the center. Ranges from -1 to 1)
cap	Capsule in the virtual world, representing the character "played" by the person
ve	Varying global coordinate system within the virtual world. cap is described relative to this system
sim	Transformed cap coordinates to make the virtual world coordinates of the capsule be described in the same orientation as the coordinate system of the Stewart platform
n	North model (one dimensional)
s	South model (one dimensional)
w	West model (one dimensional)
e	East model (one dimensional)

Table 2: Coordinate Systems

Symbols

Section (2.3), Haptic Technology

Symbol:		Description:	Unit:
V	-	Voltage	V
I	-	Current	I
Z	-	Impedance	Ω
Y	-	Admittance	S
F	-	Force	N
ΔX	-	Displacement	m
K	-	Spring Constant	$^{N}/_{m}$

Section (2.5), Method for Emulation of Acceleration

Symbol:		Description:	Unit:
F_p	-	Force from person	N
$F_{N,NW}$	-	North West normal force	N
$F_{N,NE}$	-	North East normal force	N
$F_{N,SW}$	-	South West normal force	N
$F_{N,SE}$	-	South East normal force	N
$F_{cp,NW}$	-	North West centripetal force	N
$F_{cp,NE}$	-	North East centripetal force	N
$F_{cp,SW}$	-	South West centripetal force	N
$F_{cp,SE}$	-	South East centripetal force	N

Section (2.7), Data Acquisition and Signal Conditioning

Symbol:		Description:	Unit:
V	-	Voltage	V
R	-	Resistance	Ω
I	-	Current	I

Section (3.4), Treadmill Harness Support

Symbol:		Description:	Unit:
P	-	Stress	Pa
F	-	Force	N
A	-	Cross-sectional area	m^2
F_x	-	Force in x-direction	N
F_y	-	Force in y-direction	N
F_z	-	Force in z-direction	N

Section (4.2), Initial Position

Symbol:		Description:	Unit:
$T_{X,N,b}$	-	Translation from coordinate system N to b in X direction	m
$T_{Y,N,b}$	-	Translation from coordinate system N to b in Y direction	m
$T_{Z,N,b}$	-	Translation from coordinate system N to b in Z direction	m
$R_{X,N,b}$	-	Rotation from coordinate system N to b about X axis	rad
$R_{Y,N,b}$	-	Rotation from coordinate system N to b about Y axis	rad
$R_{Z,N,b}$	-	Rotation from coordinate system N to b about Z axis	rad
s_i	-	Load cell measurement number i	V
n	-	Total number of measurements	$-$
s_{zero}	-	Load cell zeroing value	V
s_{NW}	-	Raw North West load cell measurement	V
$s_{NW,zero}$	-	North West zeroing value	V
s'_{NW}	-	North West load cell measurement conditioned once (1st order)	V
$\sum F$	-	Sum of forces	N
m	-	Mass	kg
a	-	Acceleration	$^{m}/_{s^2}$
$T_{X,b,NW}$	-	Translation from coordinate system b to NW in X direction	m
$T_{Y,b,NW}$	-	Translation from coordinate system b to NW in Y direction	m
$T_{X,b,NE}$	-	Translation from coordinate system b to NE in X direction	m
$T_{Y,b,NE}$	-	Translation from coordinate system b to NE in Y direction	m
$T_{X,b,SW}$	-	Translation from coordinate system b to SW in X direction	m
$T_{Y,b,SW}$	-	Translation from coordinate system b to SW in Y direction	m
$T_{X,b,SE}$	-	Translation from coordinate system b to SE in X direction	m
$T_{Y,b,SE}$	-	Translation from coordinate system b to SE in Y direction	m
$d_{t,NW}$	-	Tangential displacement of NW load cell	m
$d_{t,NE}$	-	Tangential displacement of NE load cell	m
$d_{t,SW}$	-	Tangential displacement of SW load cell	m
$d_{t,SE}$	-	Tangential displacement of SE load cell	m
s'_i	-	1st order conditioned load cell measurement number i	V
$d_{t,i}$	-	Tangential acceleration measurement number i	$^{m}/_{s^2}$
m_{mean}	-	Mass of the part of treadmill resting on top of a load cell	$^{Vs^2}/_{m}$
$m_{NW,mean}$	-	Mass of the part of treadmill resting on top of NW load cell	$^{Vs^2}/_{m}$
s''_{NW}	-	North West load cell measurement conditioned twice (2nd order)	V
F_g	-	Force due to gravity and mass	N
$F_{g,a}$	-	Axial component of F_g (measured by load cells)	N
$F_{g,r}$	-	Radial component of F_g (not measured by load cells)	N

Section (4.3), Idle

Symbol:		Description:	Unit:
$T_{X,N,b}$	-	Translation from coordinate system N to b in X direction	m
$T_{Y,N,b}$	-	Translation from coordinate system N to b in Y direction	m
$T_{Z,N,b}$	-	Translation from coordinate system N to b in Z direction	m
$R_{X,N,b}$	-	Rotation from coordinate system N to b about X axis	rad
$R_{Y,N,b}$	-	Rotation from coordinate system N to b about Y axis	rad
$R_{Z,N,b}$	-	Rotation from coordinate system N to b about Z axis	rad
s'''_{NW}	-	North West load cell measurement conditioned three times (3rd order)	V
s'''_{NE}	-	North East load cell measurement conditioned three times (3rd order)	V
s'''_{SW}	-	South West load cell measurement conditioned three times (3rd order)	V
s'''_{SE}	-	South East load cell measurement conditioned three times (3rd order)	V
$s''_{NW,i}$	-	North West 2nd order conditioned load cell measurement number i	V
$s''_{NE,i}$	-	North East 2nd order conditioned load cell measurement number i	V
$s''_{SW,i}$	-	South West 2nd order conditioned load cell measurement number i	V
$s''_{SE,i}$	-	South East 2nd order conditioned load cell measurement number i	V
$s''_{ref,i}$	-	Reference (average) conditioned load cell value number i	V
s''_{i}	-	2nd order conditioned load cell measurement number i	V
G_i	-	Gain factor number i	$-$
G_{mean}	-	Load cell gain factor (individual for each load cell)	$-$
n	-	Total number of measurements	$-$
$G_{NW,mean}$	-	Load cell gain factor for NW load cell	$-$
s''_{NW}	-	North West load cell measurement conditioned two times (2nd order)	V

Section (4.4), Diagnose/Rehabilitation/Simulation

Symbol:		Description:	Unit:
s'''_{NW}	-	North West load cell measurement conditioned three times (3rd order)	V
s'''_{NE}	-	North East load cell measurement conditioned three times (3rd order)	V
s'''_{SW}	-	South West load cell measurement conditioned three times (3rd order)	V
s'''_{SE}	-	South East load cell measurement conditioned three times (3rd order)	V
$L_{X,aoa,p}$	-	Location of person in coordinate system aoa in X direction (-1 to 1)	$-$
$L_{Y,aoa,p}$	-	Location of person in coordinate system aoa in Y direction (-1 to 1)	$-$
n	-	Total number of samples	$-$
$L_{X,aoa,oa}$	-	Location of coordinate system oa in aoa in X direction (-1 to 1)	$-$
$L_{Y,aoa,oa}$	-	Location of coordinate system oa in aoa in Y direction (-1 to 1)	$-$
$L_{X,oa,p}$	-	Location of person in coordinate system oa in X direction (unscaled)	$-$
$L_{Y,oa,p}$	-	Location of person in coordinate system oa in Y direction (unscaled)	$-$
W_{aoa}	-	Width of available operating area	m
W_{oa}	-	Width of operating area (same as virtual board)	m

Symbol:		Description:	Unit:
L_{aoa}	-	Length of available operating area	m
L_{oa}	-	Length of operating area (same as virtual board)	m
$L'_{X,oa,p}$	-	Location of person in coordinate system oa in X direction (-1 to 1)	$-$
$L'_{Y,oa,p}$	-	Location of person in coordinate system oa in Y direction (-1 to 1)	$-$
$W_{p,tot}$	-	Total weight of person in load cell units	V
W_p	-	Current applied weight of person in load cell units	V
$R_{X,N,ve}$	-	Rotation from coordinate system N to ve about X axis	rad
F_p	-	Force from person on one sub-model	N
z_p	-	Displacement of sub-model's mass (and person)	m
z_{sim}	-	Displacement of sub-model's virtual surface	m
m	-	Mass of sub-model	kg
k	-	Spring constant of sub-model	$^N/_m$
d	-	Damper coefficient of sub-model	$^{Ns}/_m$
F_g	-	A symbolic force representing gravitational force	N
F_k	-	Spring force	N
F_d	-	Dampening force	N
F_a	-	Acceleration force	N
L_B	-	Length of virtual board (same as oa)	m
$T_{Z,N,sim}$	-	Translation from coordinate system N to sim in Z direction	m
$R_{Y,N,sim}$	-	Rotation from coordinate system N to sim about Y axis	rad
$z_{sim,n}$	-	Displacement of virtual surface for North model	m
$z_{sim,s}$	-	Displacement of virtual surface for South model	m
$T_{X,p,Fp}$	-	Translation from coordinate system p to force Fp in X direction	m
$F_{pg,n}$	-	Force from person on North sub-model	N
$F_{pg,s}$	-	Force from person on South sub-model	N
m_p	-	Mass representing a person (parameter)	kg
g	-	Gravitation acceleration, approximately 9.81	$^m/_{s^2}$
$z_{p,n}$	-	Displacement of North sub-model's mass (and person)	m
$z_{p,s}$	-	Displacement of South sub-model's mass (and person)	m
$T_{Z,N,p}$	-	Translation from coordinate system N to p in Z direction	m
$R_{Y,N,p}$	-	Rotation from coordinate system N to p about Y axis	rad
$R_{X,N,sim}$	-	Rotation from coordinate system N to sim about X axis	rad
W_B	-	Width of virtual board (same as oa)	m
$z_{sim,w}$	-	Displacement of virtual surface for West model	m
$z_{sim,e}$	-	Displacement of virtual surface for East model	m
$T_{Y,p,Fp}$	-	Translation from coordinate system p to force Fp in Y direction	m
$F_{pg,w}$	-	Force from person on West sub-model	N
$F_{pg,s}$	-	Force from person on East sub-model	N
$z_{p,w}$	-	Displacement of West sub-model's mass (and person)	m
$z_{p,e}$	-	Displacement of East sub-model's mass (and person)	m
$R_{X,N,p}$	-	Rotation from coordinate system N to p about X axis	rad
L_{SB}	-	Length of virtual skateboard (same as oa)	m
W_{SB}	-	Width of virtual skateboard (same as oa)	m

Symbol:		Description:	Unit:
I	-	Mass moment of inertia	Nms^2
$\ddot{\theta}$	-	Rotational acceleration about center of mass	s^{-1}
$\sum M$	-	sum of moment	Nm
I_A	-	Mass moment of inertia of object A	Nms^2
$\ddot{\theta}_A$	-	Rotational acceleration about center of mass of object A	s^{-1}
F_{cp}	-	Centripetal force	N
F_N	-	Normal force	N
h	-	Height to center of mass of object A from base (y-direction)	m
w	-	Width to center of mass of object A from base (x-direction)	m
$\ddot{x}_A$	-	Acceleration of object A in x-direction	$^{m}/_{s^2}$
$\ddot{y}_A$	-	Acceleration of object A in y-direction	$^{m}/_{s^2}$
$\sum F_x$	-	sum of forces in x-direction	N
$\sum F_y$	-	sum of forces in y-direction	N
m_A	-	Mass of object A	kg
$F_{g,A}$	-	Gravitational force acting on object A	N
ϕ	-	Acceleration compensation angle	rad
v_t	-	Tangential velocity	$^{m}/_{s}$
r_T	-	Radius of turn	m
ϕ_X	-	Acceleration compensation angle about X axis (roll)	rad
θ_X	-	Roll angle which includes acc. compensation and model output	rad
ϕ_Y	-	Acceleration compensation angle about Y axis (pitch)	rad
$a_{t,sim}$	-	Tangential simlation acceleration (fowared, backwards)	$^{m}/_{s^2}$
θ_Y	-	Pitch angle which includes acc. compensation and model output	rad
Z_{mo}	-	Board model translation output in Z-direction	m

Table of Contents

1. Introduction

The introduction encapsulates the motivation behind the development of this system, the situation of relevant current technology, what the system developed in this project is, and a brief explanation of the planning for this project. Furthermore, the composition of the thesis is presented at the end to enlighten the order of what is presented throughout the report.

1.1 Motivation

As more and more complex technologies arise, the possibilities of developing new ways of tackling existing challenges arise along with it. Today, smart gadgets such as lawnmowers, vacuum cleaners, dish-washers, lights, carports, and many, many more are being implemented in homes to enhance the quality of life. This is an immense aid for people with handicaps who struggle to do ordinary chores. However, another, more time-consuming and effortful method to enhance the quality of life, is training and rehabilitation [1]. This method, as opposed to buying assistive devices, tackles the problem at its core. Rehabilitation is proved by statistics to be rewarding to those who participate [2], but unfortunately, statistics also show that approximately one-third of those who is part of a program show low participation [3]. This could imply that the motivation for attending rehabilitation is not sufficient. After all, rehabilitation is time-consuming and requires great effort.

There is no direct way of making rehabilitation effortless and less time-consuming. However, the time spent can be used more efficiently and interesting for the patient, which could cause time-consumption and effort to be less of an issue. For example, during the process of relearning how to walk or run, a stationary treadmill within a room is often used nowadays which can feel boring and unnatural. A realistic environment with realistic movements could be beneficial, both for increasing the motivation of the patient and making a smoother transition into daily life. Hence, a tool for rehabilitation that provides the opportunity to emulate realistic scenarios and relevant movements could prove to be very motivational and beneficial in terms of the overall rehabilitation process.

This is relevant to several target groups such as people subjected to brain damage, stroke, spinal damage, amputees, muscle diseases, etc. Many of these people are in need of rehabilitating their ability to walk to reduce their struggle in daily life. Additionally, athletes may find this environment engaging and competitive compared to difficult training conditions in remote areas.

For example, Zwift is a program which allows bikers and runners to train in a virtual world, while using a stationary bike or treadmill. According to an article by Forbes written December 21st 2018, Zwift had at that time over 1 000 000 users, and had an estimated value of 180 million US dollars [4]. This indicates that engaging with a virtual world, and possibly other users, is motivating and keeps the user interested and invested in the rehabilitation or training program. A setup of Swift can be seen in Figure (1.1).

Figure 1.1: Illustration of Zwift indoor training [5]

By combining existing technologies, it is possible to emulate being in one place while actually being somewhere else. Using simulated environments properly, one can do the rehabilitation within controlled circumstances where monitoring and diagnosing are available, while the patient is wandering in the woods, surfing the waves, skiing downhill, or whatever is of interest within a virtual world. Also, since the circumstances are fully controlled, the difficulty can be adjusted to best suit the skill-level of the patient for optimal rehabilitation.

After a conversation with expert personnel regarding rehabilitation, there are a few main challenges regarding the technological progress within the field. It seems in a simulated environment, or in a game scenario, that the focus of the patient shifts from the objective of rehabilitation to the objective of winning the game. This may be counterproductive, causing the patient to develop undesirable features rather than developing them correctly. In future iterations of VR, simulations should reward the correct movement in addition to the pure objective of the game in order to ensure qualitative training. This will then have to include a real-time feedback system using camera sensors, providing information to the user regarding how they are doing.

Additionally, the use of sensors and actuators enables the possibility to store, process, and analyze information about the patient's condition and progress in a more efficient way as opposed to manual evaluation. Today, this information is manually gathered by visual observation and stored by taking notes [6]. By using a computer system for this information, progress and follow-up can be done more accurately and efficiently.

1.2 Current Technology

As mentioned, the current situation of lower limb rehabilitation involves mostly manual labor. With that being said, there are some pilot projects regarding the usages of treadmills combined with virtual reality models for rehabilitation purposes, where Motek's CAREN system being the most prominent.

CAREN, short for Computer Assisted Rehabilitation Environment, is an ongoing project by Motek where the simulation of virtual reality is a key feature used to enhance the efficiency of rehabilitation. By the use of a 6 degree-of-freedom (DOF) platform with an integrated treadmill, CAREN can imitate movements occurring in real-life scenarios. Also, with a camera system, real-time analysis of the biomechanical movement of the patient is possible. CAREN also utilizes a 180° screen to visualize the simulated environment. Figure (1.2) shows the CAREN system. More information can be found on Motek's home page. [7].

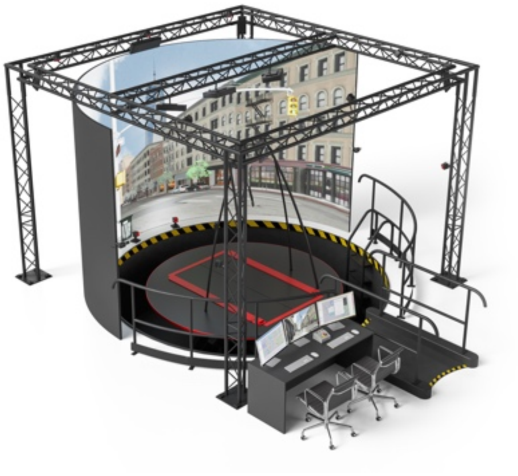


Figure 1.2: CAREN [8]

The CAREN system is a comprehensive and expensive system that makes it unobtainable by most in its current state. Instead, most of the institutions for rehabilitation relies on the more primitive method of visual inspection and auditory feedback by a professional. Such a method is relatively inexpensive and affordable by most, but it lacks the opportunity to vary settings in terms of environments while maintaining a fully controlled rehabilitation process. Moreover, the efficiency of such a manual method is highly dependent on the experiences of the therapist performing the analysis. Due to the inconsistency of similarities in knowledge and perception of humans, some therapists suit some patients better than others. Such inconsistency results in some patients having bad experiences with rehabilitation, and perhaps stop attending it. Therefore, developing a system resembling CAREN with some tweaks, is desirable. The main motivations being to make it more mobile and more accessible to facilities offering rehabilitation services.

1.3 CELNA

CELNA is short for "Computer-based Environment for Lower limb Neurorehabilitation and Assessment". The purpose of the system is to serve as a platform for diagnosis and rehabilitation within the realm of biomechanics, specifically aimed toward relearning how to balance, walk and run. By placing a treadmill on top of a 6 DOF platform, a scenario can be created where users can experience the same movements as in day-to-day life. This is done by implementing VR-goggles, where the user is engaged in a virtual reality whilst moving on top of the platform, interacting with the virtual world. Additionally, depending on the skill level of the user, the system can increase the difficulty in order to stimulate the user for efficient rehabilitation and training. Further, through motion capture, exact biomechanical data can be collected and analyzed, both for rehabilitation or performance enhancements for athletes. As the motion is being recorded, real-time data is transmitted to the operator and/or therapist, making it possible to present constructive feedback to the user for him/her to focus on in order to make progress. This information about the user's condition and progress can be stored, making it possible to monitor the progress of a user's ability to walk, run, or interact with certain objectives and challenges. Figure (1.3) illustrates the functionality of such a system.

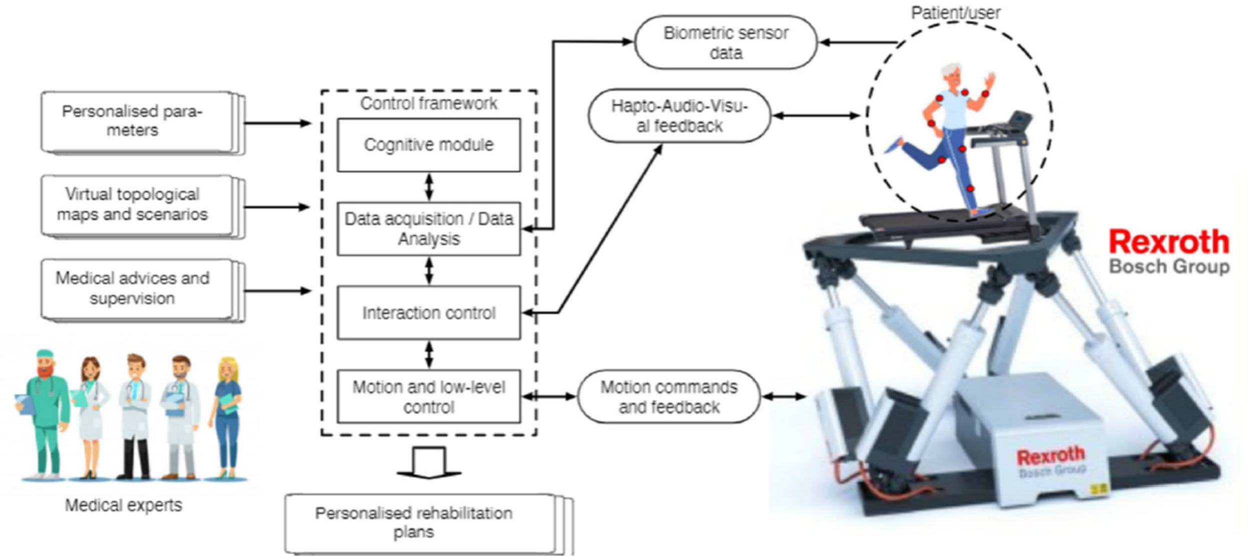

Patient/user
Biometric sensor data
Personalised para-meters
Control framework
Cognitive module
Hapto-Audio-Visu-al feedback
Virtual topological maps and scenarios
Data acquisition / Data Analysis
Rexroth
Bosch Group
Medical advices and supervision
Interaction control
Motion and low-level control
Motion commands and feedback
Medical experts
Personalised rehabilitation plans

Figure 1.3: Overview of a motion platform rehabilitation system

1.4 Societal Impact and Economy

Impact on Society

As mentioned previously in Section (1.2), similar systems are developed using slightly different approaches. The objective of this system is to be a semi-mobile system, which can be transported between different clinics. By doing this, treatment using this sort of technology can meet the patients where they are, reducing the negative impacts related to transportation of patients. Additionally, VR-technology has improved massively within the last decade, removing the need for a large screen and speaker systems. Everything can be contained within a pair of VR-goggles, without using external sensors. This, in conjunction with a smaller Stewart-platform, opens up the opportunity to produce a semi-mobile setup which can be transported between different medical institutions, dependent on needs.

According to the department of assistive technology at NAV, in 2016 $NOK\,2\,657\,942\,000$ was spent on assistive devices, equivalent to $NOK\,511$ per 1 000 inhabitants [9]. This is just on the assistive devices, excluding the money invested in rehabilitation and in nursing homes. By investing in more efficient rehabilitation devices, reducing the need for assistive devices, the annual costs would decrease significantly.

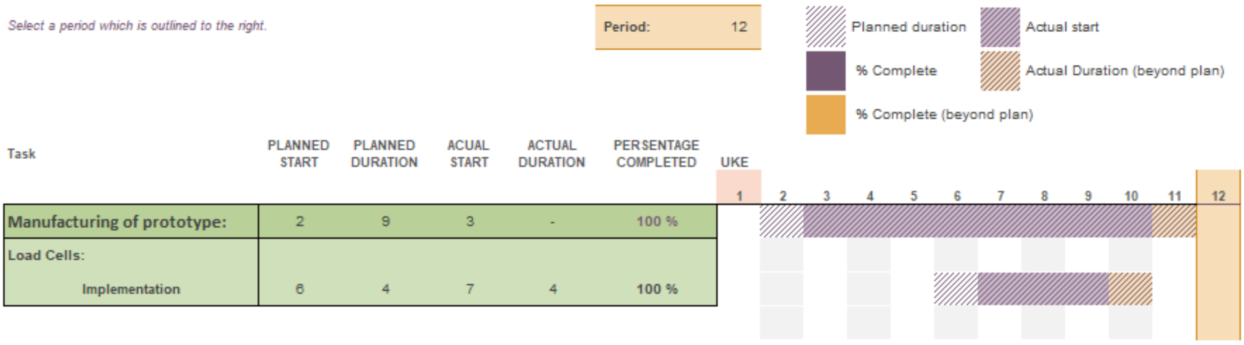
Economy and Cost

Since computer and entertainment system technology have progressed greatly the previous decades, the costs of such technologies have dropped. As an example, even though the Stewart platform used in this project is capable of lifting and manipulating 1500 kg, a smaller, less expensive platform is more than suitable. This is further the case when looking into the price of motion capture systems, treadmills, VR-equipment, and workstation computers. This has implications on the overall cost of a build like this. Since this is definitely a prototype system, using existing lab-equipment without using third-party services to deliver complete systems, a definite cost is difficult to compute. However, compared to existing products, a lower price tag is expected.

1.5 Planning

As with any long-term project, a structured plan is beneficial to keep track of the progress relating to short-term and long-term goals. As soon as the main objectives of the thesis were outlined, a Gantt-chart was developed to get an overview of milestones and the timeline. This serves as a guideline of when to hit certain milestones as a part of the main objectives. However, note that it serves only as a guideline, and is not considered as a schedule or deadline. It can be found in Appendix (A.1).

Figures (1.4) and (1.5) illustrate how the Gantt-chart works in excel, which is very intuitive and simple to work with along the project. Note that the figures only serve as illustrations, and is not a true representation of the true Gantt-chart. The tasks are planned and presented in the left-hand column, and the week-numbers from start to finish are written in a single row. Each task has a planned start and duration, and is entered in the spreadsheet. Actual start, actual duration and completion percentage are input as the tasks are being completed. Then, the spreadsheet visualizes the progress. This serves as a great tool in order to keep track of the progress in relation to the main objectives.

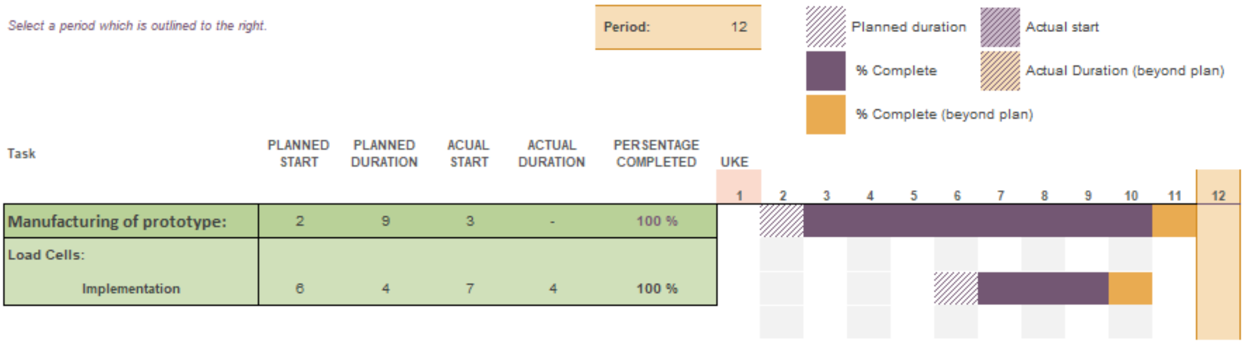


Select a period which is outlined to the right.

Period: 12

Task	PLANNED START	PLANNED DURATION	ACUAL START	ACTUAL DURATION	PERSENTAGE COMPLETED
Manufacturing of prototype:	2	9	3	-	100 %
Load Cells:					
Implementation	6	4	7	4	100 %

Figure 1.4: The Gantt-chart outline, using planned start and duration compared to actual start and duration



Select a period which is outlined to the right.

Period: 12

Task	PLANNED START	PLANNED DURATION	ACUAL START	ACTUAL DURATION	PERSENTAGE COMPLETED
Manufacturing of prototype:	2	9	3	-	100 %
Load Cells:					
Implementation	6	4	7	4	100 %

Figure 1.5: The same Gantt-chart outline, using the same plan, as 100% complete

1.6 Composition of Thesis

Throughout this report, the process of manufacturing a working prototype will be explained. First, an overview of the key components is presented, explaining the use and utility of all the different components. Then, the treadmill hardware modifications are described, which includes removing unnecessary parts, adding sensors and control units, along with the development of a safety frame for harness system support. Next, an explanation of the data acquisition and signal conditioning from each sensing component is presented. As there are many components to include in this system, defining each signal separately and determining how they should be obtained and utilized by the controllers is essential. After all components and signals are described and communication is established, the design of the modeling control follows. This encloses the approach in designing and development of haptic and impedance control for the cooperation between human, simulated environment and Stewart platform, and also velocity control for the treadmill. Then, everything is tied together to obtain a complete system for simulation, rehabilitation, and biomechanic diagnosis.

The results are presented before the discussion and recommendations for future work, while the concclusion is given in the end.

2. Background & Theory

This chapter will cover the technologies and functions which will be used as the basis for this project. This includes the neurorehabilitation process, the information regarding the hardware used for this build as well as the information of the interaction model used for the user experience.

2.1 Neuro-Rehabilitation Process

As humans experience injury, disease, or dysfunction a rehabilitation process is initiated in order to get patients functioning properly as soon as possible. This is done differently depending on the diagnosis of the patient, be it injury, stroke, disease, or other causes. Regardless, the movements have to be re-learned, through physical therapy and exercise. This is the job of physical therapist and doctors, working manually with patients. Figure (2.1) illustrates all the different components comprising a neurorehabilitation process.

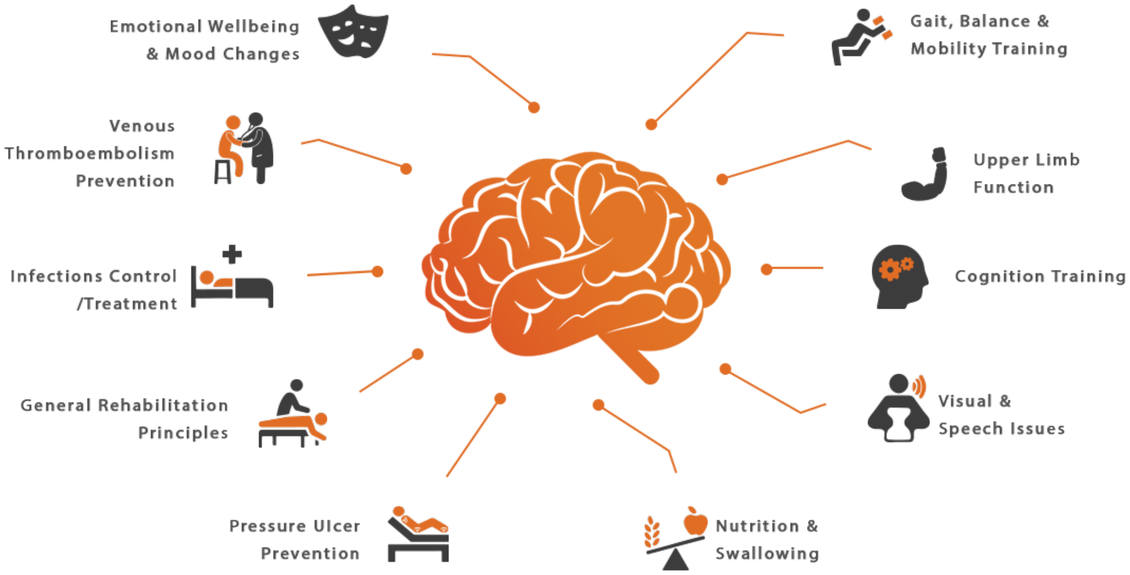

Emotional Wellbeing & Mood Changes
Gait, Balance & Mobility Training
Venous Thromboembolism Prevention
Upper Limb Function
Infections Control /Treatment
Cognition Training
General Rehabilitation Principles
Visual & Speech Issues
Pressure Ulcer Prevention
Nutrition & Swallowing

Figure 2.1: Stroke rehabilitation illustration [10]

A typical example is stroke rehabilitation, where the purpose of a systematic rehabilitation plan is to relearn skills by physical training and motion repetition. This is done to accomplish one of two things. Either to relearn the movement done before the cause of the disability or to learn a compensating motion which effectively does the same job. The former is typical for brain damage injuries, and the latter is typical for patients getting used to prosthetic limbs. There are several types of exercises available depending on the need of the particular situation, where some are very common. Motor-skill exercises, mobility training, constraint-induced therapy, and range-of-motion training are the most common rehabilitation exercises concerning the physical aspect of the process. Motor-skill exercises are used to help improve muscle strength and coordination. Mobility training aids the learning of using a prosthetic limb, canes, or ankle brace. This is often done during the process of relearning to walk. Constraint-induced training, also called force-use therapy, restrains an unaffected limb, while the affected limb is used to improve function. Finally, range-of-motion training is utilized to ease muscle tension and aids to regain range of motion. [11]

Most of these training exercises are currently carried out by a team of experts including doctors, nurses, physical therapists, and other medical personnel. Simplifying the physical tasks in this process by using new technology to identify and solve complex issues, the rehabilitation process can be improved not only for the patients but also for the medical staff. Using robotics in the rehabilitation process has two major purposes. Either to serve as a labor-saving device allowing for more rehabilitation without requiring additional staff or to provide therapy superior to that of conventional, physical rehabilitation. Figure (2.2) shows the type of manual therapy typically carried out during the manual rehabilitation process.

Figure 2.2: Traditional physical rehabilitation therapy [12]

In relation to this thesis, the objective is to use the knowledge of effective rehabilitation to enhance this process using new technology. As mentioned in Chapter (1), Introduction, using camera motion tracking, a treadmill, and a six-degree-of-freedom platform, the possibilities of significantly enhancing the rehabilitation process are great.

Lower limb rehabilitation is more complicated than upper limb rehabilitation due to generally more complex tasks. Balancing and walking require more refined movements and cooperation of several muscles and limbs. Additionally, lower limb strength and core strength is crucial in order to balance and walk properly. Compare that to the simpler movements often associated with upper limb motions, and the challenges become apparent. Typical balance exercises include the same basic principles. By making the surface on which the patient is standing or walking inherently unstable, the patient will have to continuously adapt in order to balance. By input from the virtual world or from the operator, perturbation can be introduced in the form of a jerk of the platform or slip. The purpose for this is to force the user to react to an unexpected event while standing or walking. This can be done for different skill levels by adapting the surface. As examples, for patient which suffers from a severe injury, a soft exercise mat might be sufficient, while a Swiss ball or balance board is more suitable for athletes.

Physical therapists will still have the main responsibility regarding the rehabilitation of the patients. As mentioned in Section (1.3), with the use of motion capture and load cell data, the therapist can obtain accurate data regarding the movements of the user. This includes joint placement/angle, pose and weight distribution, etc. These data can be used to diagnose the patient, and determine an individual plan for the patient. Rehabilitation is a process which has to be tailored to the individual as patients will react to treatment differently.

2.2 Technical functioning

To be able to simulate real scenarios in an artificial environment, more degrees of freedom (DOF) than what an ordinary treadmill provides are needed. By placing the treadmill on a Stewart platform, which has six DOF's (translation and rotation about three axes), one obtains the possibility to imitate the physical structure of an environment while moving through it. Then, by implementing the visual aspect of the simulated environment through VR-goggles, the feeling of actually being there is greatly improved.

By placing patients on this device, in a known artificial environment, a diagnostic analysis can be made about the biomechanical condition of the patient. If the patient is not responding as expected, either due to physical disabilities, coordination, or cognitive attributes, a diagnosis can be made. From there, a specific rehabilitation plan can be developed according to the patient's needs and desires. Figure (2.3) illustrates the concept of a treadmill mounted on top of a Stewart platform. On this treadmill, safely within a harness system, the patient is placed for training or rehabilitation purposes.

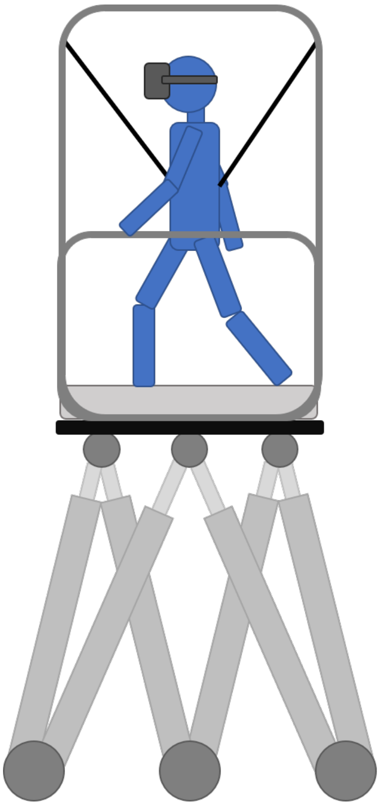


Figure 2.3: Concept illustration

The main advantage of a platform like this one, regarding the rehabilitation process, is the ability to simulate a variety of scenarios. This allows the patient to walk, run, or surf through different artificial environments, testing all coordinate abilities. According to a previous study regarding rehabilitation using a virtual reality [13], this type of platform shows promising results regarding the improvement of the gait of the patient. This study shows that increased motivation contributes to the willingness of the patient to relearn.

By using motion tracking technology, extremely precise measurements can be made regarding the gait of a person. This allows for corrections in walking patterns for rehabilitating patients, but can also be used to enhancing a current skill for sports athletes since fine adjustments can be made based on accurate points from the motion capture system.

There are typically two simulation modes for such a system - balance mode and walking/running mode. What separates these two modes is the fact that in balance mode, an object such as a board is located beneath the user's feet, whereas in walking/running mode the user us located directly on top of the terrain. Examples of "balance mode" might be snowboarding, surfboarding, skateboarding, balancing on a balance board, standing in a kayak, etc.

2.2.1 Balance mode

As mentioned, balance mode concerns the scenarios where the user is virtually located on top of a board-like object. Depending on the scenario, the user is able to move through the simulated environment by shifting his/her weight around on the virtual board. When doing so, the Stewart platform should respond in a relatively expected manner to emulate the interaction between the user, the ground surface and the board.

Figure (2.4) shows the front view of the system when in balancing mode. As a starting position, shown left in the figure, the platform is flat and the patient's center of gravity is right above the center of the treadmill. As the patient starts to move his/her center of gravity, the platform should angle respectively in order to represent the movement of the board. Of course the virtual surface orientation affects the platform orientation as well. In the figure example below, the virtual surface is flat.

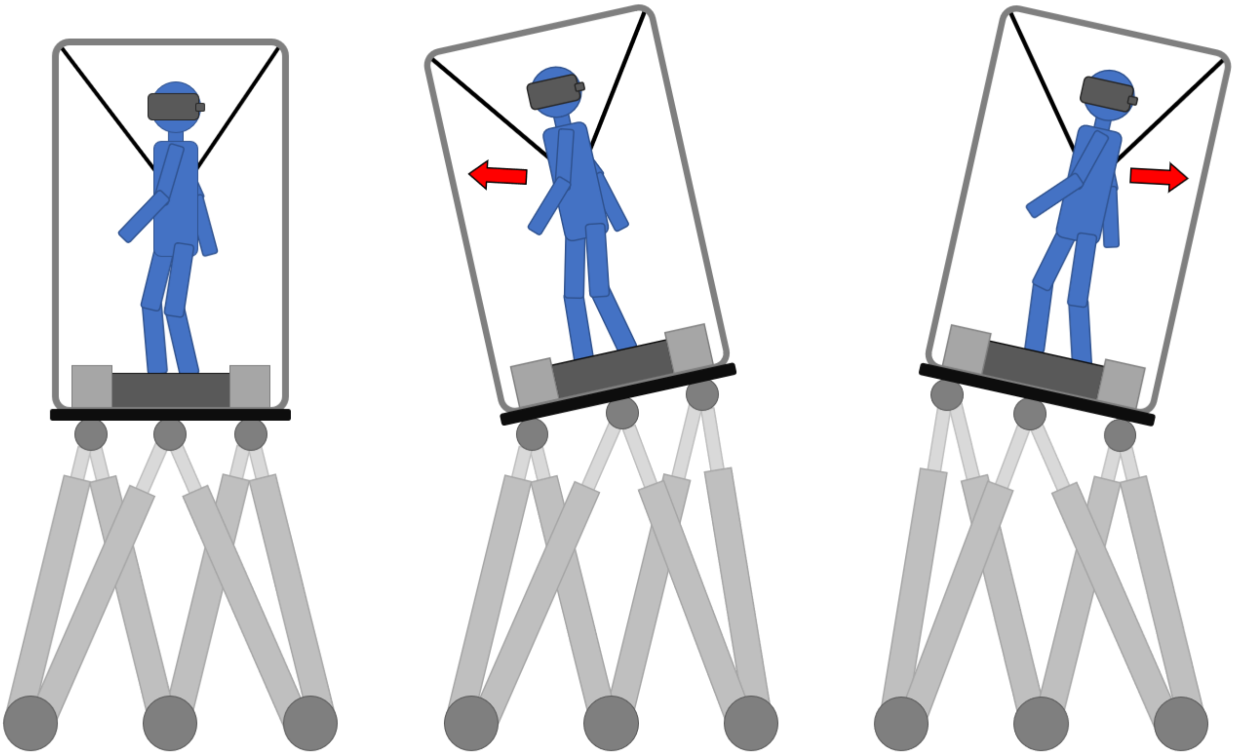


Figure 2.4: Sideways interaction between the patient and platform

As illustrated in the figure, the patient is positioned sideways on the treadmill as if standing on a skateboard. In order to turn the skateboard to the right, the patient leans forward. Doing so makes the platform tilt toward the same side in order to simulate a skateboard. Simultaneously, in virtual reality, the avatar should turn right while moving over the terrain. Conversely, when the patient leans backward, the platform should simulate a skateboard performing a left turn.

2.2.2 Walking mode

A separate mode is made for the walking simulation. In a real scenario, the person walking determines his/her own walking pace. To be able to emulate this freedom of pace control in a simulated environment, the speed of the treadmill will have to self-adjust based on the user's pace. This can be done using motion tracking by locating the user and controlling the velocity of the striding belt based on his/her position relative to the belt surface length. As the usable belt area is restricted, having the user centred on the belt is essential for safety and flexibility. As a result, the walking mode velocity controller will strive to keep the user at the center of the treadmill belt at all times by controlling the integrated AC-motor.

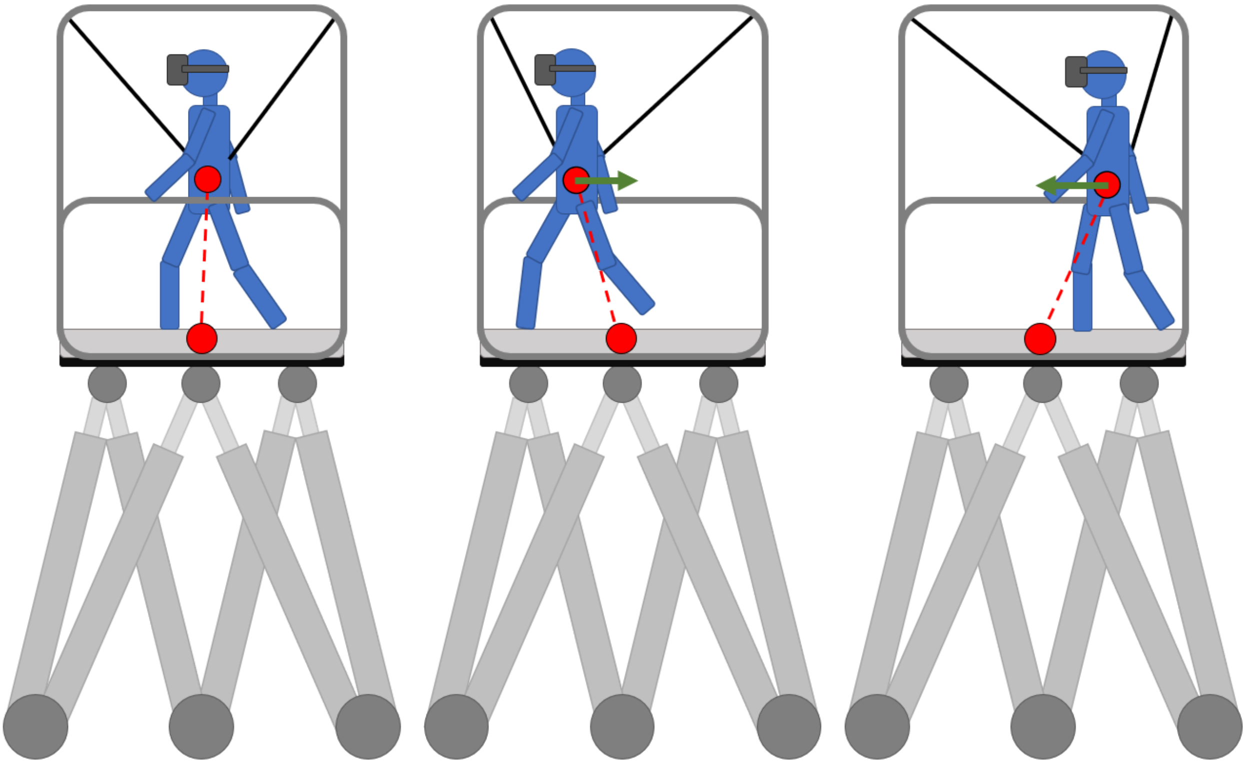


Figure 2.5: An illustration of the walking mode control principle

2.2.3 Data and Information Flow

The whole system can be divided into four essential elements:

- Human
- Haptic interface/mechanical system
- Virtual coupler/suspension model
- Artificial environment/topography model

By dividing the system into these elements, a visualisation of the information flow can be easily described. A flow chart can be seen in Figure (2.6) with a descriptive text below.

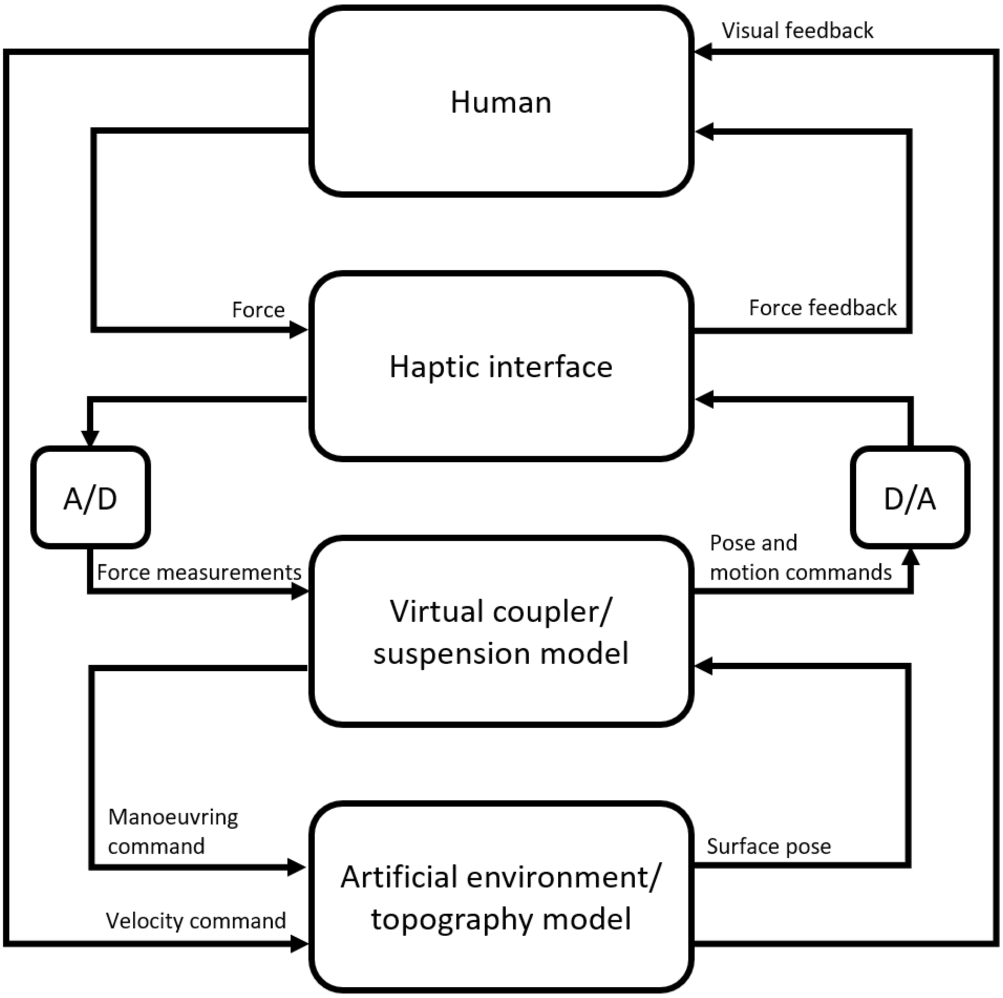


Figure 2.6: Information flow

The human has two main outputs: a velocity command and a force impact due to his/her weight distribution on the treadmill. The force is applied to a haptic interface, which is the mechanical contraption consisting of sensors and actuators, i.e. the treadmill unit mounted on the Stewart platform. This force is measured and used as one of the multiple inputs to a virtual coupler/suspension model. The virtual coupler/suspension model also gets inputs describing the topography of the artificial environment from the artificial environment/topography model. The virtual coupler/suspension model is the element containing a mathematical model that describes the real-time dynamic behaviour of the interaction between the human's feet and the surface below. In other words, it is the coupling between the virtual world and the real world. From this model, pose and motion commands are sent to the haptic interface, i.e. the Stewart platform and treadmill, which applies a force on the human as the pose changes. Simultaneously, a maneuvering command is sent back to the artificial environment/topography model in order to interact with the virtual world when for example tilting a skateboard to make it turn. If applicable for the mode, a velocity command can be sent directly from the human to the artificial environment/topography model in order to set the velocity within the virtual world. Lastly, of course, visual feedback of the virtual world is received by the human directly from the artificial environment/topography model through VR goggles.

Haptic interface and virtual coupler/suspension model are part of the terminology haptic technology, which is explained more in-depth later in this chapter. The artificial environment/topography model is also addressed later in this chapter.

2.3 Haptic Technology

Haptics is the science of touch, i.e. how we interface with the world. The way we distinguish shapes and material properties through touch in addition to the feeling of resistance when pushing against a yielding structure such as a branch falls within this category. To recreate this sensation of touch by the use of technology such as a mechanical structure with actuators is referred to as haptic technology. Haptic technology is an essential term to address as it is exactly this type of technology that is relevant for a system such as CELNA.

An important feature of CELNA is to emulate a virtual reality in a logical and intentional manner. As a person walks across surfaces with different material properties, the mind creates an expectation of how the surfaces will feel beneath the feet based on experience. As an example, you know from experience that grass feels softer to the touch than concrete. Therefore, it is important to distinguish these materials within the artificial environment as well, which can be done by the use of haptic technology.

A haptic controller is delivering force feedback to the user through a haptic interface by the use of a physical actuator, also called 3D-touch. A classic example of this is the vibrating mechanism in handheld controllers such as Playstation DualShock controllers, using unbalanced weights to vibrate the controller [14]. Haptics have several purposes in different industries. For instance, to control robots in environments unfit for humans, for medical practice in simulators, or for 3D-drawing. A typical information flow for haptic technology can be seen Figure (2.7). Within the term haptics, there are two main types of feedback - kinestethics and tactile. Tactile feedback is feedback related to the feel of touch [15]. For example, the difference between the feel of grass compared to concrete. Kinesthetic feedback is related to the force feedback experienced when moving objects. For example, compressing a spring will increasingly apply a force the more the spring is compressed. In this system, kinesthetics are utilized.

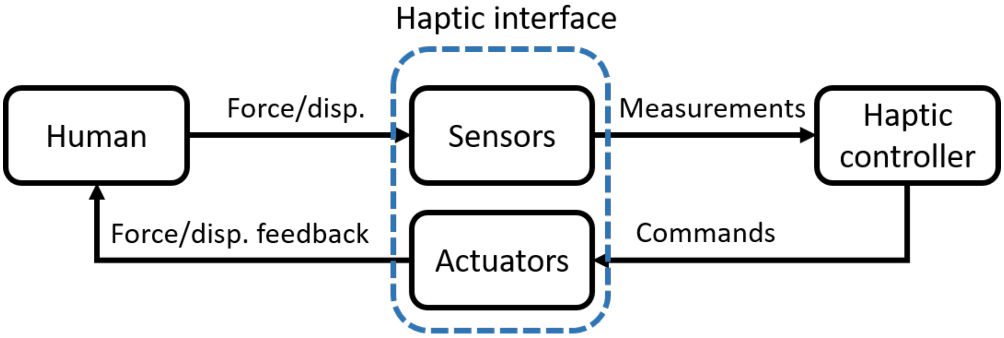


Figure 2.7: Haptic technology information flow

Studies suggest that rehabilitation including robotic therapy is potentially superior compared to conventional methods [6]. Additionally, robotic therapy can be utilized in different ways to suit the needs of the patient. By manipulating the actuators, the robotic element can be set to actively assist the patient in order to rebuild the muscle-memory of a movement. If the patient has made some progress, the robotic element can be set to passive, neither assisting or resisting the motion. Lastly, if the patient is in a state which requires exercise, the robotic element can be set to actively resist the motion of the patient in order to increase strength.

Most of the development of robotic therapy has been done in relation to upper body rehabilitation, especially arms and hands. This has something to do with the size and complexity needed of the apparatus, but also due to the generally simpler task of grasping and manipulating objects compared to walking. Using robotic therapy in relation to lower limbs is therefore rarer due to the more complex tasks of walking and balancing own body weight at the same time.

There are mainly two types of haptic control strategies within the kinesthetic realm - impedance control, and admittance control. The former controls the force feedback depending on the motion and displacement of the user input in regard to the simulated environment. Conversly, admittance control controls the motion and displacement feedback depending on the force applied by the user, also in regard to the simulated environment. The names, impedance and admittance control, originates from the electricity domain and Ohm's law, stating that for a circuit of alternating current, Equation (2.1) applies.

$$V = I \cdot Z \tag{2.1}$$

Where V is voltage, I is current and Z is impedance. Admittance Y is formulated as:

$$Y = \frac{1}{Z} = \frac{I}{V}$$

$$\implies I = V \cdot Y$$

As an example, Force F for a spring can be described as displacement ΔX times spring constant K.

$$F = \Delta X \cdot K$$

By assuming that voltage is analogous to force, and the current is analogous to displacement, one can make out that for impedance control, the force feedback is determined by the displacement input times system impedance. For admittance, the displacement feedback is determined by the force input times system admittance. Impedance control and admittance control are related as being inverse (or reciprocal) to one another.

2.3.1 Impedance control

A flow chart describing the information flow in a general impedance controlled haptic interface is shown in Figure (2.8).

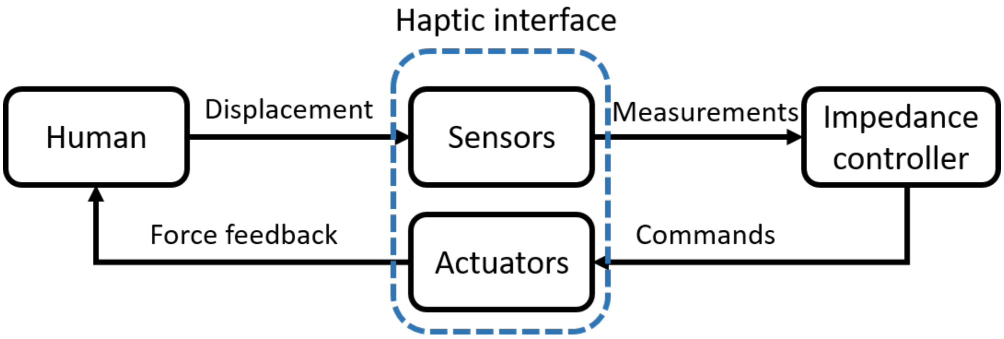


Figure 2.8: Block diagram of a impedance controlled haptic interface

An impedance control system determines output force based on input motion. Figure (2.9) illustrates the relationship between input motion and output force. If the human input, in this case, is downward motion causing a displacement of the end-effector of the robotic arm, the actuator is expected to generate an opposing force to the human depending on the amount of displacement. This is assuming that a human is working against a robotic actuator as shown in Figure (2.9). A different scenario is expected when a robotic arm is controlled using a joystick or similar inputs. Depending on the environment in which the actuator is placed, the force feedback to the user will vary. For example, the force acting on the arm of an excavator will be very different when moving through sand as opposed to rocks, thus the force feedback to the user will also vary. By adding a velocity-dependent reaction force to the displacement dependent one, a damped resistance is obtained. It should be mentioned when a response depends on spring and damper coefficients, terms such as eigenfrequency, damping factor, etc. must be accounted for in order to make a safe and reliable system. Impedance control is typically used for robotic arms which operate against admittance objects, i.e. masses. How the robotic arm responds is very dependent on the properties of the given mass. For example, the manipulator of a marble placed on a table will behave differently to a manipulator moving logs on water.

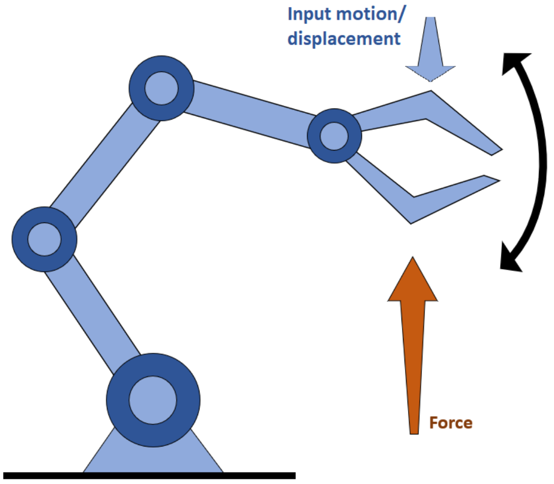


Figure 2.9: Illustration of how an impedance controlled robotic element behaves

Impedance control can be used in simulating a multitude of daily scenarios within artificial environments. An example of this can be the simulation of an object moving freely within a confined space such as a room surrounded by walls. Illustrated in Figure (2.10) is the force/displacement relationship that occurs in an instance where the object is freely moved and suddenly interacts with an infinitely stiff wall. Figure (2.11) on the other hand illustrates an impedance controlled haptic interface emulating the force of a linear spring relative to its displacement.

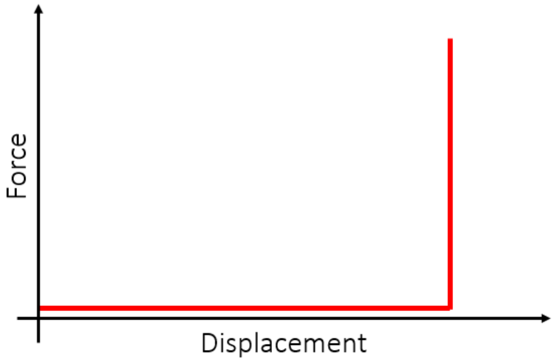


Figure 2.10: Impedance control limited space example

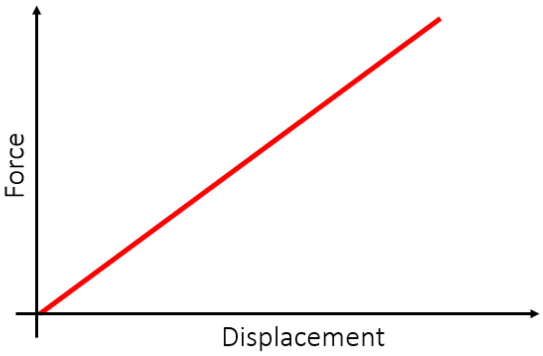


Figure 2.11: Impedance control linear spring example

Most springs are linearly proportional to the displacement of the spring, as described in Equation (2.2).

$$F = \Delta X \cdot K \qquad (2.2)$$

2.3.2 Admittance control

The other form for haptic controller - admittance control, determines the robotic motion and displacement depending on the input force from the operator. An example of admittance control is when the velocity, or motion, is proportional and in the same direction as the force input, i.e. greater force applied - greater velocity. A flow chart describing the information flow in a general admittance controlled haptic interface is shown in Figure (2.12).

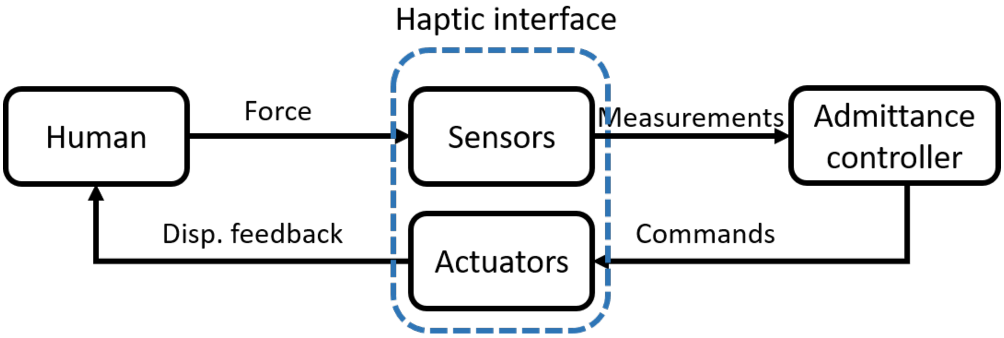


Figure 2.12: Block diagram of a simple admittance control system

Figure (2.13) shows the relationship between input force and output motion. When the input force from the operator increases, the output motion of the robot end-effector increase as well. As with impedance control, the system will have to be regulated in order to function safely and well with human interaction.

Examples of this are flight simulators and exoskeletons. As the user applies a force to whatever the input interface is, the actuators should adapt to reach the reference position. In recent industry this is used for manipulating robotic arms, in various fields.

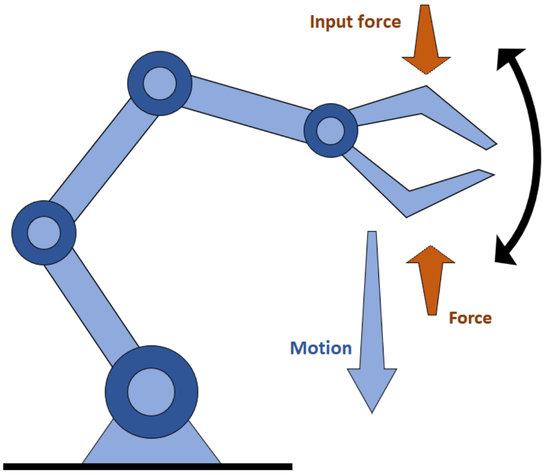


Figure 2.13: Illustration of how an admittance controlled robotic element behaves

In this project, an admittance controller will be used to simulate the sensation of balancing on a board. The load cells placed in each corner of the treadmill serve as force sensors to estimate the center of gravity of the user and the applied weight. These data are then used as inputs to a board model that puts out the desired pose and dynamics of the haptic interface (Stewart platform and treadmill) in order to emulate the virtual board. A board model falls into the virtual coupler category, as its main task is to couple the virtual world with the real world through a mathematical model. Figure (2.14) shows the I/O (input/output) relationship of the virtual coupling. It receives information about the topology from the artificial environment and information about the force applied by the human. All this data enters a MIMO (Multiple Input Multiple Output) system (board model) and a desired pose and motion for the haptic interface is generated. The orientation of the modeled board is also used to send commands to the VR software in to gain a two-way interaction between simulated environment and human.

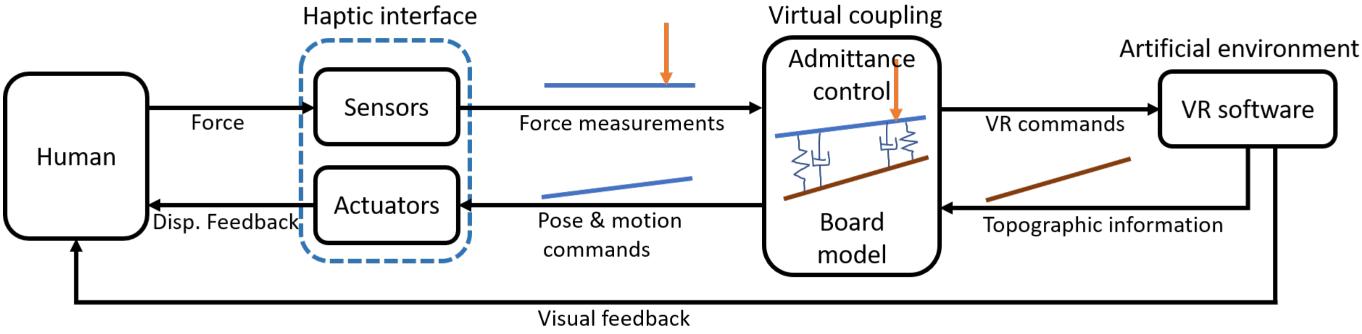


Figure 2.14: Virtual coupling I/O

2.3.3 Haptic Rendering

Haptic rendering is described as the process of computing and generating forces in response to human user interactions with virtual objects. The name comes from the Greek word

"Haptesthai", meaning "to touch". Haptic is to touch as visual is to seeing and what audity is to hearing. It was generated to interact with objects in a virtual world as if they were real. Previous studies show that the ability to touch virtual objects increases the sense of presence in simulated environments. As real-world examples, when we eat with a fork, write with a pen or open a lock with a key, we are moving an object in 3D and we feel the interaction with other objects. In essence, this is 6-DOF manipulation with force and torque feedback [16]. Figure (2.15) illustrates the typical flow chart regarding a haptic model.

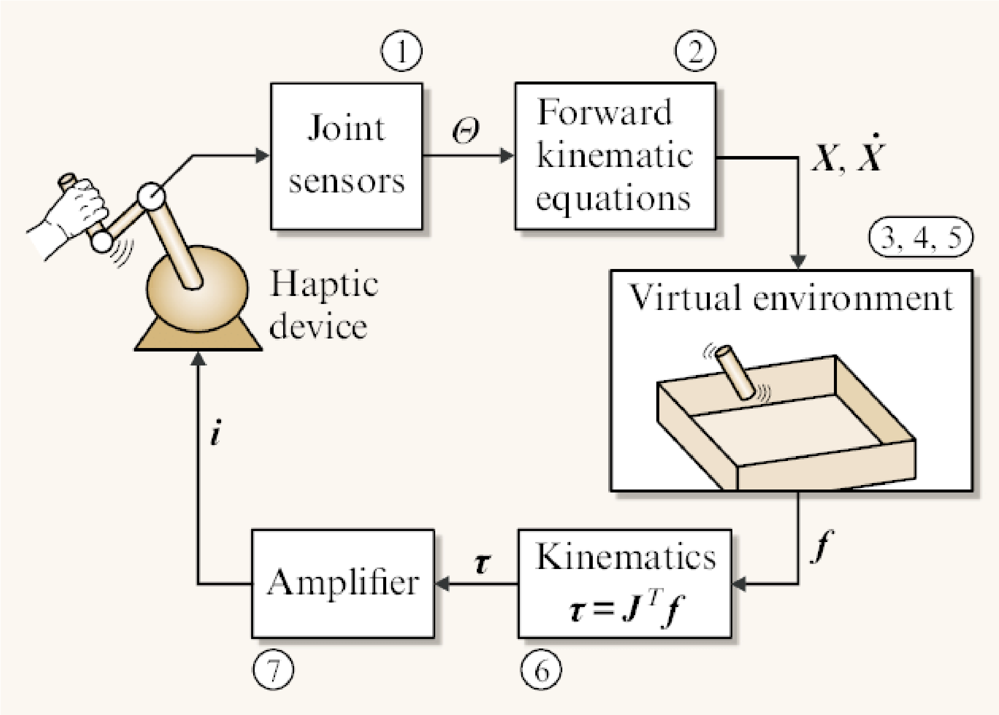


Figure 2.15: Diagram illustrating the process of haptic rendering [17]

There are two types of haptic rendering, 3-DOF rendering, and 6-DOF rendering. 3-DOF rendering is manipulation in 3D-space, while 6-DOF rendering uses 3D position and orientation in addition to the force and torque feedback to the user.

Two main tasks are comprised of haptic rendering. One is the computation of the position and/or orientation of the actuator. The other is the computation of contact force and/or torque of the actuator. These tasks are processed in two main groups called *Direct rendering* and *Virtual coupling* [18].

In direct rendering, the position and/or orientation are directly applied to the haptic device. In order for the virtual scenario to be consistent with a real scenario, the model for collision detection is performed between the virtual manipulator of the user, and virtual objects. Either done as object separation, often hard contact for most usages or as a model using penetration depth, often used in medical purposes.

Using virtual coupling, the position and/or orientation is set as goals for the haptic device, followed by a viscoelastic force applied in the actuator. That coupling force combined with collision detection is used to compute the position and/or orientation of the probe. The same coupling is used for force and torque feedback.

2.4 Artificial environment

An artificial environment in this context, is a simulated environment developed by the use of a game engine. The point of incorporating an artificial environment is to both visually and physically emulate scenarios of the real world while rehabilitating. The topography of the artificial environment is used to influence the virtual coupler, thus also the response of the Stewart platform in order to enhance the immersive feeling of simulation. In other words, it is used to emulate the sensation of touch through haptic technology. An illustrative mesh of a typical artificial environment with varying topography is shown in Figure (2.16).

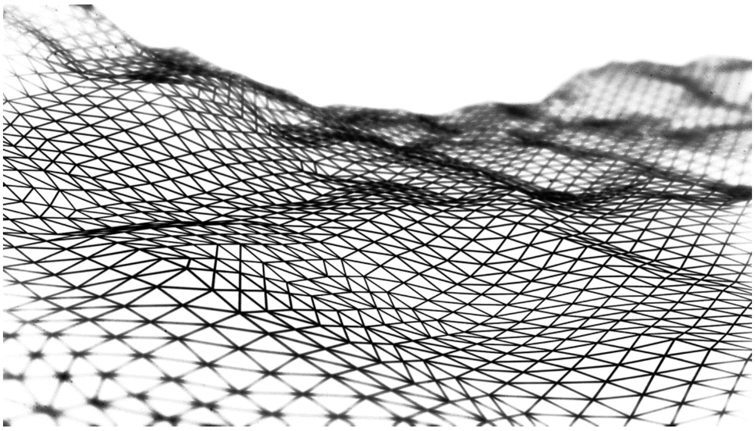


Figure 2.16: Terrain mesh [19]

Certain applications benefit from incorporating an artificial environment, and some applications even depend on it to work as intended. CELNA is a system with multiple areas of operation, where some types of operation require an artificial environment to be visualized for the user. If the user is to snowboard or skateboard down a hill consisting of obstacles and uneven topography, it is necessary for the user to visually see the environment in which they are moving in order to prepare themselves for the approaching obstacles. There are types of operations where it might be beneficial to not include any artificial environments, such as during perturbation training. Perhaps some randomly generated bumps are applied which the user can not prepare for in order to engage and improve reflexes. Everything depends on the needs of the user to make biomechanical progress. However, since there are some operations that require an artificial environment and a visualization of it, it must be addressed.

2.4.1 Type of environments

The number of different environments that could be used for this application is numerous, but some are more suitable than others. There are a few key points that are important to consider in the selection of a suitable environment for rehabilitation and training:

- **Familiarity of scenario.**
  During the process of rehabilitation, it might be inconvenient for the user to adapt to a totally unfamiliar scenario while rehabilitating. Although it should not be expected that the user has any direct experience with it, some familiarity with the working principle goes a long way

- **Complexity of tasks.**
  A considerable amount of the potential users of this system are stroke patients, which may have been subjected to a reduction in cognitive abilities. This may affect their

ability to process information and learn new things. Therefore, it is important to include relatively simple scenarios. Of course, this does not necessarily mean that some types of scenarios have to be discarded from others, as long as they are internally adaptable, which brings us to the next point.

- **Adaptive scenario.**
  Patients in need of rehabilitation have different degrees of handicap. It will not be sufficient to create one scenario, or map, that is identical for all patients. It is crucial that the environments and scenarios are adaptive in terms of difficulty, i.e. it should be possible for the operator, or therapist, to select a suiting difficulty for any user.

- **Variety of tasks.**
  To contribute to the patient's motivation, a variety of different tasks should be available. It is beneficial to have the opportunity to include several tasks of different difficulties, where the user gets notification feedback of the accomplishment of any task. A task could be to complete certain routes, collect objects, or perform certain movements.

- **Fun factor.**
  Another contribution to the patient's motivation is the fun factor. The fun factor is heavily dependant on the interests of the user, but a study done by Nicole Lazzaro from Big Think shows that there actually exists a common understanding of what makes a game fun[20]. Software generated artificial environments with tasks and scenarios can be considered a game, which is how this study is relevant for this project. According to the study, the fun factor of a game can be divided into four categories; Easy fun, hard fun, serious fun, and people's fun. The main type of fun for this application is serious fun. This is the type of fun people get from learning new skills and improve existing skills, achieving personal goals affecting their daily lives. Easy fun on the other hand is the fun people get from exploring environments, mechanics, and general basic features within the game. This suggests that the user should have the ability to free-roam over a larges space within the environment. Perhaps the addition of a few easter-eggs and surprises can be beneficial. Hard fun is the fun people get from achieving something that required repeated tries and failures. This is not a priority in the first place, as repeated failures and frustration might prove counterintuitive when rehabilitation is the main objective. With that being said, the task should prove challenging, but accomplishable. People's fun is the fun people get from social bonding, which is not a priority either. Conclusion: The artificial environment should have the possibility for serious fun, easy fun and some degree of hard fun for optimal contribution to the patient's motivation and improvement.

Following is a couple of examples of environment and scenarios that matches the previously listed key points.

Snowboarding

Snowboarding is a commonly known sport and takes place in an environment where exploring large landscapes, completing multiple different objectives, and selecting suiting difficulty is possible. In addition, the stiffness of snow can range from soft to hard depending on its condition. This makes it very suited for balance training as the stiffness can be changed without compromising the realistic feeling. Figure (2.17) is a screenshot of the landscape from the famous game SSX 3, showing a typical snowboarding scenario with varying snow density, trees, bumps, and jumps.

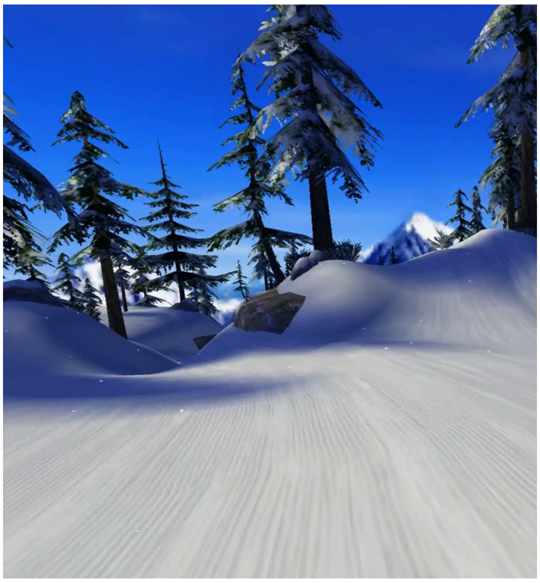


Figure 2.17: SSX 3 landscape [21]

Skateboarding

Like snowboarding, most people have some familiarity with skateboarding. Either it is from riding or observing others ride. The working principle of a skateboard is known. A skateboard can be ridden on a variety of materials, as long as it is hard. Therefore, the environments and scenarios available to skate within are vast. The stiffness of the trucks can be adjusted on a real skateboard, which gives the opportunity to adjust the difficulty in balance to suit the patient. This scenario yields the opportunity to demount the skateboard and walk around within the same environment. Figure (2.18) is a screenshot from True Skate which is an Android game, showing a typical skateboarding scenario - on top of a ramp. However, it could be anything from a city street to a country road.

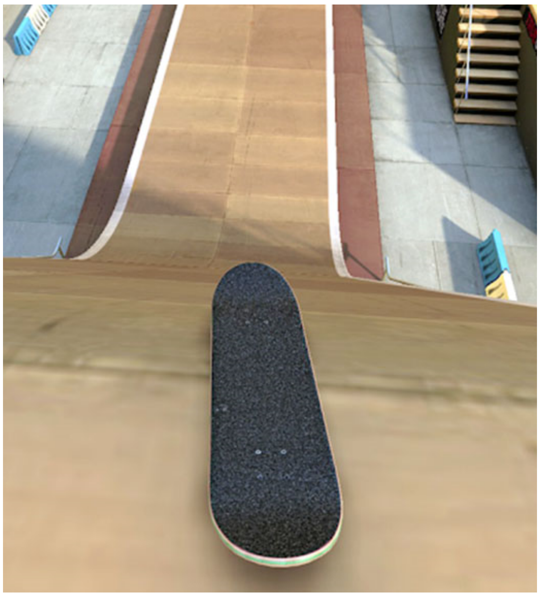


Figure 2.18: True Skate environment [22]

2.4.2 Interaction between Artificial Environment and Virtual Coupler

Topographic information

In order to couple the software-based artificial environment with the haptic interface, specific topographic information is required from the software. It is desired that the dynamic behavior of the interaction between the human and the virtually represented material, for instance, snow, is generated outside the software and not within. In other words, the software executing the visual part of the simulation should only generate data describing the pose of the surface right beneath the virtual location of the person. To recall, Figure (2.19) is the flow of information between the human and the artificial environment, where the topographic information flow is highlighted.

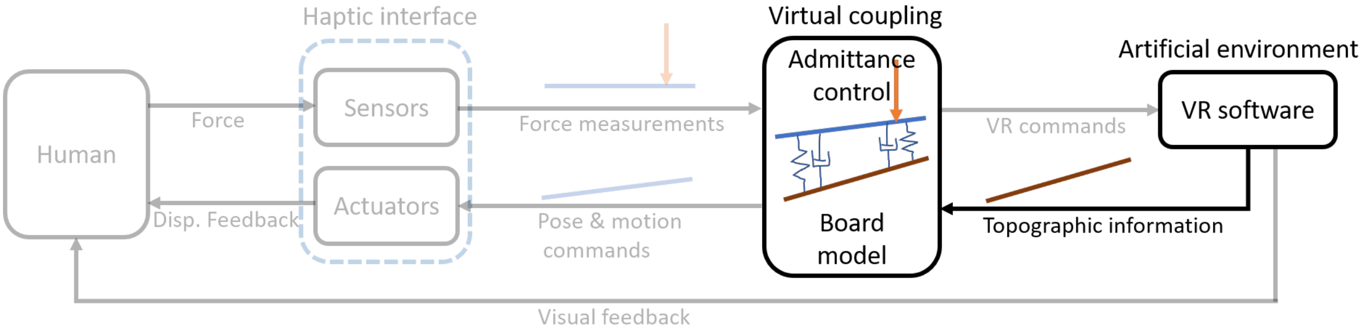


Figure 2.19: Topographic information flow highlighted

The mathematical model describing the dynamic behaviour of the material on which the person is virtually located, is part of the admittance controller. By doing so, the core functioning of the haptic interface (Stewart platform and treadmill), which is generating dynamic behaviour, is still available even without a software that generates an artificial environment.

There are three main variables that are crucial to know from the virtual pose of the surface beneath the person. See Figure (2.20) for reference. Note that coordinate system [x,y,z] is the coordinate system of a point in the surface the board is interacting with. The board and person are added for illustrative purposes.

- Heave (Translation i Y)
- Roll (Rotation about X)
- Pitch (Rotation about Z)

Yaw is not necessary as it does not affect the haptic sensation beneath the persons feet. Surge and sway are not needed since information about where in the surge-sway plane the avatar is located relative to the origin is irrelevant when the physical person is operating in the same spot.

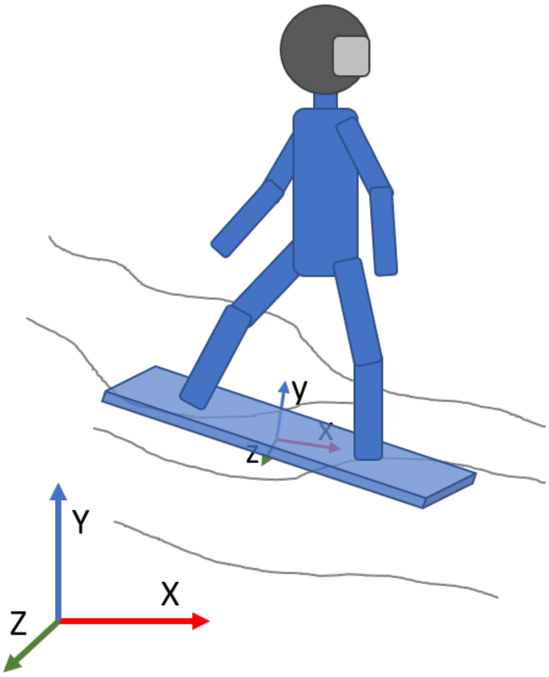


Figure 2.20: Topography coordinates

VR Commands

The topographical information transmitted to the admittance control connects the user to the virtual reality in a one-way type communication. This makes the actions executed within the VR generating software independent of the actions of the user. In order to enable the user to interact with the artificial environment, VR commands have to be generated. Similar to the topographic information, an illustration of the VR command flow can be seen highlighted in Figure (2.21).

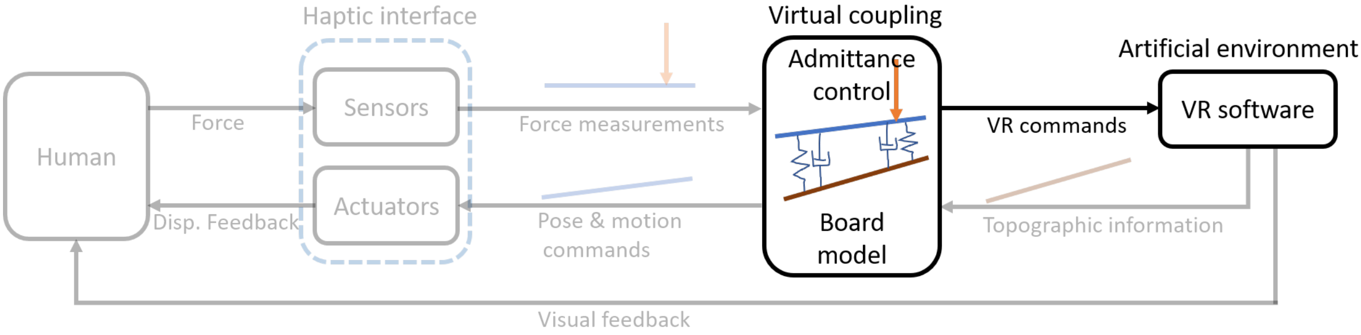


Figure 2.21: VR command flow highlighted

Typical VR commands can be velocity control, maneuvering control, or button interaction. Based on a dialogue with game developers using the Unity engine, velocity in all six degrees can be used as control signals. Velocity control is the most common way of moving a character about within a virtual world by the use of a controller, such as a PlayStation controller or a keyboard. The controlled velocities are in the direction of the characters coordinate system, i.e. velocity in x-direction is always forward regardless of the characters orientation relative to the virtual world's global coordinates. Virtual velocity control is ideal to use for this application.

Figure (2.22) illustrates two scenarios - snowboarding and skateboarding. In both scenarios, the topography is similar and both avatars are moving parallel to the YX-plane, i.e. they are not turning. Notice how the boards are angled differently. For a skateboard to turn, the board must have an angle relative to the axial vector between the wheels. Assuming that the axial vector between the wheels has the same orientation as the surface below, which is true as long as there is contact, the turning rate of a skateboard depends on the board's orientation relative to the surface. On the other hand, the turning rate of a snowboard is assumed to be related to the angle between the board and the ZX-plane. Although not entirely correct as it also has to do with pressure distribution between the rear and front leg, it is considered a viable assumption.

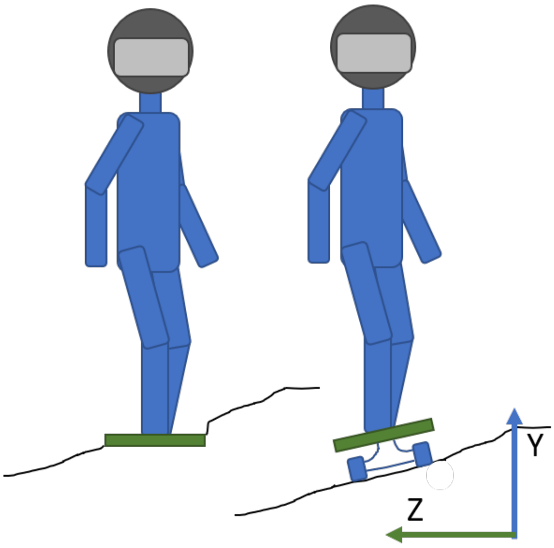


Figure 2.22: Snowboard vs. skateboard

The turning rate is the rate of rotation about the Y-axis. This particular parameter is a needed addition to make the user feel like a part of the environment. Forward and backward velocities, meaning the virtual velocity in the direction of the board, is also a parameter that could be controlled by the user. Realistically, however, the velocity is not directly controlled when riding a snowboard or skateboard but rather influenced by actions taken while riding, such as skidding and/or performing a slalom-like movement down a hill. Therefore, the velocity control is not depending on the model and board angle but is rather directly controlled by either the user or operator by a parameter control.

2.5 Method for Emulation of Acceleration

Newton's first law states that an object moving in a straight line will remain going in a straight line unless subjected to external forces that act to change the motion. Usually, for humans riding bikes, skateboards, snowboards, motorcycles, and other types of vehicles, this external force comes from the ground beneath.

As an example, in order to make a skateboard turn, an external force acting from the ground on the wheels must be generated. This is done by tilting the board which causes an angle between the axles of the wheels, as illustrated in Figure (2.23). F_p is the force from the person on the board, F_N and F_{cp} are the normal and centripetal forces, respectively, from the ground on the wheels. NW, NE, SW, and SE describe the locations of the wheels on the board. When the force from the person on the board is shifted to the right, the trucks are designed to generate an angle between the axles, thus making the board turn. Due to friction between the ground and the wheels, a force is applied to each wheel parallel to its axles. These friction forces combined causes an acceleration of the skateboard toward the center of the turn, also called radial acceleration. The yellow line illustrates the resulting trajectory of the board when assuming no slip.

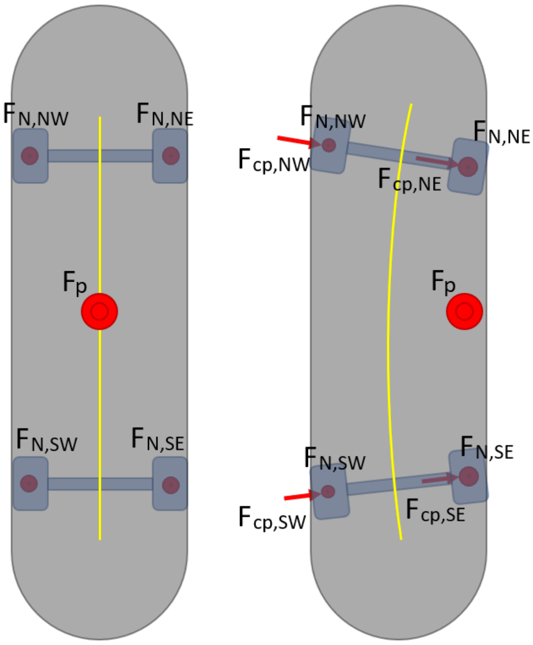


Figure 2.23: Force on skateboard

The force vector pointing toward the center of the turn, which is also responsible for accelerating the board in that direction, is called the centripetal force. Therefore, the radial acceleration directed towards the center of the turn is also referred to as centripetal acceleration. Now, let's say that a person is standing on a skateboard and moves his/her ankles to turn the board, but remains standing straight up. If the board and person has a velocity, a centripetal force will cause the board to accelerate in a radial direction. If the person does not counter that acceleration, he/she will risk being thrown over. Countering centripetal acceleration is a natural instinct learned at an early stage of life. When running around corners, turning a bicycle, swinging in the playground, and when snowboarding and skateboarding, this acceleration is countered by shifting the center of mass, i.e. leaning into the turn. When riding a bicycle, it is not common to calculate the angle one should execute the turn at. Instead, experience, muscle-memory and continuous adjustments prevent us from falling over. In order to simulate centripetal forces by the use of a Stewart platform, calculations are however needed. Where should the center of mass be located when acceleration occurs in another known place of the same object? These calculations are covered in Chapter (4), System Behaviour, Modeling and Control, in addition to how it is implemented to make the Stewart platform emulate the feeling of being subjected to centripetal acceleration. However, it should be clarified that this should be an optional addition, as it might not serve any beneficial purpose when performing certain rehabilitation routines.

2.6 Component Overview

Since the system consists of multiple subsystems and components, it is convenient to highlight some general information about them as they will be further utilized later in this report. Figure (2.24) shows an overview of the total system and components of significance.

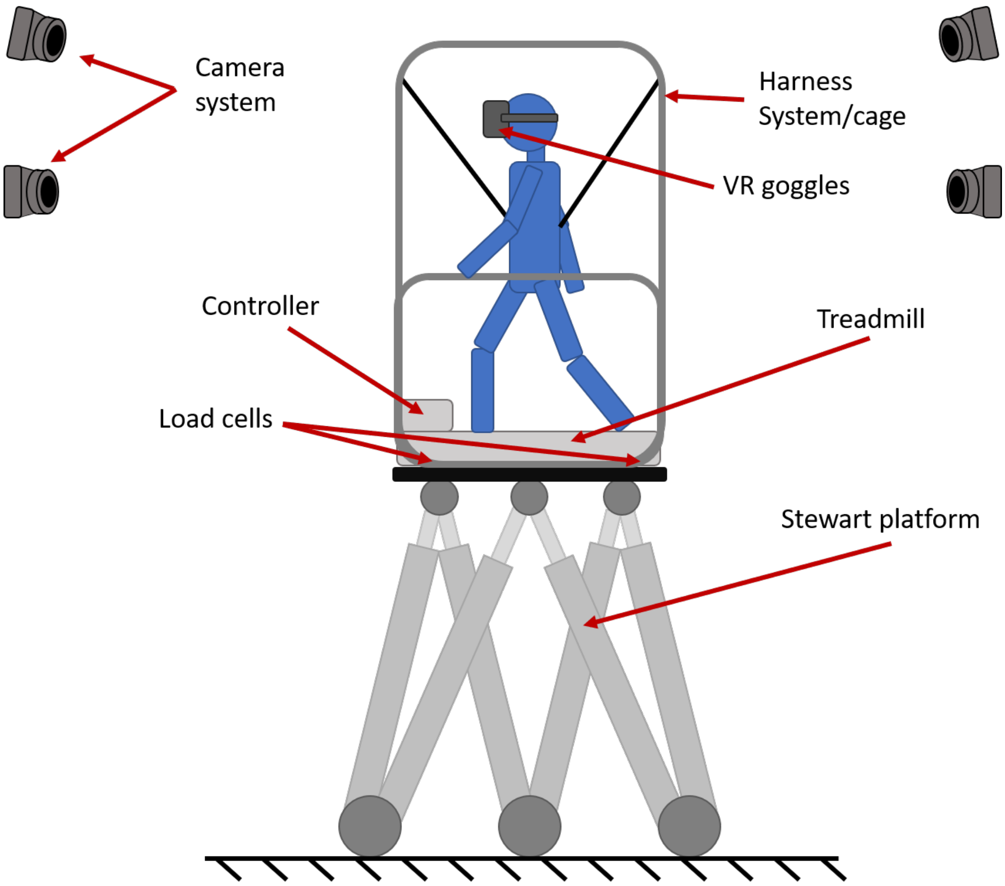


Figure 2.24: System & component overview

By combining these components in a proper way, the possibility of performing a comprehensive and thorough analysis of how an individual biomechanically responds to different scenarios is obtained. Scenarios are artificially created by the use of a simulated environment visualized with VR goggles and physically imitated by the treadmill and Stewart platform using a variety of sensors, such as the load cells and camera system.

2.6.1 Stewart Platform

A Stewart Platform is a manipulator consisting of six prismatic actuators and twelve universal joints connected in a specific way. The setup gives the opportunity to manipulate the object located on top of the platform in six degrees of freedom; surge, sway, heave, roll, pitch, yaw. Rexroth Bosch Group is a producer of such a platform and has a Stewart platform series called eMotion. The Stewart platform available for this project is a Rexroth eMotion 1500, which is the smallest among their Stewart platform capable of manipulating up to 1500 kg. Rexroth eMotion 1500 has an update rate up to 250 Hz which corresponds to an update of signal every 4 ms[23].

To avoid self-destruction, the products come with an internal feedback controller that handles all the kinematics with reference to the center position of the upper base. As a result, the eMotion platform has six inputs; surge, sway, heave, roll, pitch, yaw of the center of the upper base. The platform can be seen in Figure (2.25). This platform will act as the main actuator for simulating the dynamic behaviour of a board and the orientation of the virtual ground.

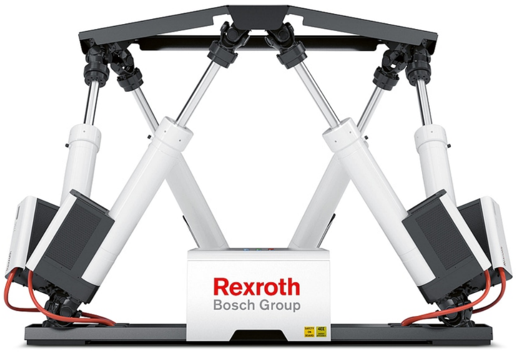


Figure 2.25: 6 DOF Stewart Platform [23]

2.6.2 Qualisys Motion Tracking

A major part of the biomechatronic lab is the implementation of a motion capture system that records the running or walking motion in real-time. This allows for sub-millimeter precision of the position and movement of the user utilizing CELNA. The high accuracy and frame rate of the cameras benefits both the striding belt velocity control and the biomechanical analysis of the patient.

The Qualisys system at the available facility is made up of 17 Qualisys 7+ cameras, shown in Figure (2.26), spread out to obtain very accurate biomechanical information about the subject of interest. To get the most precise readings, small reflectors are added to the body of whoever is using the system, in critical joint areas such as knees, hips, shoulders, etc.

Figure 2.26: Qualisys motion capture camera [24]

These cameras are equipped with 12-megapixel sensors, capable of capture speeds of up to 10 000 frames per second (fps). However, at these speeds, the resolution and field of view (FOV) is reduced. Using the full FOV at full resolution, the maximum capture rate is 500 fps. Additionally, the maximum capture distance, using 16 mm markers, is 35 meters. Because this is a 7+ series model, this camera includes motorized lens control, adjusting focus, and aperture directly in the QTM software. According to Qualisys' product specification this system is engineered for minimum latency and excellent real-time performance, with latency down to 4 ms. A data sheet of appropriate information can be found in Appendix (B.2) and at Qualisys' website.

2.6.3 LifeFitness Treadmill

An industrial level treadmill will be used in conjunction with the Stewart platform in order to simulate an environment for walking or running. The treadmill, shown in Figure (2.27), is a Life Fitness 95T, and is powered by a 2.1 kW AC motor and has striding belt dimensions 152 cm by 55 cm (length by width). The built-in computer which originally enables the user to control and engage different exercise programs is irrelevant and will therefore be removed along with the handlebars, handlebar support, control panel, control board and elevation mechanism. The only parts needed are the power train to the striding belt, the striding belt itself, and the base structural frame. Datasheet can be found in Appendix (B.3).

Figure 2.27: LifeFitness 95T Treadmill [25]

2.6.4 MyRio

The National Instruments myRIO-1900, Figure (2.28), is a student embedded device developed by National Instrument for implementing multiple design concepts with one device [26]. For this project, the myRIO is used as the main control unit. NI myRIO has a processor speed of 667 MHz, and a Field Programmable Gate Array (FPGA), which is an integrated circuit consisting of reprogrammable logic switches [27]. FPGA allows for programs to execute much faster and in parallel, totally independent of each other. However, FPGA is limited in terms of programmability and might only be used for specific tasks to reduce the number of processes handled by the processor.

Figure 2.28: National Instruments MyRio [26]

2.6.5 Load Cells

A set of four load cells is mounted beneath the treadmill. The purpose of these load cells is to determine how much weight is applied by the user and where it is applied. A $24\,V$ power supply is connected to a weight transmitter which amplifies the sensor readings to a voltage in the range $0-10\,V$. By conditioning and processing the load cell signals, an estimation of magnitude and location of applied load can be made. The load cells used are similar to the model shown in Figure (2.29), DYLF-102, but rated for 200 kg instead of 1000 kg. The load cells are mounted in a rectangular pattern underneath the treadmill.

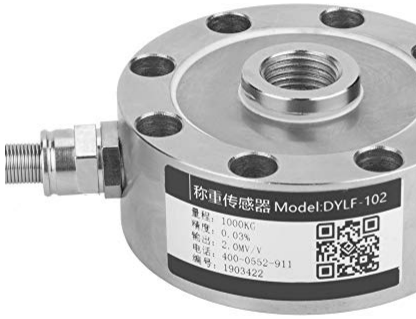


Figure 2.29: Load Cell [28]

2.6.6 Wii Balance Board

Although not a physical part of CAREN, the Wii Balance Board must be covered as it is a crucial tool in testing the control system on a smaller scale. Its small size and light weight allows for program testing other places than at the laboratory facility, which is a great benefit. Similar to the load cell locations beneath the treadmill, the Wii board has four sensors located in the same orientation and therefore works in the exact same way. This is convenient as it ultimately allows for a quick hardware swap from Wii board to the treadmill-with-load-cells when finished with dry run testing.

The Wii balance board is an accessory for Nintendo's Wii game console developed by Nintendo. It was introduced in 2007 together with the Wii Fit game [29], designed for enhancing people's body awareness [30].

The board, seen in Figure (2.30), has one load cell in each corner. These cells output an electrical signal corresponding to the load they are subjected too and the data from all four cells are converted to binary values and transmitted via Bluetooth to a host device. Then, by software programming, the data from the four load cells can be used to determine the weight magnitude and location of the object on top. This information is used for human interaction within the virtual reality which also affects the orientation of the Stewart platform, depending on the simulation mode.

Figure 2.30: Wii Balance Board [31]

2.6.7 Speed Controller

Originally, the treadmill contains a control board with an integrated AC motor controller. However, the documentation on the original controller is confidential and information about what data and communication protocol is used as inputs to this controller is therefore impossible to obtain. As a consequence, a substitute for the original AC motor controller had to be implemented. The substitute AC motor controller is a SEW EURODRIVE MOVITRAC MC07A022-2B1-4-D0 which is an industrial frequency converter. This frequency converter enables full velocity control of the treadmill belt and is well suited for supplying the AC motor as the AC motor is rated to 2.1 kW and the frequency converter output is rated to 2.2 kW. Additional general information about the frequency converter is presented in Appendix (B.1). The frequency converter can be seen in Figure (2.31). By removing the old control board and elevation motor, the frequency converter fits snugly in the treadmill compartment at the front.

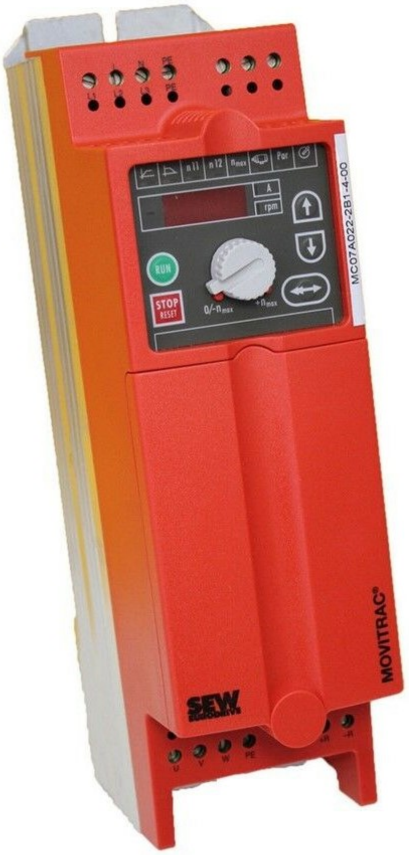


Figure 2.31: Speed controller of the AC motor in the treadmill [32]

2.6.8 VR Goggles

The Oculus Quest VR-goggles is a pair of all-in-one VR-goggles, where no external sensors are necessary. The required information to orient correctly in the virtual world is contained within the goggles, making it suitable for a small, compact system. Additionally, it is uncomplicated to operate, and the user experience is more than adequate for the intended use of this project.

Figure 2.32: The pair of VR-goggles used to experience the virtual world [33]

2.7 Data Acquisition and Signal Conditioning

There are multiple different sensors utilized in the whole system configuration, and it is essential to clarify how to acquire the information needed from all sensors and how to utilize them. Figure (2.33) shows the communication flow between the different components within the system configuration.

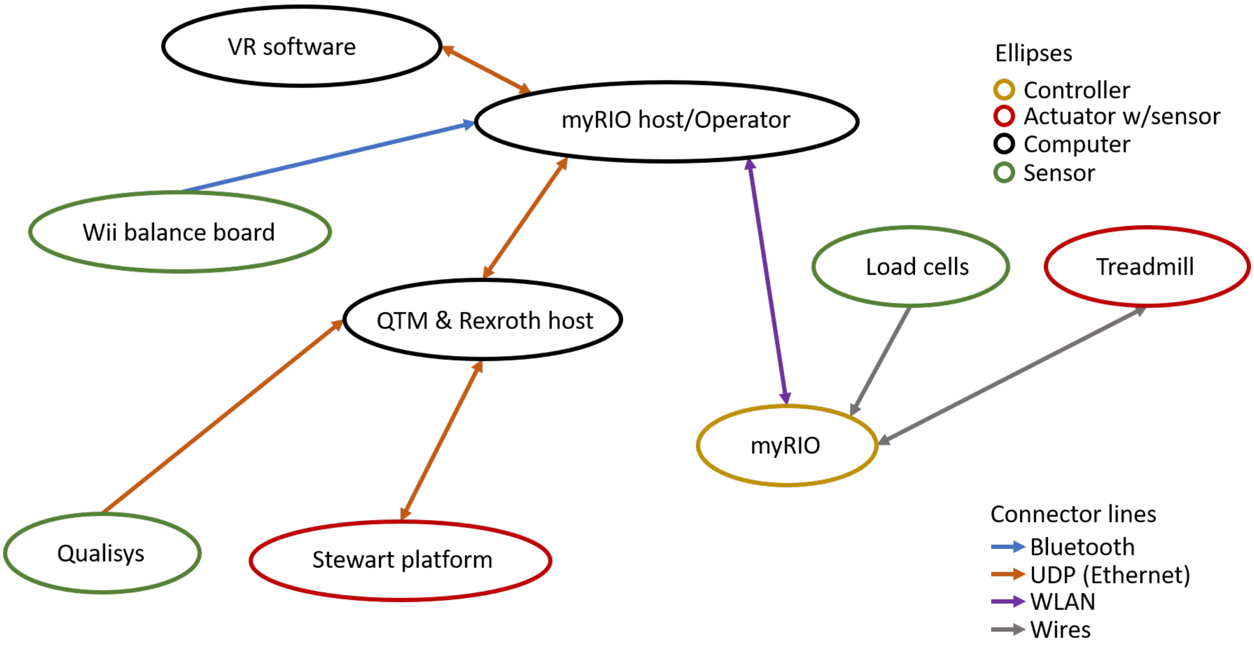


Figure 2.33: Communication flow

Table (2.1) shows an descriptive overview over the different sensors and actuators within CELNA and their individual purposes.

Sensors	**Purpose**
Load cells	-Determine user pressure location & magnitude for simulation -Gait & balance analysis
Wii balance board	-Substitute for load cells during dry run testing
Qualisys motion tracking system	-Determine user location for treadmill velocity control -Gait & balance analysis -Dynamic analysis of joints and skeleton
Treadmill velocity sensor	-Used in velocity control as feedback
Stewart platform	-Used for state handling

Actuators	**Purpose**
Stewart platform	-Responsible for physically simulating the pose and velocities generated by the haptic model
Treadmill	-Responsible for generating striding pace

Table 2.1: Sensors and actuators overview

Information gathered from the sensors should be available for the myRIO for control purposes and for the operator computer for storage and analyzing purposes. It should be clarified that further coverage of how and what data is acquired from Qualisys motion tracking system and treadmill velocity sensor is limited as it has not been included in the prototype due to cameras being serviced and the COVID-19 situation. Before addressing how the data from the sensors are acquired and conditioned, some information about the communication protocol used between the computers is provided.

2.7.1 User Datagram Protocol

User Datagram Protocol (UDP) is a transport layer protocol used to transfer data from a sender to a receiver on an internet protocol (IP) network. This transportation protocol has attributes that make it suitable for applications that require rapid transportation of information but that can tolerate some loss of data. Packets (datagrams) are small packets of information transported within the transportation layer. There is a chance that some of these packets can be dropped or received in a different order than they were transmitted for the benefit of achieving a faster communication, which is useful in some real-time controlled systems. The communication speed is a direct result of the lack of handshaking other communication protocols utilizes. Regarding communication protocols, handshaking is a term where the transmitter waits for a response from the receiver whether or not the package was received before either resending the same package or sending the next package. By doing so, one can be certain that all information has been transferred successfully, but at the cost of data exchange speed.

UDP uses the Internet Protocol (IP) to transmit a datagram from one computer to another. UDP encapsulates data in a UDP packet and adds its own header information. The header information contains the source and destination ports of communication, the packet length, and a checksum. Each category within the header has 16 bits available for its information, or in other words, a 16 digit long binary number. How a UDP packet is built is illustrated in Figure (2.34).

The source port describes what port the packet is sent from, while the destination port describes where the packet is sent. Packet length contains the information about how long the package is, including both header and data. Lastly, the checksum holds information used to determine whether or not the package has been received with errors or not. After a UDP packet is enclosed in an IP packet, they are sent off to their destination.

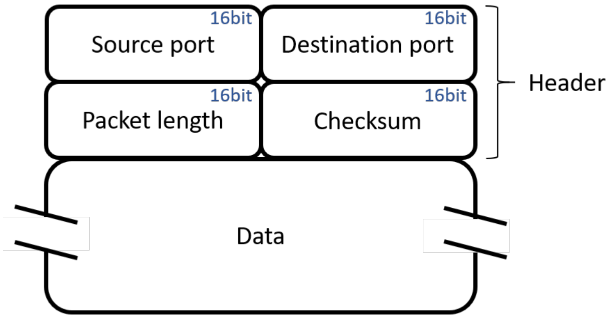


Figure 2.34: UDP packet build

The opposing transportation layer protocol, Transmission Control Protocol (TCP), establishes a connection between the transmitter and receiver. In addition, TCP utilized handshaking, making sure all data is received and in the right order. However, handshaking causes the exchange of data to be executed slower than UDP, and latency spikes can occur. Therefore, UDP is favorable to use for this application, since latency in data transfer is considered a bigger issue than the loss of some data packets. Where UDP is used in this application can be seen in the communication flow chart, Figure (2.33).

There are two parts in a UDP communication - a transmitting part and a receiving part. LabVIEW supports UDP communication and provides elements that make setting up this protocol very easy.

2.7.2 Stewart platform

The Stewart platform, which is of type Rexroth eMotion-1500, is preinstalled with associating control system and host computer. The host computer is set up to give the opportunity to remotely control the Stewart platform via UDP communication. In addition to receive and utilize UDP packets describing the desired pose of the Stewart platform, the host computer can transmit the actual real-time pose of the Stewart platform, also in UDP packets. The data describing the desired and actual pose of the Stewart platform is transferred as a string of bits which is unflattened to an array of positions, velocities and even accelerations in all six degrees of freedom. In order to remotely control the Stewart platform, it is important that the commands transmitted from the myRIO describes the desired pose and velocities in the correct format. Rexroth eMotion-1500 host computer transmits and receives data in SI-units - radians for rotation and meters for translation.

2.7.3 VR software

Since the VR software is running on a separate computer, the needed data from the artificial environment is transmitted via UDP. The VR software was developed by another group of students, and they also figured a way to gather and transmit data from the Unity engine by the use of a python-based middleware. The middleware transmits information about the pose of the virtual ground surface and receives velocity and turn rate commands, both via UDP. Data transmitted and received by the middleware and Unity engine is in SI units - radians for rotation and meters for translation. Further information about how the middleware was created and how it communicates with Unity is not covered in this report as it is part of another project.

2.7.4 Wii balance board

A Wii Balance Board is used as a tool for dry run testing the developed control program for this application. According to WiiBrew, this Wii balance board utilizes a Bluetooth Human Interface Device (HID) profile [34]. HID is a profile which assembles data from a device such as a keyboard when a key is pressed, converts it to scan code in a raw HID report, sends it as a packet via Bluetooth or USB to the host which decodes the HID report [35].

For this application, the Wii balance board sends its data packages to the operators computer which decodes the packages. The information is then packed again and forwarded to the myRIO via WLAN (labVIEW Shared Variables).

A fan base developed LabView VI package is used to obtain the data from the Wii board into LabView [36]. It requires a .dll file which is obtained from another fanbase developed package [37]. By utilizing the Wii Balance Board VI package, the data from all four sensors within the balance board are easily obtained and can be transmitted via WLAN to the myRIO and be utilized in the exact same way as the four load cells would be.

2.7.5 Load cells

There are four load cells located beneath the treadmill, one at each corner. Each load cell is connected to a weight transmitter, JY-S60, in order to obtain reasonable signal readings. A schematic of the connection between the load cells and their corresponding weight transmitters can be seen in Figure (2.35). The weight transmitter gives the opportunity to manually adjust the setpoint (zero value) and the amplification of the sensor readings. The weight transmitter is capable of transmitting either voltage or current measurements, depending on the need.

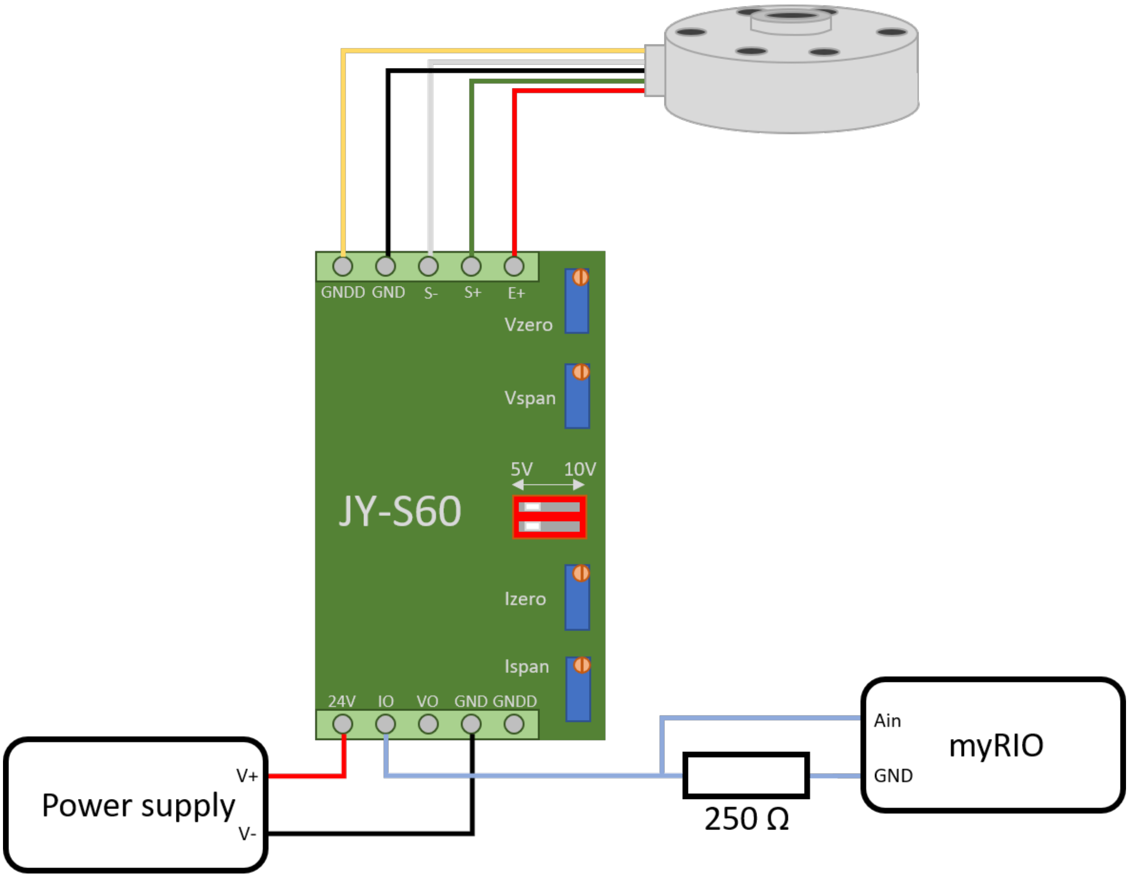


Figure 2.35: Load cell w/ weight transmitter schematic

Although the analog input of the myRIO only supports voltage readings, current is the preferred setting for the transmitter for a few reasons, but especially since current is more resistant to electromagnetic interference (EMI)[38]. EMI can cause problems as both an AC induction motor and an Inverter is operating within the same, confined space where part of the wires run through. The current signal received from the transmitter ranges from 4-20 mA, therefore, a 250 Ω resistor is added in the circuit to obtain a voltage signal ranging from 1-5 V according to Ohm's law, Equation (2.3).

$$V = R \cdot I \tag{2.3}$$

To utilize the maximum range of the weight transmitter, the transmitters are manually adjusted to output minimum current, 4 mA, when there is no person located on the treadmill. The weight of the treadmill itself is irrelevant information, and the transmitters are adjusted with the treadmill parallel to the horizontal plane. When the treadmill is angled by the Stewart platform, some impact on the load cells is expected. However, these impacts are assumed to be relatively small compared to the impact the user's orientation on the treadmill has and are therefore neglected.

2.7.6 Qualisys motion tracking system

As mentioned earlier, the Qualisys motion tracking system is not implemented in this phase of the project. However, it is desirable to enlighten its features as they are of significant use concerning biomechanical analysis and research. Qualisys motion tracking system is supported by a variety of software programs supplied by Qualisys, each with different areas of use. For this particular system, however, the most relevant is the Qualisys Track Manager (QTM). QTM is regarded as a "User-friendly mocap software" by Qualisys themselves [39], and serves as a general all-round software where body biomechanics can be analyzed and researched. Real-time skeleton and joint animation mimicking the user can be displayed and it syncs with force plates, EMG sensors, eye trackers and other relevant rehabilitation and training sensors. By the use of the Qualisys motion tracking system, accurate, real-time descriptions of how the joints are oriented and where they are within 3D space are available. If desired, additional evaluation points can be added to the body of the patient by attaching markers. The vision of utilizing a motion capture system such as this is to perform real-time evaluation and feedback of posture or gait of the patient. The feedback may be in the form of virtual messages or objectives instructing the patient to correct posture if an error is noticed. Moreover, as the arms and hands can be accurately located within 3D space as well, they can eventually be incorporated into the simulation as virtual limbs, and thus also upper limb motion can be analyzed and evaluated. By incorporating upper limb into the simulation, additional types of rehabilitation processes are possible to perform. To name a few; eye-hand coordination, upper limb reflexes and full-body coordination by multitasking balancing while handed upper limb objectives. Motion tracking by the use of multiple cameras is a vital component of CELNA and should be implemented at a later stage of prototyping. The data of interest can be transmitted via UDP and is therefore easily available for utilization within the control program if needed.

3. Treadmill hardware modifications

As mentioned, the treadmill comes fully assembled with a control panel, control board, handlebars, power train, and striding belt. The fully assembled treadmill can be seen in Figure (3.1). However, there are some components that are not needed for this application, such as the control panel, handlebars, control board, and elevation mechanism which are removed as they might be in the way. These parts of the treadmill will not be used.

Figure (3.2) illustrates how the treadmill looks after the control panel and frame have been removed. By removing the excess frame and handlebars, the possibility of those items obstruction movements when simulating different scenarios is no longer present. A custom frame is instead built which supports a harness system, removable handlebars, and the possibility to lift and move the whole assembly as one unit.

A CAD model of the essential components after stripping down the treadmill has been created and can be seen in Figure (3.3). This model is drawn from measurements of the treadmill and has served as reference for the design of the safety cage.

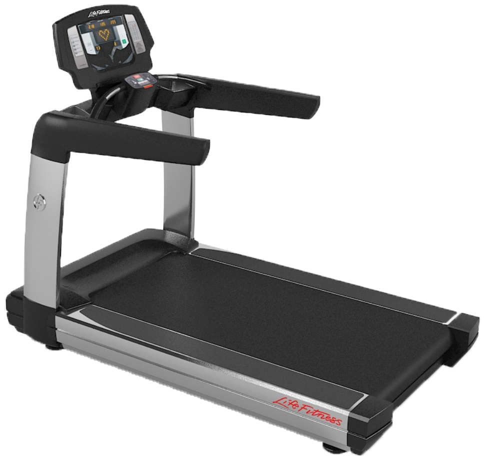


Figure 3.1: Life Fitness 95T [25]

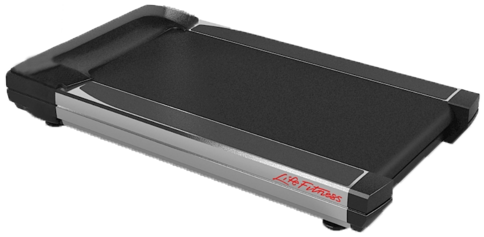


Figure 3.2: Illustration of modified treadmill

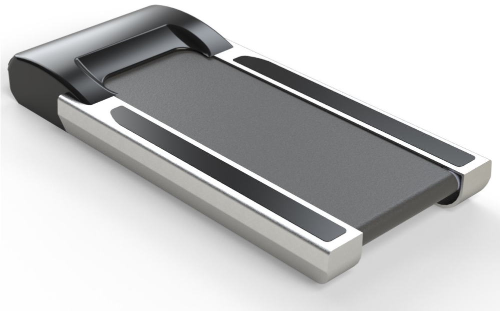


Figure 3.3: CAD model, illustrating the treadmill for use with the external components

3.1 AC motor control

As seen in Figure (3.4), the internals has been modified as well. The drive train for the striding belt (AC motor, belt, and pulleys) is the only remaining part, as the control board and the elevation actuator are removed. These components are replaced with the SEW EURODRIVE frequency converter (red) used as a speed controller for the AC motor, and a compartment for additional circuit boards (gray).

There are currently two circuit boards located inside the compartment - a relé circuit board and a current-to-voltage signal conversion circuit board. Since the frequency converter only accepts 24V inputs, a relé circuit board is needed to transmit the control signals from the myRIO as the myRIO is incapable of transmitting 24V signals. The current-to-voltage signal conversion circuit is used to convert the current measurement signals received from the load cell weight transmitters to voltage measurements receivable by the myRIO (1-5V).

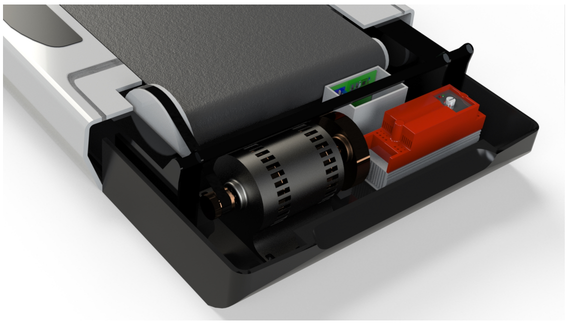


Figure 3.4: CAD model, illustrating the treadmill internals

3.2 Load cells

The load cells are fastened to the traverse structural beams of the treadmill as illustrated in Figure (3.5). The load cells act as the feet of the treadmill and are the only contact areas between the treadmill and the surface below. That can be done since each load cell is rated up to 200 kg, which is considered sufficient when assuming some degree of evenly distributed weight. It is not likely that one, or even two load cells will ever take the full load on their own. It should be noted that four fastening points such as this over defines the structure in terms of fixing points. If one load cell is slightly off in height, some of the load cells will be subject to stresses when bolted together. A relatable example is a wobbly study desk with uneven leg lengths. If you were to bolt the legs down, forcing them to the floor thus forcing the legs lengths to be even, large stresses will occur in the legs. Similarly, since the safety frame is built and welded by hand, it is unlikely that all load cells will be perfectly in touch with the frame before bolting them down. Such stresses could in fact disturb the signals to the degree they would be useless. To reduce this potential problem, a rubber bushing is sandwiched between each load cell and safety frame, as illustrated in Figure (3.6). This bushing allows for some difference in height between the load cells without causing very high stresses and disturbances in the measurements.

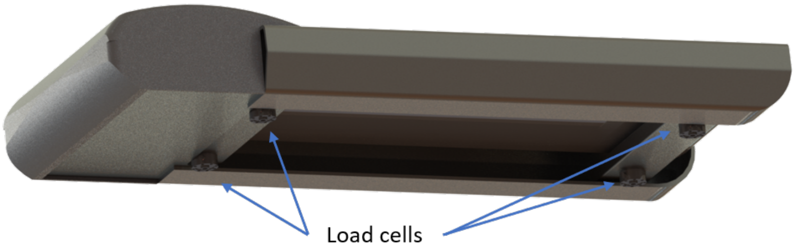


Figure 3.5: Illustration of load cells location

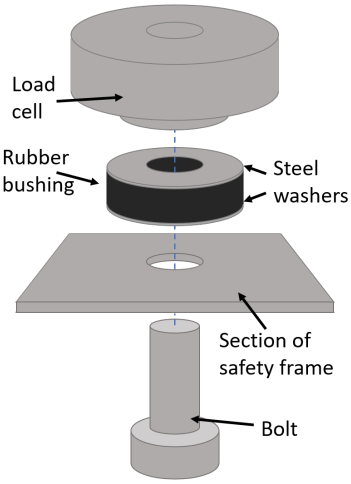


Figure 3.6: Rubber bushing location

3.3 Emergency Stops

An essential part of any human incorporated system is emergency stops. In addition to the operator's ability to shut down the system at any time, the user should also have the option to stop the system in case of an emergency. Therefore, two emergency stops are added. One strictly added as a button for the user to press in case of panic or the experience of emergency within arm's reach at any time. This could be placed anywhere with quick access to the user, for example on a game controller, or on the harness worn by the user. The second is added to shut down the system if the user suddenly falls or slips. This is added as a kill switch integrated into the safety harness worn by the user. If the user falls, this stop should automatically stop the system immediately.

3.4 Treadmill harness support

Since the purpose of the treadmill on top of a Stewart platform is rehabilitation and skill development, there is a considerable chance that the person utilizing the treadmill will, at some point, fall over. Therefore, a safety harness is implemented. The purpose of the safety harness is to catch the person using it in case of a fall, and it should not be interfering with the person when he/she is operating the system normally. The Stewart platform on which the treadmill is mounted, has six degrees of freedom. Therefore, a local frame with a harness mounting support, known as a harness system for treadmills, is the most suitable solution because it will move with the platform, while not being excessively large. To select a proper frame, multiple concepts should be evaluated and scored according to how well they fulfill the needs for this task. It should be mentioned that the structural frame of the harness system is also referred to as a safety frame throughout this report.

3.4.1 Factors of consideration

There are a few desired attributes regarding the functionality of the harness system that must be considered when weighing the individual concepts up against each other, and those are:

- Safety
- Freedom of motion (regarding the person onboard)
- Structural complexity (production and assembly)
- Usability
- Ease of transportation
- Space occupation
- Weight
- Robustness

These attributes are further used to distinguish between the different concepts.

3.4.2 Concepts

The generation of concepts is generally based on already existing concepts, as harness systems for treadmills are already developed and available on the market. However, some modifications have been done as the ability to lift and transport the structure, including the treadmill, is needed. Also, many of them are big and heavy, which conflicts with the desire for the Stewart platform to act as quickly and subtly as possible. It should be clarified that these are just concepts of the mechanical structure, and the most suited concept will be further analyzed and modified to serve as the actual safety frame to be used. Note that the modeled treadmill illustrated in the presentations of these concepts serve only as illustrations and are not representative of the actual treadmill.

Concept 1

Concept 1, seen in Figure (3.7), is a simple concept with a single transverse beam. The scope of construction is minimalistic, easing the process of constructing and assembling. However, the vertical bars on the side of the treadmill poses a risk of interfering with the arm movements of the person. The two fastening points for the harness enables moderate safety when falling, considering the potential of falling over when the treadmill is tilted. The handlebars can be unattached as wanted.

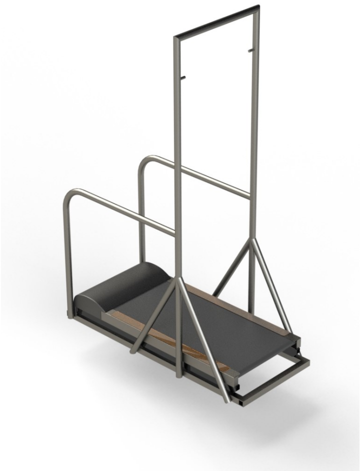


Figure 3.7: Concept 1

Concept 2

Concept 2 is more structurally complex than concept 1. This concept consists of two supporting transverse beams at each end of the treadmill with one longitudinal beam connecting them, as illustrated in Figure (3.8). This structure yields free room for the arms of the person on board when the handlebars are unattached. Two fastening points enable moderate safety when falling.

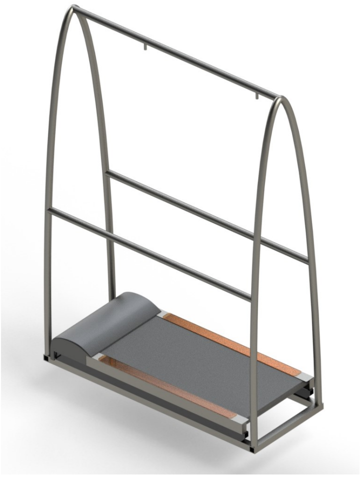


Figure 3.8: Concept 2

Concept 3

Concept 3 has two supporting beams at both longitudinal sides of the treadmill, with one beam connecting them, as seen in Figure (3.9). This concept yields free room for the arms of the person when the handlebars are unattached. Four fastening points enable good safety when falling.

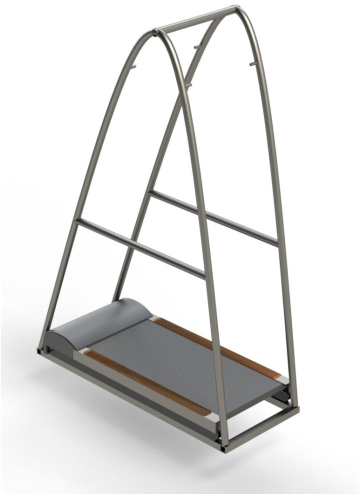


Figure 3.9: Concept 3

Concept 4

Concept 4 is, similarly to concept 1, a simple structure. The concept is shown in Figure (3.10), and consist of one longitudinal beam reaching over the treadmill. This concept yields free room for arms when the handlebars are unattached. However, this structure poses a potential risk of being pinched between the bar and the striding belt of the treadmill if the user slips of the backside of the treadmill. Two fastening points enable moderate safety when falling.

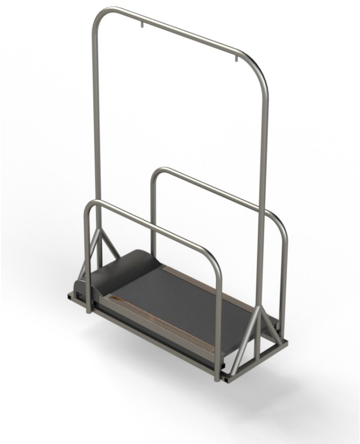


Figure 3.10: Concept 4

Concept 5

Figure (3.11) shows concept 5, which is the most market-influenced design. This structure consists of two main beams in the front that reaches over the treadmill. This concept yields free motion of the arms of the user when the handlebars are unattached. This structure has two fastening points - none from the back, which yields moderate safety when falling.

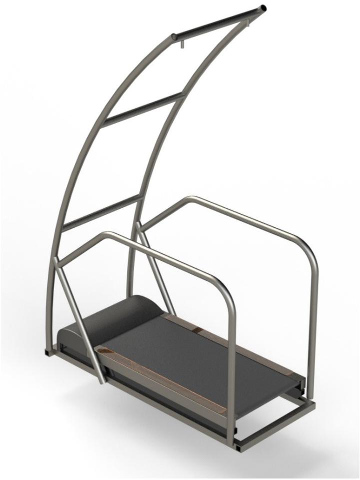


Figure 3.11: Concept 5

Concept 6

Figure (3.12) shows concept 6. Opposed to previous concepts, concept 6 utilizes only straight beams. Similar to the other concepts, the handlebars can be unattached, which yields free room for arm movements of the person. This concept has four fastening points, which yields good safety when falling.

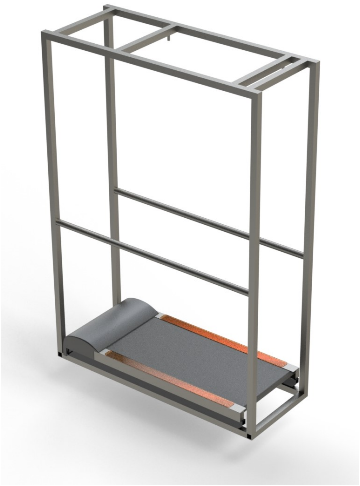


Figure 3.12: Concept 6

3.4.3 Concept scoring

To select between the six concepts of safety frames, a scoring technique is used. Concept scoring is a great method to choose between multiple concepts. It bases the results on an analysis of how well each concept fulfills the desired attributes for the harness system. There are two main terms for scoring the concepts:

- Rank
- Weight

Weight determines how important the related attribute is, while rank specifies how good the concept scores on that specific attribute. These two values are multiplied for that attribute to yield the concept's score on that attribute. For this case, weight ranges from 1 - 10, and rank ranges from 1 - 5. Table (3.1) shows the scoring of concept 1 - 6.

		Concept					
		1	2	3	4	5	6
Attribute	**Weight**	**Rank**	**Rank**	**Rank**	**Rank**	**Rank**	**Rank**
Safety	10	2	2	4	1	2	4
Freedom of motion	8	1	5	4	5	5	5
Structural simplicity	4	4	2	2	4	2	2
User friendly	7	3	4	4	3	3	4
Ability to be lifted	5	2	5	4	5	2	5
Space occupation	5	5	2	4	3	4	1
Weight	4	5	3	4	5	3	1
Robustness	10	1	3	4	1	3	5
Sum (Weight*Rank)		130	173	204	157	161	200
Continue?		**No**	**No**	**Yes**	**No**	**No**	**No**

Table 3.1: Concept scoring

As seen from the scoring, Table (3.1), concept 3 scores the most points, and is therefore considered the most suiting concept to continue developing. The main benefits of this concept are the available fastening locations for the harness hooks, along with the simple and compact, yet robust design. In addition, there are multiple locations on the frame available for lifting hooks when the construction is to be moved or transported. Also, the handlebars can be designed to be easily attached/unattached. Figure (3.13) shows concept 3.

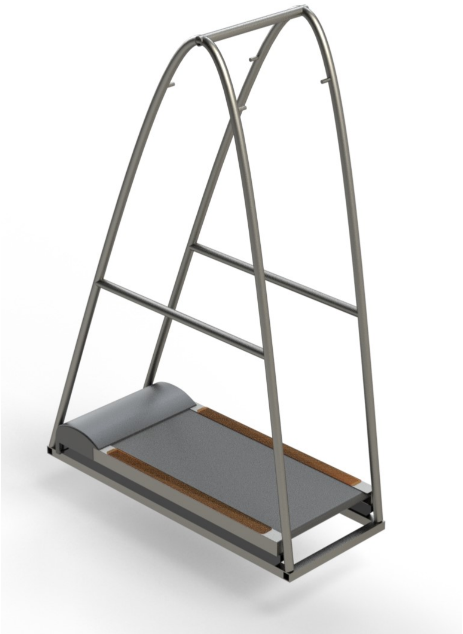


Figure 3.13: Concept 3

3.4.4 Designing and dimensioning

When designing the frame, it is important to take into account the available materials and tools. Although concept 3 is the basis of design, some changes have to be made for it to be produced quickly and efficiently in the available workshop. In addition, the dimensions used for the beams should be consciously selected based on the number of forces the structure is subjected to during operation. For this project, simulation software is used to efficiently determine if the dimensions selected are suitable for this application.

After applying some slight modifications to concept 3 and selecting beam dimensions, the final safety frame design became as shown in Figure (3.14). This model consists of:

- 40 x 80 mm rectangular profiles as base platform
- ø 50 mm circular profiles as handlebars
- 40 x 40 mm square profiles for the rest
- 4 x 0.34 T lifting eyes for harness
- 2 x 0.7 T lifting eyes for frame

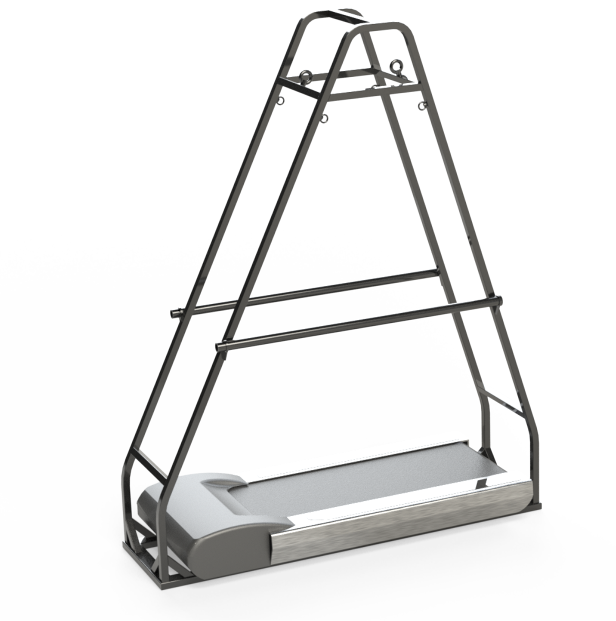


Figure 3.14: Final cage design

As can be seen, the side support consists of multiple point bends instead of one continuous bend as in concept 3. Point bends make constructing two similar supports easier. Also, the final frame includes more supports to reduce the magnitude of stresses in the material. It is important that the stresses occurring during operation are safely below the yield strength of the material used. The material used for this application is S355-steel, which is a weldable construction steel with a yield strength of 355 MPa. Blueprints of the final safety frame can be found in Appendix (C.1).

3.4.5 Abaqus stress analysis

Abaqus is a stress analysis simulation program used to simulate the stress levels in structures. By using such a program, an evaluation can be done regarding the safety and functionality of a given model. A simple model is imported to get a reference of what stresses can be expected during operation. Additionally, this model is imported as a surface model, rather than a solid model due to computation power and complexity. The thicknesses of each section are then implemented in Abaqus using its material description.

For the lifting scenario, a simple model consists of a fixed bottom structure. Additionally, the load is distributed equally between the two lifting brackets simulating the weight of the structure during lifting. These two simulation attributes can be seen in Figure (3.15). The lifting load is set to be a total of $6\,000\,N$, equivalent to approximately $610\,kg$. This is including a sufficient safety factor, considering only lifting operation. The mass of the actual structure is approximately $220\,kg$, which includes the safety cage and the treadmill.

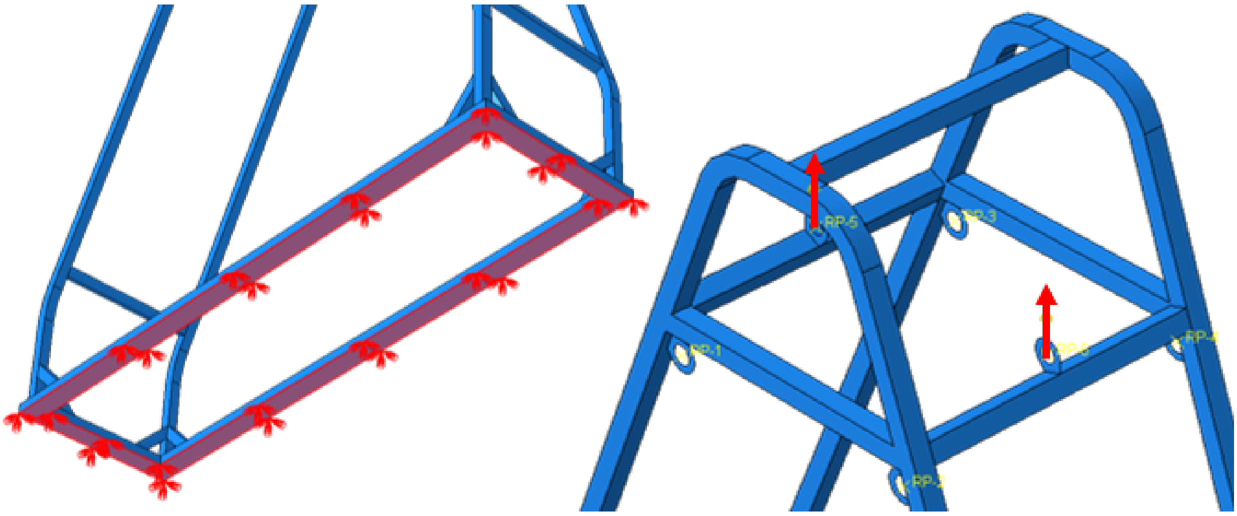


Figure 3.15: Left: Boundary conditions, fixed bottom surface. Right: Load positions during lifting operation

When reviewing the simulation results during the lifting operation, the maximum stress using a conservative approach, is approximately $278\,MPa$. This is below the yield strength of $355\,MPa$, thus considered safe for this application. The position of the maximum stress can be seen in Figure (3.16). Additionally, due to the surface simulation model, singularity issues are prone to occur as a result. This leads to excessive stress in stress concentration areas. These areas can decrease to the point where stress, $P = \frac{F}{A}$, becomes larger and larger, thus causing a singularity issue, generating greater stresses than expected. This further supports that this design, using this material is sufficient and safe for this application.

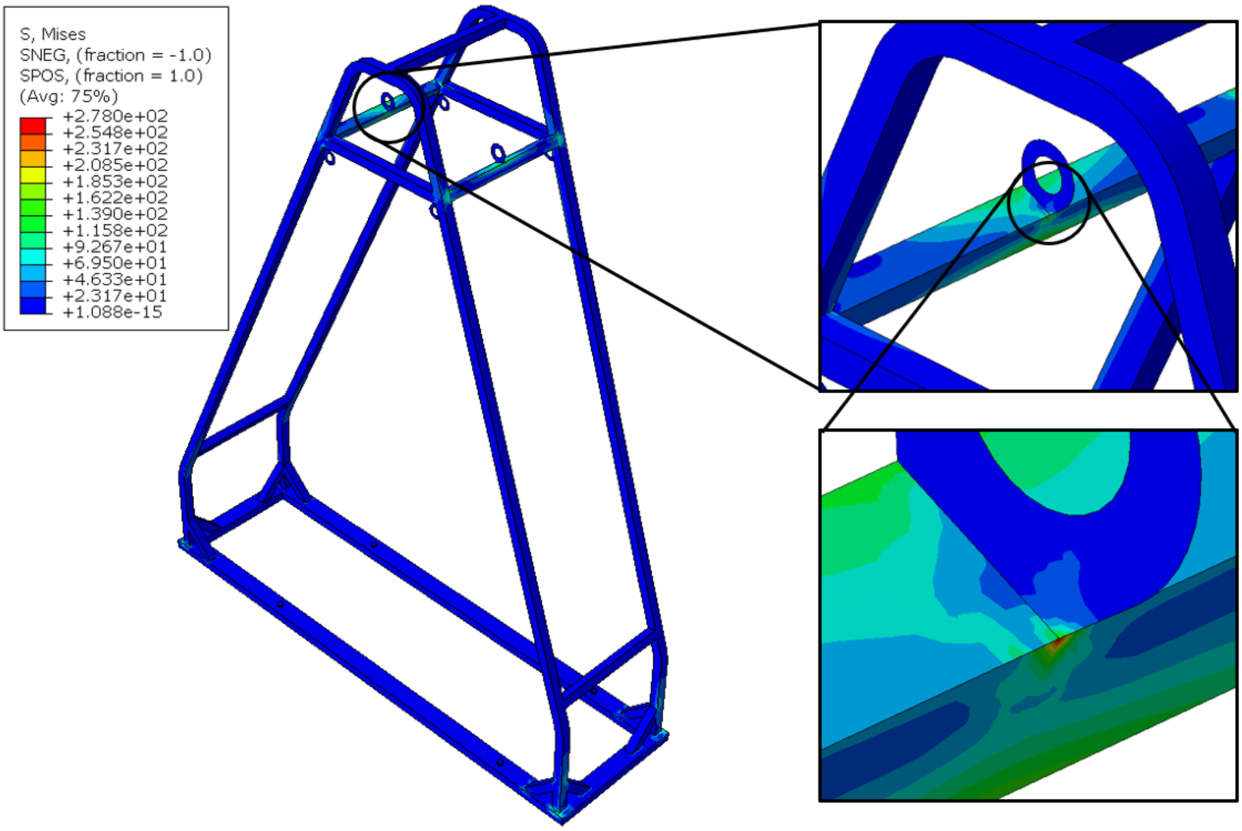


Figure 3.16: Scenario lifting the cage with the treadmill mounted.

The more important scenario regarding safety is the safety harness worn by the person on the platform. This safety harness is supposed to protect the person wearing it in case of a fall, thus a stress analysis in such a situation is required. Figure (3.17) shows the boundary conditions for this simulations, including the fixed bottom surface and the load applied in the two lifting eyes.

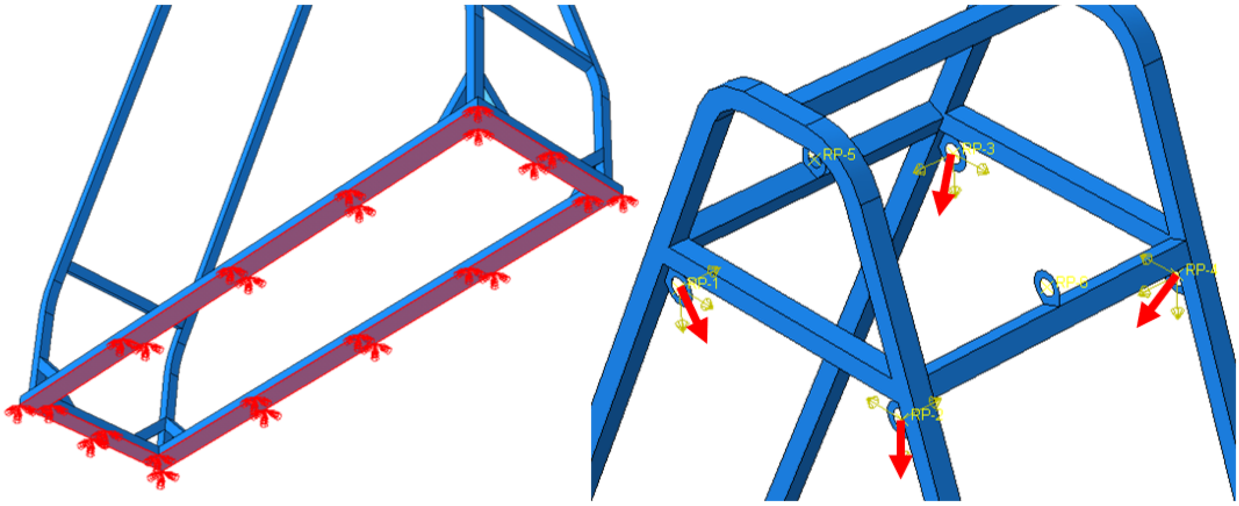


Figure 3.17: Left: Boundary conditions, fixed bottom surface. Right: Load positions during falling scenario

A similar situation is simulated with a fixed bottom structure. Additionally, four equal loads are applied to the four harness brackets, with a direction towards the center of the treadmill. These forces are applied as 1700 N in the negative y-direction, 200 N in x-direction towards the center, and 200 N in z-direction towards the center. This adds up to a total force in each bracket of:

$$F_{tot} = \sqrt{{F_x}^2 + {F_y}^2 + {F_z}^2} = \sqrt{200^2 + 1700^2 + 200^2} = 1723.4N \tag{3.1}$$

This load is equivalent to 175 kg. When applying this force, the maximum stress of approximately 344 MPa occurs in a very small area of the brackets for the safety harness, as shown in Figure (3.18). As the same singularity issue is a factor here as well, and the maximum realistic stress is indicated to be roughly 250 MPa, the safety cage is deemed sufficiently designed.

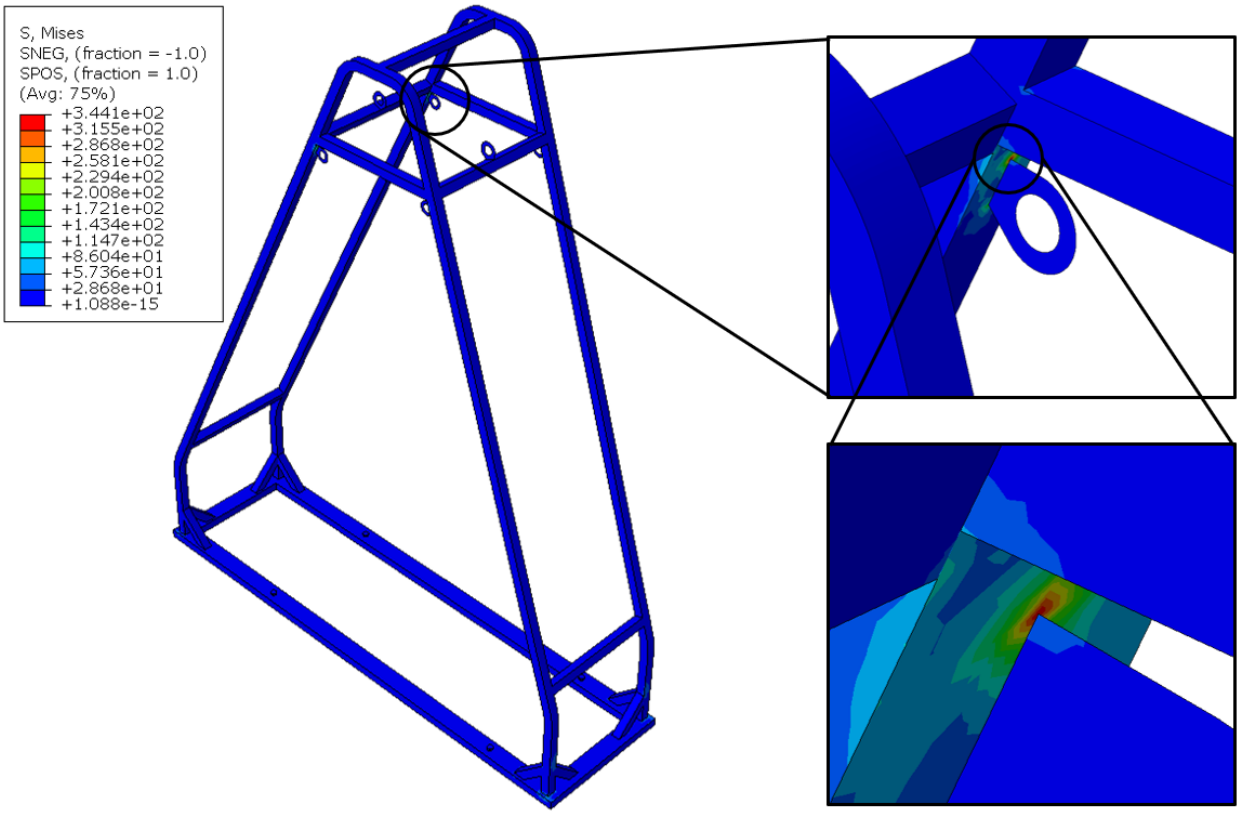


Figure 3.18: Scenario where the person is hanging form the cage.

It should be clarified that the ropes used to connect the user to the safety cage should be stretch ropes in order to prevent situations where one eye bolt takes up the whole impact force from the fall of a person.

4. System Behaviour, Modeling and Control

The Stewart platform and the treadmill are the main two actuators of CELNA. Even though the Stewart platform is built up by six individual, linear actuators, the platform has an internal controller where the only input needed is the desired translation and rotation of one point - the center of the top plate. Also, the internal controller includes a feedback system that strives to keep the actual values the same as the reference values. CELNA is, as previously mentioned, the complete system consisting of actuators and sensors. This Chapter contains all the technical information needed to understand how the human, haptic interface, admittance controller and VR software are tied together in a detailed manner. Theory about these terms is covered in Chapter (2), Theory.

Figure (4.1) is a visual description of the interconnections between the human and artificial environment, also known as haptic technology. Haptics technology, which is the process of emulating the sensation of touch by the use of sensors and actuators, can be considered the main feature of CELNA. Hence, the control system and program flow are very much built around this feature. With that being said, considering the human is part of the interconnection between the mechanical system and VR software, the user friendliness and safety of the overall system behaviour has been paid great attention to aswell.

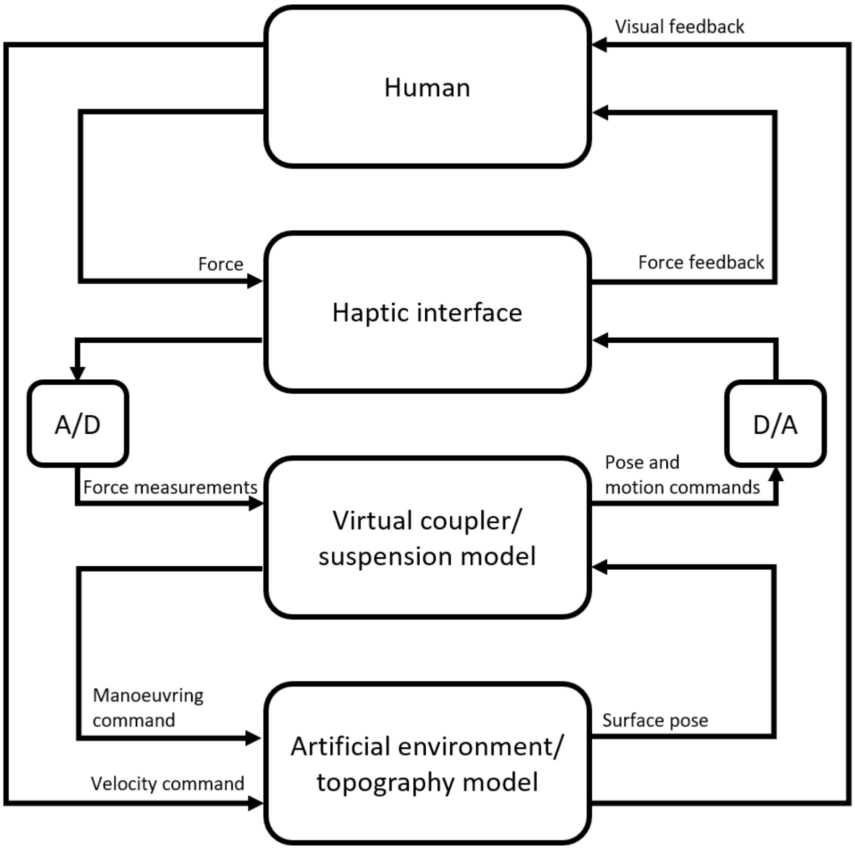


Figure 4.1: Haptics flow chart

Systems like CELNA are made to cooperate with humans in a user-friendly manner, concerning both the operator (e.g. physiotherapist) and the patient. Therefore, it is beneficial to layer the technical control flow in two layers, an outer layer (high level) and an inner layer (low level):

- **Outer layer:**
  The outer layer consists of an overview of what state the system is in. There is no in-depth information of the technical control flow within each state of the system. Such a layer is user friendly and easy to comprehend for someone that lacks necessary technical insight. This layer is interactive to both the user and the operator.

- **Inner layer:**
  The inner layer consists of the technical control flow within each state, how they cooperate and all the related algorithms and equations. Each state has an inner layer where the relevant control flow takes place. Handling this layer requires technical knowledge, and should not be directly available for unauthorized personnel.

Figure (4.2) shows the flow chart of CELNA where high level (outer layer) and low level (inner layer) are visible.

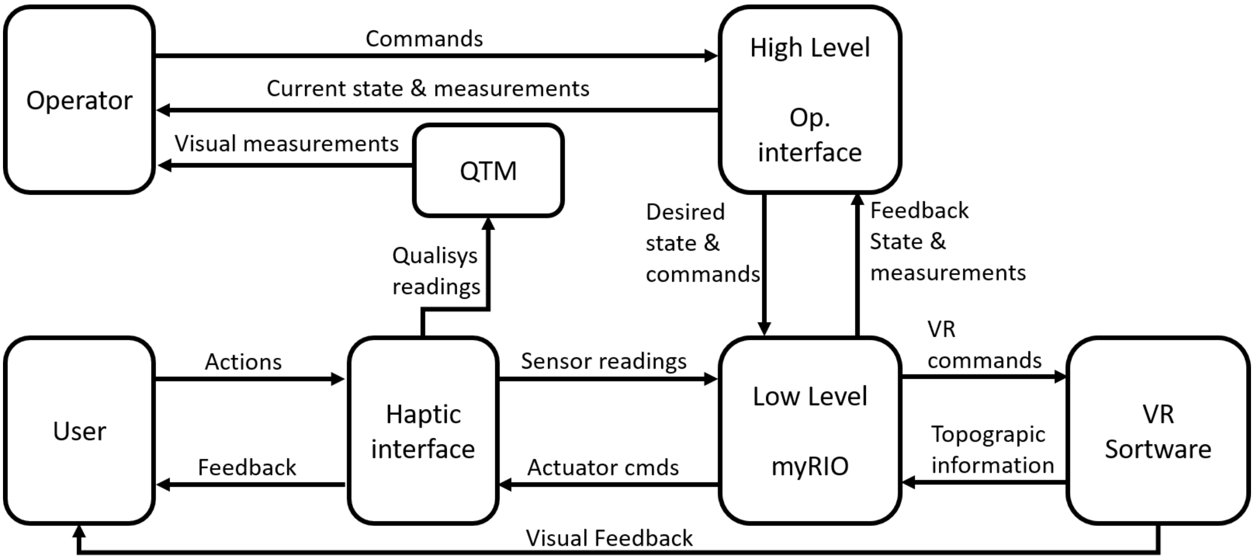


Figure 4.2: Flow chart of CELNA

There is a logic in the outer layer as well, where there are states in which the user and operator are allowed to do certain things and not allowed to do others. The states are easily navigated by the use of buttons. There is a programming strategy very suitable for state-driven systems such as CELNA, called Finite State Machine programming. As this chapter is very comprehensive, its contents follows the flow of the Finite State Machine. The states and their features, along with their inner layer control design are described in their respective sections within this chapter.

Notation

Before continuing, it may be beneficial to provide an overview of some important coordinate systems as well as how translations and rotations of one coordinate system relative to another are denoted. There are many coordinate systems, rotations, and translations, therefore it is important to have consistent forms of notations. As an example, Figure (4.3) shows the important coordinate systems regarding the physical structure. The coordinate system $[X_N \quad Y_N \quad Z_N]$ can be referred to as coordinate system N, which is the neutral coordinate system of the Stewart platform. Coordinate system b is the body of the Stewart platform, p is the projected mass center of the person on board, and NW, NE, SW, and SE are the load cells.

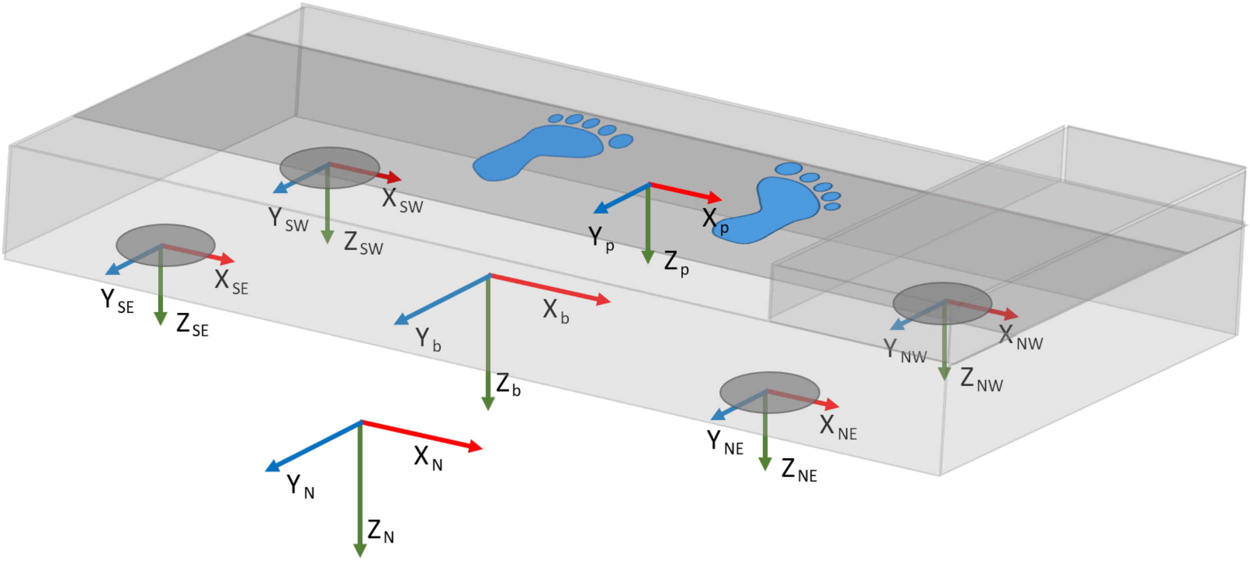


Figure 4.3: Coordinate systems

When there is/are translation(s) or/and rotation(s) of one coordinate system relative to one another, it is denoted as $T_{(along\ axis,\ from\ CS,\ to\ CS)}$ and $R_{(about\ axis,\ from\ CS,\ to\ CS)}$. As an example, when the platform is in a neutral pose, its body coordinate system relative to the neutral coordinate system is as follows:

$$[T_{X,N,b}, T_{Y,N,b}, T_{Z,N,b}, R_{X,N,b}, R_{Y,N,b}, R_{Z,N,b}] = [0,0,0,0,0,0]$$

Moreover, if there are apostrophes attached to symbols, it means that the variable/parameter has been conditioned. The number of apostrophes represents the number of conditionings done to that specific variable/parameter. Coordinate systems not described here are described where it is relevant.

4.1 Finite State Machine

A finite state machine is a behavior model that describes the relationship between a finite set of inputs and states of a system, and what output one can expect for a combination of the current state and input. State machine programming is a clean and efficient approach as the processor only handles the number of processes relevant to the current state. A system such as CELNA, which cooperates with a human, needs a direct and safety regulated system flow with no possibility of changing irrelevant parameters within any state. Finite state machine programming is very suitable for maintaining a fully functional system behavior while neglecting any impact from inputs not relevant to the current state.

4.1.1 State transition diagram

A state transition diagram is a diagram showing the signal flow through a finite state machine. It visualizes the available inputs and corresponding outputs for each state. Representing a complex system with a state transition diagram is a good way of gaining an overview of its behaviour. To establish a state transition diagram, all states of the system should be addressed and explained. First, to have some pins to hang the state's information on, the state transition diagram is shown, Figure (4.4). Transitions take place if they are true. For example, if the current state is Initial Position and if Rise.b = TRUE, proceed to Idle state.

Else, remain in Initial Position. '.b' indicates it is a button available to either the operator and/or user.

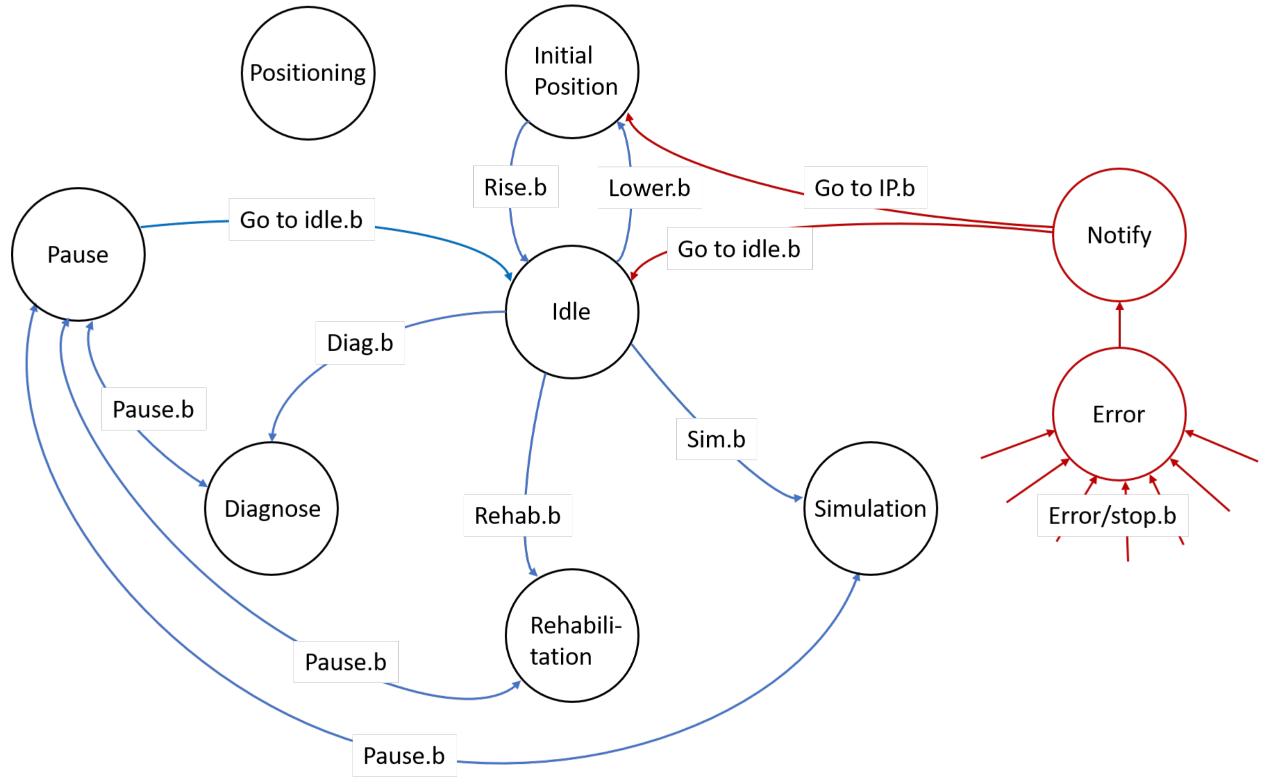


Figure 4.4: State transition diagram of CELNA

Following is a brief overview of the main overlaying characteristics of each state. See the corresponding sections for more in-depth information about each state, i.e. the inner layer.

Initial position:

- Platform is at lower position, ready for loading/deloading user
- Three options:
  - Rise - Platform travels to its neutral origo pose [0, 0, 0, 0, 0, 0]
  - Load cell zero and mass identification - identifies necessary parameters
  - Stop - Freeze the platform

Idle:

- Platform is in its neutral pose, [0, 0, 0, 0, 0, 0], waiting for further inputs
- Six options:
  - Diagnose - System goes into diagnose mode for further operation
  - Rehabilitation - System goes into rehabilitation mode for further operation
  - Simulation - System goes into simulation mode for further operation
  - Load cell two point calibration - identifies necessary signal gains

    - Lower - Lowers the platform to Initial Position
    - Stop - Freeze the platform

Diagnose/Rehabilitation/Simulation:

- Software goes to the respective mode
- Options:
    - Tailored options related to diagnose/rehabilitation/simulation
    - Pause - Pauses the platform
    - Stop - Freeze the platform

Pause:

- Pauses the platform, available from all three modes
- Three options:
    - Back to idle - Back to menu and idle position of Stewart platform
    - Continue - Continue with the current mode
    - Stop - Freeze the platform

Error:

- Stops the platform, available from all states except error and notify (indicated by multiple arrows pointing at the error state in the diagram)
- When stopped, proceeds automatically to notify

Notify:

- Goes to this state if any error occurs, such as stop button pressed
- Notifies the operator of the error
- Must reset the errors before prompted with the following two options:
    - Go back to Initial Position - Goes to Initial Position state
    - Go back to Idle - Goes to Idle state

Positioning:

- Intermediate state taking place during physical transition of the platform
- Enters this state whenever the platform has not reached the position for the next state
- One option:
    - Stop - Freeze the platform

An additional benefit of utilizing state machine programming is the ease of applying more states. To the three modes - Diagnose, Rehabilitation, and Simulation, more states can be added as necessary. It is crucial that there is a possibility to add more states in the future since the evolution of hardware can result in increasingly better ways of analyzing and rehabilitate patients, which may be desirable to add to CELNA. The following sections describe in-depth the content and functioning of each state, except Pause, Error, and Notify, as this information has been given above.

4.2 Initial Position

The initial position state has two main objectives - **Load/deload** the user and identify the zeroes and masses needed for conditioning the load cell signals. The user should not be located on the treadmill while the conditioning variables are gathered, therefore, the shortest path from deloading the user to the procedure of finding the zeroes and masses is through the initial position state. The position of the Stewart platform when in the initial position state is almost at its lowest possible position. The limits can be seen in the datasheet of Rexroth eMotion-1500 [23]. Remark that Z-axis of the Rexroth eMotion-1500 is pointing downward, hence the positive sign.

$$[T_{X,N,b}, T_{Y,N,b}, T_{Z,N,b}, R_{X,N,b}, R_{Y,N,b}, R_{Z,N,b}] \quad = \quad [0, 0, 0.3, 0, 0, 0]$$

From this state, the operator has three options - Rise to idle, perform procedure of identifying zeroes and masses, or stop. In order to rise to the idle state, both the user and operator must verify that they are ready by pressing a button. Also, when prompted to start the zeroes and mass identification procedure, a sensor must first verify that there is nothing located on top of the treadmill.

4.2.1 Force plate zeroes and mass identification

Gathering the load cell zeroes and estimated mass moved by each individual load cell is crucial for later use. The zeros and mass times acceleration should be subtracted from the raw measurements in order to prevent measuring the weight and inertia of the treadmill itself. Since the treadmill is located between the user and the load cells, as shown in Figure (4.5), measurements occurring due to the mass of the treadmill must be canceled out.

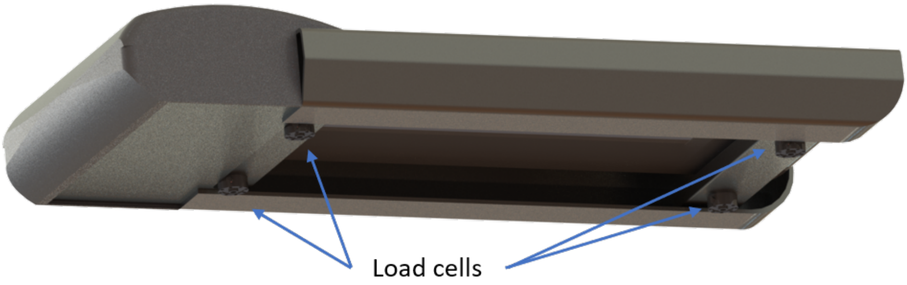


Figure 4.5: Load cell loacations

While gathering the zero-values for each load cell, the Stewart platform is stationary. Due to some sensor noise, 100 readings are done, and the mean value of these readings determines the estimated zero-value for each load cell. the zero-values are the four values (one per load cell) that must be subtracted from the respective raw load cell readings in order to obtain values of approximately zero when only subjected to the weight from the treadmill. Taking the mean value of n measurements is shown in Equation (4.1).

$$s_{zero} \quad = \quad \frac{s_1 + s_2 + s_3 + ... + s_n}{n} \tag{4.1}$$

For this case:

$$n = 100$$

The zeroes are used in the following way, with North West load cell measurement as an example:

$$s'_{NW} = s_{NW} - s_{NW,zero}$$

These zero values only zeros out the weight of the treadmill when static. However, as soon as any acceleration of the treadmill occurs, the inertia of the treadmill will affect the load readings undesirably. For constant mass or inertia, load influence varies proportionally to the acceleration according to Newton's 2nd law, Equation (4.2).

$$\sum F = m \cdot a \tag{4.2}$$

When diagnosing the patient or even when the patient is playing through simulation, this dynamic force may cause severe disturbances to the measurements. Therefore, this dynamic influence should be compensated for, i.e. subtracted from the individually measured signals. To do so, the accelerated mass and the real-time acceleration of that mass is needed. There is no direct way of obtaining the mass values, therefore, an approximation has to be done.

The load cells measure the axial forces they are subjected to, and working out their axial accelerations based on all six degrees of freedom, results in complex geometric equations. Four complex geometric equations heavily add to the overall processing power required from the myRIO, which may cause execution time issues. Again, the goal is not to completely eliminate the force occurring from accelerating the treadmill, but to reduce it. Too many factors may cause other types of disturbances, such as dissimilarities in the accuracy of the load cells, impacts from non-axial accelerations, and signal noise. Therefore, a simplified approach is used to approximate the force measured by the load cells when accelerating the treadmill.

Figure (4.6) illustrates the coordinates of the North-West load cell in reference to the neutral coordinate system, N. Following is a set of equations describing the load cells approximated tangential displacement, velocity, and acceleration.

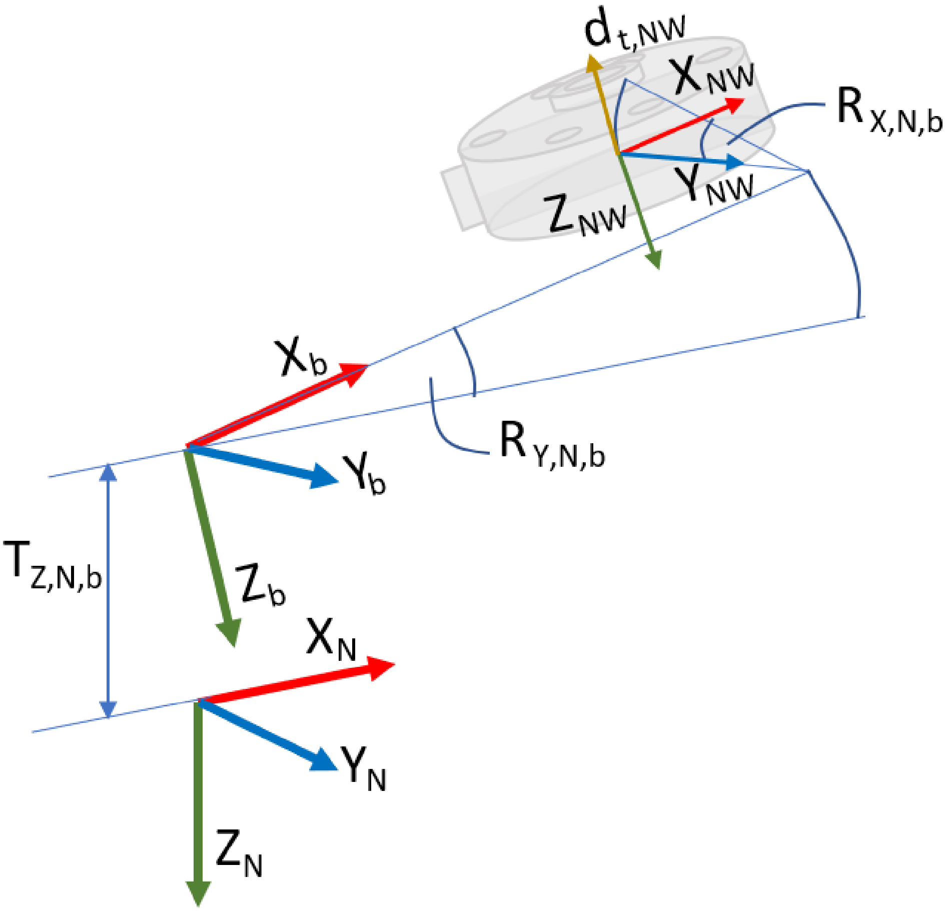


Figure 4.6: North-West load cell location

Approximated load cell tangential displacement for each load cell:

$$\begin{aligned}
d_{t,NW} &= T_{Z,N,b} - T_{X,b,NW} \cdot R_{Y,N,b} + T_{Y,b,NW} \cdot R_{X,N,b} \\
d_{t,NE} &= T_{Z,N,b} - T_{X,b,NE} \cdot R_{Y,N,b} + T_{Y,b,NE} \cdot R_{X,N,b} \\
d_{t,SW} &= T_{Z,N,b} - T_{X,b,SW} \cdot R_{Y,N,b} + T_{Y,b,SW} \cdot R_{X,N,b} \\
d_{t,SE} &= T_{Z,N,b} - T_{X,b,SE} \cdot R_{Y,N,b} + T_{Y,b,SE} \cdot R_{X,N,b}
\end{aligned}$$

Approximated load cell tangential velocity for each load cell:

$$\begin{aligned}
\dot{d}_{t,NW} &= \dot{T}_{Z,N,b} - T_{X,b,NW} \cdot \dot{R}_{Y,N,b} + T_{Y,b,NW} \cdot \dot{R}_{X,N,b} \\
\dot{d}_{t,NE} &= \dot{T}_{Z,N,b} - T_{X,b,NE} \cdot \dot{R}_{Y,N,b} + T_{Y,b,NE} \cdot \dot{R}_{X,N,b} \\
\dot{d}_{t,SW} &= \dot{T}_{Z,N,b} - T_{X,b,SW} \cdot \dot{R}_{Y,N,b} + T_{Y,b,SW} \cdot \dot{R}_{X,N,b} \\
\dot{d}_{t,SE} &= \dot{T}_{Z,N,b} - T_{X,b,SE} \cdot \dot{R}_{Y,N,b} + T_{Y,b,SE} \cdot \dot{R}_{X,N,b}
\end{aligned}$$

Approximated load cell tangential acceleration for each load cell:

$$\begin{aligned}
\ddot{d}_{t,NW} &= \ddot{T}_{Z,N,b} - T_{X,b,NW} \cdot \ddot{R}_{Y,N,b} + T_{Y,b,NW} \cdot \ddot{R}_{X,N,b} \\
\ddot{d}_{t,NE} &= \ddot{T}_{Z,N,b} - T_{X,b,NE} \cdot \ddot{R}_{Y,N,b} + T_{Y,b,NE} \cdot \ddot{R}_{X,N,b} \\
\ddot{d}_{t,SW} &= \ddot{T}_{Z,N,b} - T_{X,b,SW} \cdot \ddot{R}_{Y,N,b} + T_{Y,b,SW} \cdot \ddot{R}_{X,N,b} \\
\ddot{d}_{t,SE} &= \ddot{T}_{Z,N,b} - T_{X,b,SE} \cdot \ddot{R}_{Y,N,b} + T_{Y,b,SE} \cdot \ddot{R}_{X,N,b}
\end{aligned}$$

It is important to remark that this approach is only acceptable for relatively small angles of $R_{Y,N,b}$ and $R_{X,N,b}$. This is because the deviation in direction between the two tangential accelerations and $\ddot{T}_{Z,N,b}$ increases with increased angles, which makes summation of them increasingly inaccurate.

Now, as the equations are derived for the approximated acceleration of each load cell based on the dynamic behavior of the Stewart platform, the mass moved by each load cell can be estimated. In order to estimate these masses, a dynamic translation in the form of a sine wave will be executed by the Stewart platform in Z-direction. In doing so, a good estimation of the masses can be obtained since the acceleration equations will be accurate for such a movement as the rotation terms are zero.

One estimation of the mass is done for each loop execution for a total of n number of executions. Then, the mean mass, m_{mean}, is found by adding all the calculated masses and dividing them on the number of loop executions, n, as shown in Equation (4.3).

$$m_{mean} = \frac{\frac{s'_1}{\ddot{d}_{t,1}} + \frac{s'_2}{\ddot{d}_{t,2}} + \frac{s'_3}{\ddot{d}_{t,3}} + ... + \frac{s'_n}{\ddot{d}_{t,n}}}{n} \tag{4.3}$$

The mean mass found for each individual load cell will then be considered constants and multiplied with the continuously calculated acceleration. Then the product is subtracted from the measured forces during operation, as shown below for North West load cell as an example.

$$s''_{NW} = s'_{NW} - m_{NW,mean} \cdot \ddot{d}_{t,NW}$$

Note that mean mass, in this case, does not represent mass in kilograms, as the force measured by the load cells is not in Newton. This is only applicable because it is multiplied with acceleration again and subtracted from the same force measuring unit it was initially derived from. It should also be mentioned that with increasing angle of the treadmill, disturbances in load cell readings increases due to the ability to only measure axial forces as illustrated in Figure (4.7). F_g is the force due to gravity and mass, $F_{g,a}$ is the axial component (measured), and $F_{g,r}$ is the radial component (not measured).

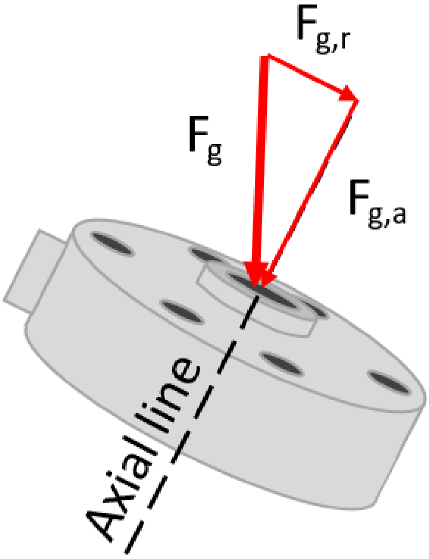


Figure 4.7: Illustration of measures force

Again, it is crucial to emphasize that the subtraction of estimated force disturbance due to acceleration of mass is only viable for small angles of the treadmill orientation. Lastly, a flow chart of the Initial Position state is presented in Figure (4.8).

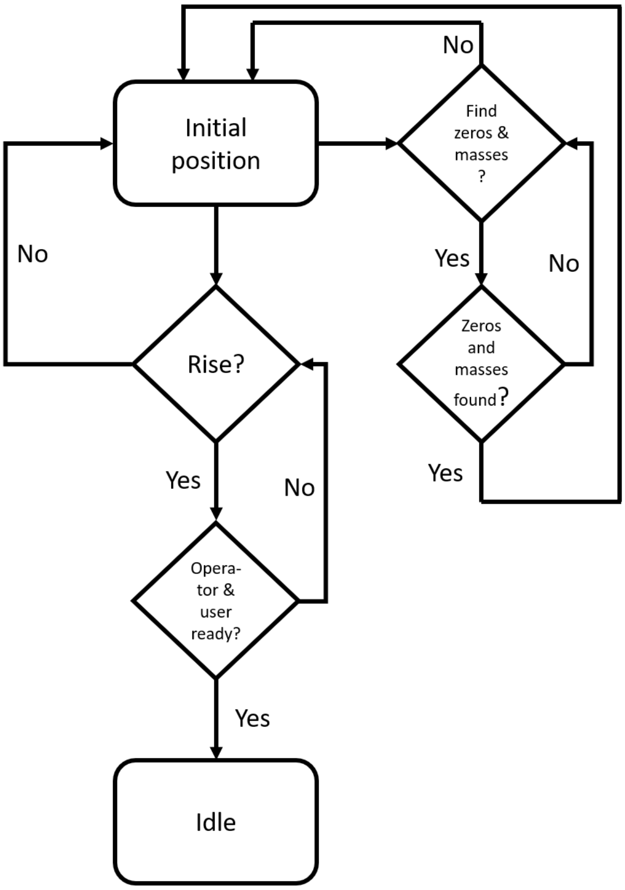


Figure 4.8: Flow chart of Initial Position state

4.3 Idle

The idle state can be viewed as a lobby state, as the main states, or modes, are progressions from this state. In this state, the operator has the opportunity to select either one of the modes, lower the platform, or push the stop button. In addition, the operator can start a load cell two point calibration, as it is convenient to do so in this state. While in Idle state, the Stewart platform is located in its neutral position:

$$[T_{X,N,b}, T_{Y,N,b}, T_{Z,N,b}, R_{X,N,b}, R_{Y,N,b}, R_{Z,N,b}] \quad = \quad [0, 0, 0, 0, 0, 0]$$

4.3.1 Load cell two point calibration

Two point calibration is the procedure of making the I/O-relation of a signal behave as a linear slope between two reference points. Such a calibration is possible assuming that the I/O-relation of the sensor is also linear. For this case, the force/voltage or force/current relationship (depending on settings of transmitter) is assumed to be linear. The first reference point, which is common for all load cells after the first step of conditioning, has value zero, i.e. no applied force - zero measurement. The second reference point however has to be derived somehow, which should also be a common point for all load cells in order to satisfy the following criteria. See Figure (4.9) for reference. s''' is the load cell signal with zeros subtracted, mass times acceleration subtracted, and multiplied with a gain, i.e. processed/-conditioned three times.

The criteria for load cell measurements after two point calibration are:

- Person located right between East and West load cells:

$$\begin{aligned} s'''_{NW} &= s'''_{NE} \\ s'''_{SW} &= s'''_{SE} \end{aligned}$$

- Person located right between North and South load cells:

$$\begin{aligned} s'''_{NW} &= s'''_{SW} \\ s'''_{NE} &= s'''_{SE} \end{aligned}$$

- Person located in the center/origo:

$$s'''_{NW} = s'''_{NE} = s'''_{SW} = s'''_{SE}$$

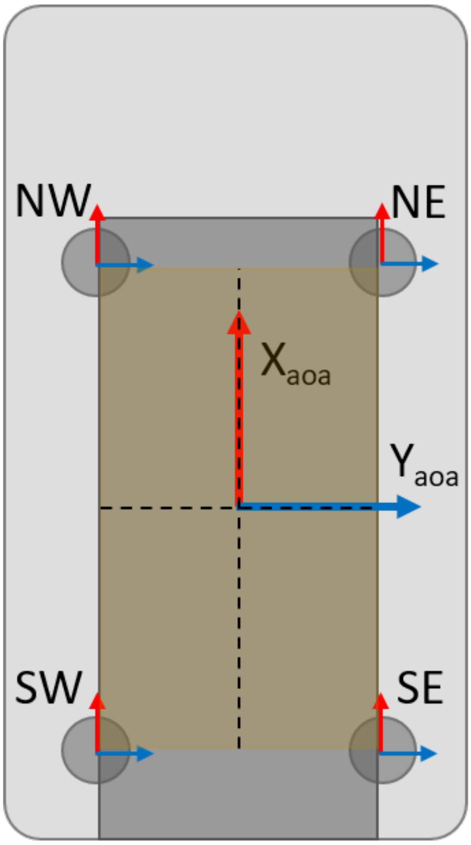


Figure 4.9: Illustration of load cell locations

For four ideal, equal load cells and weight transmitters with perfect positioning, these criteria will be true considering the initial zero and mass identification procedure of the load cells are done. In reality, however, the amount of force needed to read similar measurements from multiple combinations of load cells and weight transmitters varies. Although the load cells are zeroed, the slope of input versus output is different for each combination of load cell and weight transmitter. Therefore, additional information about the load cell measurements is needed; the individual output/input relations for each load cell.

Since the person is already located on the treadmill, the simplest and most efficient way of identifying the load cell gains is to utilize the weight of the user. The procedure is to place the user in the center of the rectangle formed by the load cells (marked on the physical treadmill) and observe the new load cell readings. All four new load cell readings should be similar if the person is in the dead center, which can be achieved by multiplying each individual reading with a respective gain. It should be mentioned that the sum of all four load cells when the user is statically located on the treadmill is stored for later use, as $W_{p,tot}$. The approach of using the person standing on the treadmill as a calibration reference may not be ideal in terms of accuracy, but it is efficient and an approximation is good enough for prototype testing. The gains found are observed by the operator, and if they seem to be incorrect, recalibration is quick and easy to do.

To have as little deviation from a unit gain as possible, the average of all four load cell measurements is used as the second reference point of the two point calibration. A large deviation from a unit gain can cause an increase in disturbances. In order for each load cell's I/O-relation to match the linear slope between the two reference points, their individual measurements must be multiplied with a corresponding gain G_{mean}, see Equation (4.4). The procedure for estimating the required gain is as follows.

Finding the n averages between the four load cells:

$$s''_{ref,i} = \frac{s''_{NW,i} + s''_{NE,i} + s''_{SW,i} + s''_{SE,i}}{4}$$

Calculating the required gain for one load cell to reach the average measurement of all four load cells for i measurement:

$$G_i = \frac{s''_{ref,i}}{s''_i}$$

Then the mean gain for each load cell is as shown in Equation (4.4).

$$G_{mean} = \frac{G_1 + G_2 + G_3 + ... + G_n}{n} \tag{4.4}$$

Current measurement:

$$i = 1 : n$$

Total number of measurements:

$$n = 100$$

The four load cell signals are then multiplied by their respective gains during operation in order to estimate the patient's location. An example with the North West load cell signal is shown below:

$$s'''_{NW} = G_{NW,mean} \cdot s''_{NW}$$

Figure (4.10) shows the flow chart of the Idle state.

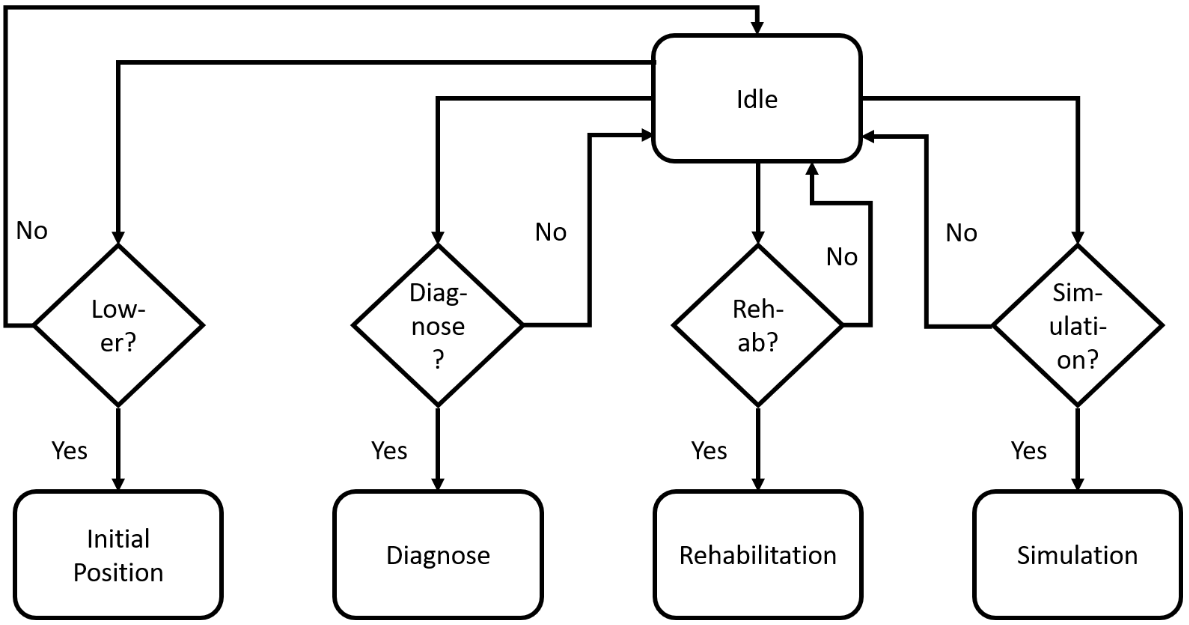


Figure 4.10: Flow chart of Idle state

4.4 Diagnose/Rehabilitation/Simulation

Diagnose, rehabilitation, and simulation are states with multiple common technical features that need addressing. Therefore, these common features are explained in this section. These three states are also referred to as modes since they are the fundamental features of CELNA - diagnose, rehabilitate, and simulate. The main common feature among these three modes is the integration of virtual reality. The feeling of being within virtual reality is what makes CELNA special, and should be available for utilization when diagnosing the patient, when rehabilitating the patient and of course when simulating games.

This Section is naturally comprehensive, as it covers how the load cells, Stewart platform, treadmill, and virtual reality are combined to form a system that enhances the feeling of being within a virtual reality. The procedure is chronologically organized in the following order, where each subsection is a description of:

- **First:**
  How the load cells are combined to form an operating area that gives the location of the user within this area and the force the person applies to it.

- **Second:**
  How the topographical information received from the virtual reality software is transformed to suit the haptic model.

- **Third:**
  The construction of a model responsible for mimicking the natural, dynamic response of the surface beneath the user's feet in different scenarios. The model behavior should be mimicked by the Stewart platform, making it respond to both pressure orientation from the user and the input from the simulation software. Such a method of controlling the Stewart platform is also referred to as Haptic control which is explained in Chapter 2, Background & Theory.

- **Fourth:**
  The emulation of accelerations.

- **Fifth:**
  How the desired pose underneath the user's feet is obtained by altering the model output transmitted to the Stewart platform. Doing so is necessary in order to achieve a realistic feeling of roll and pitch as there is a considerable distance between the original control point of the Stewart platform to the surface beneath the user's feet.

- **Sixth:**
  The necessary information transmitted to the virtual reality software in order to interact with it.

The steps and their place in the program flow within the myRIO are visualized within the blue dotted square in Figure (4.11). The conditioning of the load cell signals is grayed out since it has been covered earlier in this chapter.

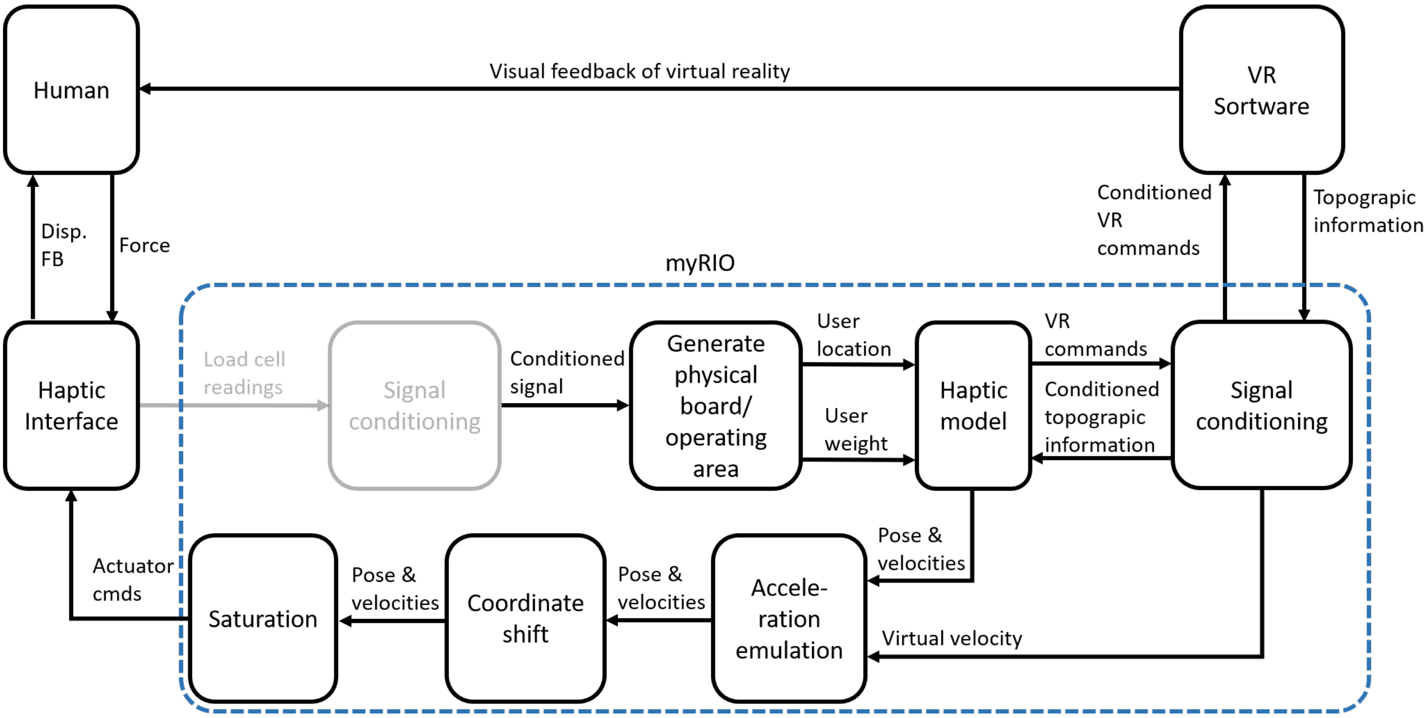


Figure 4.11: Visualization of program flow executed within the myRIO

4.4.1 Establish operating area for user localization

The load cells mounted beneath the treadmill can estimate the location of the user on top of the treadmill as well as the force the user applies using the proper relations. These features are essential for diagnosing the patient in terms of balance and weight distribution when walking, running, etc. Also, knowing the patient's location and weight distribution is required for simulating board games such as skateboarding.

Depending on where the person is located on the treadmill, each load cell will measure a specific load it is subjected to. By utilizing four load cells with similar characteristics, one can estimate where the person is located and how much force the person is applying to the treadmill. There is one area that is formed by the load cell locations - available operating area. This area is considered, as the name suggests, the available projected space of which the user can move within. The available operating area is the rectangle formed by the physical location of the four load cells located beneath the treadmill. The coordinate system of this area is two dimensional, $X_{aoa}Y_{aoa}$-plane, with origo in the dead center as visualized in Figure (4.12). Also, notice there is another coordinate system - $X_{oa}Y_{oa}$. This is the coordinate system of the area referred to as operating area, which is an area representing the physical dimensions of a virtual board. The size of this area is selectable by the operator, and its origo is determined by the user. The origo of the operating area can be anywhere within the available operating area, and its size can span beyond the physical limits of the available operating area. However, the patient is not allowed to step beyond the limits of available operating area as the formula for identifying location does not support it. One reason for expanding the operating area beyond the limits of available operating area could be when simulating a boat floating on water. The dynamics of a boat in relation to force applied is highly influenced by its size, and the projected surface area of a boat is usually larger than the available operating area. As the surface area of the haptic board models developed later in this section corresponds directly to the operating area, it is desired that the dimensions of operating area can correspond to the simulated object. Again, the only physical limitation for the user in terms of location is the available operating area.

As mentioned, the operating area is the physical area on the treadmill corresponding to the surface area of the virtual object located beneath the user's feet, typically a board of some sort. Available operating area is the same as the rectangle formed by the load cell locations. The operating area varies depending on the scenario, whether it is a snowboard, a skateboard, or a tiny boat. The reason for generating an operating area the size of the virtual board is to emulate the dynamic behavior of the board as realistic as possible. Since the material beneath the board is assumed to be uniform, it is the boards width/length ratio that distinguishes the dynamic behavior of roll from pitch and vice versa. Load cell locations, available operating area (yellow) and operating area (red) are illustrated in Figure (4.12). Both coordinate systems are unitless ranging from -1 to 1 in both directions within their regions.

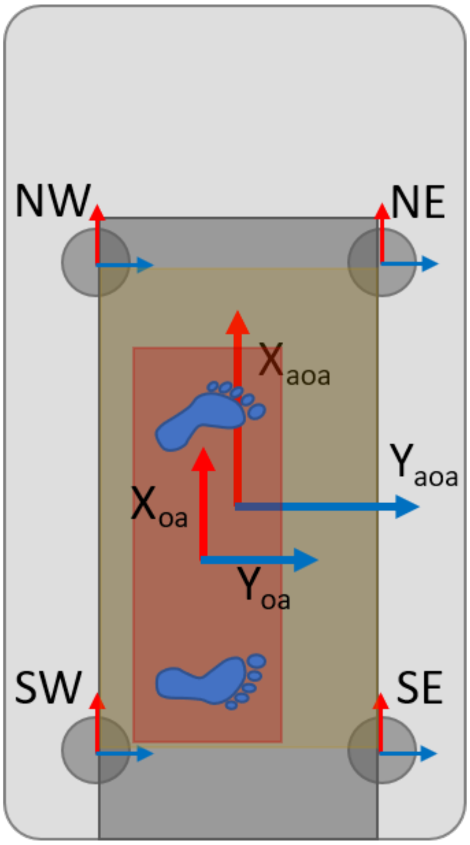


Figure 4.12: Illustration of operating area

Where the operating area is positioned within the available operating area depends on the user. It is preferred that the user sets the origo of the operating area by current posture as it might be difficult to find if it is prefixed or equal to the origo of the available operating area. One way of doing so is to first convert the load cell readings to X and Y coordinates of the person within the available operating area coordinate system, followed by shifting the coordinates to the operating area selected by the user. Equations (4.5) and (4.6) show how the X and Y coordinates of the user within the available operating area is calculated. It should be mentioned that if the person is not located on the treadmill by the load cells, for example, due to a little jump, the denominator will be very small, resulting in very sensitive calculations. This is simply prevented by setting a suiting lower limit for the denominator (limit value depends on the range and disturbances of the load cell measurements s''').

$$L_{X,aoa,p} = \frac{s'''_{NW} + s''_{NE} - s'''_{SW} - s'''_{SE}}{s'''_{NW} + s'''_{NE} + s'''_{SW} + s'''_{SE}} \tag{4.5}$$

$$L_{Y,aoa,p} = \frac{s'''_{NE} + s'''_{SE} - s'''_{NW} - s'''_{SW}}{s'''_{NW} + s'''_{NE} + s'''_{SW} + s'''_{SE}} \tag{4.6}$$

$L_{X,aoa,p}$ and $L_{Y,aoa,p}$ are unitless numbers, or factors, that lies between -1 and 1 when operated within the available operating area. To select origo for the operating area, the user positions him/herself comfortably in a suiting posture for the specific scenario. As an example, if the scenario is snowboarding, the user should stand sideways with the legs shoulder width apart. Read the user's position within the available operating area coordinates and make it the origo for the operating area coordinates. This is done by taking the mean of 100 samples as disturbances will take place, see Equations (4.7) and (4.8).

$$L_{X,aoa,oa} = \frac{L_{X,aoa,p,1} + L_{X,aoa,p,2} + L_{X,aoa,p,3} + ... + L_{X,aoa,p,n}}{n} \tag{4.7}$$

$$L_{Y,aoa,oa} = \frac{L_{Y,aoa,p,1} + L_{Y,aoa,p,2} + L_{Y,aoa,p,3} + ... + L_{Y,aoa,p,n}}{n} \tag{4.8}$$

$$n = 100$$

Now, the user's location within the operating area can be estimated by the use of Equations (4.9) and (4.9).

$$L_{X,oa,p} = L_{X,aoa,p} - L_{X,aoa,oa} \tag{4.9}$$

$$L_{Y,oa,p} = L_{Y,aoa,p} - L_{Y,aoa,oa} \tag{4.10}$$

At this point, the origo of the user's operating area is found. However, by only shifting the original coordinates, the scale of the operating area is similar to the available operating area and not necessary the simulated board. Assuming that the load cell measurements are proportional to the location of the person (remember, weight does not affect the estimated location), that issue is corrected by scaling $L_{X,oa,p}$ and $L_{Y,oa,p}$ by a geometric related factor. The geometric factor is determined by the relation between the available operating area and the desired operating area. This relationship is illustrated in Figure (4.13), where $L'_{X,oa,p}$ and $L'_{Y,oa,p}$ ranges all the way from -1 to 1 within the operating area. It is important to saturate the measurements as exceeding the boundaries of the operating area will cause the measurements to also exceed -1 to 1. This should instead cause a warning that the user is located outside the board, and must get back on before continuing. Alternatively, re-establish the operating area if the operating area can't be relocated.

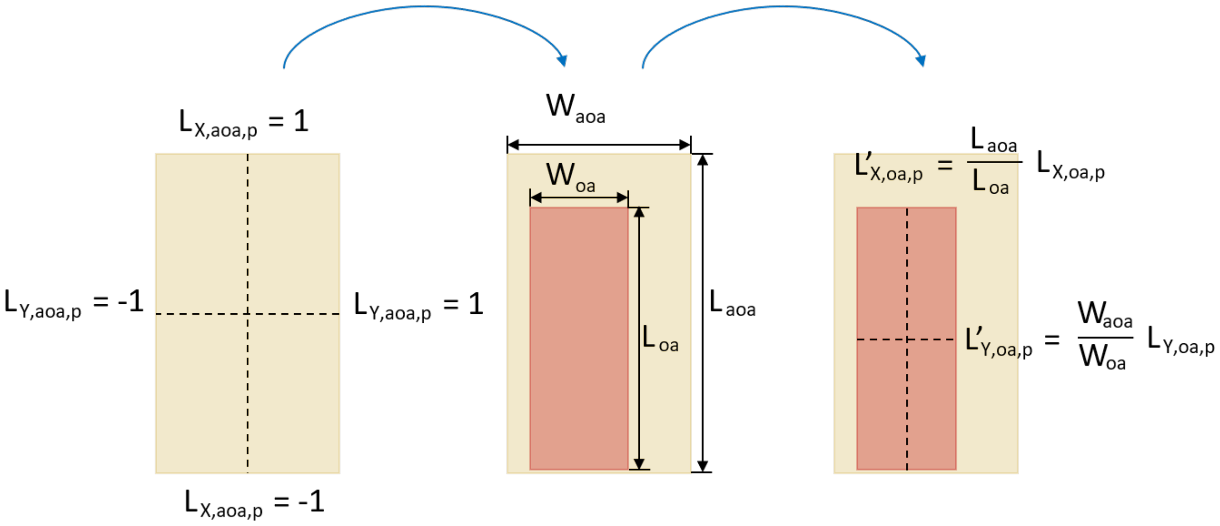


Figure 4.13: Illustration of geometric relationship between available operating area and operating area

Regarding the weight estimation, it is preferred that the weight of the person also is a unitless factor as the actual mass of the person should not have an impact on the behavior of the haptic model. This is to prevent the necessity of changing parameters for each individual using CELNA. Equation (4.11) shows how the weight is estimated by the use of three-time-conditioned load cell signals and the measured total static weight of the user. Total static weight is measured and stored during the two point calibration.

$$W_p = \frac{s'''_{NW} + s'''_{NE} + s'''_{SW} + s'''_{SE}}{W_{p,tot}} \tag{4.11}$$

If W_p is 0, no person is located on the treadmill. If W_p is 1, the person is standing still on top of the treadmill. W_p will vary around 1 depending on the dynamic influence of the person.

4.4.2 Conditioning topographic information

It is essential to know how the coordinates from the simulation software are obtained in order to utilize them properly in the models. The following conditioning is done for coordinates gathered from a game demo developed in the Unity engine due to some cooperation with a couple of game developers who uses this engine. Within this game environment, a capsule which is a controllable object represents the virtual location of the human. The capsule projects an area on the surface beneath, which yields the average surface pose in that specific area relative to global coordinates. An illustration of the capsule and coordinate orientation is shown in Figure (4.14).

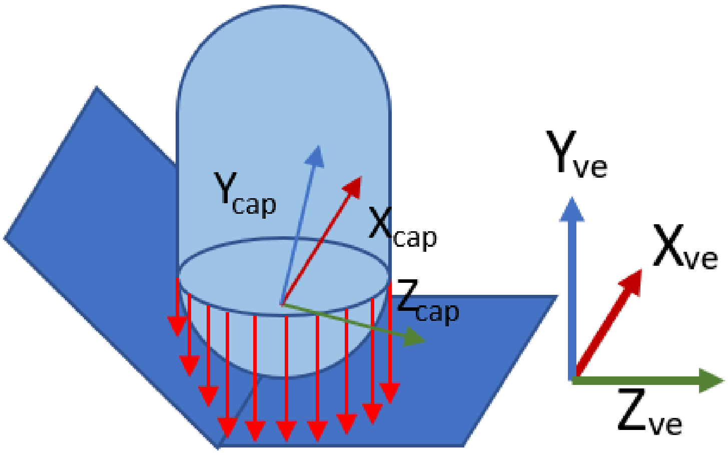


Figure 4.14: Unity capsule illustration

It should be emphasized that coordinate system *ve* is not a fixed coordinate system as it translates along the base contour of the virtual ground. It is easier explained with the use of an illustration. Figure (4.15) shows two lines - one linear (gray) and one nonlinear (black). The gray one is the "base contour" of the virtual ground of which the *va* coordinate system translates along. The black line is the actual surface contour, which is described by the orientation of the *cap* coordinate system in relation to the *va* coordinate system.

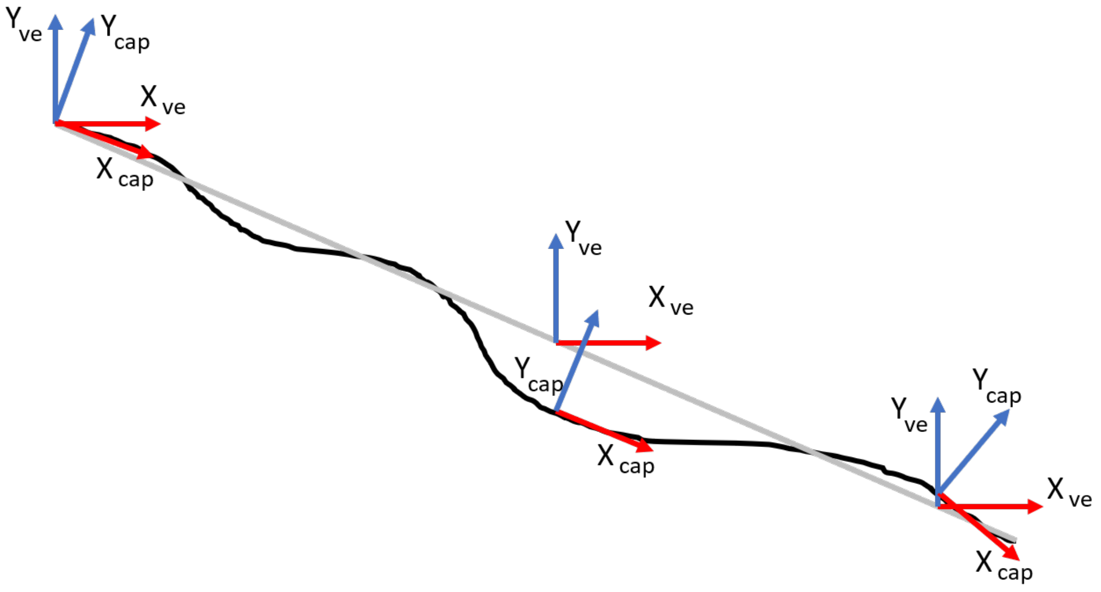


Figure 4.15: Example of "base contour" (gray) and "surface contour" (black)

The reason for not having a fixed reference is due to the heave limitations of the Stewart platform. For an artificial environment that consists of large variations in height, such as a snowboard course, the height relative to a fixed coordinate system will quickly exceed the physical limitations of the platform. Therefore, only the height of bumps and humps relative to the "base contour", gray line in Figure (4.15), is considered. It is assumed that this parameter is available from the virtual reality software.

Since the capsule is parallel with Y_{ve}-axis at all times, the orientation of the surface about Z_{ve}- and X_{ve}-axis are always known, as well as the translation in Y_{ve} direction. X_{cap}, Y_{cap}, and Z_{cap} is the coordinate system of the projected area beneath the capsule.

$T_{X,ve,cap}$, $T_{Y,ve,cap}$, $T_{Z,ve,cap}$ and $R_{X,ve,cap}$, $R_{Y,ve,cap}$, $R_{Z,ve,cap}$ are the relative translations and rotations of the surface obtained from the artificial environment. These coordinates are not oriented the same as the coordinate system of the Stewart platform, as shown in Figure (4.16). X_{ve}, Y_{ve}, and Z_{ve} is the "base contour" coordinate system of the artificial environment within Unity, whereas X_N, Y_N and Z_N is the neutral coordinate system of the Stewart platform. To obtain similarly oriented coordinates, which are necessary for the platform to mimic the virtual topography, the surface pose must be transformed to make it relative to the coordinate system of the Stewart platform.

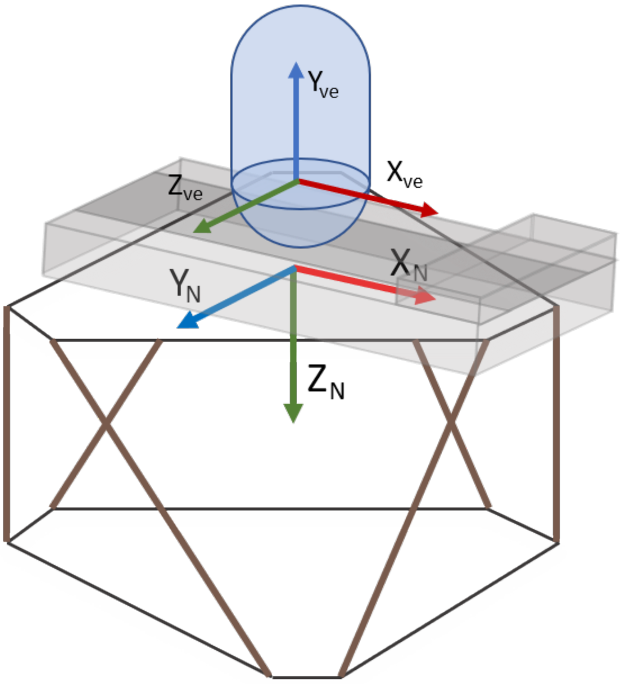


Figure 4.16: Comparing coordinate systems

Performing a rotational transformation, in this case, is quite simple, as it is a 90 degree CW rotation about the common X axis. This transformation of X_{cap}, Y_{cap} and Z_{cap} coordinates will yield the simulation coordinates needed as input to the models, X_{sim}, Y_{sim} and Z_{sim}. The relation between the two coordinate systems is shown in Equation (4.12).

$$\begin{bmatrix} X_{N,sim} \\ Y_{N,sim} \\ Z_{N,sim} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(R_{X,N,ve}) & -sin(R_{X,N,ve}) \\ 0 & sin(R_{X,N,ve}) & cos(R_{X,N,ve}) \end{bmatrix} \begin{bmatrix} X_{ve,cap} \\ Y_{ve,cap} \\ Z_{ve,cap} \end{bmatrix} \tag{4.12}$$

For this case, where $R_{X,N,ve} = -\frac{\pi}{2}$, the transformation becomes:

$$\begin{bmatrix} X_{N,sim} \\ Y_{N,sim} \\ Z_{N,sim} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} X_{ve,cap} \\ Y_{ve,cap} \\ Z_{ve,cap} \end{bmatrix} \tag{4.13}$$

Equation (4.13) concerns both translation and rotation transformation from ve-coordinates to N-coordinates. The N,sim-coordinates describe the contours of the simulated surface right beneath the capsule relative to the Stewart platform's neutral coordinate system. These coordinates are used as an input to the dynamic surface model which should in turn give the orientation of the board, or the operating area, beneath the user's feet, X_p, Y_p, and Z_p.

4.4.3 Haptic Modeling

When interacting with objects in the real world, each object has a unique feeling to the touch characterized by its shape, material properties, and inertia. For a simulation to appear realistic and give the feeling of being present within the virtual world, the sensation of touch when interacting with objects is necessary. Such emulation of touch is called haptic technology and is explained in Chapter (2), Background & Theory. Regarding CELNA, the only interaction with the virtual world is the virtual surface beneath the user's feet, which is represented by the Stewart platform in the real world. If our feet are in direct contact with the ground, the interaction is characterized by the shape and material properties of the ground. However, if there is an object between our feet and the ground, the interaction is characterized by the characteristics and material properties of the object and its interaction with the ground. How an object or surface behaves when interacted with can be approximated by a mass-spring-damper model. However, the accuracy in representation lies not only within the complexity of the model but also within the ability to recreate the model behavior with physical system. A complex model may be too power-demanding to be simulated by a real-time target, which prevents the overall physical system to recreate the proper behavior.

For example, when skateboarding down a hill, the interaction between the ground and the wheels is defined by the stiffness and damping of the skateboard's wheels and the ground itself. Since the wheels of an ordinary skateboard are very stiff, their dynamic behavior can be neglected in order to simplify the overall dynamics of the model. Many dynamic elements within a model have the potential of slowing down the processing speed, which can cause the simulation to feel unnatural no matter how good the model representation is. In addition to the ground dynamics, the trucks between the board and the wheels also have a specific behavior that must be addressed. Based on the weight distribution on top of the board as well as the trucks stiffness and damping, the amount of tilt can be modeled. Combined, a skateboard can be represented by a model as illustrated in Figure (4.17). Note that the model differs for each plane.

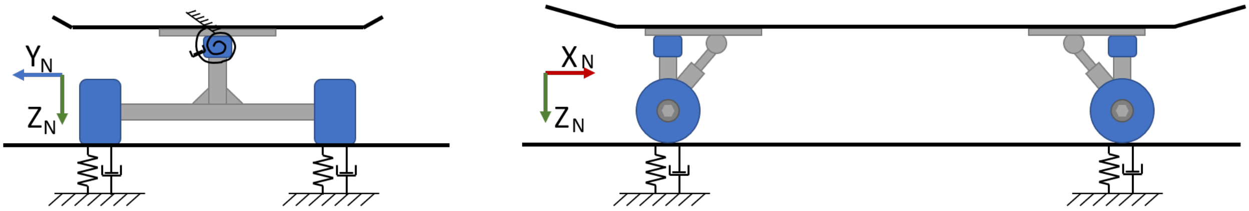


Figure 4.17: Skateboard model

Other similar examples of typically ridden boards are snowboards, surfboards, wakeboards, etc. These types of boards are very simple in design, where the dynamic behavior is mainly characterized by the material beneath the board and the board's contact area. In all cases, the contact surface beneath the board and the weight location and magnitude on top of the board determine the board's tilt angle. These types of boards can be represented by the simplified model shown in Figure (4.18). The model is simplified in such a way that the uniform pressure beneath the board is represented by two springs and dampers at each side as shown. This is a viable approach since the exact stiffness and damping of the ground beneath is not known and must be approximated through experimental testing.

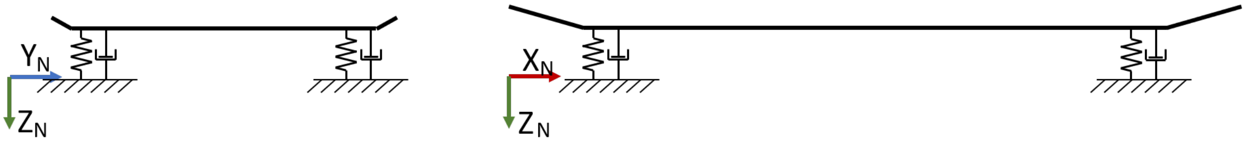


Figure 4.18: Board model

Even though the goal is to develop a model which is as efficient as possible in terms of rehabilitation and training, there is no way of proving that its efficiency is optimal. This is often the case with systems that incorporates humans, since humans have a varying and unpredictable way of responding and behaving. For such systems, a heuristic technique is often used to improve efficiency. Heuristic technique for improvement is the principle of improving something by practical tweaking and testing until sufficient results are obtained, as the optimal way is impossible to know. CELNA incorporates a human and thus a heuristic technique might be necessary to use in order to improve its efficiency concerning rehabilitation and training progress. Therefore, the main goal for the haptic model is only to resemble the behaviour of a board in a reasonable and logical manner, not to perfectly mimic it (since it is not known if perfectly mimicking is optimal). It is important to keep in mind that progression in rehabilitation and balance training is obtainable by any sort of dynamic activation of the surface beneath the patient's feet, especially under supervision by a professional. It does not require a dynamic behaviour identical to something in the real world, although some resemblance might be beneficial and worth striving for.

Modelling interaction between the surface of a board and the ground

When a person is located on a board laying on a uniform and flat ground surface, the board will be pushed into the ground due to the not-infinite-stiffness of the ground. Depending on where the person is located on the board, the amount of descendence and inclination of the board varies. If the person is located in the dead center of the board, angle between the curvature of the ground and board should ideally not occur, as illustrated in Figure (4.19).

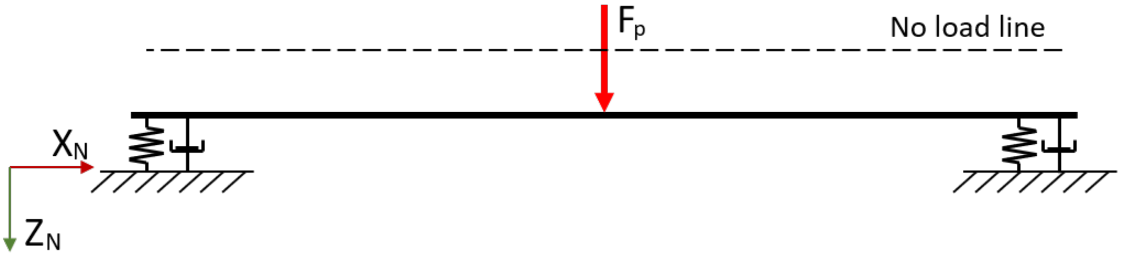


Figure 4.19: Force applied from the person to the center of the board

As soon as the applied load starts shifting along the board, variations in inclination and descendence will occur, as illustrated in Figure (4.20).

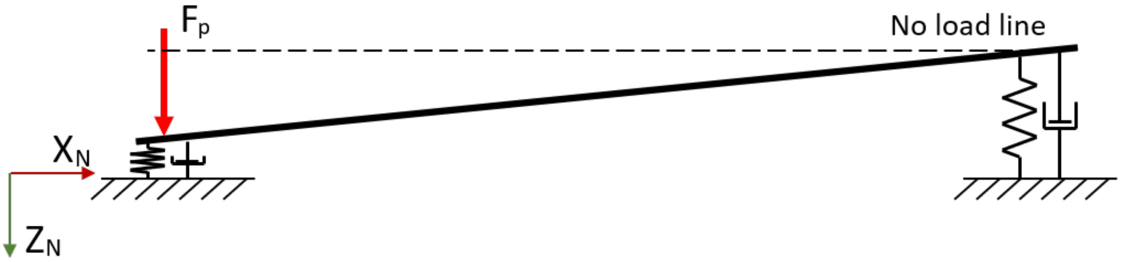


Figure 4.20: Force applied from the person to one end of the board

At this point, there are two differential equations needed to describe this system in this specific plane. One way to approach this problem is to consider the system as two individual systems independent of each other. Figure (4.21) shows a mass-spring-damper system that represents one of the two individual systems. Note that the simulated ground, z_{sim}, is a varying reference for the system. Lowercase, z_p and z_{sim}, are used to distinguish them from the coordinates of the center of the board, which has variables Z_p and Z_{sim}. z_p and z_{sim} applies locally for each of the sub-models.

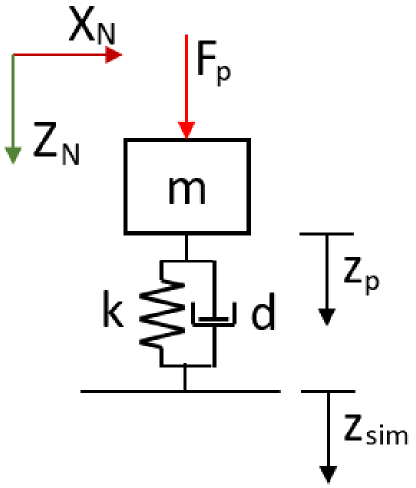


Figure 4.21: Individual system

A simplification that needs addressing is how the person's mass impacts the system. As the person moves across the board, the mass that is accelerated by each of the four systems varies, i.e. m is essentially varying. However, such a system is non-linear, meaning that it is not solvable by linear operators. A non-linear system generally requires more computing-power to simulate compared to a linear system, which could quickly cause execution time issues when those individual systems are added up to form the board-to-ground model. Hence the simplified approach of adding the external force F_p, which varies in magnitude based on the location of the person on the board and the weight applied with the purpose of emulating a varying mass. However, the inertia of the system remains fixed.

A Free Body Diagram (FBD) and Kinetic Diagram (KD) of the mass-spring-damper system illustrated in Figure (4.21) is shown in Figure (4.22). A description of the forces is presented below.

- $\mathbf{F_g}$ is a gravitational force acting in the direction of Z_N. This force has no actual relation to the mass and is merely there to provide a counterforce to the spring when there is no load applied from the person. By doing so, noise generated from the load cells will have less impact on the model's dynamic behaviour.
- $\mathbf{F_p}$ is the force dependent on the weight applied by the person and his/her location on the board. The magnitude of this force has a positive value and nonlinear characteristics, as it depends on two factors.
- $\mathbf{F_k}$ and $\mathbf{F_d}$ is the spring force and the damper force, respectively. The value of these forces is determined by the spring and damper coefficients and the difference in displacement and velocity between the simulated ground and board. The spring and damper coefficients are experimentally determined to mimic different ground materials.
- $\mathbf{F_a}$ is simply the resulting force determined by the system's mass and acceleration.

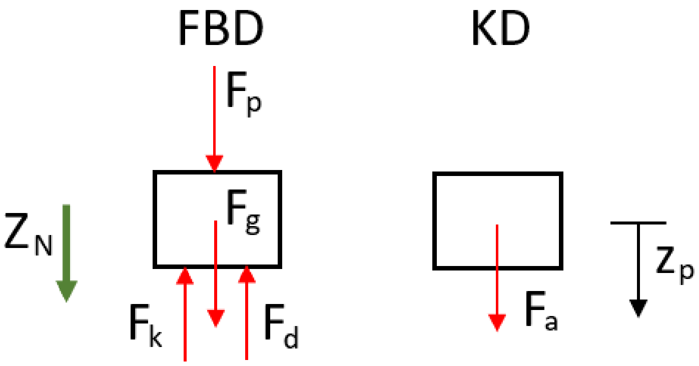


Figure 4.22: FLD and KD of mass-spring-damper system

The differential equation describing this system is shown in Equation (4.14).

$$F_p + F_g - F_k - F_d = F_a \tag{4.14}$$

Where

$$\begin{aligned} F_k &= k \cdot (z_p - z_{sim}) \\ F_d &= d \cdot (\dot{z}_p - \dot{z}_{sim}) \\ F_a &= m \cdot \ddot{z} \end{aligned}$$

Such a system is a Multiple Input Multiple Output (MIMO) system, where the inputs are the variable weight of the person, F_p, as well as the vertical velocity and position of the simulated ground, $\dot{z}_{sim}$, z_{sim} respectively. Outputs are the velocity and position of the interaction with the person, $\dot{z}_p$, z_p respectively. A MIMO system can be efficiently represented by a state-space model, which is a model type that utilizes multiple first-order linear differential equations instead of one higher-order linear differential equation to describe system behavior.

Currently, the second-order differential equation can be written as shown in Equation (4.15).

$$m \cdot \ddot{z}_p + d \cdot \dot{z}_p + k \cdot z_p \quad = \quad F_p + F_g + d \cdot \dot{z}_{sim} + k \cdot z_{sim} \tag{4.15}$$

Since F_g is constant and acts in the same direction as F_p, they are combined as F_{pg} to reduce the number of inputs to the model. Rewriting Equation (4.15) with respect to $\ddot{z}_p$ and replacing F_g and F_p with F_{pg}:

$$\ddot{z}_p \quad = \quad \frac{1}{m}(F_{pg} + d \cdot \dot{z}_{sim} + k \cdot z_{sim} - d \cdot \dot{z}_p - k \cdot z_p)$$

The states $\mathbf{x}$ and inputs $\mathbf{u}$ are as follows.

$$\mathbf{x} \quad = \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad = \quad \begin{bmatrix} z_p \\ \dot{z}_p \end{bmatrix} \qquad\qquad \mathbf{u} \quad = \quad \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \quad = \quad \begin{bmatrix} F_{pg} \\ z_{sim} \\ \dot{z}_{sim} \end{bmatrix}$$

Then, the differentiation of the states with respect to time becomes:

$$\dot{x}_1 \quad = \quad x_2$$

$$\dot{x}_2 \quad = \quad \frac{1}{m}(u_1 + k \cdot u_2 + d \cdot u_3 - d \cdot x_2 - k \cdot x_1)$$

The two first-order differential equations $\dot{x_1}$ and $\dot{x_2}$ can be assembled in a state-space model as shown in Equation (4.16).

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} \quad = \quad \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{d}{m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{m} & \frac{k}{m} & \frac{d}{m} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \tag{4.16}$$

As both states represent position and velocity directly, the C matrix should be a two-by-two identity matrix for both states to be outputted. Moreover, there is no feed-forward term for this model, which means the D matrix is a zero matrix. The whole state-space model becomes:

State matrix:

$$\mathbf{A} \quad = \quad \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{d}{m} \end{bmatrix}$$

Input matrix:

$$\mathbf{B} \quad = \quad \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{m} & \frac{k}{m} & \frac{d}{m} \end{bmatrix}$$

Output matrix:

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

Feed-forward matrix:

$$\mathbf{D} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

State equation:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$

Output equation:

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$$

To recall, the individual sub-models have three inputs; F_{pg}, z_{sim} and $\dot{z}_{sim}$, and are separated by their locations, North, South, East, and West. Figure (4.23) illustrates the location of the models, where the board is represented by the blue rectangle.

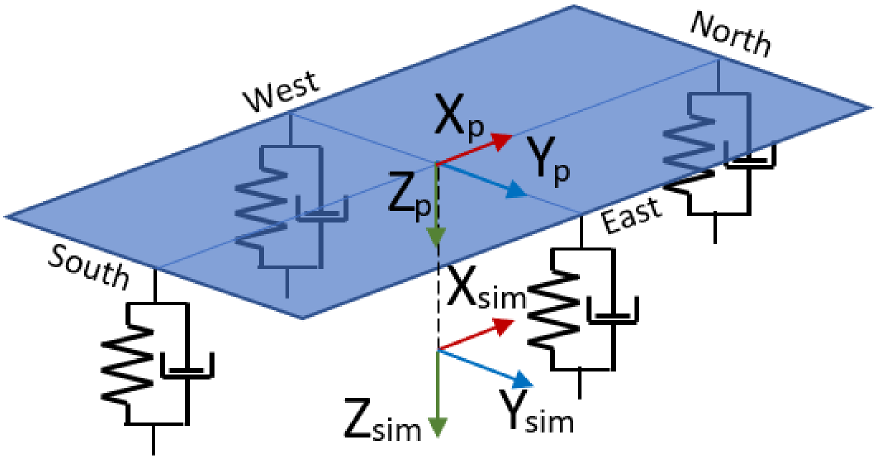


Figure 4.23: Board model

The capital noted sim-coordinates describe the ground surface right beneath the center of the board, as shown in Figure (4.23). Furthermore, it is assumed that there is a linear slope in the ground surface from North to South, and from East to West. Then, by performing some geometric calculations, it is possible to determine the individual inputs from VR software for each sub-model, z_{sim} and $\dot{z}_{sim}$. It should be noted that the force from the person and gravitation, F_{pg}, is not addressed yet. Figure (4.24) shows the geometric relations between the sim-coordinates and the sub-models input. Note that north and south are indicated by subscript addition, n and s respectively.

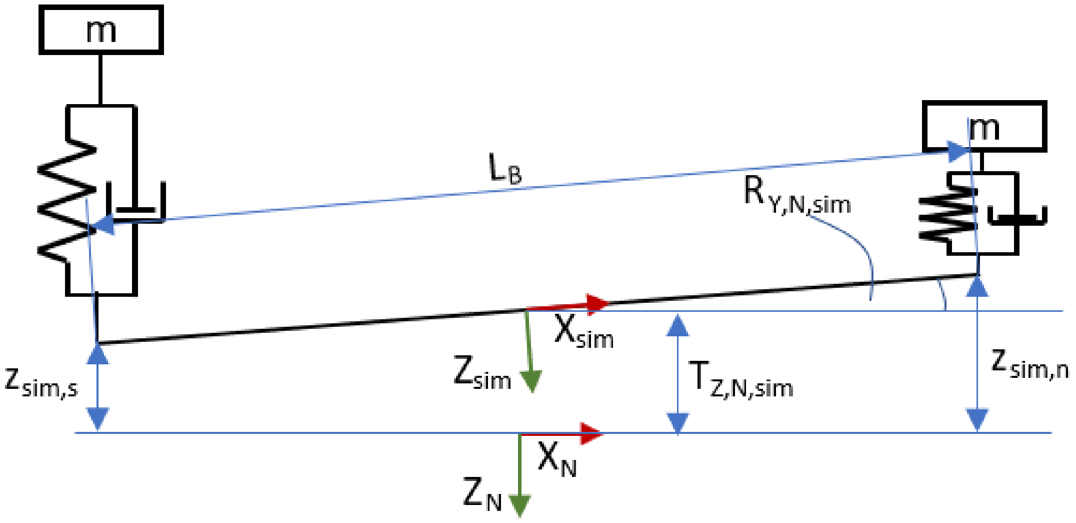


Figure 4.24: Sim to model input relation

Inputs relevant to $X_N Z_N$-plane available from the simulation are:

- Translation of sim coordinates in Z_N-direction, $T_{Z,N,sim}$
- Rotation of sim coordinates about Y_N-axis, $R_{Y,N,sim}$

The length of the board, L_B, is a predetermined length. By knowing these values, the translation and velocity of $z_{sim,s}$ and $z_{sim,n}$ can be found by utilizing Equations (4.17)-(4.20).

$$z_{sim,n} = T_{Z,N,sim} - \frac{L_B}{2} \cdot sin(R_{Y,N,sim}) \tag{4.17}$$

$$\dot{z}_{sim,n} = \dot{T}_{Z,N,sim} - \dot{R}_{Y,N,sim} \cdot \frac{L_B}{2} \cdot cos(R_{Y,N,sim}) \tag{4.18}$$

$$z_{sim,s} = T_{Z,N,sim} + \frac{L_B}{2} \cdot sin(R_{Y,N,sim}) \tag{4.19}$$

$$\dot{z}_{sim,s} = \dot{T}_{Z,N,sim} + \dot{R}_{Y,N,sim} \cdot \frac{L_B}{2} \cdot cos(R_{Y,N,sim}) \tag{4.20}$$

The velocities are not directly available from the simulation, but can be obtained by numerical derivation. At this point, two out of three inputs needed by each sub-model are defined. The remaining input signal is the force applied from the user, which is a function of the location and magnitude of the weight applied. Figure (4.25) illustrates how the magnitude and location of the force vector affect the board.

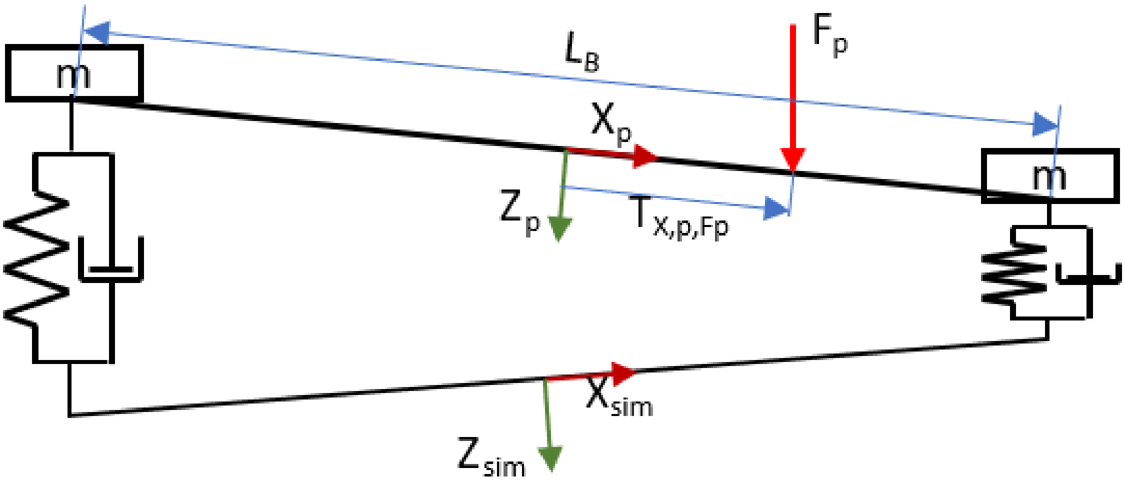


Figure 4.25: User to model input relation

How the location and magnitude of the force applied by the person, F_p, determines the forces applied to the sub-models, F_{pg}, is by two linear relationships. First, if the magnitude of F_p is constant, the magnitude of F_{pg} has a linear relationship with the location of F_p in X_p direction. If F_p is located on one of the farthest side of the board, maximum force is applied to that side and zero force is applied to the opposite side. On the other hand, if the location of F_p is constant and its magnitude varies, the magnitude of F_{pg} for both sub-models in the respective plane varies with a linear relationship to $|F_p|$. Together, these two factors form a nonlinear relationship as the force applied to the sub-models depends on two variables, $F_{pg}(|F_p|, T_{X,p,Fp})$. An illustration of F_{pg} acting on the north sub-model is shown in Figure (4.26).

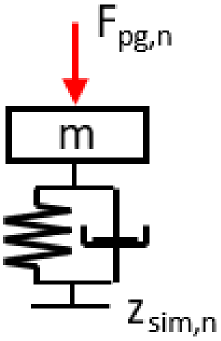


Figure 4.26: Force applied to model

One way to combine the two variables - magnitude and location, is to multiply the magnitude of the force applied from the user with a 0-1 value describing the location of the force, as done in Equation (4.21) and (4.22).

$$F_{pg,n} = \left(\frac{1}{2} + \frac{T_{X,p,Fp}}{L_B}\right) \cdot |F_p| + F_g \tag{4.21}$$

$$F_{pg,s} = \left(\frac{1}{2} - \frac{T_{X,p,Fp}}{L_B}\right) \cdot |F_p| + F_g \tag{4.22}$$

$$|F_p| = \frac{1}{2} \cdot m_p \cdot \left(\frac{W_p}{W_{p,tot}}\right) \cdot g$$

Where W_p is the real-time sum of current load cell measurements and $W_{p,tot}$ is the aum of load cell measurements when the person is standing still. m_p is a predetermined value in $[kg]$ that represents the mass of a person as the user's actual mass is irrelevant. g is the gravitational acceleration in $[{}^{m}/{}_{s^2}]$. The magnitude of the force $|F_p|$ includes a division by two assuming that the other plane (YZ) takes up approximately half of the total force at all times. $T_{X,p,Fp}$ is a function of the board model dimensions and the previously calculated factor, $L'_{Y,oa,p}$, which ranges from -1 to 1 depending on where the person is located across the board. The function is shown in Equation (4.23), notice how L_B will cancel out when inserted into Equation (4.21) and (4.22).

$$T_{X,p,Fp} = \frac{L_B}{2} \cdot L'_{X,oa,p} \tag{4.23}$$

Finally, the outputs of the model, $T_{Z,N,p}$ and $R_{Y,N,p}$, can be found. See Figure (4.27) for reference.

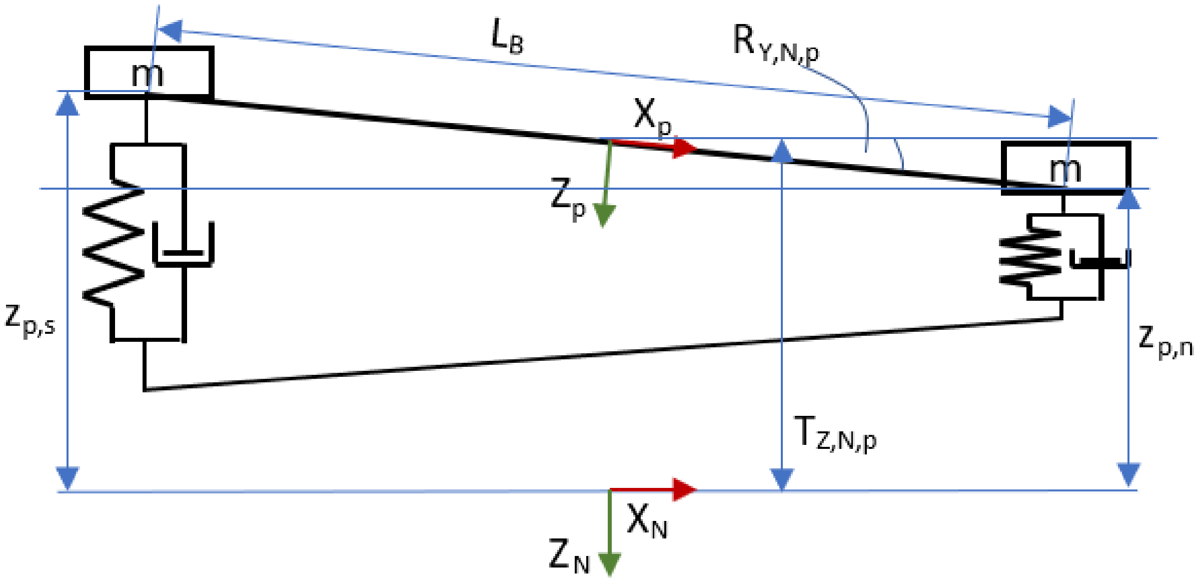


Figure 4.27: Output coordinates

Translation along Z_N-axis, $T_{Z,N,b}$, is assumed to be the average of $z_{p,s}$ and $z_{p,n}$, since the board is considered infinitely stiff. The rotation about Y_N-axis, $R_{Y,N,p}$, can be derived by the use of basic geometric relations, see Equations (4.24) - (4.27).

$$T_{Z,N,p} = \frac{z_{p,s} + z_{p,n}}{2} \tag{4.24}$$

$$\dot{T}_{Z,N,p} = \frac{\dot{z}_{p,s} + \dot{z}_{p,n}}{2} \tag{4.25}$$

$$R_{Y,N,p} = sin^{-1}\left(\frac{z_{p,s} - z_{p,n}}{L_B}\right) \tag{4.26}$$

$$\dot{R}_{Y,N,p} = \frac{\dot{z}_{p,s} - \dot{z}_{p,n}}{L_B \cdot \sqrt{1 - \left(\frac{z_{ps} - z_{p,n}}{L_B}\right)^2}} \tag{4.27}$$

Since the system is similar in the Z_N-Y_N plane, the procedure of determining the inputs and outputs of the models is the same but with different coordinates. A less explanatory procedure follows for determining the inputs and outputs of the two models in the Z_N-Y_N plane. It is important that the rotation about X_N axis complies with the right hand rule for rotation, which yields an orientation of coordinate systems as shown in Figure (4.28).

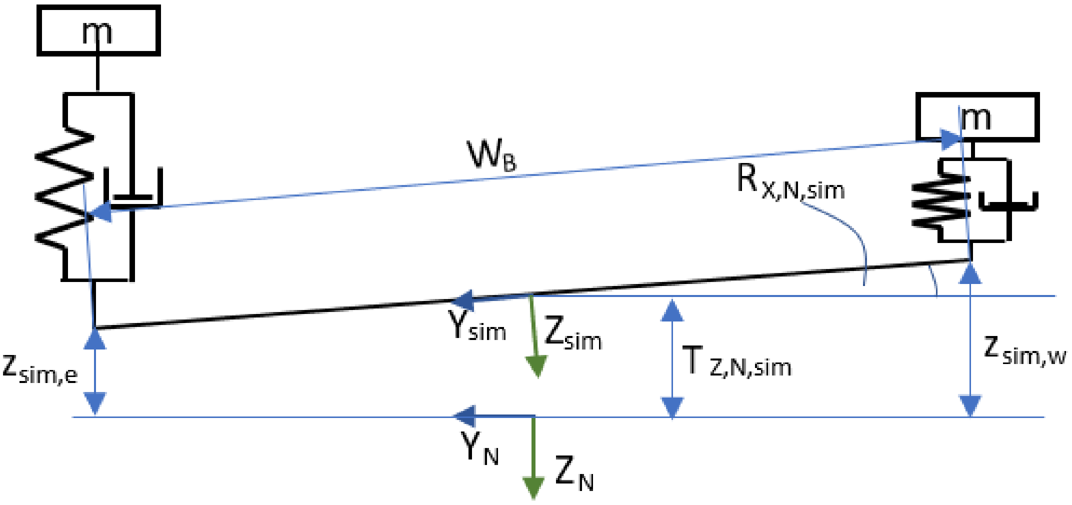


Figure 4.28: Simulation to model input relation in Y_N-Z_N plane

Inputs related to $Y_N Z_N$-plane available from the simulation are:

- Translation of sim coordinates in Z_N-direction, $T_{Z,N,sim}$
- Rotation of sim coordinates about X_N-axis, $R_{X,N,sim}$

The width of the board, W_B, is a predetermined value. By knowing these values, the translation and velocity of $z_{sim,e}$ and $z_{sim,w}$ can be found by utilizing Equations (4.28)-(4.31).

$$z_{sim,w} = T_{Z,N,sim} - \frac{W_B}{2} \cdot sin(R_{X,N,sim}) \tag{4.28}$$

$$\dot{z}_{sim,w} = \dot{T}_{Z,N,sim} - \dot{R}_{X,N,sim} \cdot \frac{W_B}{2} \cdot cos(R_{X,N,sim}) \tag{4.29}$$

$$z_{sim,e} = T_{Z,N,sim} + \frac{W_B}{2} \cdot sin(R_{X,N,sim}) \tag{4.30}$$

$$\dot{z}_{sim,e} = \dot{T}_{Z,N,sim} + \dot{R}_{X,N,sim} \cdot \frac{W_B}{2} \cdot cos(R_{X,N,sim}) \tag{4.31}$$

Defining force in the Y_N-Z_N plane. See Figure (4.29) for reference.

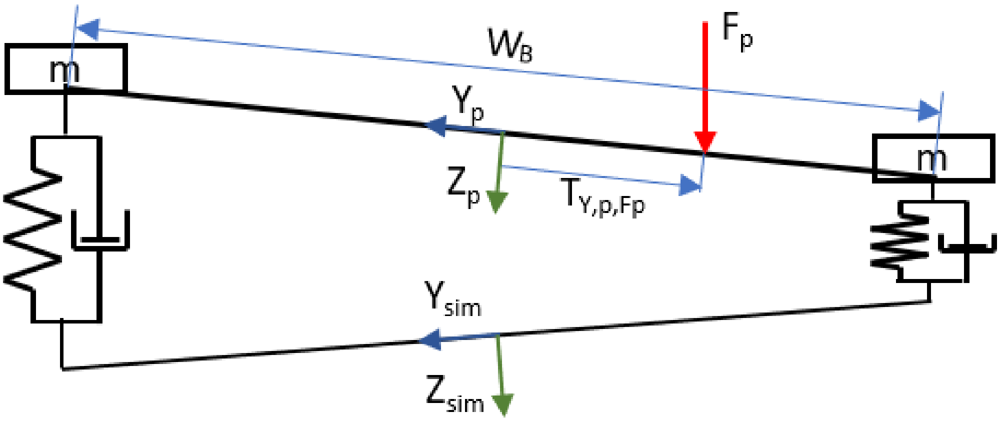


Figure 4.29: User to model input relation in Y_N-Z_N plane

Equations (4.32) and (4.33) describe how much force each model is subjected to based on the location and magnitude of the force acting from the person.

$$F_{pg,w} = \left(\frac{1}{2} - \frac{T_{Y,p,Fp}}{W_B}\right) \cdot |F_p| + F_g \tag{4.32}$$

$$F_{pg,e} = \left(\frac{1}{2} + \frac{T_{Y,p,Fp}}{W_B}\right) \cdot |F_p| + F_g \tag{4.33}$$

$$|F_p| = \frac{1}{2} \cdot m_p \cdot \left(\frac{W_p}{W_{p,tot}}\right) \cdot g$$

$$T_{Y,p,Fp} = \frac{W_B}{2} \cdot L'_{Y,oa,p}$$

As all the inputs are known, equations describing the output coordinates can be established. Use Figure (4.30) as reference for following Equations (4.34) - (4.37).

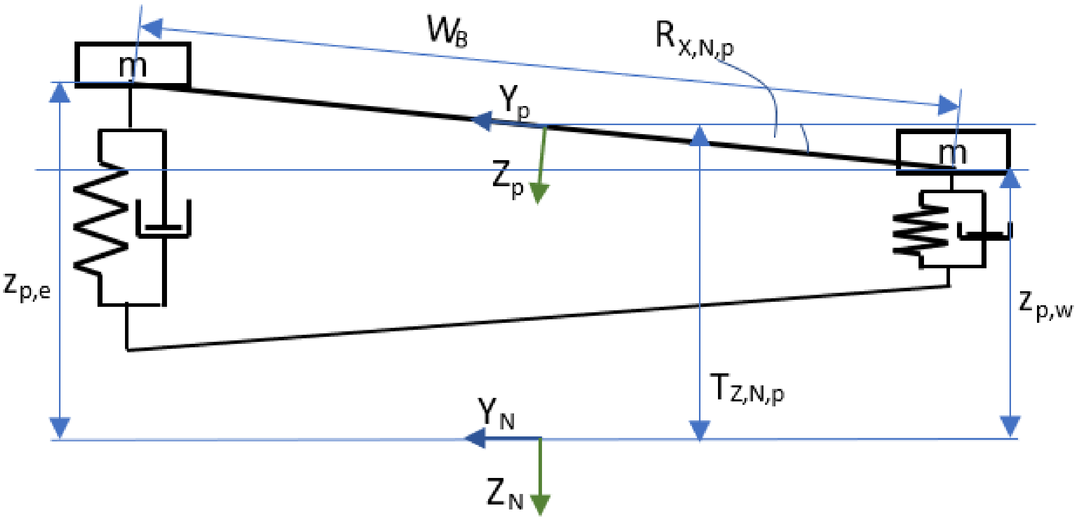


Figure 4.30: Output coordinates in Y_N-Z_N plane

$$T_{Z,N,p} = T_{Z,N,p} \tag{4.34}$$

$$\dot{T}_{Z,N,p} = \dot{T}_{Z,N,p} \tag{4.35}$$

$$R_{X,N,p} = sin^{-1}\left(\frac{z_{p,e} - z_{p,w}}{W_B}\right) \tag{4.36}$$

$$\dot{R}_{X,N,p} = \frac{\dot{z}_{p,e} - \dot{z}_{p,w}}{W_B \cdot \sqrt{1 - \left(\frac{z_{pe} - z_{p,w}}{W_B}\right)^2}} \tag{4.37}$$

Notice how $T_{Z,N,p} = T_{Z,N,p}$, that means it is equal to the one calculated in the other plane to prevent conflicting coordinates.

To summarize, input equations for each model can be seen below, followed by a flow chart of the full board model.

North model:

$$\begin{aligned}
z_{sim,n} &= T_{Z,N,sim} - \frac{L_B}{2} \cdot sin(R_{Y,N,sim}) \\
\dot{z}_{sim,n} &= \dot{T}_{Z,N,sim} - \dot{R}_{Y,N,sim} \cdot \frac{L_B}{2} \cdot cos(R_{Y,N,sim}) \\
F_{pg,n} &= \left(\frac{1}{2} + \frac{T_{X,p,Fp}}{L_B}\right) \cdot |F_p| + F_g
\end{aligned}$$

South model:

$$\begin{aligned}
z_{sim,s} &= T_{Z,N,sim} + \frac{L_B}{2} \cdot sin(R_{Y,N,sim}) \\
\dot{z}_{sim,s} &= \dot{T}_{Z,N,sim} + \dot{R}_{Y,N,sim} \cdot \frac{L_B}{2} \cdot cos(R_{Y,N,sim}) \\
F_{pg,s} &= \left(\frac{1}{2} - \frac{T_{X,p,Fp}}{L_B}\right) \cdot |F_p| + F_g
\end{aligned}$$

East model:

$$\begin{aligned}
z_{sim,e} &= T_{Z,N,sim} + \frac{W_B}{2} \cdot sin(R_{X,N,sim}) \\
\dot{z}_{sim,e} &= \dot{T}_{Z,N,sim} + \dot{R}_{X,N,sim} \cdot \frac{W_B}{2} \cdot cos(R_{X,N,sim}) \\
F_{pg,e} &= \left(\frac{1}{2} + \frac{T_{Y,p,Fp}}{W_B}\right) \cdot |F_p| + F_g
\end{aligned}$$

West model:

$$\begin{aligned}
z_{sim,w} &= T_{Z,N,sim} - \frac{W_B}{2} \cdot sin(R_{X,N,sim}) \\
\dot{z}_{sim,w} &= \dot{T}_{Z,N,sim} - \dot{R}_{X,N,sim} \cdot \frac{W_B}{2} \cdot cos(R_{X,N,sim}) \\
F_{pg,w} &= \left(\frac{1}{2} - \frac{T_{Y,p,Fp}}{W_B}\right) \cdot |F_p| + F_g
\end{aligned}$$

An illustration of the board models internal data flow is shown in Figure (4.31).

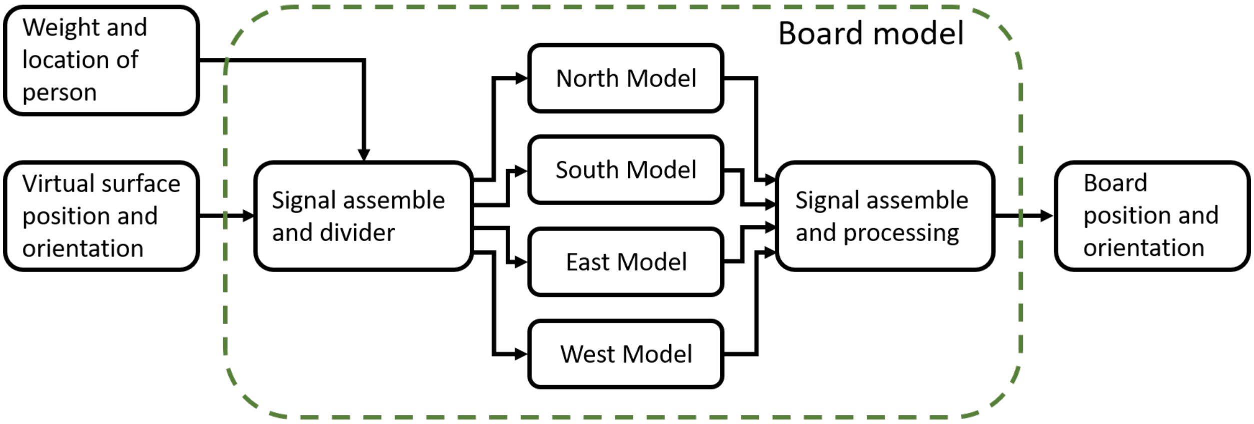

Weight and location of person
Virtual surface position and orientation
Board model
Signal assemble and divider
North Model
South Model
East Model
West Model
Signal assemble and processing
Board position and orientation

Figure 4.31: Flow chart of board model

This board model was verified in Matlab using dynamic 3D-plot. A few steady-state examples are shown in Figure (4.32). The black rectangle represents the contour of the virtual ground gathered from the virtual reality software. The blue rectangle represents the board and the red line represents the location and magnitude of the applied force. Even though not shown in this specific figure, the dynamic orientation of the ground also affects the board orientation as expected. The coordinates from the simulation and the models are transformed to suit the coordinate system of the Matlab 3D-plot. The script used for the Matlab simulation can be seen in Appendix (E.1).

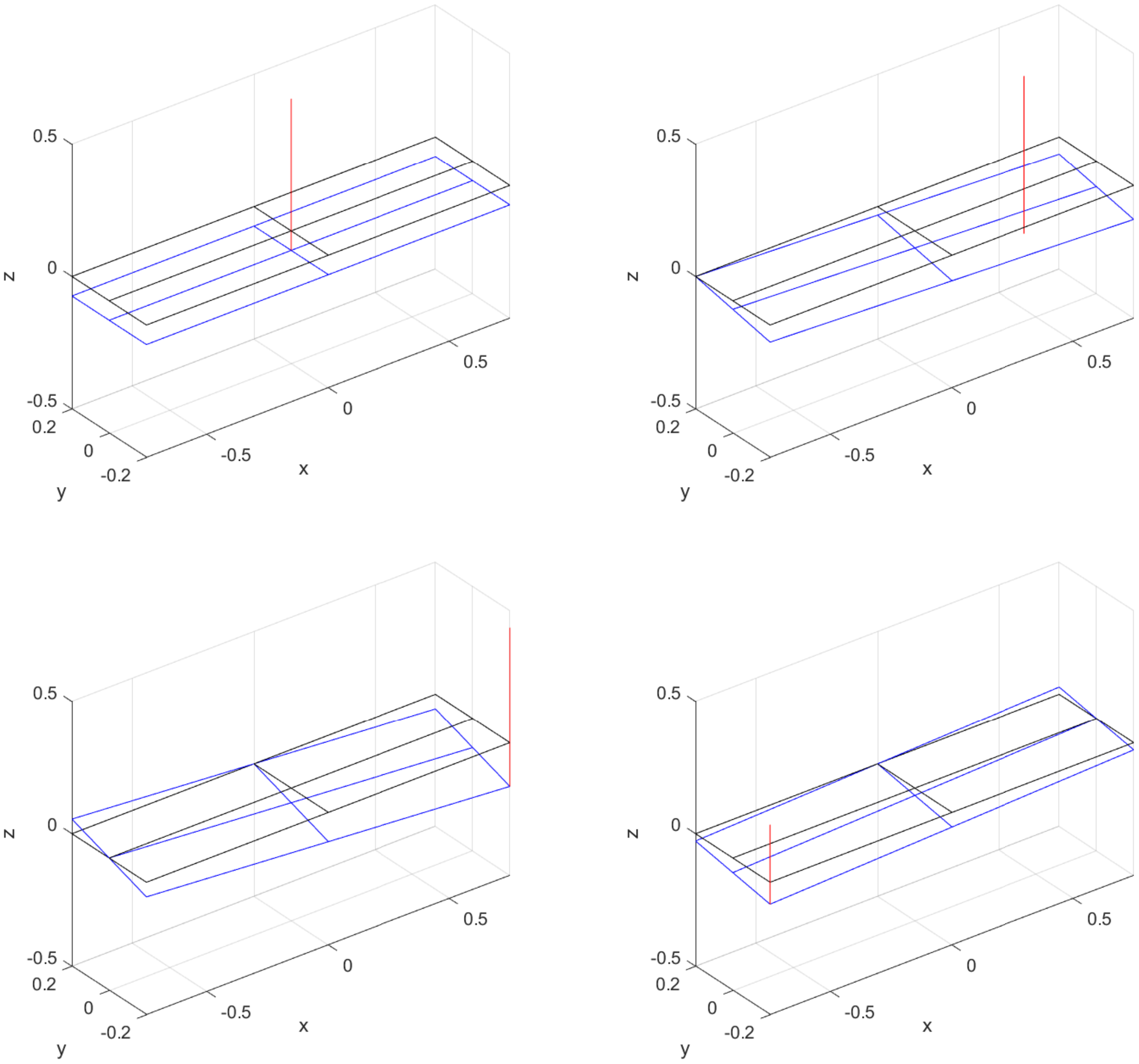


Figure 4.32: Verification of model in Matlab

The model is a general board model and is very versatile concerning its ability to simulate different types of scenarios. It has a direct relation to scenarios such as snowboarding, wakeboarding, and surfboarding. In addition, it can be very useful for balance rehabilitation training since its parameters are easily changed to adjust the level of difficulty to suit the patient.

Skateboard model

A skateboard behaves slightly differently from a plain board which is in direct contact with the ground. Therefore, a separate model is made for this scenario. A skateboard can be described as shown in the introduction to this subsection, Figure (4.33).

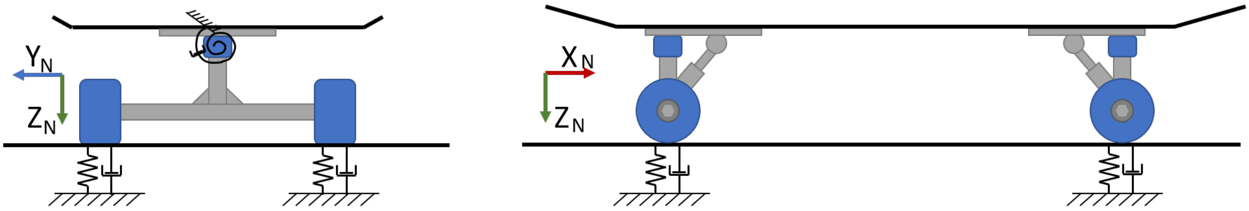


Figure 4.33: Skateboard model

However, by assuming that the skateboard runs hard wheels on a hard ground surface such as asphalt, it is possible to further simplify the skateboard model. Figure (4.34) illustrates how the skateboard can be simplified by assuming an infinitely stiff ground surface.

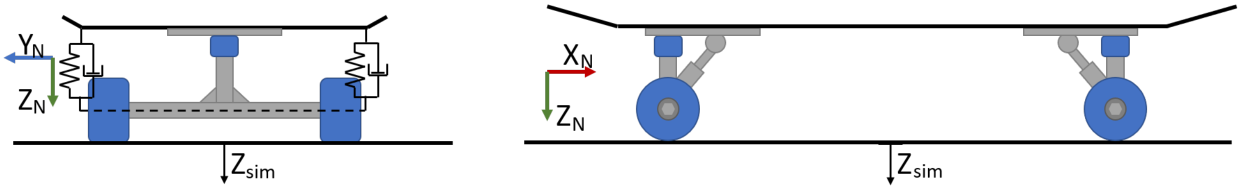


Figure 4.34: Skateboard model with assumptions of stiffness

One key feature that distinguishes a skateboard model from the general board model is the pivot point at the trucks of which the board rotates about. By assuming infinitely stiff ground surface, wheels and trucks, this pivot point will follow the contour of the ground. Also, in X_N, Z_N-plane the skateboard will follow the contour of the ground due to stiff interaction between the wheels and the ground. Since the pivot point follows the contour of the ground and the board itself is considered infinitely stiff, there will be a linear and opposite relationship between the behavior of west and east side of the board around the pivot. Therefore, even further simplifications can be done by removing one of the two sub-models. By doing so, only one sub-model has to be simulated to generate the dynamic behavior of the skateboard. Figure (4.35) shows the further simplified model of the skateboard.

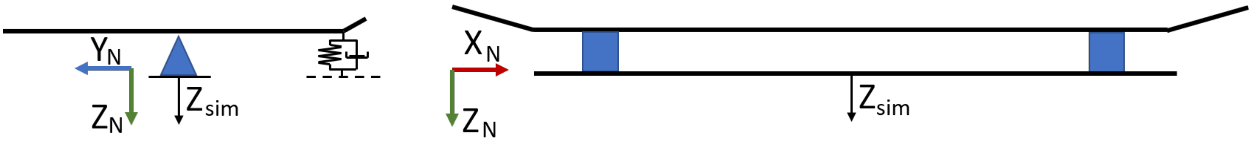


Figure 4.35: Simplified skateboard model

For this system, the state-space model representation of the mass-spring-damper system remains the same. However, it is important to keep in mind that even though the number of sub-models is reduced from two to one, the one sub-model must have parameters that make it equivalent to two. Opposed to the general board model, the translation in Z direction gathered from the virtual reality software will not affect the dynamic behavior of the sub-model, since it works against the stiff pivot point. However, the rotation of the virtual ground will on the other hand have an influence on the sub-model similar to the general board model. Figure (4.36) is an illustration of the sub-model and some geometric relations used to form the skateboard model. Note that H_{SB} represents the height of the skateboard, but as it has no major impact on the sole experience of riding a skateboard, it is treated as zero.

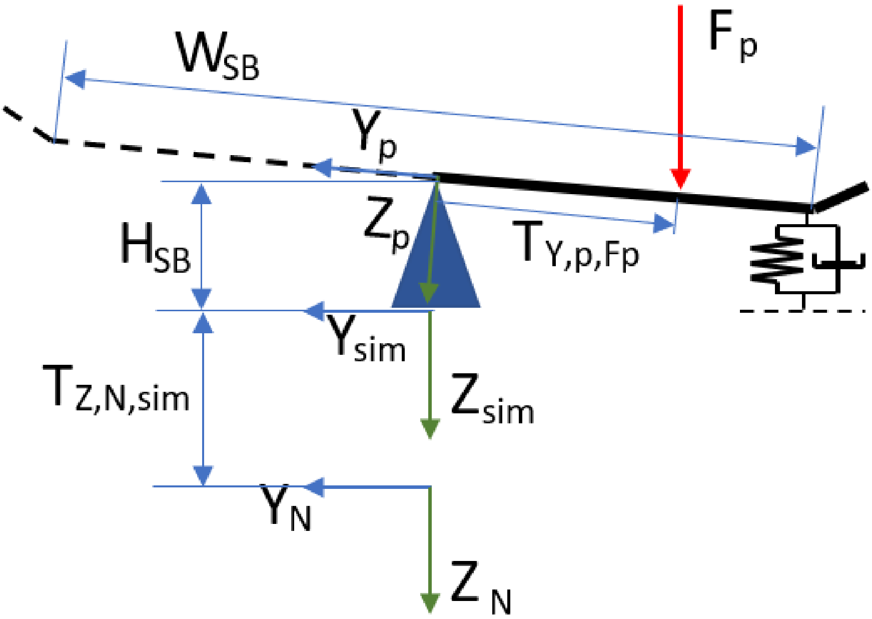


Figure 4.36: Further simplified skateboard model

The inputs needed by the skateboard's sub-model are similar to those of the general model, except translation in Z-direction as well as its velocity is not necessary. Neither is the rotation about the Y-axis, as there is no dynamic behavior characterized by a mass-spring-damper in the Y_N, Z_N-plane.

Figure (4.37) shows the west mass spring damper model. The force input, $F_{pg,w}$, is derived slightly differently from the general board model due to the pivot point. It is wanted that when the load from the person is applied at the center of the board, none is applied to the spring. Also, a negative force shall act on it when the person is on the opposite side of the pivot point. The three inputs to the skateboard model can be seen in Equations (4.38)-(4.40).

$$z_{sim,w} = -\frac{W_{SB}}{2} \cdot sin(R_{X,N,sim}) \tag{4.38}$$

$$\dot{z}_{sim,w} = -\dot{R}_{X,N,sim}\frac{W_{SB}}{2} \cdot cos(R_{X,N,sim}) \tag{4.39}$$

$$F_{pg,w} = -\frac{T_{Y,p,Fp}}{W_{SB}} \cdot |F_p| + F_g \tag{4.40}$$

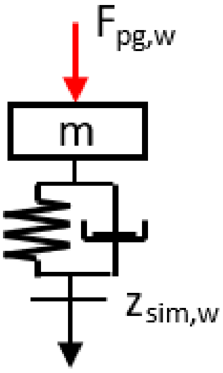


Figure 4.37: Force applied to sub-model

The output of the model in the Y_N, Z_N-plane then becomes as shown in Equations (4.41) - (4.44).

$$T_{Z,N,p} = T_{Z,N,sim} \tag{4.41}$$

$$\dot{T}_{Z,N,p} = \dot{T}_{Z,N,sim} \tag{4.42}$$

$$R_{X,N,p} = sin^{-1}\Big(\frac{2 \cdot (-z_{p,w})}{W_{SB}}\Big) \tag{4.43}$$

$$\dot{R}_{X,N,p} = \frac{2 \cdot (-\dot{z}_{p,w})}{W_{SB} \cdot \sqrt{1 - \Big(\frac{2 \cdot (-z_{p,w})}{W_{SB}}\Big)^2}} \tag{4.44}$$

In X_N, Z_N-plane, the skateboard is stiff and its orientation therefore becomes the same as the software simulated ground surface, see Equations (4.45) - (4.48).

$$T_{Z,N,p} = T_{Z,N,sim} \tag{4.45}$$

$$\dot{T}_{Z,N,p} = \dot{T}_{Z,N,sim} \tag{4.46}$$

$$R_{Y,N,p} = R_{Y,N,sim} \tag{4.47}$$

$$\dot{R}_{Y,N,p} = \dot{R}_{Y,N,sim} \tag{4.48}$$

4.4.4 Acceleration emulation

This subsection will cover the technical approach of emulating the accelerations by the use of gravitational acceleration and the Stewart platform. There are mainly two types of accelerations occurring under normal operation of a skateboard, snowboard, and similar - centripetal acceleration and acceleration in the riding direction. Accelerations of a board where a person is located on top, forces the person to lean toward the direction of acceleration in order to prevent falling over.

Taking a human as an example, where the feet are not located at the mass center. To prevent falling over, it is necessary that the acceleration of his/her mass center is the same as their feet. Acceleration of their feet depends on the acceleration of the board, which has a direct relationship to its roll angle.

Shown in Figure (4.38) is an FBD of a mass attached to a mass-less beam. The bottom of the beam is subjected to a horizontal force, in this case centripetal, and a normal force. The force due to gravitational accelerations, occurs at the mass center, as shown. To figure out the proper tilt angle, Newton's 2nd law for rotating objects is used, Equation (4.49).

$$\sum M = I \cdot \ddot{\theta} \tag{4.49}$$

For this example:

$$F_{cp} \cdot h - F_N \cdot w \quad = \quad I_A \cdot \ddot{\theta}_A$$

I_A is not zero as the object has a mass and a volume. For the rotational acceleration around the mass center, $\ddot{\theta}_A$, to be zero, Equation (4.50) must be true.:

$$\begin{aligned} F_{cp} \cdot h &= F_N \cdot w \\ \Longrightarrow \frac{w}{h} &= \frac{F_{cp}}{F_N} \end{aligned} \tag{4.50}$$

Then, from the force equations, neglecting the influence from acceleration in y-direction (simplification done in order to prevent the need for differentiating the simulation output with respect to time twice):

$$\begin{aligned} \sum F_x &= F_{cp} = m_A \cdot \ddot{x}_A \\ \sum F_y &= F_N - F_{g,A} = m_A \cdot \ddot{y}_A = 0 \\ \Longrightarrow F_N &= F_{g,A} = m_A \cdot g \end{aligned}$$

Substituting F_{cp} and F_N into Equation (4.50):

$$\begin{aligned} \frac{w}{h} &= \frac{\cancel{m_A} \cdot \ddot{x}_A}{\cancel{m_A} \cdot g} \\ \Longrightarrow \frac{w}{h} &= \frac{\ddot{x}_A}{g} \end{aligned}$$

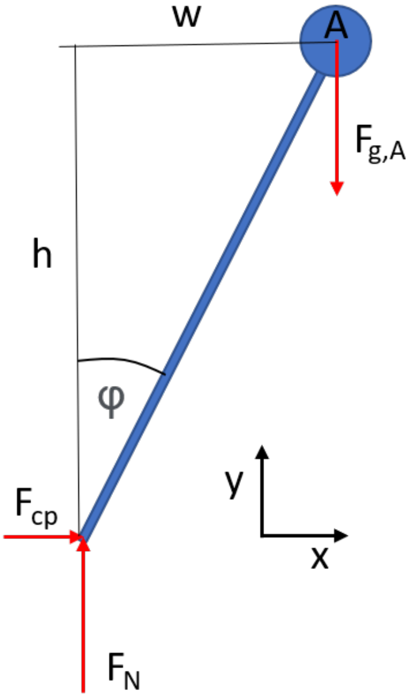


Figure 4.38: Illustration of forces

g is gravitational acceleration. Tilt angle needed to prevent the object from rotating i.e. falling over can be found by using geometric relationship.

$$\begin{aligned} tan(\phi) &= \frac{w}{h} \\ \Longrightarrow tan(\phi) &= \frac{\ddot{x_A}}{g} \end{aligned}$$

$$\Longrightarrow \phi = tan^{-1}\left(\frac{\ddot{x}_A}{g}\right) \tag{4.51}$$

As can be seen from Equation (4.51), the tilt angle is directly determined by the acceleration of object A, which should be similar to the acceleration happening at the person's feet to prevent falling over. But what is the acceleration at the person's feet?

The formula for centripetal acceleration can be seen in Equation (4.52), [40].

$$\ddot{x}_{cp} = \frac{v_t^2}{r_T} \tag{4.52}$$

Where v_t is the tangential velocity and r_T is the radius of the turn.

By substituting $\ddot{x}_{cp}$ for $\ddot{x}_A$ in Equation (4.51), Equation (4.53) is obtained which describes the tilt angle with respect to velocity, turn radius and gravitational acceleration. Notice that ϕ describes rotation about X-axis (roll), which connects it to the coordinates of the Stewart platform.

$$\phi_X \quad = \quad tan^{-1}\left(\frac{v_t^2}{r_T \cdot g}\right) \tag{4.53}$$

Turn radius is, for most boards, assumed to be proportionally related to tilt angle. The accuracy of the relationship factor is not of importance since every board has to be tested anyway in order to know exactly how it responds. Therefore, a suiting relationship factor is determined by experimental testing. The tangential velocity, v_T, is a known variable and gravitational acceleration is a known constant. Following is an illustration, Figure (4.39), that illustrates an example where a person is riding a skateboard and leaning towards the center of the turn to compensate for the centripetal acceleration (left). Note that the centrifugal force, F_{cf}, is a fictitious force, i.e. it does not lead to motion, but is rather a force that appears to act on a mass whose motion is described using a non-inertial frame of reference [41]. However, it is visually included as it illustrates what a person feels when accelerating.

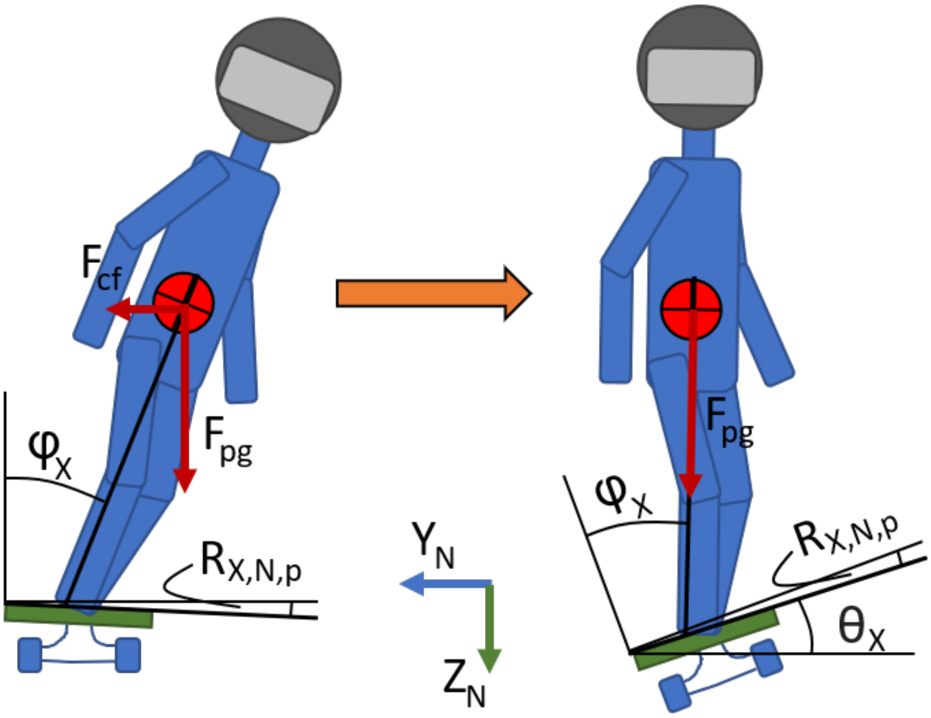


Figure 4.39: Fictitious force emulation

A body orientation similar to the one left in the illustration is not physically possible without falling over while at the Stewart platform since it can not emulate the centripetal acceleration by accelerating to in X or Y, as its movement is limited. Instead, it is possible to utilize the already present gravitational acceleration by making it point in the same direction as the sum of F_{pg} and F_{cf}.

ϕ_X is the needed tilt angle to compensate for the centripetal acceleration, $R_{X,N,p}$ is the rotation about X_N axis gathered from the model, and θ_X is the angle needed from the Stewart platform in order to emulate the feeling of leaning into the turn. However, the feeling will deviate slightly from the real world as the magnitude of the force is constant, whereas the magnitude of the actually combined force vector when performing a turn at velocity will increase. From the illustration, angle θ_X can be derived as shown in Equation (4.54).

$$
\begin{aligned}
\theta_X &= \frac{\pi}{2} - (\frac{\pi}{2} + \phi_X) + R_{X,N,p} \\
\Longrightarrow \theta_X &= R_{X,N,p} - \phi_X
\end{aligned}
\tag{4.54}
$$

Note that positive rotation is counterclockwise. Now, if it is desired, fictitious centrifugal force emulation can be activated by using θ_X to describe the desired rotation about X_N axis instead of just $R_{X,N,p}$.

A similar approach can be used to emulate accelerations along the ridden path, i.e. the Stewart platforms X_N direction. When accelerating and deaccelerating, a fictitious force similar to the centrifugal force is felt by the person. Figure (4.40) illustrates how this fictitious force can be emulated in the same way by rotating the Stewart platform to the point where the gravitational force vector and the sum of F_{pg} and F_{fict} have the same direction.

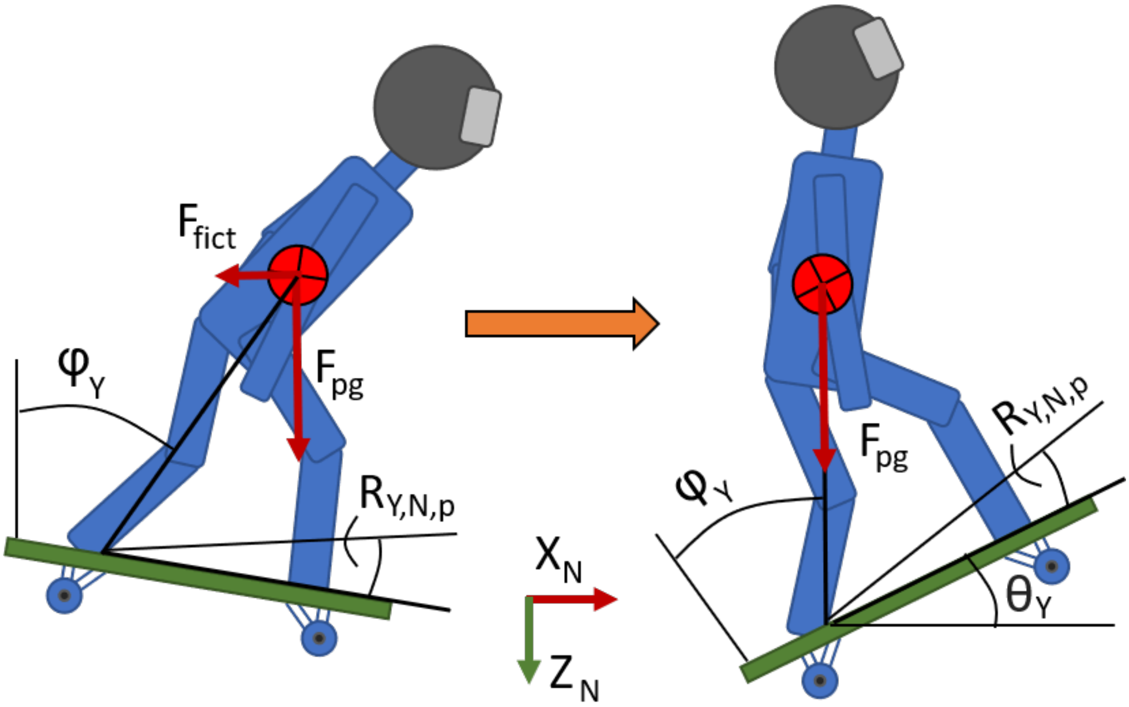


Figure 4.40: Fictitious force emulation

Opposed to the case with centripetal acceleration, the acceleration in travel direction is a known variable. Therefore Equation (4.55) is sufficient to use.

$$
\phi_Y = tan^{-1}\left(\frac{a_{t,sim}}{g}\right) \tag{4.55}
$$

Then, the rotation about the Y_N-axis needed from the Stewart platform to emulate the fictitious force due to acceleration and deacceleration is described by θ_Y in Equation (4.56).

$$
\theta_Y = R_{Y,N,p} - \phi_Y \tag{4.56}
$$

4.4.5 Shifting Stewart Platform control point

As previously mentioned, the pose of the Stewart platform depends on the input vector it receives. Of course, the input vector, which contains pose, velocities, and even accelerations, describes the desired pose and dynamic behavior of one specific point of the platform. This point can be considered the Stewart platforms control point, which lies in a location predetermined by Rexroth. This point lies somewhere around the center of the top plate.

Since the virtual ground for the user is the operation area which lies on top of the treadmill striding belt, rotation about the original control point would feel very unnatural. Therefore, some kinematics must be done between the original control point and the desired control point to achieve control of the correct point. Figure (4.41) illustrates the two points and their locations. Although it seems like the desired control point is right above the original, it might not be the case. Thus, the opportunity to shift it in all directions is necessary. This is not only to account for the variable operating area but also due to the fact that there is no guarantee that the center of the treadmill lines up with the control point when installed.

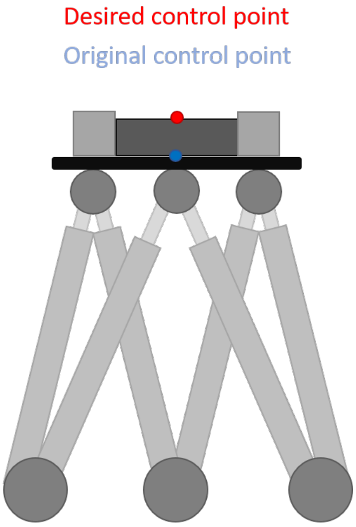


Figure 4.41: Shifted point of rotation

Figure (4.42) shows an illustration of a Stewart platform pose in $X_N Z_N$-coordinates. X_p and Z_p is the coordinate system of the point beneath the user's feet - desired control point, and X_b and Z_b is the coordinate system of the point in the center of the platforms top plate - original control point. Although it is represented as a two-dimensional coordinate system, imagine the Y_N axis pointing out of the paper.

The other parameters are:

- a, b, and c are the fixed translations $T_{X,p,b}$, $T_{Y,p,b}$ and $T_{Z,p,b}$ respectively, between the desired control point and the original. **b** is pointing out of the paper. Their values are positive or negative depending on their directions within the neutral, N, coordinate system.
- Z_{mo} is the model output. The model is developed to translate only in Z_N. See Subsection (4.4.3) for model development.

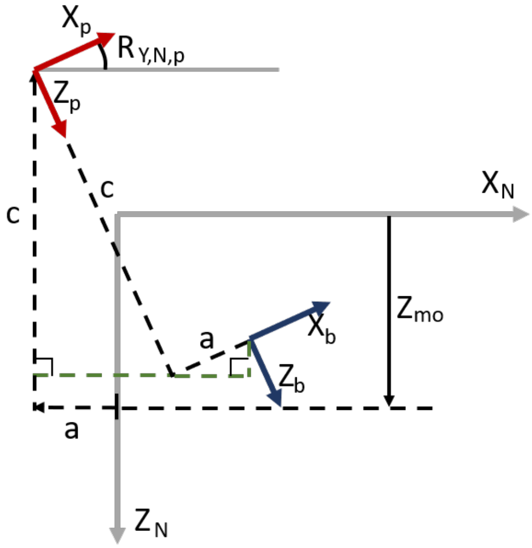


Figure 4.42: Control point shift

Note that when $R_{Y,N,p} = 0$, $R_{X,N,p} = 0$ and $Z_{mo} = 0$, i.e. no model output, the pose of the Stewart platform will be zero. This is due to the a, b, and c shifts of X_p, Y_p and Z_p relative to the neutral coordinate system. Following are the derivations of the transformation matrices for each of the two planes where rotation occurs - $X_N Z_N$ and $Y_N Z_N$. For positive a, b, and c values, the rotation matrix about Y_N becomes as shown in Equation (4.57).

$$\begin{aligned} X_{N,b} &= c \cdot sin(R_{Y,N,p}) + a \cdot cos(R_{Y,N,p}) \\ Y_{N,b} &= b \\ Z_{N_b} &= c \cdot cos(R_{Y,N,p}) - a \cdot sin(R_{Y,N,p}) \end{aligned}$$

In matrix form:

$$\begin{bmatrix} X_{N,b} \\ Y_{N,b} \\ Z_{N,b} \end{bmatrix} = \underbrace{\begin{bmatrix} cos(R_{Y,N,p}) & 0 & sin(R_{Y,N,p}) \\ 0 & 1 & 0 \\ -sin(R_{Y,N,p}) & 0 & cos(R_{Y,N,p}) \end{bmatrix}}_{\text{Transformation matrix, } \mathbf{TR_{Y,p,b}}} \begin{bmatrix} a \\ b \\ c \end{bmatrix} \tag{4.57}$$

Similarly, Equation (4.58) shows the transformation matrix for rotation about X_N. Illustration is not included as the approach is the same.

$$\begin{aligned} X_{N,b} &= a \\ Y_{N,b} &= -c \cdot sin(R_{X,N,p}) + b \cdot cos(R_{X,N,p}) \\ Z_{N_b} &= c \cdot cos(R_{X,N,p}) + b \cdot sin(R_{X,N,p}) \end{aligned}$$

In matrix form:

$$\begin{bmatrix} X_{N,b} \\ Y_{N,b} \\ Z_{N,b} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(R_{X,N,p}) & -sin(R_{X,N,p}) \\ 0 & sin(R_{X,N,p}) & cos(R_{X,N,p}) \end{bmatrix}}_{\text{Transformation matrix, } \mathbf{TR_{X,p,b}}} \begin{bmatrix} a \\ b \\ c \end{bmatrix} \tag{4.58}$$

By multiplying transformation matrix $\mathbf{TR_{X,p,b}}$ with $\mathbf{TR_{Y,p,b}}$, the transformation of coordinate system b relative to N for any rotation about X_N and Y_N can be described in 3D, Equation (4.59). The multiplication order does not matter for this case as both rotations are relative to the neutral coordinate system N.

$$\mathbf{TR_{XY,p,b}} = \mathbf{TR_{X,p,b}} \cdot \mathbf{TR_{Y,p,b}} \tag{4.59}$$

However, for the case illustrated in Figure (4.42), a, b and c have negative values. Hence the negative signs of a, b, and c in the matrix multiplication within Equation (4.60), which describes the final coordinates of coordinates system b relative to N, including the translational offset.

$$\begin{bmatrix} X_{N,b} \\ Y_{N,b} \\ Z_{N,b} \end{bmatrix} = \mathbf{TR_{XY,p,b}} \begin{bmatrix} -a \\ -b \\ -c \end{bmatrix} + \begin{bmatrix} a \\ b \\ c + Z_{mo} \end{bmatrix} \tag{4.60}$$

Similar to position, the translational velocity of the original control point is also affected by the shift and is required to derive in order to prevent conflicting position and velocity commands. The velocity of the original control point is a function of rotation, rotational velocity and velocity in Z_N direction provided by the model output. The function can be derived by differentiating Equation (4.60) with respect to time, as shown in Equation (4.61). Both transformation matrices $\mathbf{TR_{XY,p,b}}$ and $\dot{\mathbf{TR}}\mathbf{_{XY,p,b}}$ can be seen in Appendix (D.1).

$$\begin{bmatrix} \dot{X}_{N,b} \\ \dot{Y}_{N,b} \\ \dot{Z}_{N,b} \end{bmatrix} = \dot{\mathbf{TR}}\mathbf{_{XY,p,b}} \begin{bmatrix} -a \\ -b \\ -c \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{Z}_{mo} \end{bmatrix} \tag{4.61}$$

4.5 Postitioning

Positioning is an intermediate state which is entered as soon as the pose of the platform does not match the position of the desired state. This state utilizes the current pose and the desired pose to generate a sloped path for the transition to follow. Since the system is in interaction with humans, it is desired that the platform operates in a smooth and continuous manner. Creating a continuous curve for the platform to follow when moving from one pose to another is therefore needed. There is a part of the sine wave - when going from either maximum value to minimum value or vice versa, which is suitable to use as a transition reference. What part and how it is used is illustrated in Figure (4.43).

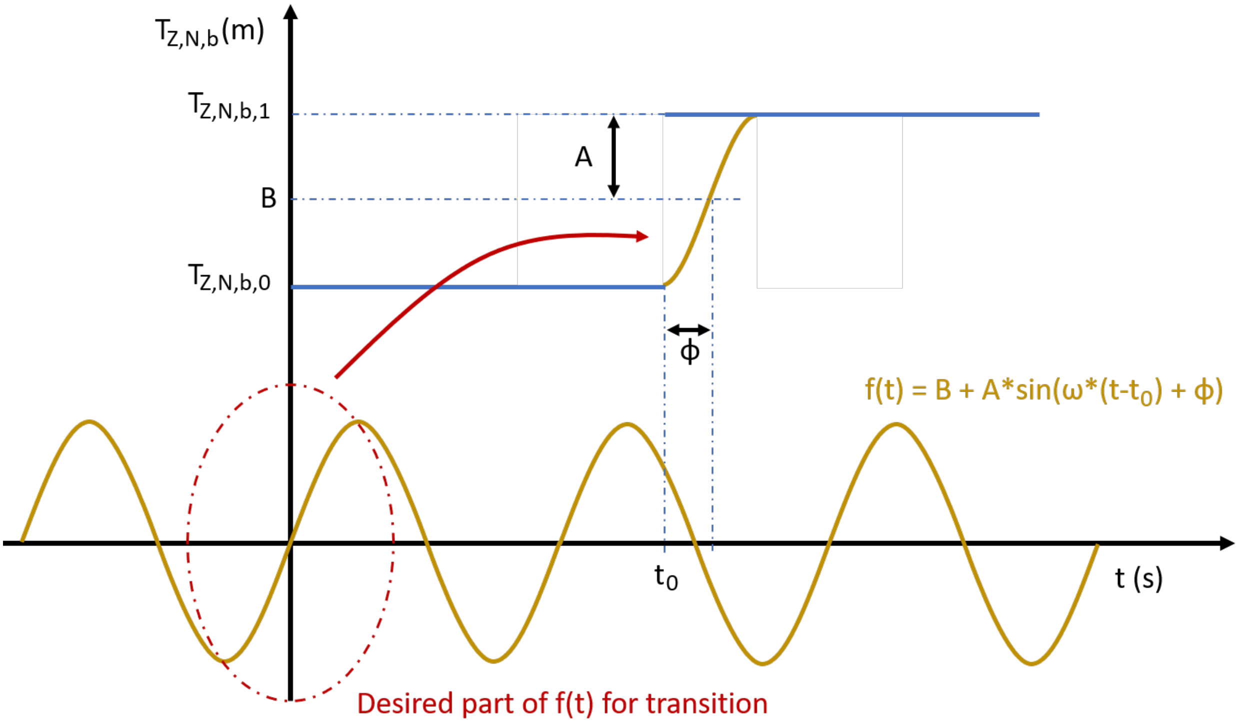

T_{Z,N,b}(m)
T_{Z,N,b,1}
A
B
T_{Z,N,b,0}
ϕ
f(t) = B + A*sin(ω*(t-t_0) + ϕ)
t_0
t (s)
Desired part of f(t) for transition

Figure 4.43: Illustration of sine wave transition

The section circled in red within Figure (4.43) is the part of a sine wave which is of interest, and is described by f(t) (in figure). The variables A, B and ϕ needed to create that curve from current pose, $T_{Z,N,b,0}$ to the desired pose, $T_{Z,N,b,1}$, are shown in Equations (4.63) - (4.64). Note that $T_{Z,N,b}$ is the translation of b coordinates relative to N coordinates in Z direction, but the same procedure is applicable for all six degrees of freedom.

$$A = \frac{T_{Z,N,b,1} - T_{Z,N,b,0}}{2} \tag{4.62}$$

$$B = T_{Z,N,b,0} + A \tag{4.63}$$

$$\phi = -\frac{\pi}{2} \tag{4.64}$$

t_0 is the moment in time when the transition command is given, and the value of ω is selected by testing as it determines how rapid the transition should be. These equations work for downward transitions as well, as long as signs are included.

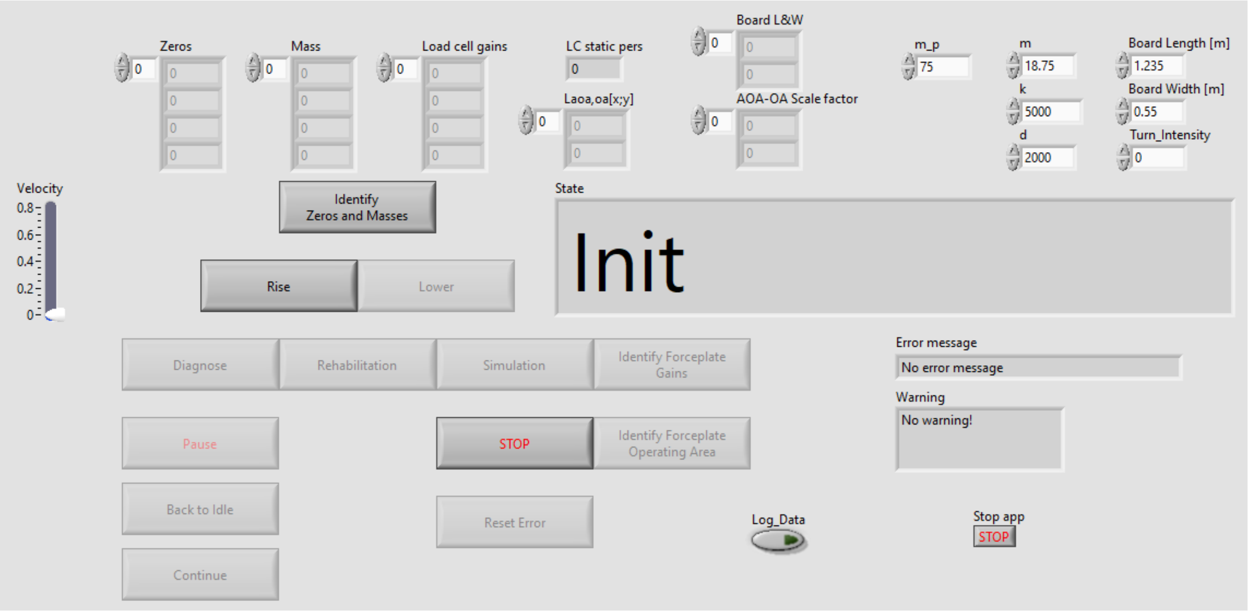
Since the system state should change from positioning to the desired state as soon as the pose of the platform is within a certain set of tolerances, it is not needed to set an ending time for the sine wave point generation. It will stop generating points as soon as it reaches the right pose for the state since the positioning state is no longer active.

5. Application and Testing

This chapter shows how the developed control system was tested and implemented on the physical system. Pictures of the physical system, plots, and graphs of relevant data are also included. First, the program, which is designed in NI LabVIEW, is dry run tested to check if the transmitted signals to the Stewart platform is reasonable and will not cause irrational behavior. Further on, when dry run testing is complete, data is transmitted to the Stewart platform with the physical structure partly assembled. In the end, the whole physical system is assembled with the program implemented and tested as one combined unit.

5.1 User interface

The user interface, or more accurately for this system; the operator interface, was created with the purpose of establishing a good base for further development. The operator interface acts as a state selector, process actuator and data observer, and communicates with the myRIO (running the main program) via a wireless network. The operator interface is developed in a way that prevents the operator from clicking buttons which could disrupt the program flow. This is done by deactivating interactive buttons not relevant to the current state. Figure (5.1) shows the operator interface while the system is in its initial condition. Notice how buttons relevant for that state are the only interactive buttons.



Zeros: 0 | 0, 0, 0, 0
Mass: 0 | 0, 0, 0, 0
Load cell gains: 0 | 0, 0, 0, 0
LC static pers: 0
Laoa,oa[x;y]: 0 | 0, 0
Board L&W: 0 | 0, 0
AOA-OA Scale factor: 0 | 0, 0
m_p: 75
m: 18.75
k: 5000
d: 2000
Board Length [m]: 1.235
Board Width [m]: 0.55
Turn_Intensity: 0
State: Init
Error message: No error message
Warning: No warning!

Figure 5.1: Operator interface

The top indicators show the data gathered from load cell identification processes such as identifying load cell zeroes and masses. When these processes are carried through, data relevant to the process will be displayed in these indicators. In the top right corner are the parameters affecting the simulation. These parameters concern the models and can be changed according to the simulated scenario. On the far left side is the velocity parameter, which sets the user's velocity within the virtual reality. Then, there is the state display which displays the current state of the myRIO program, buttons, as well as warning and error displays. If the signals transmitted to the Stewart platform has been saturated, descriptive warnings are displayed.

5.2 Dry run testing program with Wii-board

Using the Wii-board as a force plate was a convenient solution to the first stage of dry run testing the program, as it eliminated the need for laboratory access which would otherwise be needed in order to use the actual treadmill with the attached load cells. Similarly to the treadmill with attached load cells, the Wii-board has one load sensor in each of its four corners, so the working principle is exactly the same. The only difference in using the Wii-board is the sensor values. However, the program is designed to support sensors with different ranges and resolutions which eliminates that potential issue. The goal of dry run testing is to eliminate any potential bugs in the program and ensure that the calculations yield reasonable and expected results that are safe to transfer to the actual system during manned testing. To receive continuous logged data during labVIEW program execution, two logging blocks had to run parallel with different saving intervals since saving caused logging of data to halt for a few second. Therefore, the two files (Excel sheets) had to be merged in order to obtain one, continuous set of data points. The first problem was to fill in the gaps of each data set with correctly spaces data points. This was done by a developed matlab function, shown in Appendix (E.2). Secondly, a script was developed which merges the two data sets automatically by taking parts of one data set to overlap the missing parts of the other, shown in Appendix (E.3).

Transmitted signal during transition between states

The program flow, which has a finite state machine structure, depends on feedback from the Stewart platform. This is to avoid entering another state without the platform being in the respective position for that specific state. To by-pass this during dry run testing, the feedback is equal to the signals that are supposed to be transmitted to the Stewart platform. The only two adjustments that are made to gain the possibility to dry run test the program without laboratory access are therefore the load input source and the feedback source.

First, it is desired to analyze the transmitted signals when moving between states. As previously described, the states require the Stewart platform to be located in different positions. It is wanted that these transitions are smooth to optimize the experience of the user. The physical transition between states concerns only translation in Z_N directions when simulation mode is not engaged. When testing these transitions, it is important to test all possible actions to prevent any undesired behavior. An example of dry run testing the transmitted signals for a few different actions done by the operator is shown in Figure (5.2). The black line is the transmitted signal in Z_N direction, the red and blue dotted lines indicate the correct positions for the initial and idle state, respectively.

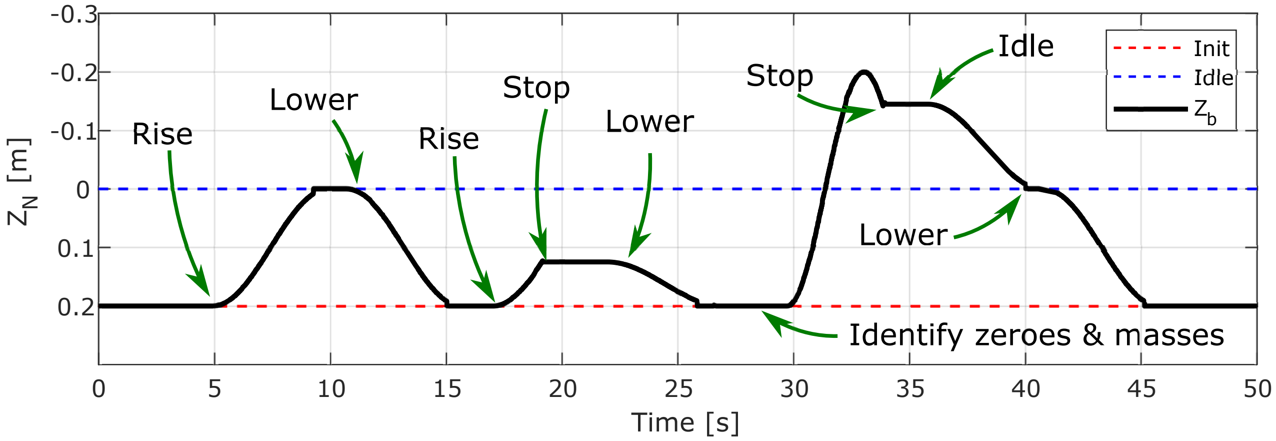


Figure 5.2: Dry run testing transmitted signals during state transitions

Notice how minor steps occur in the transmitted signal as it approaches the different states. These steps occur due to a set of applied tolerances which makes the system enter a state as soon as the platform position is within these tolerances. Such an implementation is necessary when operating with a physical system where disturbances are present. The tolerances are set to be ±10 mm in all directions and ±0.01 rad about all axes. Due to their relatively small value, these steps will not be noticed by the person located on the platform.

Transmitted signal during board simulation

Similarly to state transitions, it is important to check for any irrational behavior of the transmitted signal during the simulation as well. The transmitted signal is preferred to be continuous, where only minor steps are acceptable. During simulation, however, five degrees of freedom are used. Heave, roll, and pitch, which are the three outputs of the board model, and also minor surge and sway values to compensate for the moved control point. All these values are of interest to analyze when entering simulation, during simulation and also when leaving simulation. These transitions should be smooth and as continuous as practically possible. The transmitted signals during simulation can be seen in Figure (5.3). Similarly to dry run testing the state transitions, the actions available such as stop and pause are tested. It should be clarified that there are implemented limits to rotation at $\pm 0.9 rad \approx 5^\circ$, which is why some saturation of the transmitted rotation about X_N axis is visible.

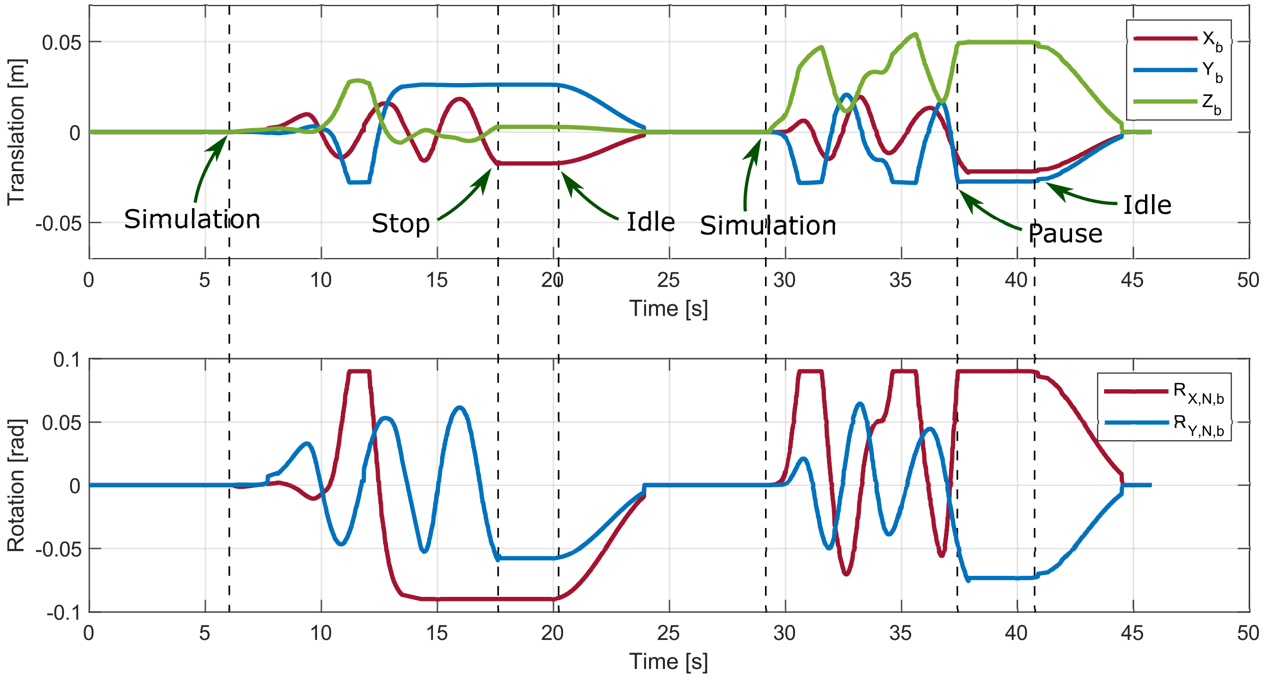


Figure 5.3: Dry run testing transmitted signals during simulation

Likewise to testing the state transitions, some steps in the signals are present when approaching the idle state, which is due to the tolerances set. This is expected behavior, and will not cause any trouble.

Verifying the load cell conditioning

To obtain a certain feeling of control when operating on the force plate, it is required that the load cell conditioning has been carried through correctly. Otherwise what you expect to happen when applying load at a certain spot might prove entirely wrong, and you quickly lose the sense of how to control the simulation. There are three conditioning steps regarding the load cell signals:

- Zeroing signals.
- Reduce inertial impact from the acceleration of treadmill.
- Apply individual gains to obtain equal I/O relation.

To verify that each step conditions the signals to something usable, a simple test can be done. However, it should be mentioned that the inertial impact part can not be verified by dry run testing with a Wii-board. The test to verify the conditioning is as follows:

1. **Read raw signals values, no load**
   Expected: Signal values are steady, but different for each sensor.

2. **Subtract zero values**
   Expected: Signal values are steady at zero.

3. **Apply increasing load at center of forceplate and hold steady**
   Expected: Signal values increases at different rates, ending up at steady, but different values.

4. **Multiply signal values with affiliating gains**
   Expected: Signal values are steady and same.

5. **Deload gradually**
   Expected: Signal values decreases at the same rate and ends up at steady, zero value.

When carrying through the sequential test above, the conditioned signals turned out as expected. The plot is shown in Figure (5.4), where each step of the test is marked by a number. Both plots are from the same test, the only difference is the range of the 'Sensor values' axis. Since the zero values of the Wii-board sensors are very high in magnitude compared to the change in values when operated, the effect of the gain conditioning is not as apparent. Hence, the right-hand plot having a reduced range of the 'Sensor values' axis.

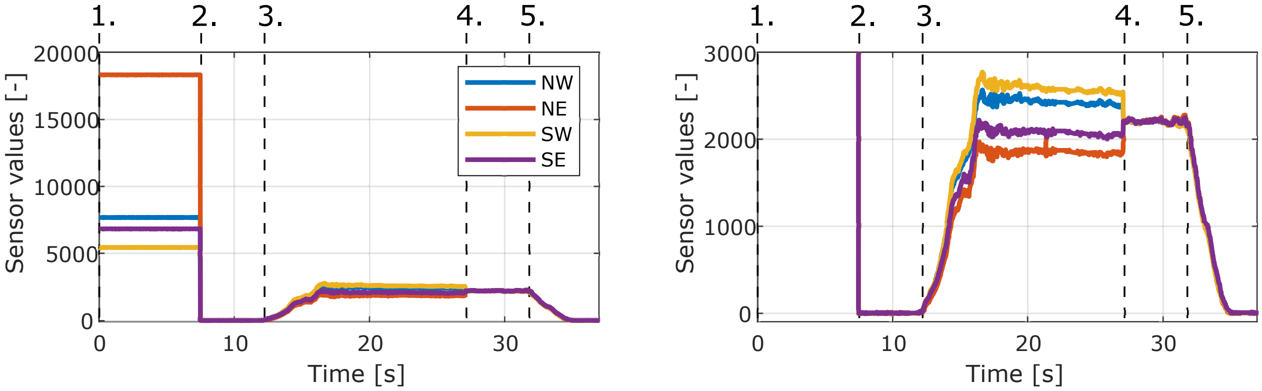


Figure 5.4: Verification of load cell conditioning

Verifying operating area

The operating area is a physical area which represents the size of the virtual board, or the board to be simulated. To map this area onto the surface of the treadmill, a couple of processes are done with the conditioned signals received from the load cells. These processes must be separately checked to make sure the operating area becomes as expected. The main steps in going from four conditioned load cell signals to operating area are as follows:

- Calculate location of the applied load within available operating area, i.e. the rectangle formed by the location of the load cells

- Shift origo to the current location of the applied load and resize area to the size of the virtual or simulated board.

The location of the load within these operating areas should be described by a value between -1 and 1 in X_N and Y_N direction. In this case, two separate tests are done - one for the location of applied load within the available operating area, and one for location of applied load within the operating area. First, the test for locating the load within the available operating area is as follows.

1. **Load center of force plate**
   Expected: Both $L_{X,aoa,p}$ and $L_{Y,aoa,p}$ is approximately zero.

2. **Gradually load North-West corner while deloading center**
   Expected: $L_{X,aoa,p}$ goes towards 1 and $L_{Y,aoa,p}$ goes toward -1.

3. **Gradually load North-East corner while deloading North-West corner**
   Expected: $L_{X,aoa,p}$ stays approximately at 1 and $L_{Y,aoa,p}$ goes toward 1.

4. **Gradually load South-West corner while deloading North-East corner**
   Expected: Both $L_{X,aoa,p}$ and $L_{Y,aoa,p}$ goes toward -1.

5. **Gradually load a near-center spot while deloading South-West corner, then alternate magnitude of applied load at near-center location**
   Expected: $L_{X,aoa,p}$ and $L_{Y,aoa,p}$ goes toward some values near center, and remains approximately same during the alternation of load magnitude. This is to verify that magnitude and location of load are successfully separated.

When performing the test, relevant data was logged and three plots were created. These plots are shown in Figure (5.5), where the top subplot shows the conditioned load cell data (from Wii-board), the middle subplot shows the calculated location of applied load within available operating area, and the bottom subplot shows the magnitude of the load applied (same units as sensor values). Each step of the sequential test procedure is numbered on the plot.

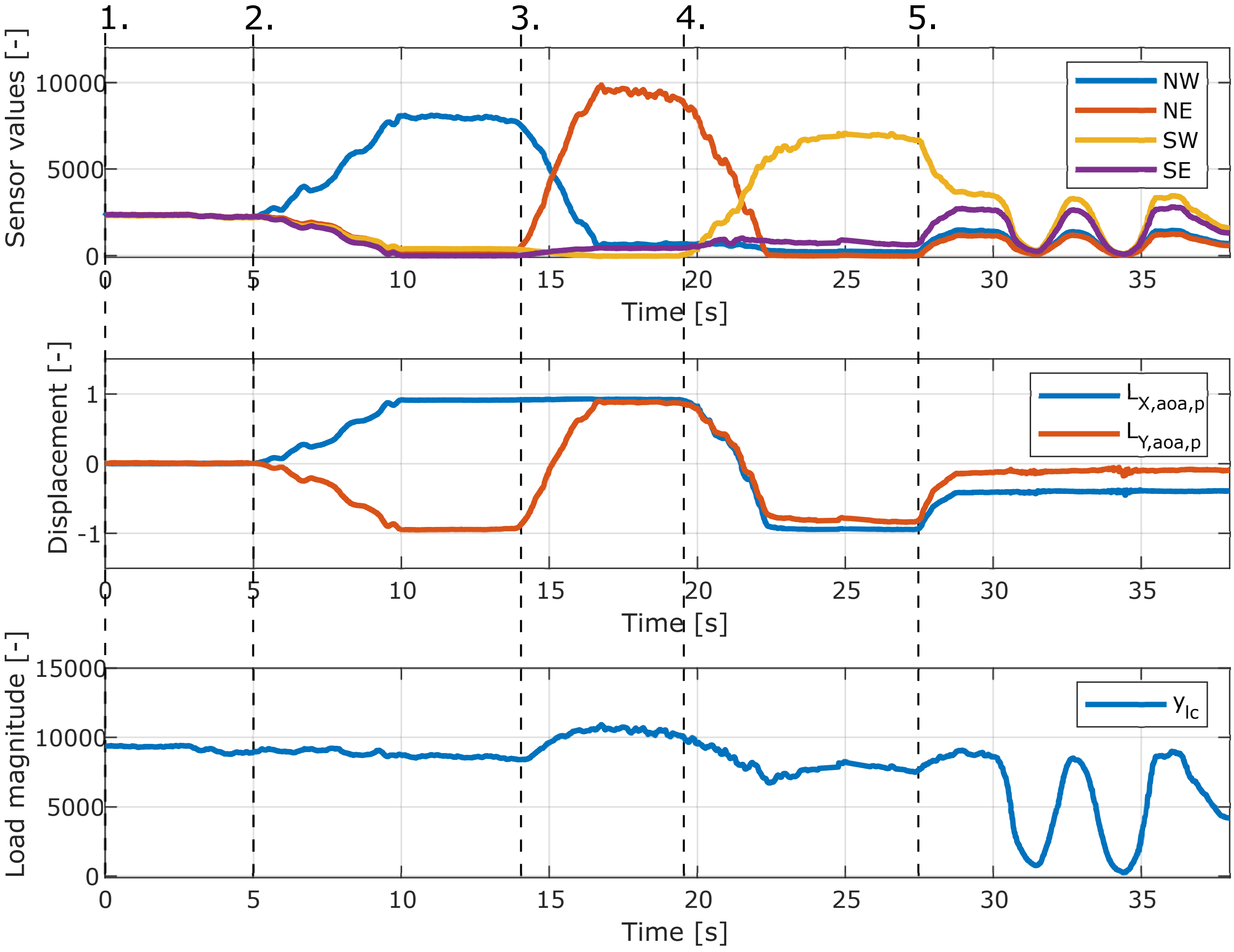


Figure 5.5: Verifying load location within available operating area

The localization of applied load within available operating behaved as expected, which means that this information can be further used in the process of locating the applied load within the operating area. As mentioned, this process establishes a new origo, depending on the current location of the load, and resizes the -1 to 1 area to an area similar to the virtual or simulated board. The procedure is exactly the same as for $L_{X,aoa,p}$ and $L_{Y,aoa,p}$, except this time it is the location within the operating area, $L_{X,oa,p}$ and $L_{Y,oa,p}$, that ia expected to behave as described in the previous test. However, one additional expectation is generated:

Expected: For an operating area half the length and half the width of the available operating area, it is desired that for any displacement of $L_{X,oa,p}$ or $L_{Y,oa,p}$, the displacement of $L_{X,aoa,p}$ or $L_{Y,aoa,p}$ is half their values.

For testing, it is important to specify where the origo for the operating area should be, and how large the board should be. The size of the board should be half the length and half the width of the available operating area, as shown in Figure (5.6). This is usually entirely up to the scenario, but for testing, it makes for an easy verification that the operating area is where it is supposed to be. Moreover, it is desired that the origo of the operating area is right in the center of the third quadrant, as illustrated in the figure. Of course, the origo is selected by the location of the load when the identification of operating area process is started, which means that placing the origo in the exact center is not likely. However, approximate is good enough for testing. The sequential test procedure is also illustrated in Figure (5.6), where the numbers corresponds to the test for available operating area.

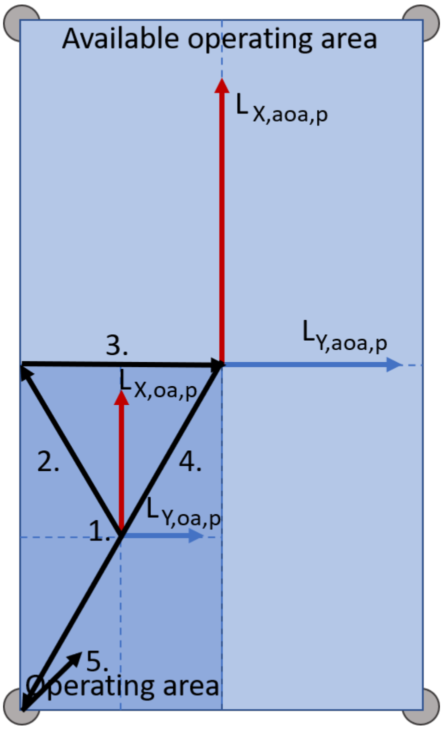


Figure 5.6: Oa vs. aoa

The result of the test can be seen in Figure (5.7). The numbered sections correspond to the sequential testing procedure for the available operating area. However, for this test, the load was applied in the corners of the operating area, and not at the load cells. Notice how $L_{X,aoa,p}$ and $L_{Y,aoa,p}$ displaces approximately half the value of $L_{X,oa,p}$ and $L_{Y,oa,p}$, respectively. This verifies that the operating area is indeed half the length and half the width of the available operating area. In addition, $L_{X,oa,p}$ and $L_{Y,oa,p}$ both starts at zero value even though both $L_{X,aoa,p}$ and $L_{Y,aoa,p}$ starts at approximately -0.5, which verifies that a new origo was successfully been created in the center of the third quadrant. Also, $L_{X,oa,p}$ and $L_{Y,oa,p}$ reaches approximately -1 and 1 when load is applied at the corners of the operating area.

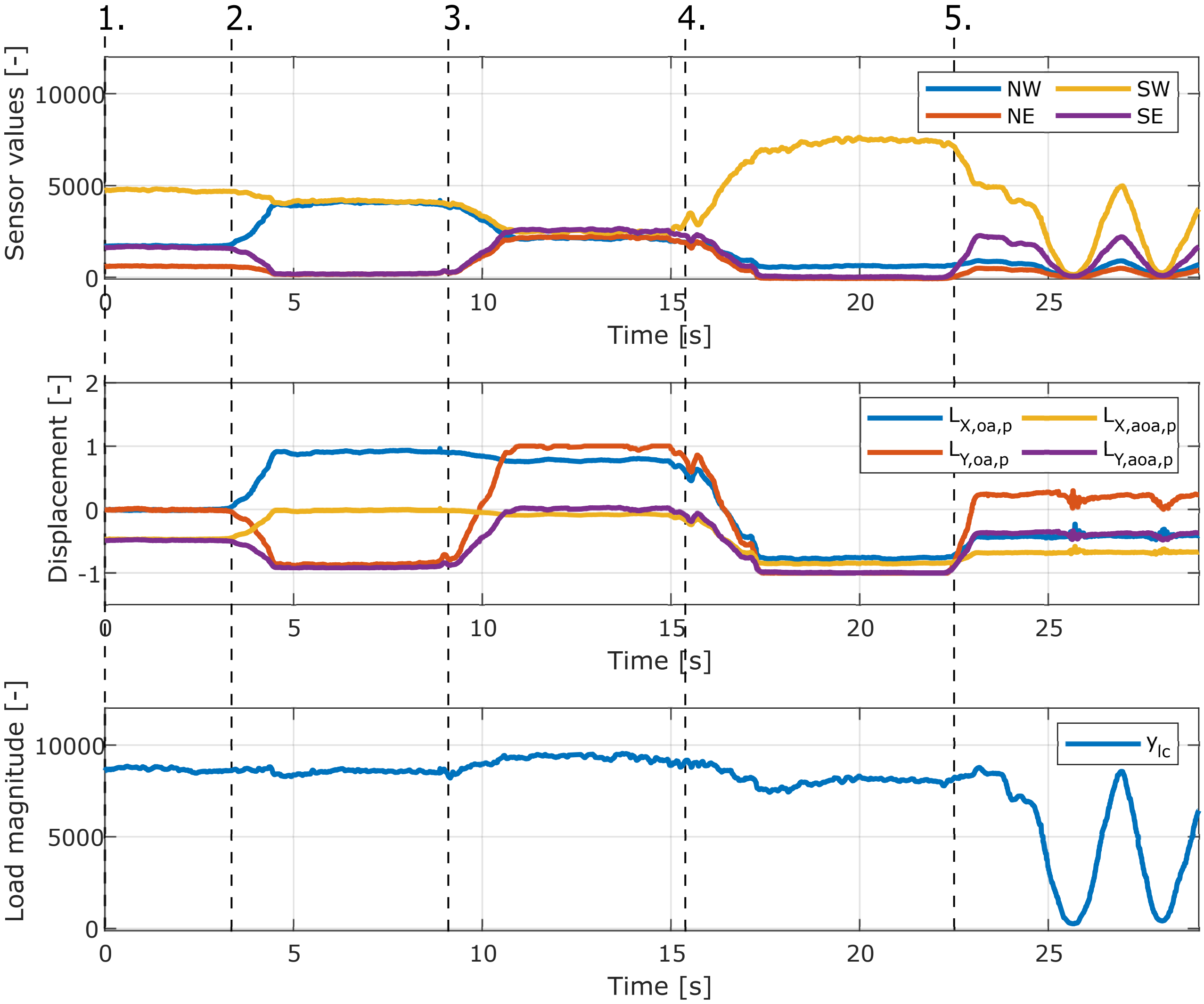


Figure 5.7: Verifying load location within operating area

These tests do indeed verify that the procedure of creating an operating area within the available operating area based on a current location of the load and the size of the virtual or simulated board works as intended.

Verifying directions of movement

Although the continuity and flow of the transmitted signal during simulation has been tested, the directions and magnitudes of the transmitted signal relative to the inputs from the force plate have not yet been tested. How the model responds to inputs from the force plate and the shift of control point is important to analyze in order to verify that the directions of movements make sense. The control point shift depends on the physical distance between the original control point and the center of the available operating area. Due to the height of the treadmill, there is a shift in the negative Z_N direction. In addition, the shift is related to the origo of the operating area, which depends on the location of the applied load during the operating area identification procedure. However, for the following verification test, the origo of the operating area is located roughly at the center of the available operating area. I.e, the shift can be considered only in a negative Z_N direction. The test procedure for verifying the directions of movement in the transmitted signal with respect to applied load is as follows:

1. **Deload Wii-board**
   Expected: Platform rises, no rotation, no translations in $X_N Y_N$-plane.

2. **Gradually load and deload North-West corner**
   Expected: Platform descends slightly, negative rotation about X_N and Y_N, slight translation in negative X_N and positive Y_N (to make the axis of rotation locate right beneath users feet).

3. **Gradually load and deload North-East corner**
   Expected: Platform descends slightly, positive rotation about X_N and negative about Y_N, slight translation in negative X_N and Y_N.

4. **Gradually load and deload South-West corner**
   Expected: Platform descends slightly, negative rotation about X_N and positive about Y_N, slight translation in positive X_N and Y_N.

5. **Gradually load and deload South-East corner**
   Expected: Platform descends slightly, positive rotation about X_N and Y_N, slight translation in positive X_N and negative Y_N.

6. **Gradually load and deload North end**
   Expected: Platform descends slightly, negative rotation about Y_N and none about X_N, slight translation in negative X_N and none in Y_N.

7. **Gradually load and deload East end**
   Expected: Platform descends slightly, positive rotation about X_N and none about Y_N, slight translation in negative Y_N and none in X_N.

When performing this test, the data of interest is the conditioned load cell signals - to visualize at which load cell the force is applied and its magnitude. Also, it is of course important to visualize the translation along and rotation about the X_N, Y_N, and Z_N axis. Figure (5.8) shows the conditioned load cell signals and the transmitted signals during the test procedure. As can be seen, the directions are as expected. It should be mentioned that the magnitude in rotation about X_N axis, thus also translation along Y_N axis is greater than rotation about Y_N axis and related translation along X_N axis. This is due to the length/width ratio of the operating area being two to one. The numbers correspond to the steps in the test procedure.

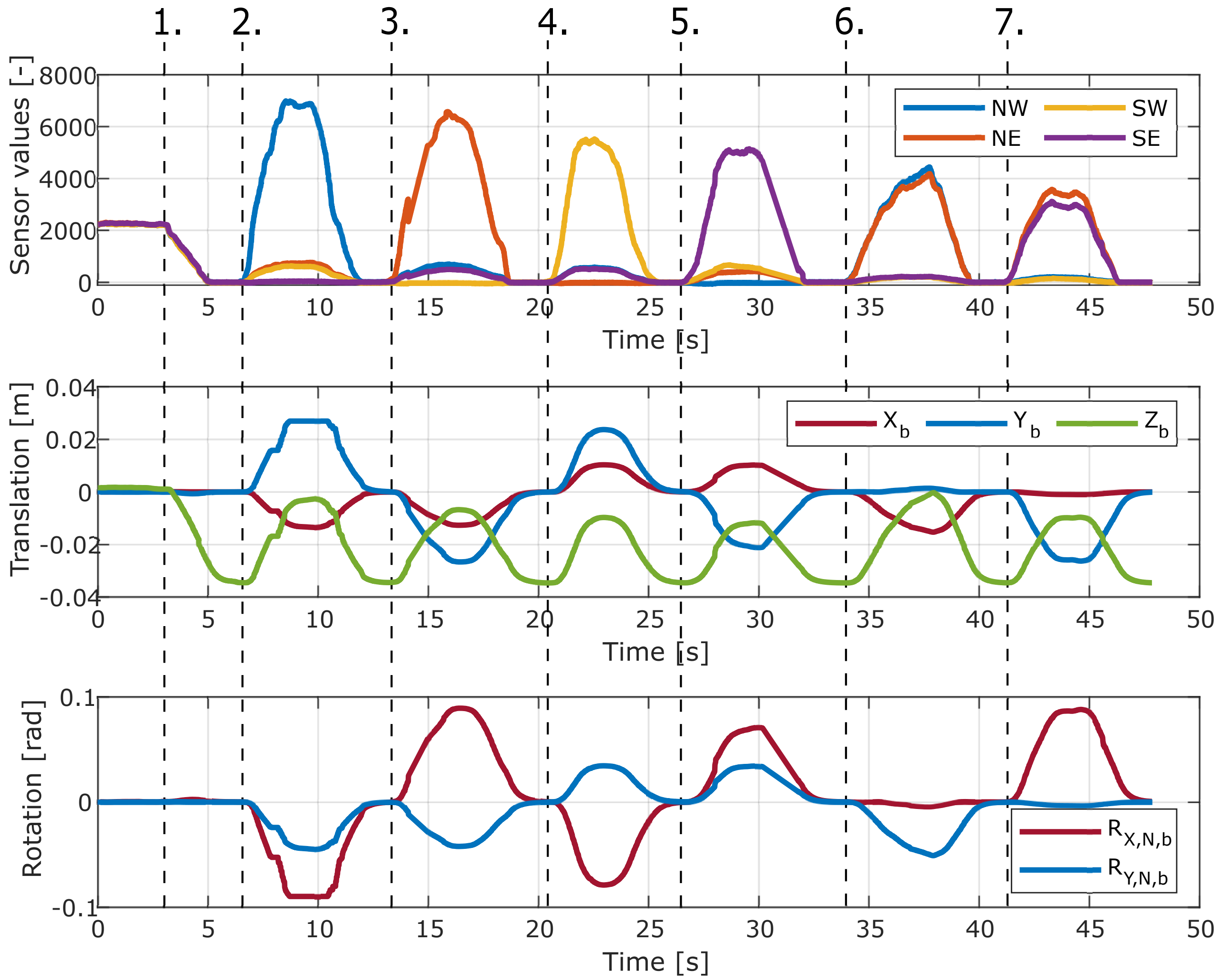


Figure 5.8: Verification of directions

Test of dynamic response

To ensure safe operation, it is desired that the response of the sub-models is stable and overdamped. It is the relationship between the models mass, spring constant and damper coefficient that determines the dynamic response of the models. During the early stages of testing, these parameters were experimentally determined by dry run testing. Determining parameters related to the overall dynamic response of the system is considered a viable approach as CELNA resembles a holistic system. A holistic system is a system that can not be fully explained by evaluating each interconnecting part but rather by referring to the whole system as one [42]. This is a relevant philosophy in technologies where the behavior of one interconnecting part that is not fully explicable affects the overall behavior of the system. Human behavior is unpredictable and since human interconnection plays a major role in the overall behavior of CELNA, CELNA also resembles a holistic system.

It is desired that the sub-models are stiff to emulate ground surface, but also responsive to emulate a relatively light board. Hence, the following values for the early-stage testing:

$$\begin{aligned} m &= 18.75 \\ k &= 5000 \\ d &= 1500 \end{aligned}$$

These values bring the behavior of the sub-models to a stable, overdamped behavior, which is described by its poles and zeroes. If the poles and zeroes are located in the left hand plane (LHP) of a pole-zero map, the system is stable. If the poles and zeroes are located in the LHP and on the real axis, the system is stable and overdamped. Figure (5.9) shows the pole-zero map of one sub-model.

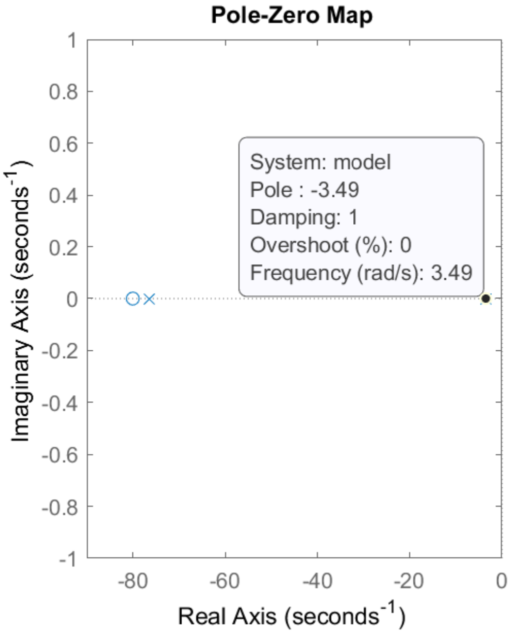


Figure 5.9: pole-zero map sub-model

To test the response of the system, the force plate should be subjected to impacts at different rates and magnitudes, and also at different locations. Moreover, timed back and forth movements should also be tested to provoke increasing oscillations. Figure (5.10) shows the relationship between inputs (upper sub-figure) and transmitted signals (bottom two sub-figures). Although the figure does not yield much information, the information gathered is that the system behaves stably for inputs from the stationary force plate. Ensuring stable response of transmitted signals before actually transmitting them to the Stewart platform is good practice.

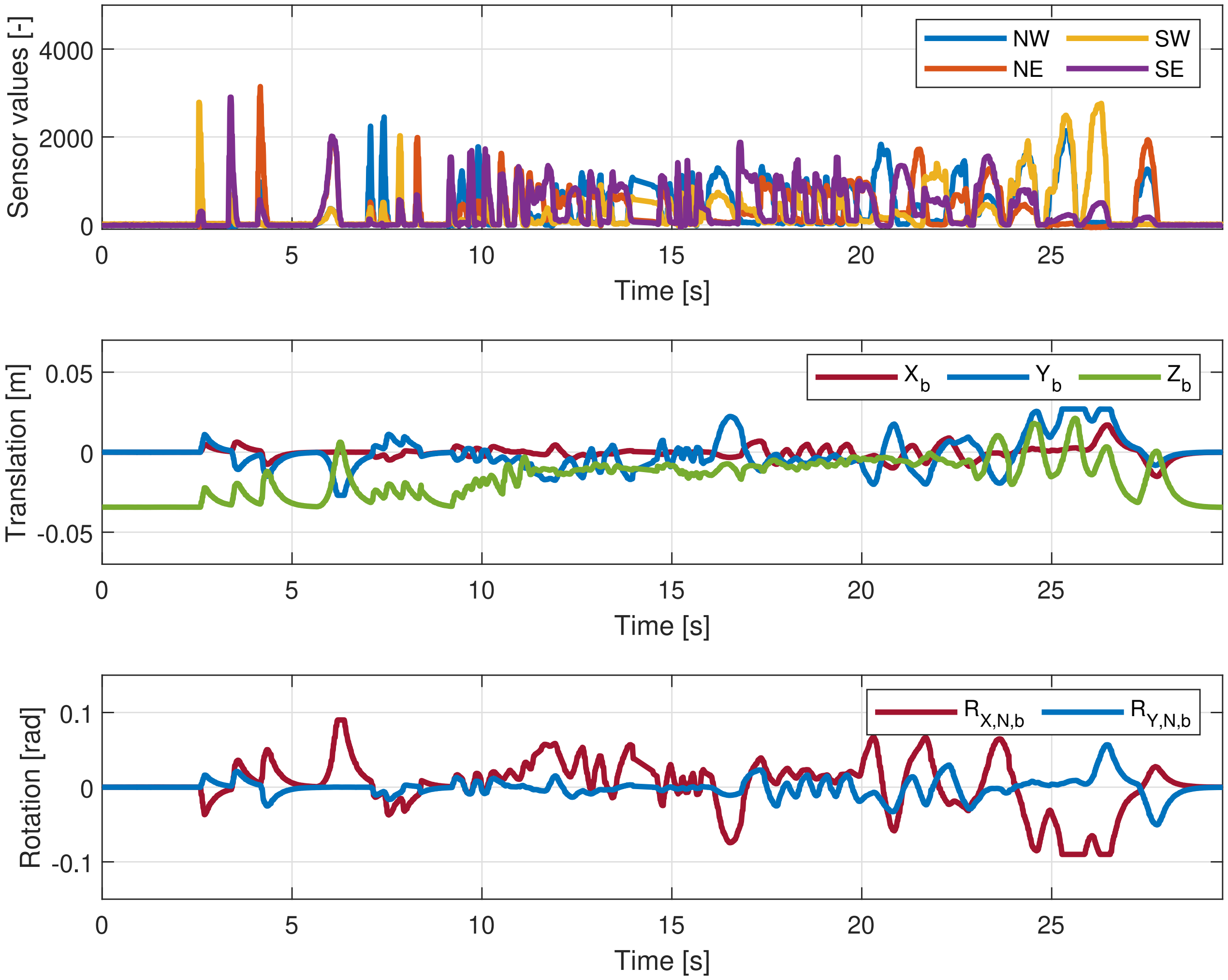


Figure 5.10: Testing dynamic behaviour

Remark that disturbances from the acceleration of mass will occur when the force plate is mounted on the Stewart platform. Although an acceleration-related value is subtracted to reduce the disturbance, it will not be completely removed. Since the force due to the acceleration of mass counteracts the force applied from the person, there is a risk of the system going unstable. Remember, increasing applied load causes the mass-spring-damper models to retract causing downward acceleration for a brief moment. This action deloads the load cells where the person applied load, which causes the models to expand again, and an oscillation may initiate. It is important to keep this in mind when reaching that stage of testing. Precautions should always be made even though dry run testing has been successfully carried through.

5.3 Prototype testing

Although dry run testing has been successfully carried through until the point where it feels safe to transmit the signals to the Stewart platform, it is wise to do further testing in an iterative manner. Doing so makes locating bugs and issues with the program a lot easier. Therefore, the process of assembling CELNA happens in the following way:

- Testing with the force plate (treadmill with load cells) stationary on the ground, observing the response of the Stewart platform relative to the location and magnitude of the applied load.

- Mount the treadmill unit (harness supporting cage, load cells and treadmill) on the platform and perform some unmanned response tests by pulling the corners of the treadmill.

- Final test by manned operation.

Of course, any encountered problems must be resolved before continuing to the next phase of testing.

Stationary forceplate

The next stage after dry run testing is to transmit the previously tested and verified signals describing the desired pose of the Stewart platform to the actual Stewart platform itself. At this stage, the Wii-board is replaced by the treadmill which is supported by four load cells. The working principle is exactly the same, and the signal conditioning, load location identification (operating area), and force magnitude do not depend on the measurement unit nor magnitude. As long as there is a linear relationship between the rate of change in measurements and applied load, any load measuring unit can be used. Therefore, the transition from using a Wii-board as a force plate to using the treadmill with load cells as a force plate is done by swapping hardware.

First, the treadmill unit - harness supporting cage, load cells, and the treadmill was stationed on the floor in front of the Stewart platform as shown in Figure (5.11), which is a photo of the setup. Doing so prevents disturbances from the acceleration of the platform to influence the transmitted signals and the platform should behave as expected, assuming that the transmitted signals are within the limits of the platform. To ensure that the transmitted signals do not exceed the physical limitations of the Rexroth eMotion-1500, the very last operation done to the transmitted signals before they are sent over UDP to the platform is to limit them.

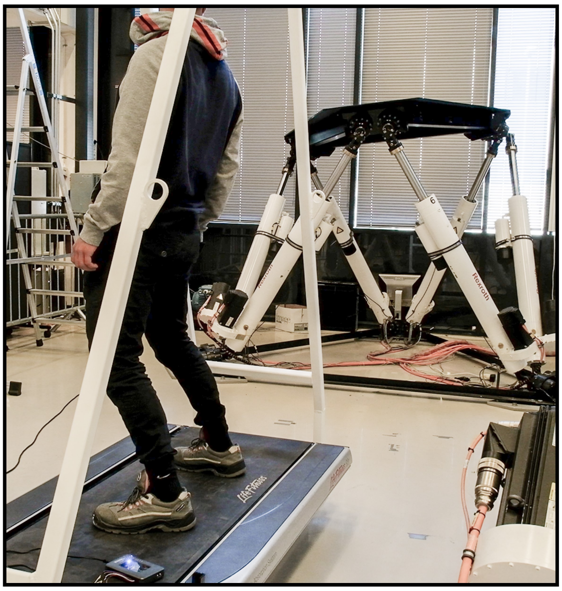


Figure 5.11: Stationary forceplate

Unfortunately, no logging was done during this phase of testing. Anyway, the response of the Stewart platform in terms of orientations and positions relative to the location and magnitude of the load turned out as expected. Concerning the response in terms of time, the delay between applying load and corresponding movement of the platform was barely noticeable. Delay can quickly become an issue when operated by a person as he/she will naturally try to increase the load applied if the platform does not respond as expected when applying a certain load. Then the increased load will have an effect just a moment later, and the person will try to counter it, which again, will be delayed. Therefore, it is crucial for the operation that the system response in terms of time is kept as low as possible. However, as this test showed promising results concerning the dynamic behavior and time delay, progression to the next stage of testing could be done.

Mounting treadmill onto platform and perform unmanned tests

The treadmill unit was loaded onto the Stewart platform using an overhead crane and fastened to the platform using four M16 bolts. The structure (treadmill unit mounted on the platform) is shown in Figure (5.12). As seen, the treadmill unit is mounted in such a way that the front is sticking out past the surface of the platform. This is done intentionally to have the available operating area's origo approximately right above the original control point of the Stewart platform. In doing so, less movement is required from the platform to shift the control point to the point right beneath the user's feet.

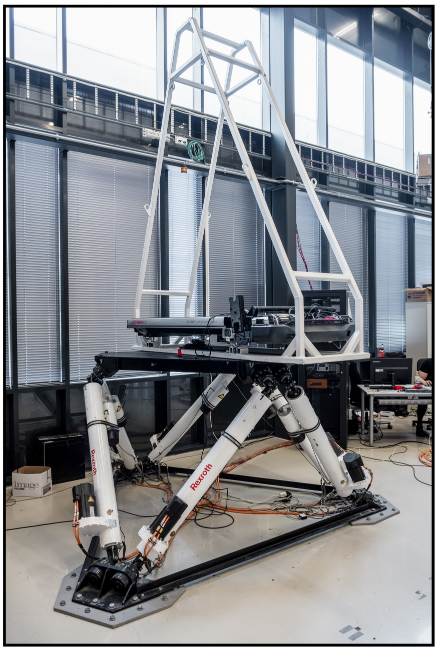


Figure 5.12: Treamill mounted on platform

When this setup was first tested unmanned, everything worked as intended considering the translation between states and stop buttons. The simulation on the other hand, showed some irrational behavior. However, this was not entirely unexpected, as it most likely occurred due to the acceleration of masses being measured by the load cells which in turn affected the board model. This resulted in increasingly oscillating behavior which is unacceptable for this system. Being on a tight schedule due to the COVID-19 situation, with very limited access to the laboratory, this issue had to be resolved quickly in order to have a chance performing manned testing. To sort out this issue, the raw signals gathered from the load cells were filtered using four mass-spring-damper models - one for each load cell signal. Although it is not the best-suited type of filter for such an application as it will introduce a noticeable delay in the overall system response, it gives the possibility to quickly tailor the filtering to at least stabilize the system.

As mentioned, the filter is a simple mass-spring-damper model, one for each load cell signal, where the raw load cell signal acts as a force on the model. Depending on the magnitude and dynamic behavior of the force applied, the mass of the model will displace correspondingly. How it will respond to the force applied is determined by its parameters - m, k, and d. Then, the displacement of the mass should be multiplied with a gain to have the same magnitude as the input when in steady-state. How the load cell signals are filtered by a mass-spring-damper model is illustrated in Figure (5.13).

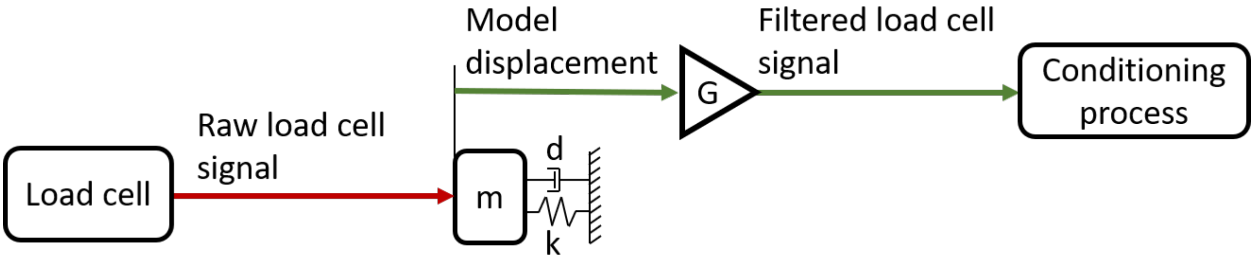


Figure 5.13: Load cell signal filtering

The parameters used for these filters are:

$$\begin{aligned} m &= 3 \\ k &= 50 \\ d &= 20 \end{aligned}$$

To find a suitable gain, G, the differential equation for a mass-spring-damper system must be evaluated at steady state, i.e. velocity and acceleration term is zero. For this example, Model displacement is z, Raw load cell signal is x and filtered load cell signal is y:

$$\begin{aligned} k \cdot x &= z \\ z &= \frac{y}{G} \\ \implies k \cdot x &= \frac{y}{G} \end{aligned}$$

As it is desired that the Filtered load cell signal, y, should be equal to the Raw load cell signal, x, when in steady state, i.e. $x = y$

$$\implies G = k = 50$$

The effect of filtering the raw load cell signals by the use of these mass-spring-damper models can be seen in Figure (5.14). The top subplot shows the filtering effect on the raw measurements of one load cell, while the two bottom subplots show the effect this filtering process has on the behavior of one sub-model.

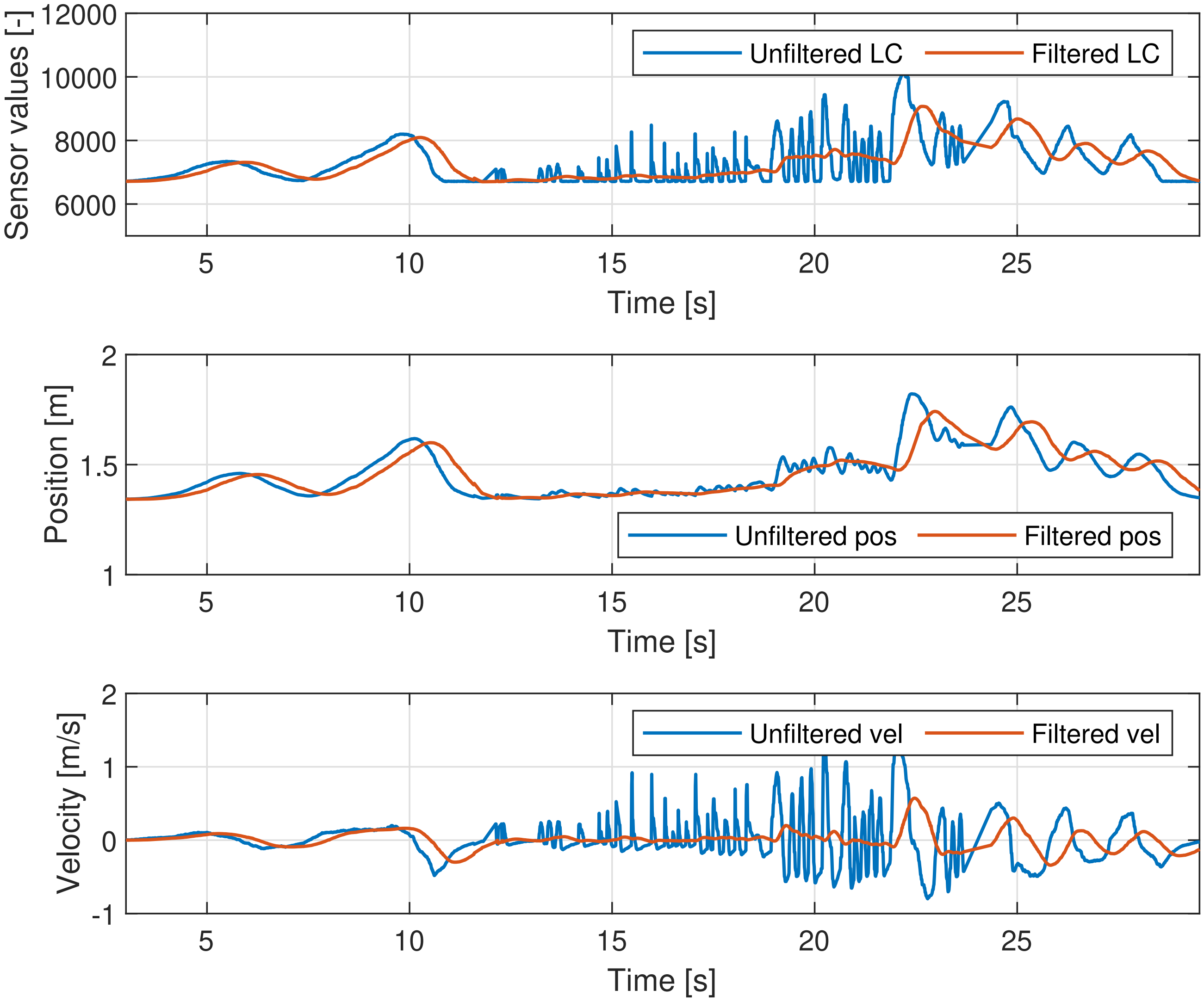


Figure 5.14: Effect of mass-spring-damper filter

As previously mentioned, this type of filtering increases response time from the load is applied until the model responds. In fact, by analyzing the plot shown in Figure (5.14), the filtered position is approximately 400 ms slower than the unfiltered position. Adding that on top of the already 200 ms delayed unfiltered model response compared to the raw load cell measurement, the delay between applying load until the model responds is over half a second. The parameters could be changed to gain a more rapid response, however, it was decided to go with these values to be able to do some testing of a functioning system before the laboratory became unavailable.

When the raw load cell signals were filtered, the increasing oscillations did not occur, and the platform was stable in simulation mode with no external influences. Then, to test how it would respond to external influences in a safe manner, a person pulled on each corner of the treadmill and the response was analyzed. Figure (5.15) shows a picture taken of the pull-test. As can be seen, when the North-East corner of the treadmill is pulled down while simulating a general board, the platform angles towards it. Rapid and intense pulls were also tried to test its resistance against instability provoking actions. As expected, the system responded a bit slow and felt a little like trying to move a small boat, but in turn, the previous influence acceleration of mass had on the model were no longer noticeable.

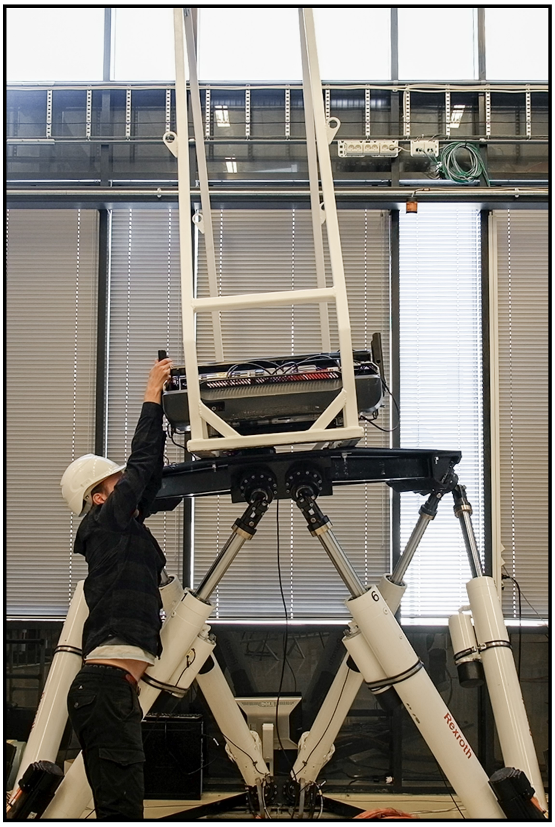


Figure 5.15: Pull Test

Parts of the pull test were logged and can be seen in Figure (5.16). The upper sub-plot is the raw, unfiltered load cell signals, and the two bottom plots are the transmitted signals where filtered load cell signals have been used. The reason for visualizing the raw unfiltered load cell signals in comparison with the model output is to analyze how big of an impact disturbances have on the model output. Moreover, it is possible to check the response time between registered sensor values and model output. Notice how it is just above half a second as previously mentioned.

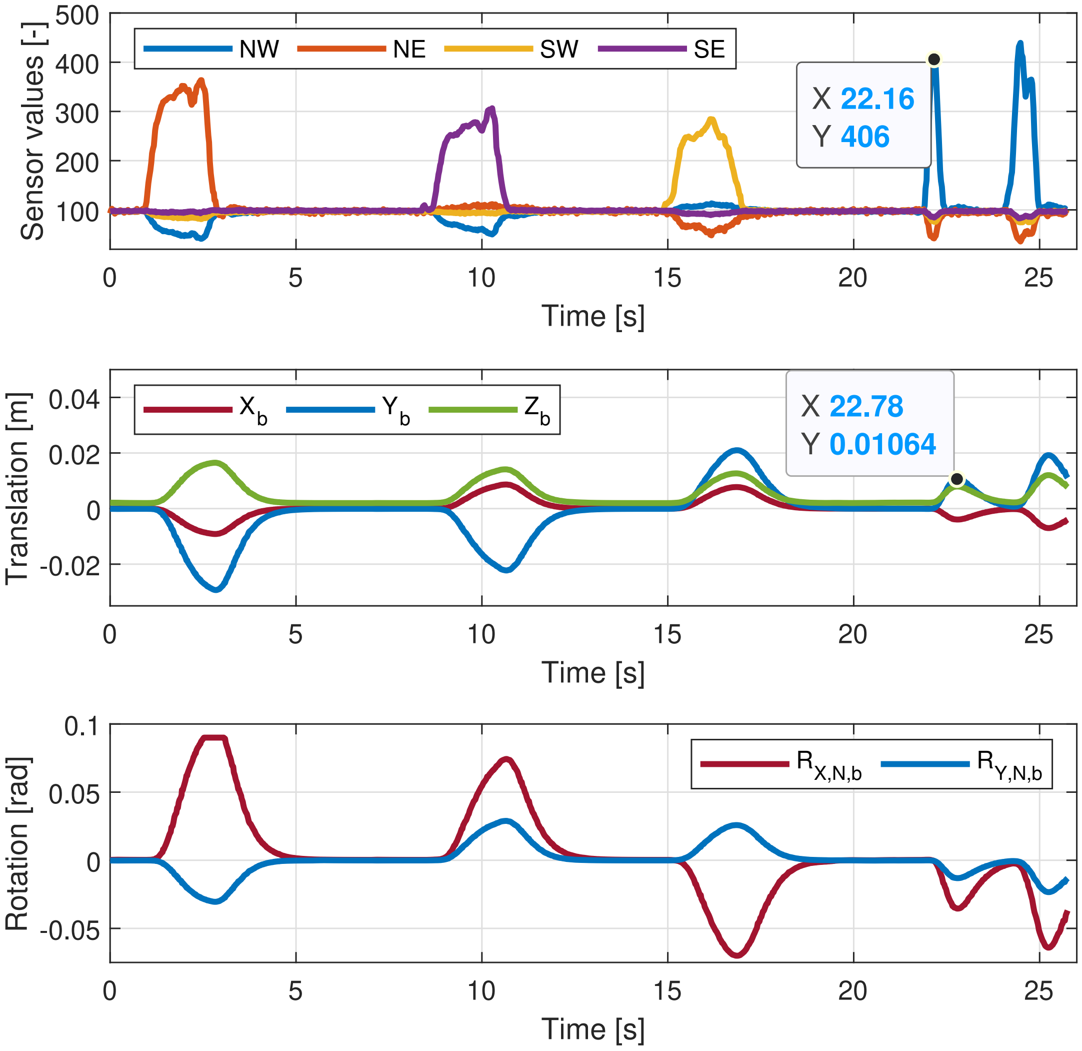


Figure 5.16: Pull test plot

The pull test proved that the filter worked as intended although it slowed the overall system response down significantly. A slow but stable system is good enough for testing purposes, as improvements are easy to add at later stages.

Manned testing

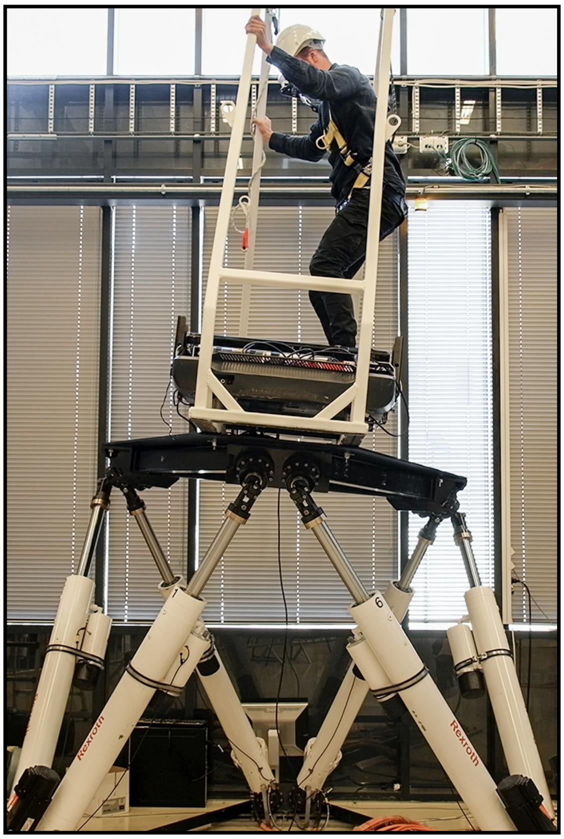


Figure 5.17: Manned Test

Finally, the testing which involves a person located on top of the treadmill can start. Since a human is introduced into the system, it is important to tread with care. First of all, it is essential that the user wears a helmet and a safety harness attached to the cage. Then the signals to transmit should be observed and verified before they are sent to the controllers that control the actuators, in order to ensure they behave as expected. When opening the communication and transmitting the signals, the user should carefully start shifting weight and get a feel for the platform response. Of course, standby operators should be ready with an emergency stop. Figure (5.17) shows a photo of CELNA manned with a person. Notice how the platform rolls toward the load location.

When this test was carried through, the person on board quickly gained a feeling and trust over the system and started to move more intensely and rapidly. This was done to test if the system would show signs of going unstable or behave irrationally. However, the system responded with subtle and smooth movements and did not show any signs of unwanted oscillations. Figure (5.18) shows some data logged of manned testing for a brief amount of time. The upper sub-plot shows that raw, unfiltered load cell signals, and the bottom two sub-plots shows the transmitted signals where filtered and conditioned load cell signals have been used. Notice how the transmitted signals flatten out when reaching the specified upper and lower limits, which is at ±0.9 rad, or approximately ±5° for rotation.

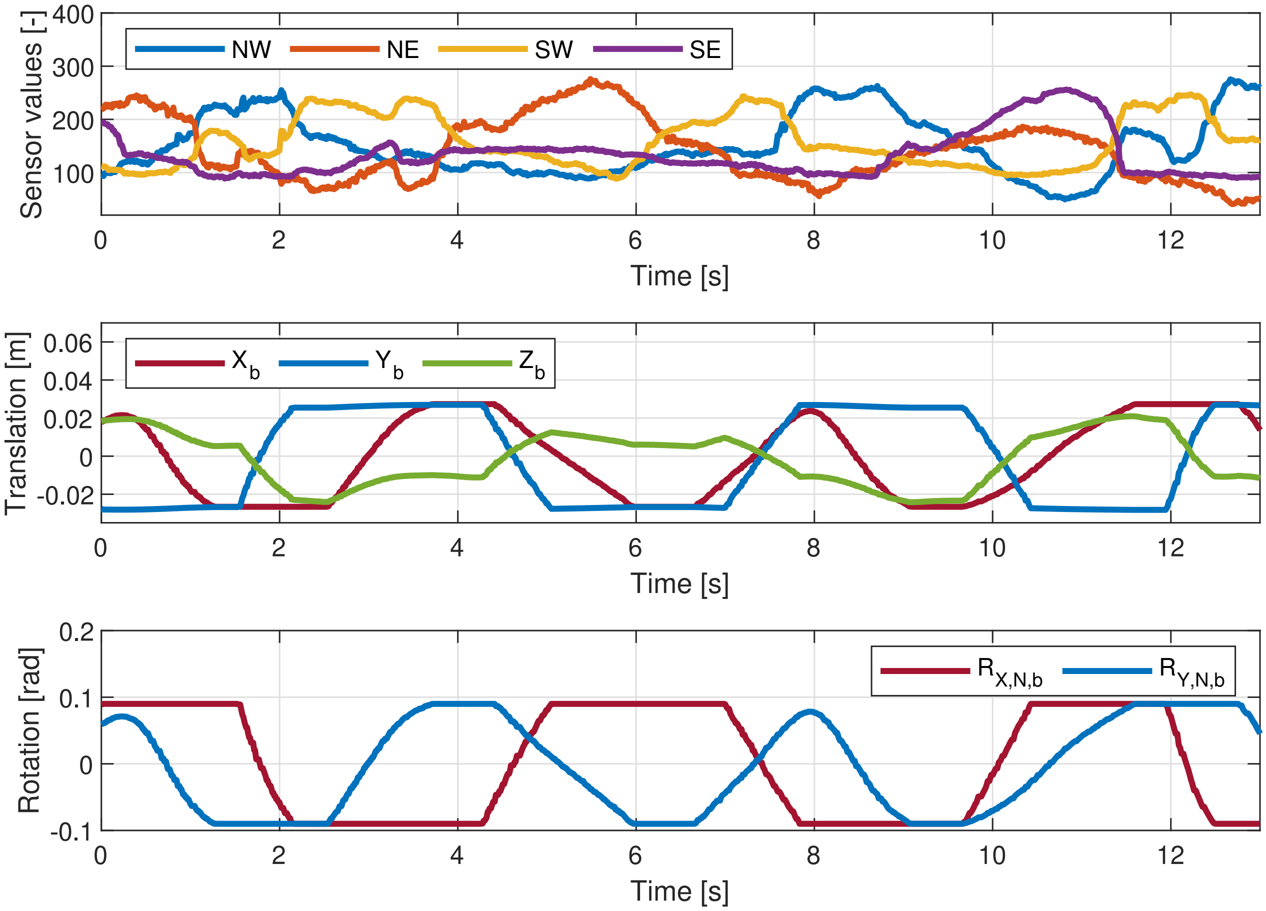


Figure 5.18: Manned test plot

Testing interaction with VR

The implementation of interactive virtual reality adds a new set of inputs to the board model - the virtual surfaces orientation about X_N and Y_N and displacement along Z_N, denoted as $R_{X,N,sim}$, $R_{Y,N,sim}$ and $T_{Z,N,sim}$ respectively. In addition, if available, the velocities $\dot{R}_{X,N,sim}$, $\dot{R}_{Y,N,sim}$ and $\dot{T}_{Z,N,sim}$ can and should be used.

Concerning the generated artificial environment available for us, only the pose - not the velocities, were available. Although numerical derivation was a possible solution to obtain velocity signals, it was decided to do the first tests without them. This decision was made based on the fact that numerical derivation tends to introduce noise which could require some sort of signal processing such as low-pass filtering, and there was little time to test it. Since the virtual reality software was developed by another group of students, the software ran on a separate computer. Before testing could be carried through, communication had to be established. The myRIO can only be connected to one host computer via wireless LAN, which meant the virtual reality software had to be transmitted via the myRIOs host computer. A diagram showing how the communication between the units was established can be seen in Figure (5.19). Notice how the computer running the VR software does not communicate directly with the Rexroth host computer, but only with the myRIO host computer.

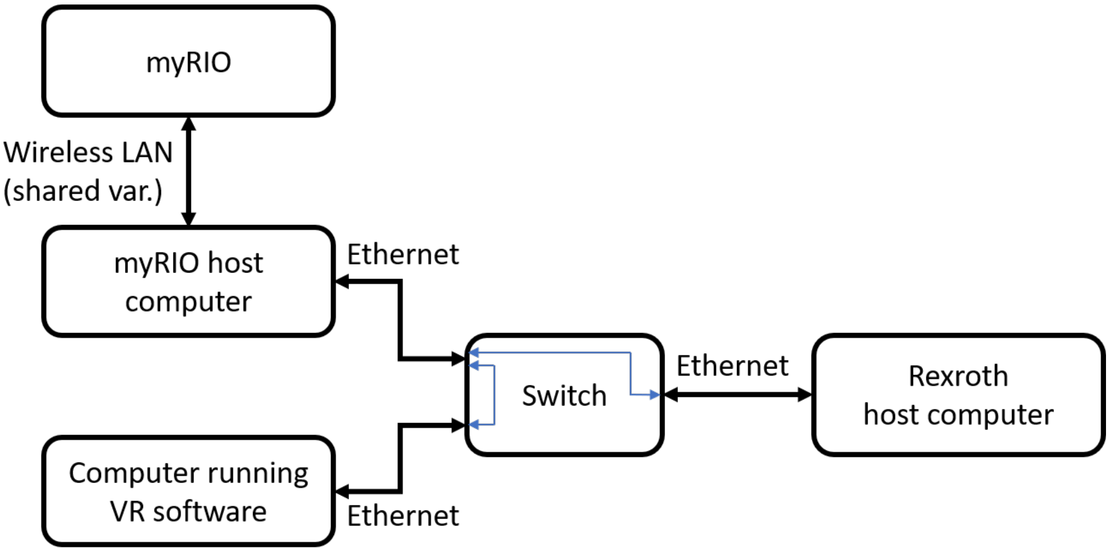


Figure 5.19: Communication diagram

When communication between all required units was established, testing of the VR interactive system could begin. When testing, the person on board the treadmill had to wear a VR headset, which is what the VR software was designed for. Within the virtual world, the subject could freely look around without influencing his virtual body posture. This is ideal for the purpose of recreating a realistic interaction with the environment. Figure (5.20) shows a photo of the setup while simulating skateboarding within virtual reality. The artificial environment developed for this test is created for skateboarding and includes a few speed bumps, obstacles, and corners to give the user opportunity to explore interact with the world. However, it is just one of many possible environments.

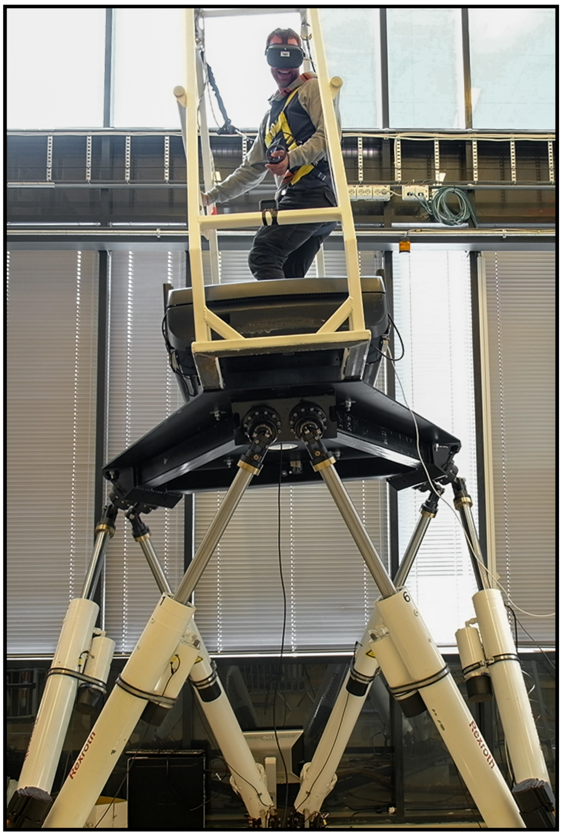


Figure 5.20: Test with VR

A snapshot of the artificial environment is shown in Figure (5.21). The shown position is the "starting position" of the track and the objective is to collect the stars scattered across the road.

Figure 5.21: Snapshot from the artificial environment as seen in the VR-goggles

To verify that the model responds as intended to inputs from the virtual ground, one must be selective when logging data. To recall, there are two board models; general board model (dynamic about X_N and Y_N axis, and along Z_N axis) and skateboard model (dynamic about X_N only). The board model used while logging is the general board model since it is much more advanced in terms of its dynamic response. Figure (5.22) shows logged data when simulating riding the general board over a bump. The upper sub-plot shows the registered location of applied load within the operating area (area of board), ranging from -1 to 1. The bottom two plots show the transmitted signals versus the input signals from the simulated environment (dashed lines). Notice how the transmitted output Z_b follows Z_{sim} almost perfectly just slightly behind (recall that negative Z is up). The simulated rotations on the other hand have less impact, yet enough to feel. The reason for why the simulated heave has more impact than the rotations is easier to understand if simulation input is neglected. When a person is located on the treadmill without holding onto something, the treadmill carries the person's whole weight. If it is assumed that the person is not jumping up and down, the magnitude of the weight will be approximately the same throughout the operation, even when the person shifts his/her weight across the board. Rotations of the board depend on the magnitude of applied load and location. Considering the magnitude is approximately constant, the orientation of the board is more or less only determined by the location of the applied load. Translation in Z direction on the other hand does not really depend on where the load is located, but rather on its magnitude, which implies that it will remain the same throughout operation considering no inputs from the simulation. This behavior is described by the general board model which is covered in Chapter (4), System Behaviour, Modeling and Control. Hence, the roll and pitch angle of the board is simpler to control by the user than its heave displacement.

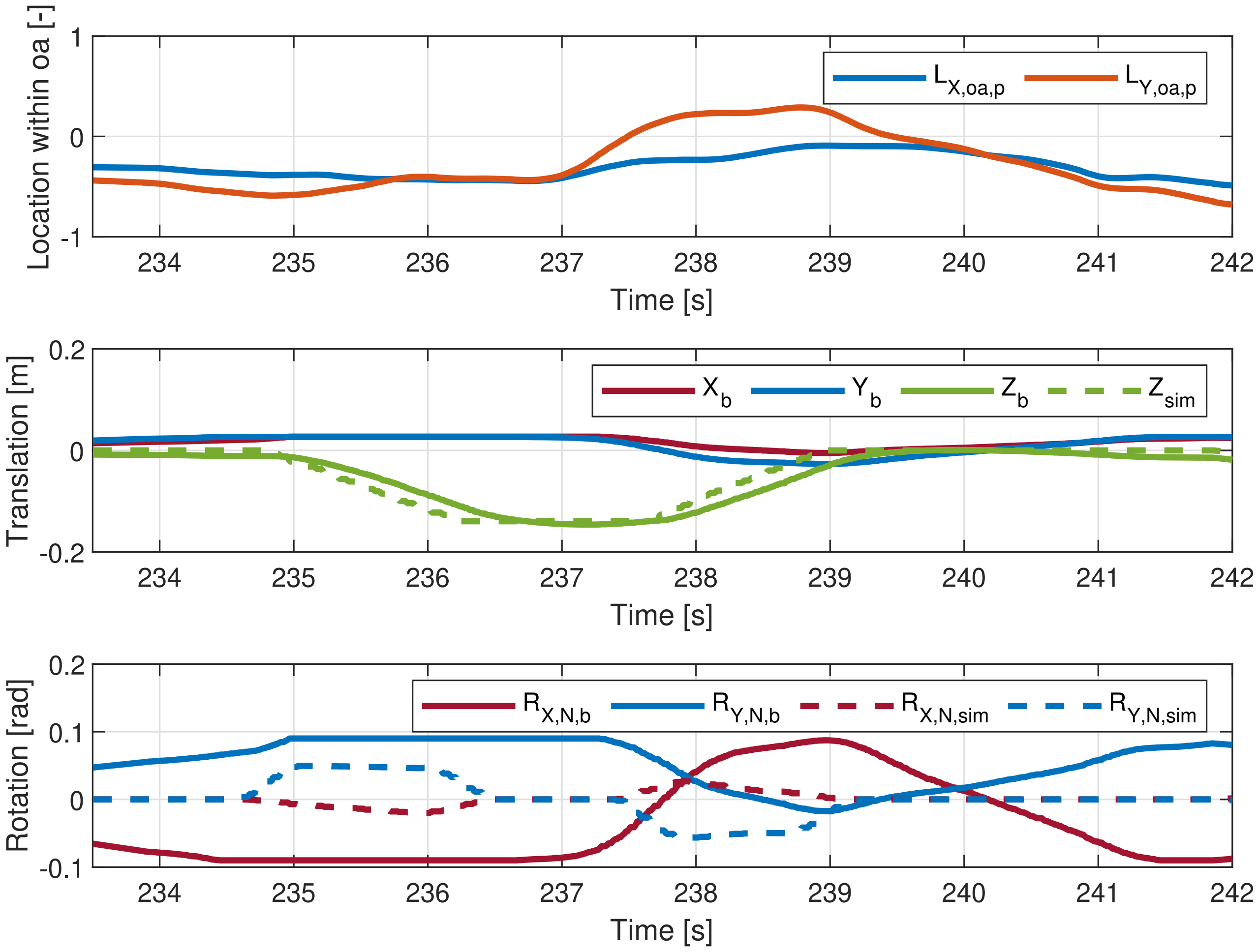


Figure 5.22: VR test plot

To interact with the virtual world, some values had to be generated by the model and transmitted back to the simulation software. When driving a car, riding a bike, or riding a board, there are two main controls responsible for getting you from point A to B - steering and velocity control. Concerning a skateboard however, the velocity control is your feet kicking off the ground to increase velocity and dragging it along to decrease velocity. This is not possible to realistically emulate on the treadmill, hence a temporary and quick solution where the operator sets the speed of the board within virtual reality was implemented instead. A handheld controller could be added at a later stage to give the subject control over the velocity.

Steering, on the other hand, is important that the subject has full control over. Conveniently, the input needed by the simulation from the model is the turn rate. In a real scenario, the turn rate of a board can be estimated to act proportionally to its roll angle. However, as described in Chapter (2), Background & Theory, there is a slightly different turning rate for a general board (snowboard, surfboard, etc) and skateboard. The two equations can be seen in Equation (5.1) and (5.2), where r_t is the turn rate of the respective board model, and f_t is a turn rate intensity factor.

General board:

$$r_{t,B} = R_{X,N,b} \cdot f_t \tag{5.1}$$

Skateboard:

$$r_{t,SB} = (R_{X,N,b} - R_{X,N,sim}) \cdot f_t \tag{5.2}$$

The inputs from the VR software influenced the signals transmitted to the Stewart platform almost immediately, which felt very natural when traveling over bumps within virtual reality. However, the relatively quick response from the simulation output in combination with the delay that was introduced when filtering the load cell signals could cause a challenge when countering the convex-shaped bumps if they were approached rapidly. With that being said, the feeling of having control over the simulation was always present, and the movements did not feel mechanical nor unnatural. When the goggles were on, the fact that you were located on a treadmill on top of a Stewart platform was quickly forgotten as you immediately felt present and engaged in the simulated environment.

5.4 Interview with external user

After initial tests, the system was ready to be tested by someone other than by the members of this group. Some questions have been asked regarding the user experience. This is what she said:

"For me, the response seemed fine, and I did not feel any particular delay. It felt quite natural. I forgot I stood that high on the Stewart platform, and it's quite immersive when you feel the ground moving. It might be even more immersive if the VR content was more realistic. I felt safe, though a thought that crossed my mind was that the safety frame could be bigger. A slip could cause the user to might hit his or her head. There are some improvements I feel can be made to enhance the user experience even more. A larger safety frame, a more realistic VR-environment, easier access to the platform itself, to be able to walk on the treadmill and the ability to control velocity myself."
- Julie M. Madshaven

6. Results

The idea behind CELNA is to develop a system with the possibility to perform an accurate analysis of the biomechanical condition of a human by the use of an advanced camera system. In addition, to rehabilitate in an efficient and motivating manner by combining a mechanically actuating structure and a virtual reality with human interconnection. Such a system has great potential in terms of its application within the sector of health care, and can be further developed to increase its field of application. The goal for this project was to create a proper base, regarding both hardware and software, for further development, and to establish a functioning human interconnection between mechanical actuators and virtual reality.

Concerning the hardware base, a treadmill was stripped of its control panel and side beams, and four load cells were attached to its underside in a rectangular pattern, as shown in Figure (6.1). This unit allows for pressure magnitude and localization analysis of the person while walking, running, jumping, or standing still. It essentially acts as a force plate and a treadmill simultaneously.

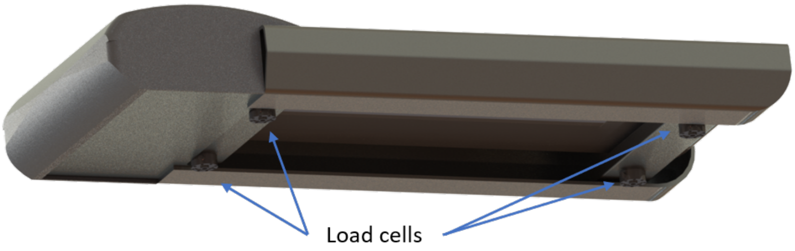


Figure 6.1: Illustration of load cells location

This unit consisting of load cells and a treadmill is then placed in a harness supporting safety frame. This frame is developed to be lightweight as it should be easily transportable, and strong enough to support a person falling with a safety harness attached to it. In addition, it was designed to be compact in order to fit on top of the Stewart platform, yet have space for the person to move. The handlebars were designed to be detachable since they could be in the way for certain scenarios. The final safety frame design is Shown in Figure (6.2).

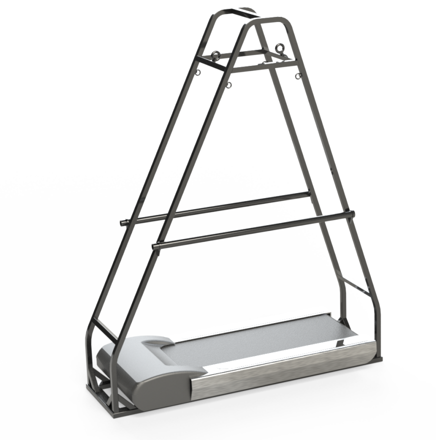


Figure 6.2: Final cage design

Stress analyses were done for the crucial parts of the safety cage - the eye bolts. To analyze the stresses within the eye bolts during the impact of a falling person, each eye bolt was subjected to a load corresponding to a mass of 175 kg within the Earth's gravitational field. This is considered a safe approach as the harness should be fastened to all four eye bolts by stretch ropes which prevent the risk of one eye bolt taking up all the force. The results of the stress analysis in Abaqus can be seen in Figure (6.3). It should be mentioned that the maximum stress calculated is just above 344 MPa. However, the location of this stress value indicates that it is affected by singularities in the calculation. Therefore, it is assumed that the realistic stress is closer to 250 MPa.

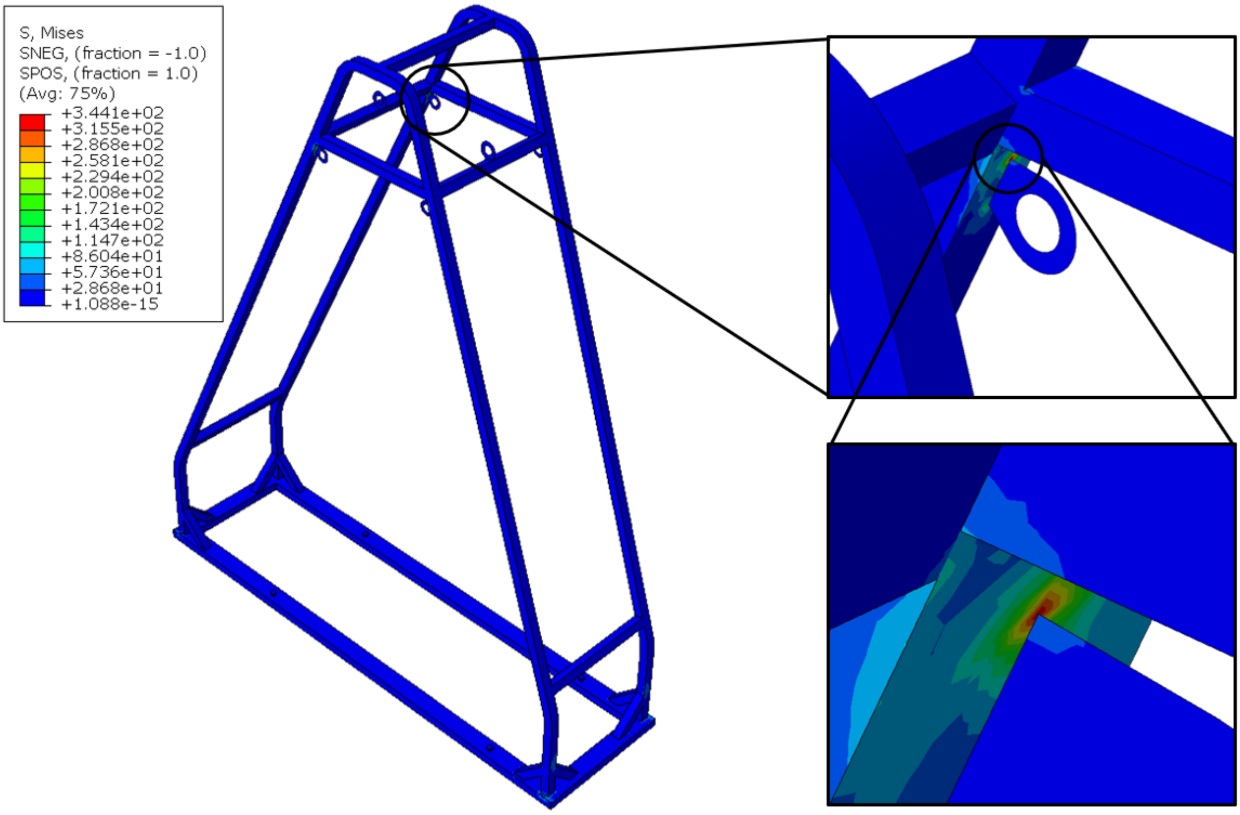


Figure 6.3: Scenario where the person is hanging form the cage.

When the treadmill with attached load cells is mounted on the frame of the safety cage, this whole unit, also previously referred to as the treadmill unit, can be transported as one. This was a requirement for this project as the Stewart platform is used for other applications as well, and a swift mounting and demounting procedure is therefore desired.

The final stage concerning the hardware base is to mount the treadmill unit onto the Stewart platform. Figure (6.4) shows a photograph of the treadmill unit mounted on top of the Stewart platform. On the walls, all around the Stewart platform (not visible in the photo), are 17 Qualisys cameras. These cameras allow for very accurate, real-time measurements of the biomechanical structure of a human being which can be used for analysis and rehabilitation. All 17 cameras are of course not needed to obtain accurate measurements (2-3 are sufficient), but due to other applications, 17 are available. Now, the hardware of CELNA has a huge potential. It has a six DOF actuator, sensors that can determine position and pressure magnitude of the person on board the treadmill, it can actuate the treadmill for walking or running, and the biomechanical structure of the person can be accurately measured and analyzed. How these features are combined depends on the application and scenario of relevance.

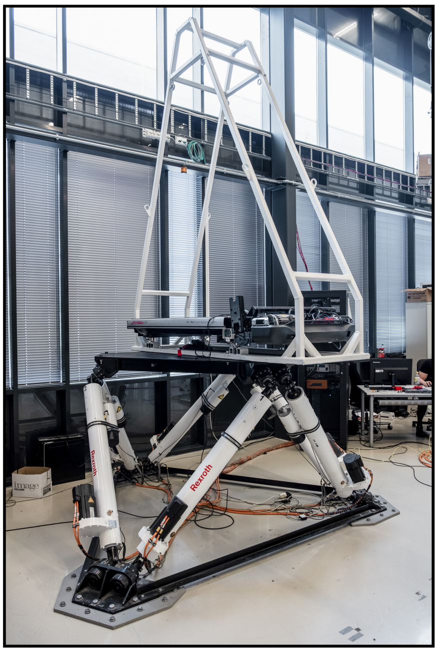


Figure 6.4: Treamill mounted on platform

In addition to make CELNA function as intended, the main objective is to create a good base for the software to facilitate further development. Hence, the program is developed based on a finite state machine architecture. This program architecture is easier to comprehend than most, since it only concerns the objectives related to the current state. The state machine developed for CELNA is illustrated in Figure (6.5) by the use of a state transition diagram.

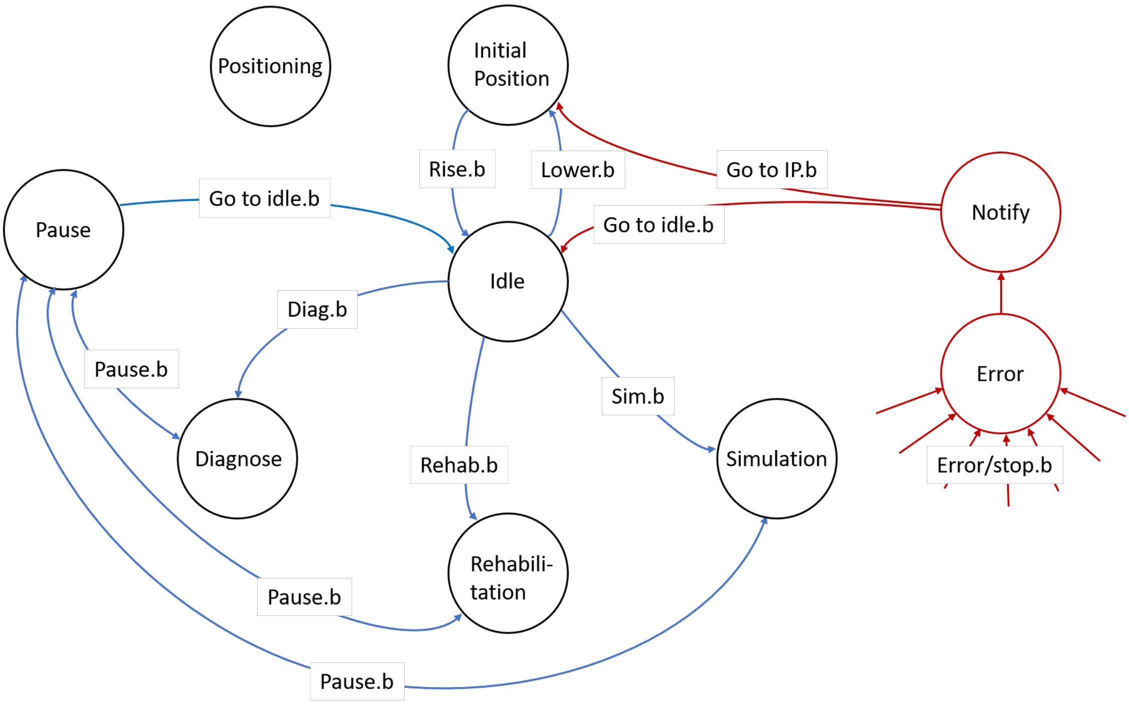


Figure 6.5: State transition diagram of Kurt

The main finite state machine program runs on a National Instruments myRIO, and communicates with a host computer over WLAN (shared variables). The program running on the host computer can be referred to as the operator and state selector program. In this program, the operator can select which states are wanted for the myRIO program, change simulation parameters and observe data. An illustration of the information flow between different units within CELNA can be seen in Figure (6.6).

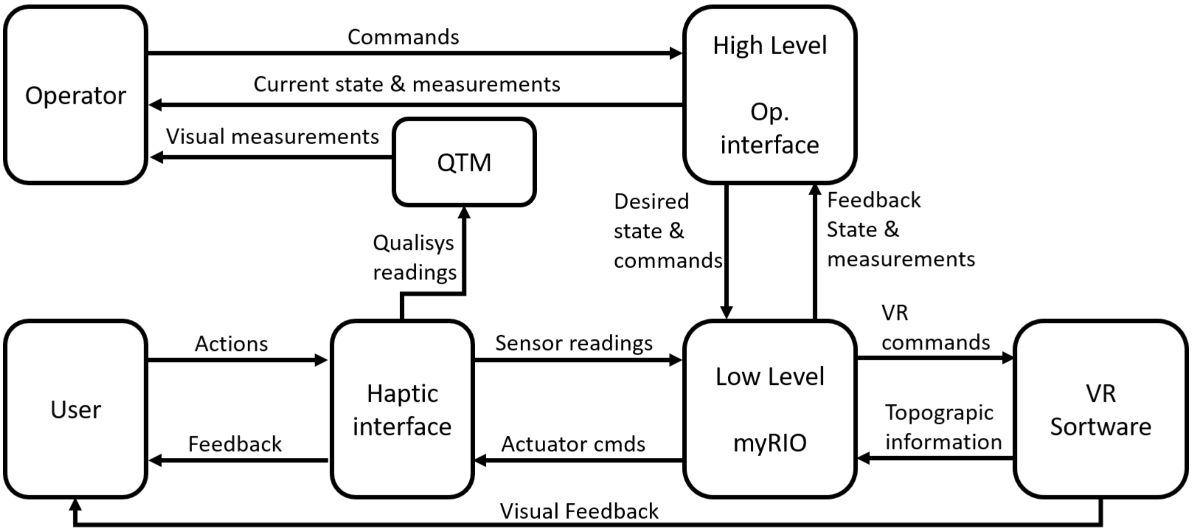


Figure 6.6: Flow chart of Kurt

The program running on the myRIO can be considered the brain of CELNA since all the real-time conditioning and calculations needed to create proper signals for the Stewart platform and VR software based on its inputs are done there. The general program flow within the myRIO is illustrated in Figure (6.7).

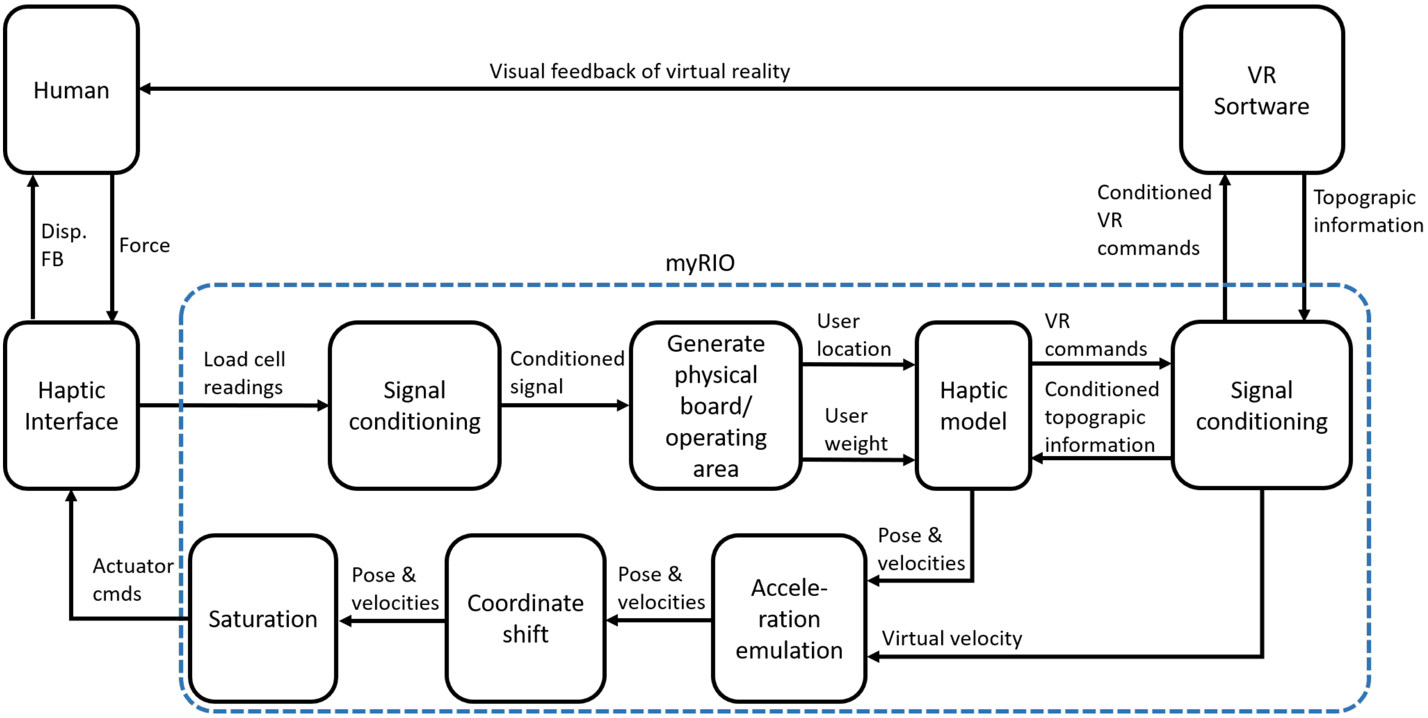


Figure 6.7: Visualisation of program flow executed within the myRIO

Two board models were developed in order to have the possibility to simulate multiple scenarios; a general board and a skateboard. The general board provides a dynamic response depending on the magnitude and location of the applied load, as it consists of four sub-models (mass-spring-damper models) - one at each side of the board (north, south, east, and west). Such a board model can be related to real boards such as snowboards, surfboards, wakeboards, balance boards, etc. A visual illustration of data flow in the general board model can be seen in Figure (6.8). It should be clarified that the model describes the dynamic behavior of the interaction between the board and the surface below and not the board alone, as the dynamic behavior mainly depends on the material it is in contact with.

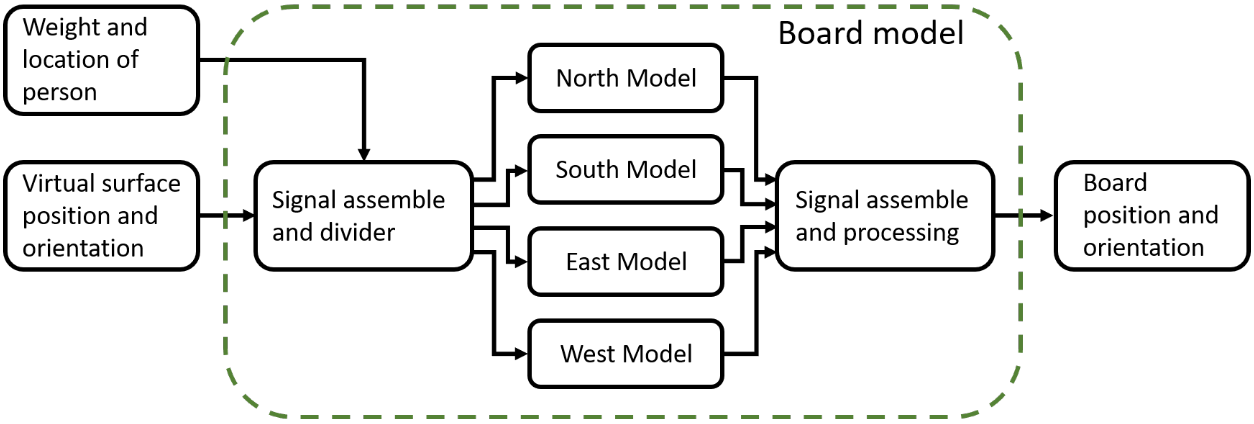
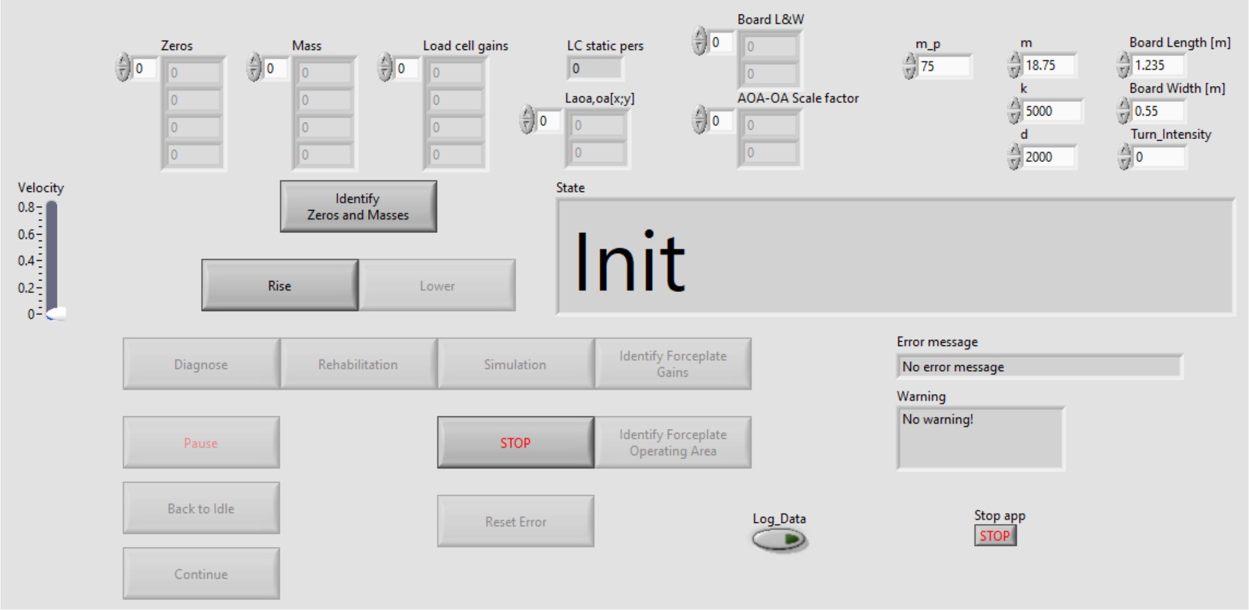

Weight and location of person
Virtual surface position and orientation
Board model
Signal assemble and divider
North Model
South Model
East Model
West Model
Signal assemble and processing
Board position and orientation

Figure 6.8: Flow chart of board model

The skateboard model is a separate board model as it has different characteristics in its dynamic behavior relative to the load input from the person onboard. By assuming that the skateboard is ridden on a very stiff ground, such as asphalt or concrete, the person can only control one degree of freedom - roll. The other degrees of freedom are determined by the pose of the surface below its wheels. The flow of information concerning the skateboard model is similar to the general board, which is shown above in Figure (6.8), except only one of the four sub-models is used - West sub-model.

When performing practical testing, an operator interface was developed. The interface is an early-stage prototype, but certain fundamental features were included. These features include disabling buttons not relevant for the current state of the system, parameter control for simulation, error and warning message display, an indicator what the current state of the system is, and conditioning indicators to validate the conditioning variables. The early-stage operator interface (also referred to as user interface, UI) developed for practical testing is shown in Figure (6.9).



Zeros 0
Mass 0
Load cell gains 0
LC static pers 0
Board L&W 0
Laoa,oa[x;y] 0
AOA-OA Scale factor 0
m_p 75
m 18.75
k 5000
d 2000
Board Length [m] 1.235
Board Width [m] 0.55
Turn_Intensity 0
State Init
Error message No error message
Warning No warning!

Figure 6.9: Operator interface

First, dry run run testing with the Wii-board used as a force plate and the transmitted signals used as feedback (pose feedback is required to enter certain states) was carried through. This was done to verify the continuity of the transmitted signals to prevent irrational behavior of the platform. And further to make sure the conditioning of load cell signals and the establishment of the operating area were correct. Below is a series of plots gathered from the verification testing. A more in-depth description of how the tests were carried through and what response to expect, can be found in Chapter 5, Application and Testing.

Figure (6.10) shows continuity testing of the state transition, i.e. going from initial position to idle position and vice versa. It also shows testing of the stop button during the transition and the identification process. Only transition in Z_N direction is visualized as this is the only degree of freedom used when not simulating.

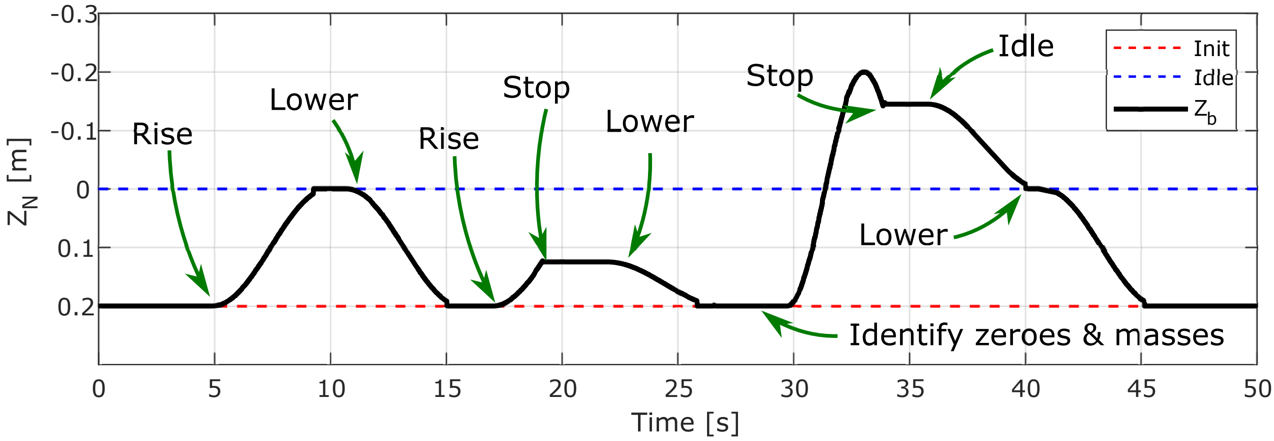


Figure 6.10: Dry run testing transmitted signals during state transitions

Figure (6.11) shows the continuity in transmitted signals during the simulation. Here, it is important to verify that the stop and pause button function as intended and that all degrees of freedom return continuously to the correct values when going to the idle state.

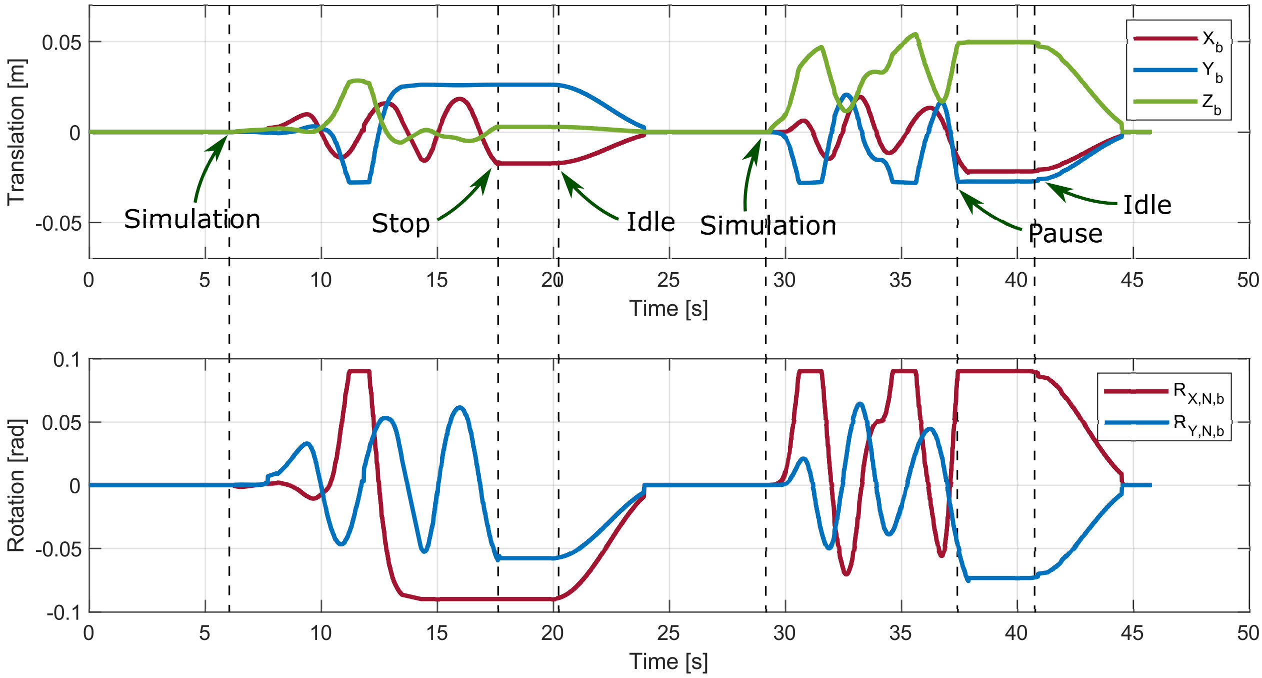


Figure 6.11: Dry run testing transmitted signals during simulation

Figure (6.12) shows the effect of conditioning the load cell signals. The numbers above the plots indicate the steps of an illustrative testing process designed to visualize the effects. 1: Raw signals (no load), 2: Zeroing signals, 3: Apply load to find averaging gains, 4: Multiply with the gains, 5: Deload to verify conditioned signals return to zero at the same rate. The two plots are similar but with different ranges in the Sensor values axis to better show the effects.

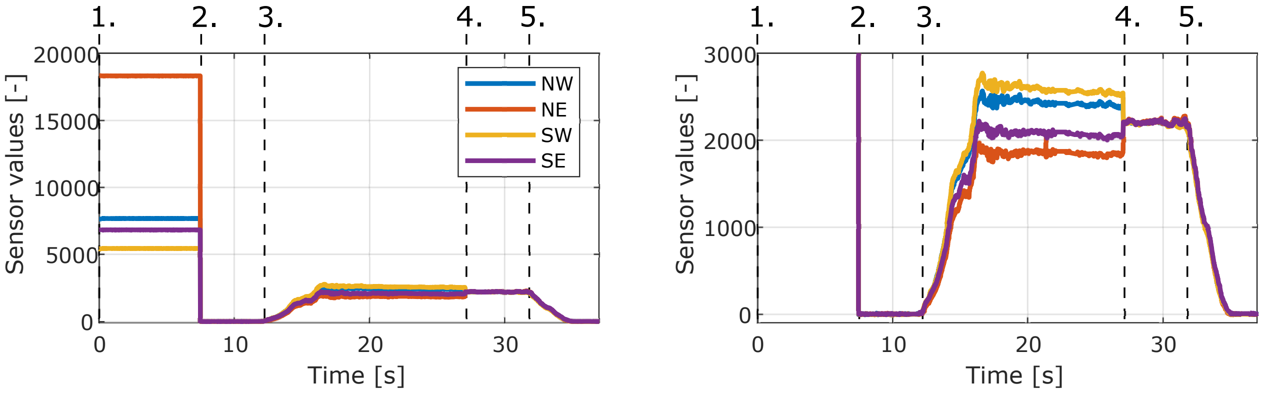


Figure 6.12: Verification of load cell conditioning

Then, a verification process of the calculations was carried through to establish an operating area. The operating area is the surface area on the treadmill representing the size of the virtual or simulated board. This is the area the user is allowed to operate on during that specific simulation, hence the name operating area, OA. Figure (6.13) shows the result of the testing process, where the steps were: 1: Load center of OA, 2: Deload center of OA and load NW corner of OA, 3: Deload NW corner of OA and load NE corner of OA, 4: Deload NE corner of OA and load SW corner of OA, 5: Deload SW corner of OA, load near the center spot of OA and alternate magnitude of the load. The top subplot shows the conditioned load cell signals, while the middle shows the location of the applied load in both the available operating area (aoa,p) and the operating area (oa,p). Both coordinates range from −1 to 1. The bottom plot shows the magnitude of the load.

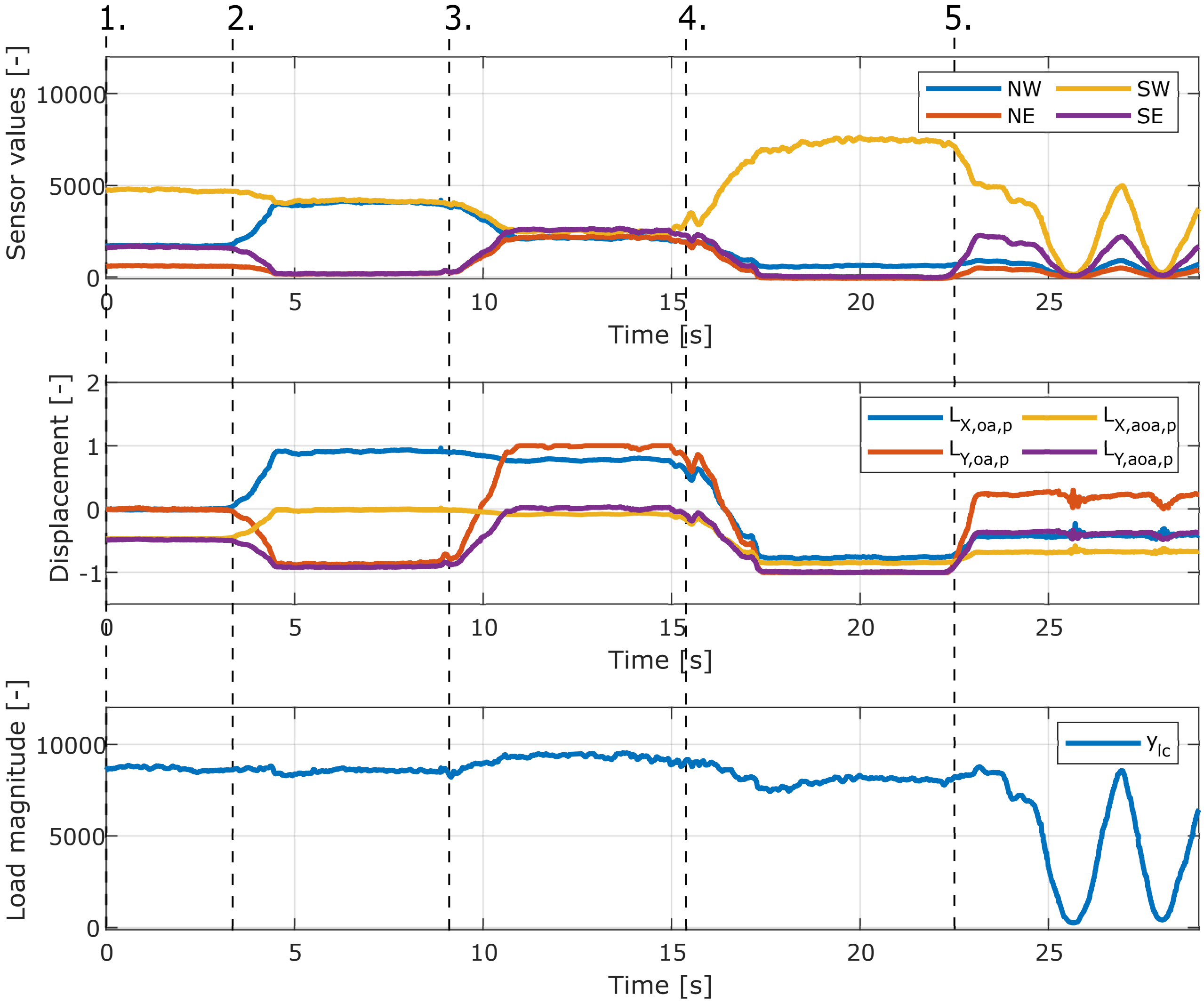


Figure 6.13: Verifying load location within operating area

The last test concerning the verification of signal conditioning and calculations was the shift of the control point. For this test, the control point was shifted some distance in negative Z_N direction (up). Figure (6.14) shows the result of the sequential test carried through. The steps of the process were: 1: Deload force plate, 2: Load and deload NW corner of OA, 3: Load and deload NE corner of OA, 4: Load and deload SW corner of OA, 5: Load and deload SE corner of OA, 6: Load and deload North side of OA, 7: Load and deload East side of OA. This test was done to verify that the platform translates in the correct direction within the $X_N Y_N$-plane for specific rotations about X_N and Y_N axis.

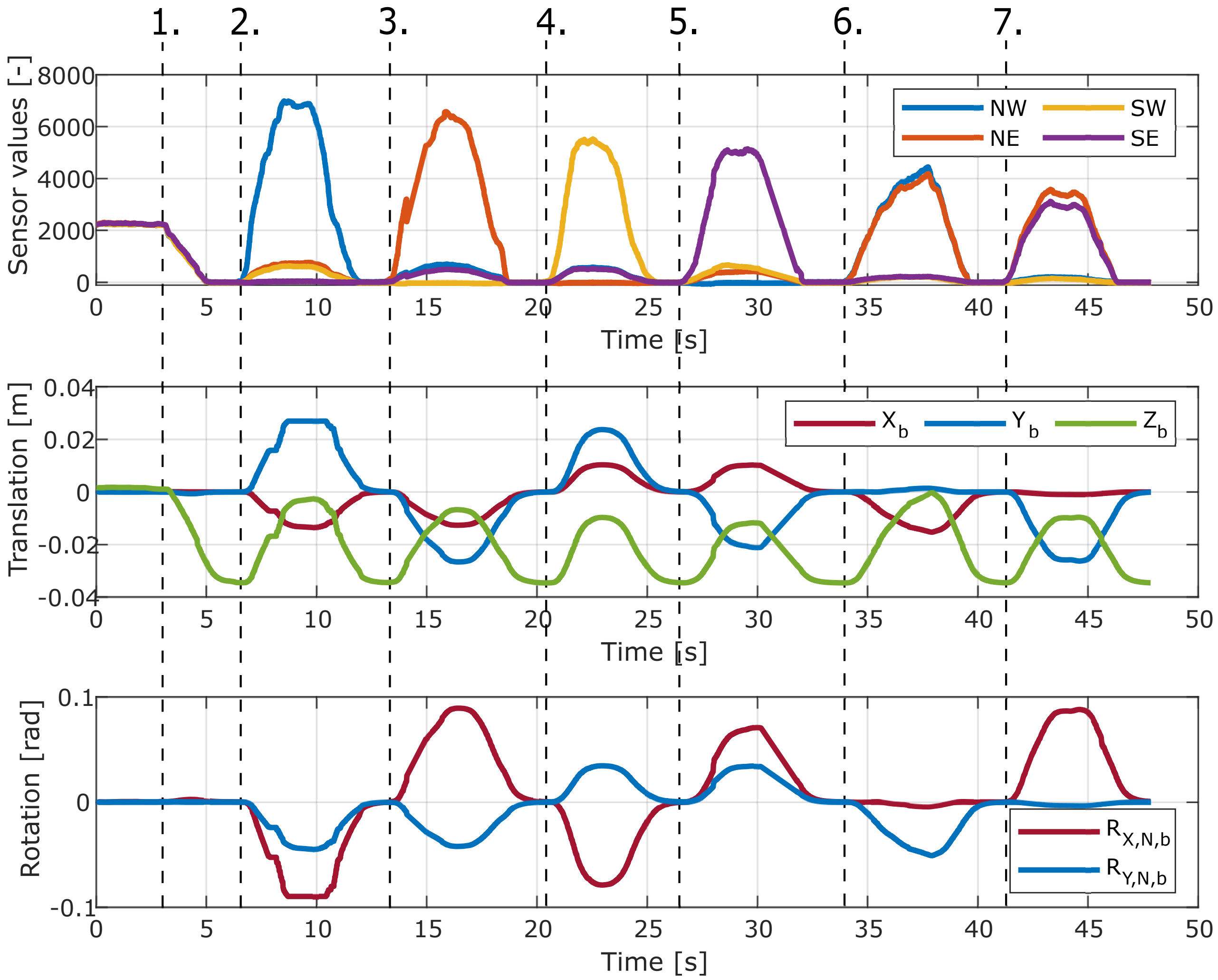


Figure 6.14: Verification of directions

When all values seemed reasonable from dry run testing, and all possible verifications were done, the program was implemented on the physical system. In other words, the Wii-board was swapped out for the treadmill and load cells, and the feedback pose was now received from the actual Stewart platform. First, the system was tested with the treadmill unit placed on the ground. At this point, the system behaved just as expected with no problems at all. However, as the treadmill unit was mounted on the Stewart platform, and simulation was engaged (unmanned), the Stewart platform showed tendencies of increasing oscillations. This was fixed by filtering the load cell signals by a low pass filter to prevent rapid disturbances to have an effect on the model. The effect of the filtering process can be seen in Figure (6.15), where only one signal is visualized for simplicity. The filter is in fact a mass-spring-damper model which was parameterized to guarantee safe operation due to very limited time for testing because of the COVID-19 situation. This filter is not optimal as it causes an additional 400 ms time delay between applying load and model response, which is a lot compared to the originally 200 ms caused by the dynamics of the board model. Therefore, the filter has potential for improvement.

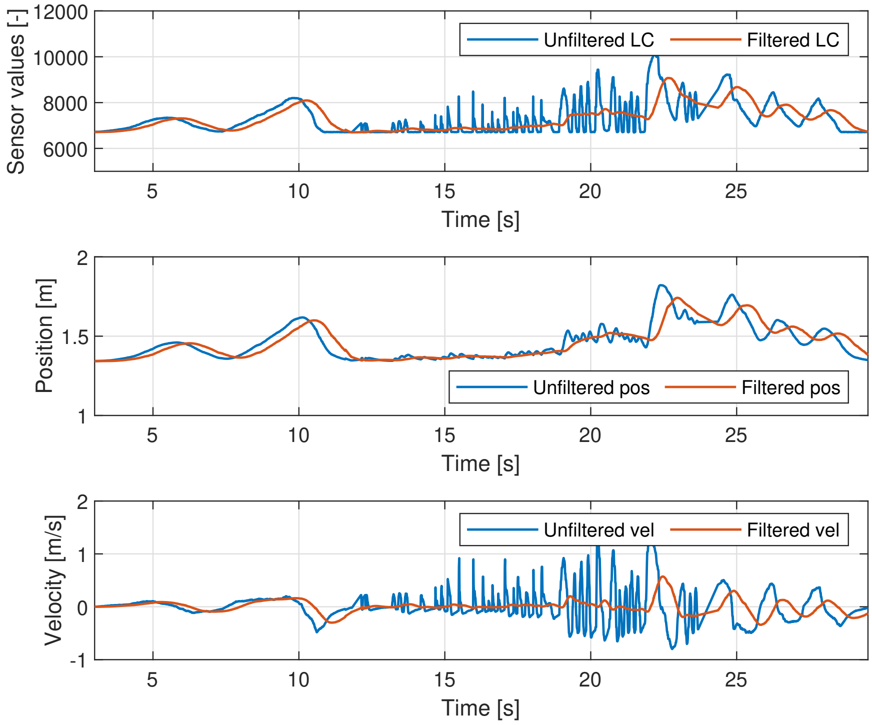


Figure 6.15: Effect of mass-spring-damper filter

Although the filtering slowed down the response dramatically, it did an excellent job of removing the oscillations. While CELNA was unmanned and in simulation mode, aggressive pull tests were carried through to ensure that the platform remained stable. There were no indications of oscillations, and CELNA was concluded safe enough for manned testing. With the person on board, there was no sign of instability, despite the user was trying to provoke it. Moreover, even though the platform responded just above half a second after the load was applied to a certain place, it did not feel unnatural. The response could resemble standing on a small boat, which of course might not be ideal as far as skateboard simulation goes, but it proves the working principle and still has its areas of applications. Having said that, the cause of the majority of this delay is known and also resolvable.

Finally, CELNA was ready to be tested in cooperation with VR software. The VR software was running on an external computer, communicating with the myRIO host computer. The myRIO host computer was also communicating with the myRIO and Rexroth host computer. The flow of information between computers is shown in Figure (6.16).

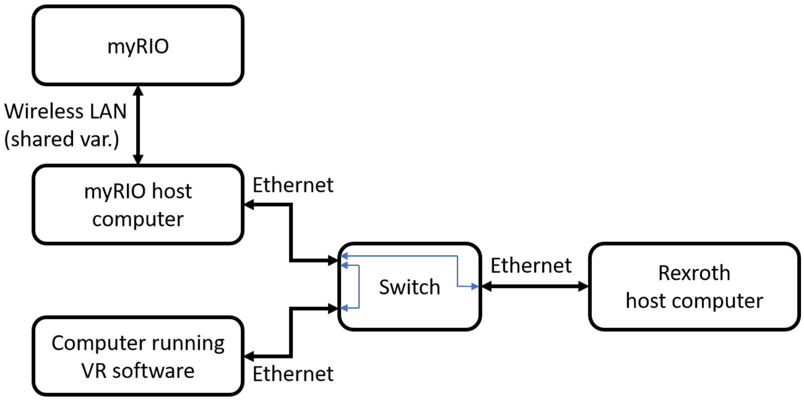


Figure 6.16: Communication diagram

The artificial environment used for the practical tests was developed by another group of students and featured an urban scenario with some bumps and obstacles - suitable for skateboard simulation. A snapshot of the environment can be seen in Figure (6.17).

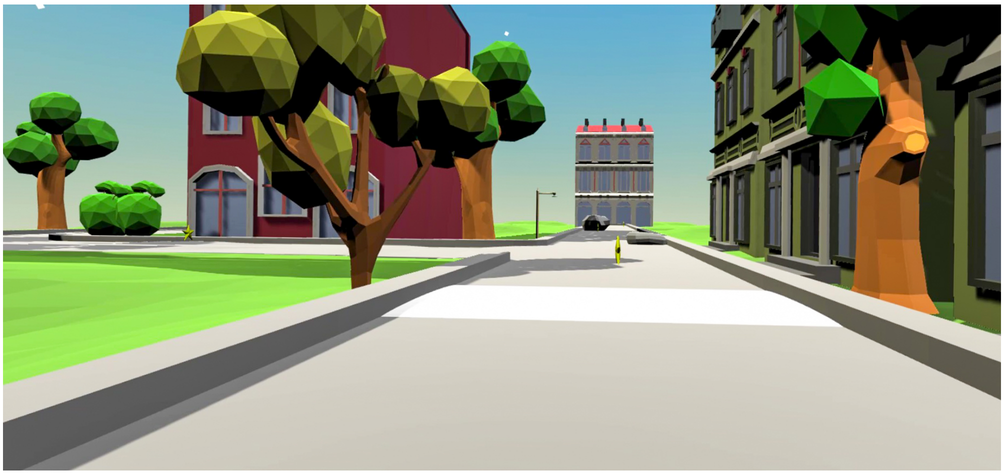


Figure 6.17: Snapshot from the virtual reality as seen in the VR-goggles

The artificial environment was visualized through a pair of Oculus Quest VR goggles which was worn by the user. This gave the user the possibility to look around and observe while moving in different directions, just like in reality. Moreover, a VR headset/goggle fully isolates the virtual world from the real world in terms of field of view, which enhances the feeling of actually being a part of the virtual world. A photo of an early-stage version of CELNA with human interaction can be seen in Figure (6.18).

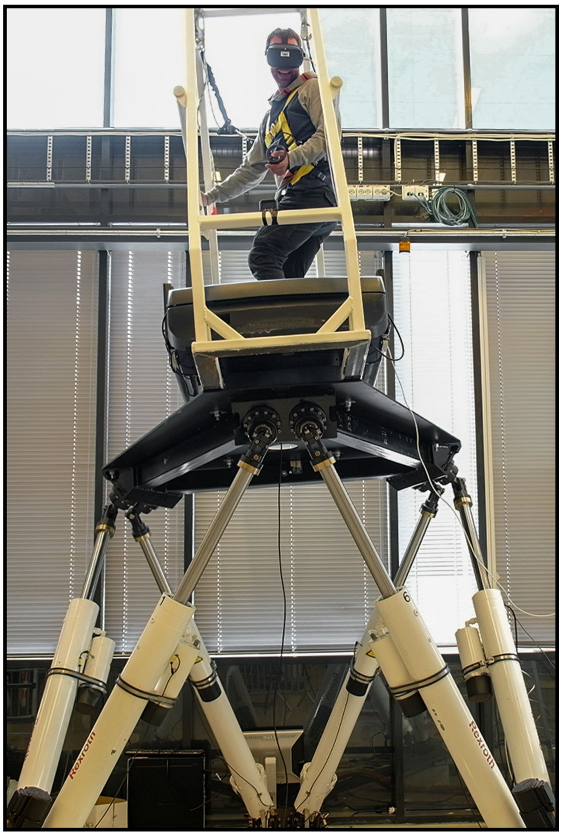


Figure 6.18: Test with VR

Figure (6.19) shows a section of the logged data where the skateboard was ridden over a bump in the road at an angle. The top subplot shows the location of the load applied by the person within the operating area, while the bottom two plots show the transmitted signals in comparison with the signals received from the VR software.

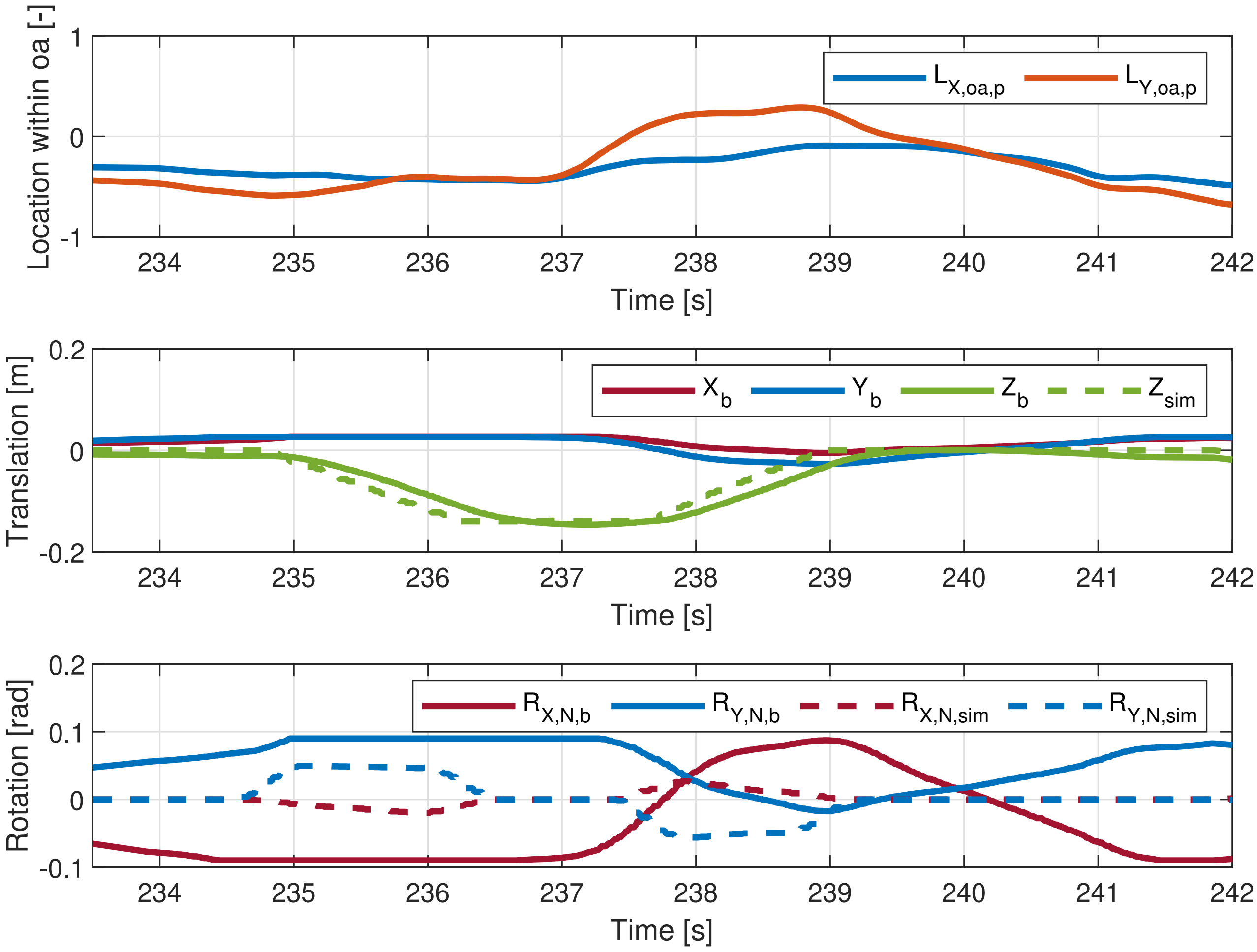


Figure 6.19: VR test plot

7. Discussion

During the time of writing this report, the COVID-19 virus has caused lockdown of necessary facilities in order to limit the spread of the virus. This limited the time for testing to five days in the period from March 12^{th} until the due date for this project. As a result, some objectives have been prioritized over others which have instead been recommended for future work. The prioritized objectives have relevance to the balance training feature of CELNA rather than walking/running. One reason for the priority of balance is its importance at the early stages of the rehabilitation process, as balancing is essential to gain control over preliminary to relearning how to walk or run. Another reason is that balance is an attribute that can always be challenged and improved by the use of the right tools. Good dynamic balance and stability are beneficial in so many ways regarding daily life, but perhaps especially in sports. The velocity control of the treadmill and Qualisys camera system are not directly connected to the balance training feature of CELNA, which is why they do not have any major part in this report.

The safety frame is an alpha prototype due to the limited time available for developing a functioning prototype of CELNA. Thus, the frame has some structural features which might be inconvenient concerning the vast variety of height and width of people. First of all, the distance from the treadmill surface to the eye bolts, which is considered the maximum operating height due to the transverse and longitudinal steel bars, is just about two meters. Moreover, since the frame slopes inward toward its top, the taller the person, the higher the risk of bumping his/her head into the frame. Such a design is especially convenient in terms of space occupation and transportation, but also regarding the efforts needed by the Stewart platform to move the frame. From a healthcare sector point of view however, the tight space in combination with steel beams might not be optimal as the chance of inflicting damage to the patient is present. That is why, especially when using this safety frame, it is very important to wear a helmet.

When performing the practical testing of CELNA at the laboratory, the program flow of the finite state machine worked as intended. However, it should be clarified that a couple of important features that it was designed for were not included during the tests - the user stop-button and confirm/pause-button. These buttons were implemented in the software to improve safe operation, where the stop-button stops CELNA in its current pose regardless of system state. The confirm/pause-button is used by the patient to confirm that he/she is ready to proceed to the state selected by the operator outside simulation and to pause the simulation while simulating. Although the hardware for these buttons was not added during the alpha testing, the software has been developed to account for these interactions. During alpha testing, these buttons had to be pressed by the operator through shared variables.

The hardware executing the main program - National Instrument's myRIO, is a student embedded device which performs the task well enough. However, it is inconvenient that the myRIO does not have an Ethernet port, as Ethernet has many benefits over a wireless connection for an application like CELNA. CELNA can be regarded as a stationary system once set up, and so is most likely the operator computer. Hence, an Ethernet connection between the operator computer and myRIO is just as convenient as wireless. However, wireless connections are in general less stable than Ethernet as wireless networks are prone to disturbances such as bad connections due to range and radio interference. Moreover, since wireless networks transmit and receive information by radio waves traveling through an open space, the information can be accessed and interfered with by unauthorized personnel. An industry-oriented controller unit such as the compactRIO would probably be better suited for this application.

When the practical testing of CELNA was done, the communication between virtual reality, control unit, operator computer, and Stewart platform was done in a rather inefficient manner. There were several communication ports needed to establish a network between all the units. Several units communicating via UDP increases the total amount of lost packages and introduces additional latency which can be avoided. The Rexroth Stewart platform is a preinstalled component at the laboratory with all its needed additions such as a control unit and host computer already equipped. By using this host computer as the operator and host computer for the CELNA control unit (myRIO) as well as running the VR software, the amount of units needed to establish the required communications is heavily reduced (relatively). Doing so is also convenient in terms of simplicity and efficiency when starting up the system.

The operator interface developed for alpha testing has its core principles included - the impossibility to click a button not relevant for the current state, displaying warning/error messages, parameter adjustments, and variable observability. However, it is desired that information of interest regarding physiotherapy can be visualized and logged in a simple yet fulfilling manner, which the alpha program does not support. Adding these features can be done by implementing code into the current program which enables the operator to observe and analyze the real-time data in an informative way, as well as store it for later use. A physiotherapist and/or medical staff should have a part in the development of the operator interface as its functioning should suit their needs.

When performing the first test with the treadmill unit mounted on the Stewart platform, increasing oscillations occurred. The system had to be brought to a stop by switching off remote control (remote control enables the platform to be controlled by data received through UDP port). This was clearly a result of disturbances affecting the load cells, which most likely originated from the acceleration of masses. When the load cell measures an increasing load, the model angles in that load cell's directions, which causes a downward acceleration which in turn decreases the measured load. Such behavior was to some degree expected and intended to be compensated for. However, since the compensation of acceleration of masses depends on feedback from the Stewart platform, rapid accelerations will not be correctly accounted for. In fact, such compensation can be counter-intuitive if the oscillations reach a frequency where the feedback (used for compensation of acceleration of masses by subtraction) is half a period behind the transmitted signals, thus amplifying the transmitted signals instead of reducing them. Hence, during testing, this compensation algorithm was not used. Moreover, a low-pass filter was constructed and added to prevent rapid accelerations to influence the load cell signals to the degree it was noticeable. The filter had to be developed quickly as the facility needed was available for only five days, and this problem was encountered on the third day. Therefore, the filter developed was four mass-spring-damper models - one for each load cell, which can be vastly modified to achieve a suitable response. However, such a filter with the parameters selected adds a noticeable delay to the overall response of CELNA, which is undesirable in the long term. With that being said, the filter did an excellent job during alpha testing concerning the given circumstances. Of course, this implies that there is a huge potential for improvements regarding the method of reducing/removing load cell measurement disturbances due to the acceleration of masses.

The user interacts with the environment as one would in the real world - turn left by pressurizing the left side of the board, and turn right by pressuring the right side of the board. However, on most standard commercial boards, the velocity is controlled by the environment (example: the slope of a hill) and can be altered by interfering with it in certain ways. When performing alpha testing, the virtual velocity was determined by the operator and the user could not interfere in any way other than by communication. This method of virtual velocity control could most likely have some use areas where it is beneficial in terms of diagnosing the patient or during a controlled rehabilitation process. However, for certain scenarios, it might be desired that the patient has control over the virtual velocity him/herself. To do so, a handheld control, very similar to those used for electric skateboards, could be used. There is usually two handheld controllers in most commercial VR kits, which is a good alternative to handheld velocity control as it has a wireless and direct connection with the VR software.

8. Future Work & Recommendations

Some objectives first considered as parts of this project have been shifted into the category of future work and recommendations. Even though the base functions of the system seem to work properly, some functions are still missing and need to be incorporated into the design.

Motion Capture System

One of the main purposes of this system is to give real-time feedback both to the user, and the therapist in charge of the rehabilitation. The Qualisys Motion Capture system is mounted in the area of the system but has not been ready for use during the time of this project due to the service of multiple cameras. Implementing gait feedback to the user and the operator will greatly aid the process of which the patient is relearning walking and/or balancing. This can be done in different ways, from simply recording the patient, to using reflectors to get a more detailed view of the movements of the user.

Treadmill Velocity Control

The hardware required for setting up a velocity controller is mounted inside the treadmill. An objective for future use is to combine the virtual reality and Stewart platform together with the motion of the treadmill to allow for walking in a virtual world. As described in Section (2.2), Technical Functioning, using the information from the motion capture system and the load cells, the speed controller should be designed to keep the user in the center of the treadmill. Contrary to conventional treadmill use, where the user follows the speed of the treadmill, the treadmill should follow the speed of the user. This is necessary to properly emulate a realistic experience of walking where the user determines the walking pace.

Additionally, as mentioned in Chapter (7), Discussion, a handheld speed controller should be added to allow the user to control his/her own virtual speed during balance mode. This is to control the virtual speed at which he/she travelling on the board through artificial environments.

Weight Support Harness System

Similar to other walking rehabilitation systems, the implementation of a harness system that can take a certain load from the user is beneficial. This opens up the opportunity for the user to do the correct movements without overloading weak muscles, bones, or joints. As the patient is making progress, the assisting load can be reduced to promote further development. Additionally, this will serve as the safety harness for the patient during operation in case of falls or for other reasons. An example of an existing type of weight support system is shown in Figure (8.1).

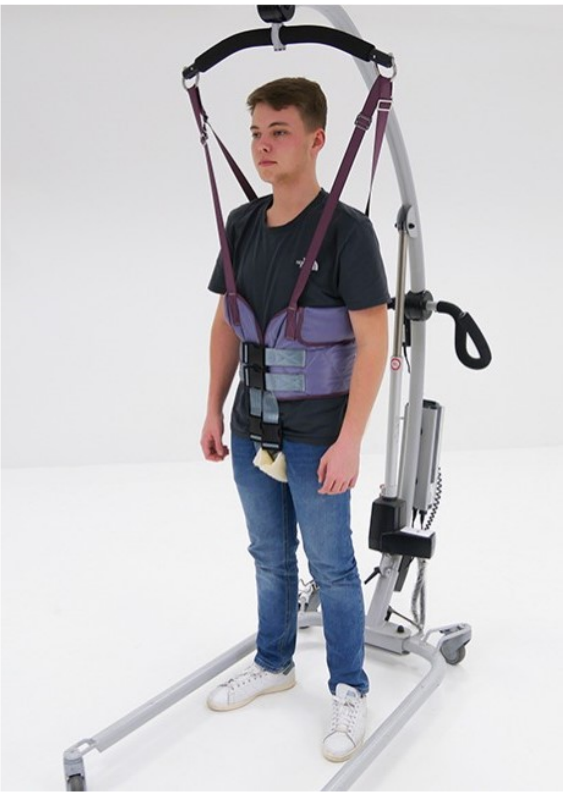


Figure 8.1: A harness system which supports weight allowing for walking under reduced weight [43]

Safety Frame

For the purpose of this project, the safety frame has worked well. However, in the future, a safety frame which is separate from the Stewart platform should be considered in order to minimize movement if the user suddenly is suspended from the safety harness. In addition, this will eliminate the risk of colliding with the frame while moving around, and remove the current height limit of the patient. This frame may also serve as mounting for a motion capture system.

Emergency Stops

A feature which is currently lacking from this system is the ability for the user to shut down the system during operation if necessary. These emergency stops are to be integrated in a few different ways. One of these is to be integrated into the harness system in order to stop the system if the user applies too much force to the harness which could potentially lead to a fall and injury. Next, a button-style emergency stop should be within arms reach for the user at any time. If at any time, the user feels the need to stop the system, an emergency button should be available for the user to press. A reasonable placement for this could be on one of the VR-controller, on the harness, or the supporting railing along the side of the treadmill.

9. Conclusion

Additional to producing a working prototype system, the main objective of this project is to create a good base for further work in the development of a biomechatronics lab. The purpose of CELNA is to eventually serve as a tool for medical staff and therapists in the process of diagnosing and rehabilitating patients suffering from a variety of different injuries, diseases and dysfunctions. On top of that, doing so in an engaging software generated artificial environment using mechanical actuators and VR-goggles.

The CELNA-system fulfills the main objectives of the project successfully, consisting of a Stewart platform, a treadmill, a safety frame, load cells, a control system, and a virtual reality visualized by a pair of VR-goggles. Communication over UDP has been an essential part of the system, making it possible to transfer data between the different components quickly and reliably. The load cells, which measure the force applied by the user, are sufficient for this application. They successfully transfer reliable readings which, along with surface pose from the VR software, provide the information required by the haptic model. The haptic model behaves rationally compared to its inputs and provides a natural, dynamic behaviour of the Stewart platform. The constructed safety frame functions as a base on which the treadmill can be permanently mounted, in addition to being the structure in which the safety harness is mounted.

By using LabVIEW, the control system interface is uncomplicated and should be simple for a therapist to operate with an introduction. It is simple to engage the different modes, and by the use of careful programming, the system is not unpredictable during operation, nor in case of emergencies where the operator presses the "stop"-button. The control system includes safety features allowing both the operator and the user to stay in control before anything is engaged further. In future projects, a more sophisticated filtering process is recommended to reduce the latency between input force and output motion.

The user can successfully operate the avatar in the provided virtual reality in an intuitive and engaging manner. As the user interacts with the virtual world, the feedback is realistic to the point where the user can experience that the motion is predictable in relation to what can be seen through the VR-goggles. This suggests that CELNA is promising for further development, where a motion capture system and a treadmill velocity control system is recommended to implement.

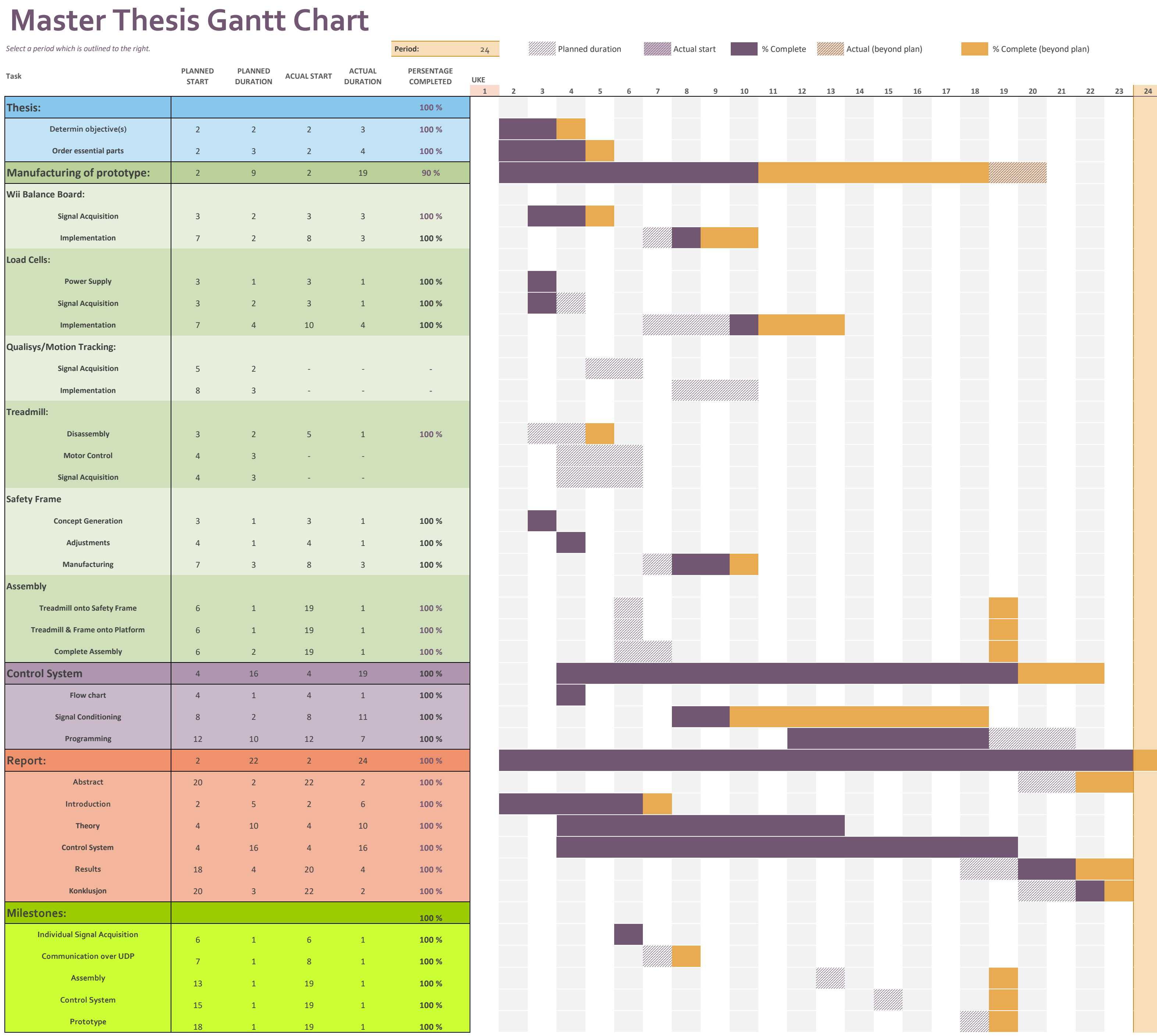
As this has been a proof of concept using mostly existing hardware located in the lab, the cost of this construction is nearly irrelevant. Some components are oversized which can be exchanged with more suitable and less expensive components. Additionally, even though the motion capture system was never incorporated during this project, 17 cameras is excessive and three cameras would suffice.

To conclude, the project is regarded as a success in terms of developing an early-stage prototype of a system that can be used as a tool for rehabilitation and training. The system functions properly and engages the user in a virtual reality where the ability to control the avatar using core movements and balance is rewarding. This opens the opportunity to further develop the virtual reality, motion capture, and more sophisticated rehabilitation methods.

Appendices

A. Project Management

A.1 Gantt Chart



Master Thesis Gantt Chart

Select a period which is outlined to the right.

Period: 24

Task	PLANNED START	PLANNED DURATION	ACUAL START	ACTUAL DURATION	PERSENTAGE COMPLETED
Thesis:					100 %
Determin objective(s)	2	2	2	3	100 %
Order essential parts	2	3	2	4	100 %
Manufacturing of prototype:	2	9	2	19	90 %
Wii Balance Board:					
Signal Acquisition	3	2	3	3	100 %
Implementation	7	2	8	3	100 %
Load Cells:					
Power Supply	3	1	3	1	100 %
Signal Acquisition	3	2	3	1	100 %
Implementation	7	4	10	4	100 %
Qualisys/Motion Tracking:					
Signal Acquisition	5	2	-	-	-
Implementation	8	3	-	-	-
Treadmill:					
Disassembly	3	2	5	1	100 %
Motor Control	4	3	-	-	
Signal Acquisition	4	3	-	-	
Safety Frame					
Concept Generation	3	1	3	1	100 %
Adjustments	4	1	4	1	100 %
Manufacturing	7	3	8	3	100 %
Assembly					
Treadmill onto Safety Frame	6	1	19	1	100 %
Treadmill & Frame onto Platform	6	1	19	1	100 %
Complete Assembly	6	2	19	1	100 %
Control System	4	16	4	19	100 %
Flow chart	4	1	4	1	100 %
Signal Conditioning	8	2	8	11	100 %
Programming	12	10	12	7	100 %
Report:	2	22	2	24	100 %
Abstract	20	2	22	2	100 %
Introduction	2	5	2	6	100 %
Theory	4	10	4	10	100 %
Control System	4	16	4	16	100 %
Results	18	4	20	4	100 %
Konklusjon	20	3	22	2	100 %
Milestones:					100 %
Individual Signal Acquisition	6	1	6	1	100 %
Communication over UDP	7	1	8	1	100 %
Assembly	13	1	19	1	100 %
Control System	15	1	19	1	100 %
Prototype	18	1	19	1	100 %

B. Component Overview

B.1 Frequency Converter

3

Technical Data
Technical data of MOVITRAC® 07

230 V_{AC} / 1-phase / size 0L / 1.1 ... 2.2 kW / 1.5 ... 3.0 HP

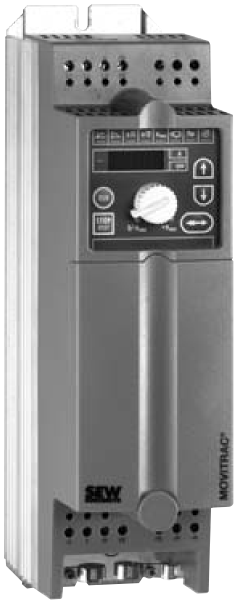


Figure 6: MOVITRAC® 07 / size 0L / 1-phase 230 V_{AC}

MOVITRAC® MC07A (1-phase supply system)		**011-2B1-4-..**	**015-2B1-4-..**	**022-2B1-4-..**
Part number		826 954 8	826 955 6	826 956 4
Part number with LOGODrive		827 188 7	827 189 5	827 190 9
INPUT				
Connection voltage Permitted range	V_{mains}	1 x 230 V_{AC} V_{mains} = 200 V_{AC} -10 % ... 240 V_{AC} +10 %		
Supply frequency	f_{mains}	50/60 Hz +/-5 %		
Rated system current, 1-phase at V_{mains} = 230 V_{AC}	100% I_{mains} 125% I_{mains}	13.4 A_{AC} 16.8 A_{AC}	16.7 A_{AC} 20.7 A_{AC}	19.7 A_{AC} 24.3 A_{AC}
OUTPUT				
Output voltage	V_N	3 x 0 ... V_{mains}		
Recommended motor power under constant load (with V_{mains} = 230 V_{AC})	P_{mot}	1.1 kW 1.5 HP	1.5 kW 2.0 HP	2.2 kW 3.0 HP
Recommended motor power under variable torque load or constant load without overload (with V_{mains} = 230 V_{AC})	P_{mot}	1.5 kW 2.0 HP	2.2 kW 3.0 HP	3.0 kW 4.0 HP
Rated output current at V_{mains} = 230 V_{AC}	I_N	5.7 A_{AC}	7.3 A_{AC}	8.6 A_{AC}
Minimum permitted braking resistor value (4-Q operation)	R_{BWmin}	27 Ω		

B.2 Qualisys Data Sheet

QUALISYS
Motion Capture Systems

DESIGNED AND PRODUCED BY QUALISYS
QUALISYS MOTION CAPTURE CAMERA
MADE IN SWEDEN
024
050
2

Capture anywhere

The platform of Qualisys premium camera series 5, 6 and 7 provides motion capture cameras suitable for all possible applications, both indoor and outdoor. They are designed to capture accurate motion capture data with very low latency and works with both passive and active markers. Select from a number of fields of view (FOV), resolutions and frame rates. Enjoy calibrated high-speed video and full-FOV mocap frame rates up to 1660 fps.

RELIABLE AND VERSATILE

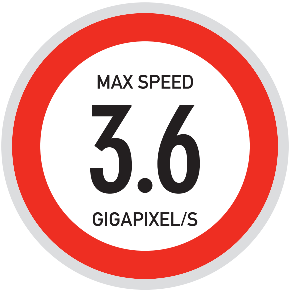
Our premium camera model is the most versatile motion capture camera in the world. Not only does it perform well indoor and outdoor, but with Qualisys underwater housing it also runs underwater. The IP67 classified housing makes it great for use in marine facilities or under rainy conditions. It can even be used next to an MRI scanner using the shielded MRI housing.

The cameras are also easily synchronized with external systems such as EMG and force plates. The entire system can be controlled by a single laptop or desktop PC.

FEATURES

- High-speed motion capture
- High-speed video[1]
- Resolution: 4, 6 & 12 MP
- Low latency for real-time applications
- Sun filter[1] and active filtering for outdoor captures
- Passive & active marker support
- Water resistant IP67 housing[1]
- Daisy-chaining (no switch required)
- Silent operation
- WiFi[1]

[1] Optional accessory/feature, not available for all camera models.



MRI shielded housing[1]

The camera is available with shielded housing that allows it to capture movement inside an MRI scanner.

Outdoor mocap

Active filtering dramatically increases the ability to capture passive markers outdoors.

Water resistant housing[1]

The IP67 classified version is perfect for marine or outdoor applications.

HIGH-SPEED SENSOR MODE

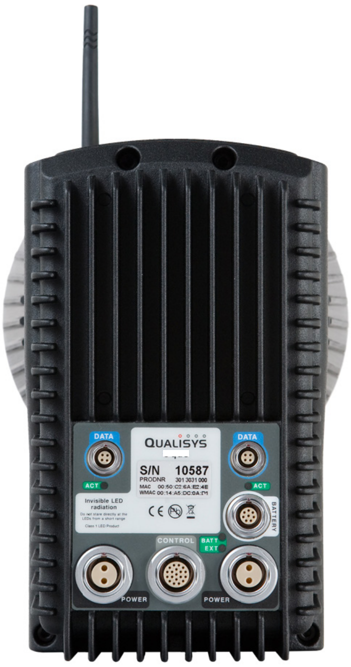
High-speed sensor mode[2] is a sub-sampling mode that gives increased frame rate without sacrificing field-of-view. For example, the 6+ is able to capture at 1660 fps, with full FOV and a sensor resolution of 1536 × 992. Even more impressive is the 7+, which is able to capture at 1100 fps with full FOV and a resolution of 3 megapixels. It is also possible to measure up to 10,000 fps with reduced field of view.

SPORTS, GAIT AND OTHER ANALYSIS MODULES

With the cameras, you can take the full advantage of the analysis modules for QTM, our motion capture software. Select from a wide range of modules, including ones for gait, golf, baseball and running. The modules are designed for clinical and sport environments, where speed and ease of use is of great importance. The modules are complete with pre-defined marker sets, report templates and documentation.

REAL-TIME SDK & MATLAB

With ths camera you can take full advantage of the real-time SDK of QTM, our data acquisition software. It allows any number of 3D coordinates or 6DOF transformations to be streamed in real-time to the host computer. The data is visualized in the software and can be accessible either in MATLAB, LabVIEW or any other software using custom-written scripts and applications that implements Qualisys' real-time protocol.



Input/output.
The power, data, sync in/out & trigger ports on the back of the camera.

Motorized lens
Control focus and aperture on 7+ cameras with motorized lenses. The motorized lens is controlled from QTM, our motion capture software.

Low latency
The platform is engineered for minimum latency with complete camera and full software pipeline latency down to 4 ms.

3D video overlay
With a calibrated video 210c reference, synchronized video overlay is a good way of fleshing out the bare bones of 3D motion capture data.

High-speed video[1]
Full-frame high-speed video capability is available as an option. Get frame rates up to 355 fps (full FOV) and up to 10,000 fps with reduced FOV.

External synchronization
We provides configurable sync in and out ports for syncing with external hardware including EMG, force plates and eye tracking devices among others.

Field of view
Get horizontal field of view of up to 70° with our wide angle lenses or choose from numerous lenses with various field of views

Daisy-chaining
The cameras are daisy-chained which means less cables and easier setup. Plus, small systems can be set up without switches.

Extreme pixel rate
Get 12 MP resolution captured at a frequency of 300 fps, or 3 MP resolution at 1100 fps (full FOV) with the 7+, our fastest camera yet.

Quick lens access
When a motorized lens is not the available option, easily adjust aperture and focus manually by rotating the strobe to gain lens access.

[1] Optional accessory/feature, not available for all camera models.

With the sun filter, 5+, 6+ and 7+ perform well even in extreme sunlight.

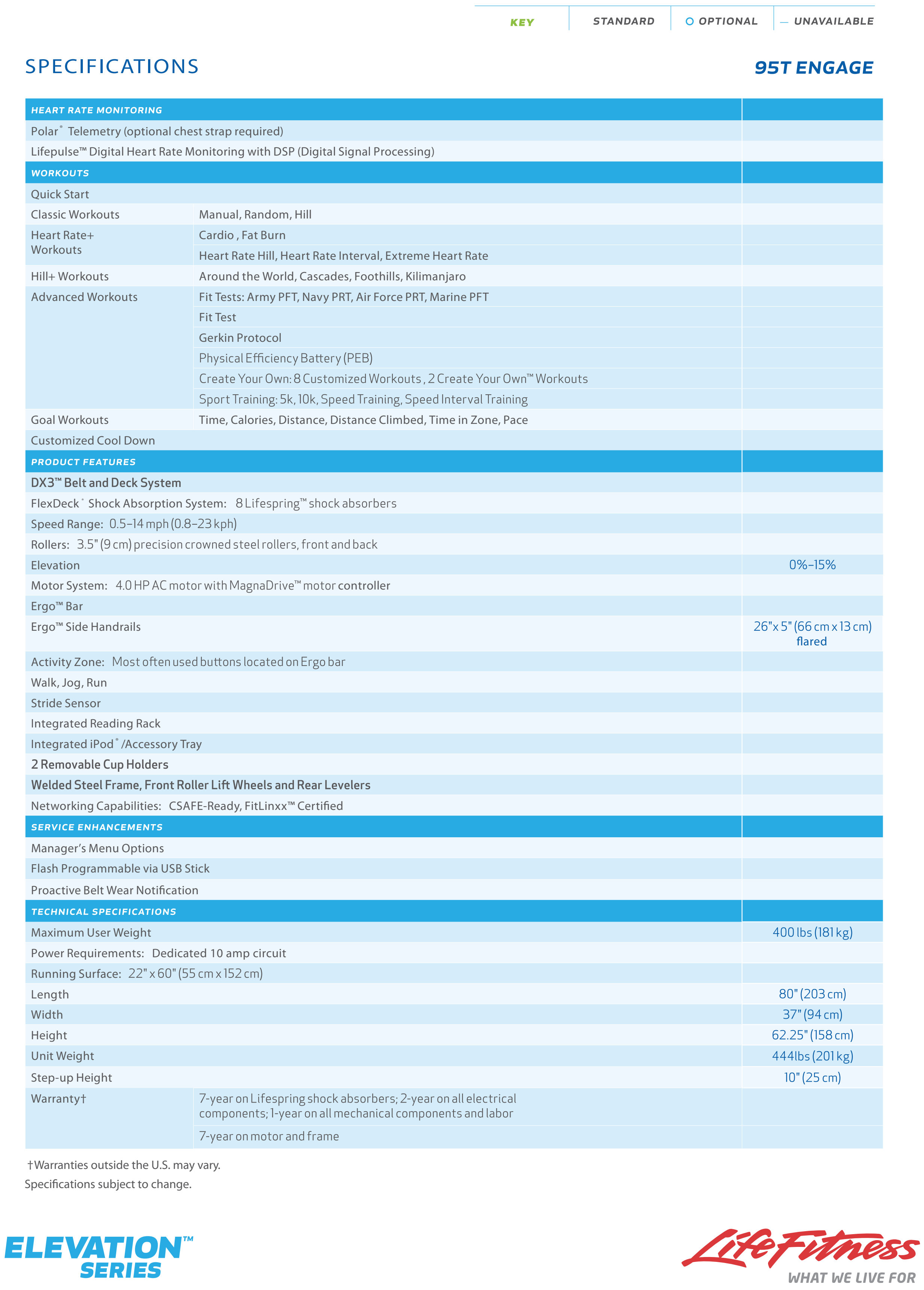
TECHNICAL SPECIFICATIONS

Camera output modes	Marker coordinates / high speed video[1]
Built-in camera display	128 x 64 graphical high contrast OLED
Marker support	Both passive and active
Camera body	Convection cooled, custom die-cast aluminum
Synchronization options	Internal 1 ppm clock/ext. freq. output/ext. word clock input/smpte input, PTPv2
Strobe	Invisible infrared light
Cabling	Hybrid cable with Ethernet and power
Wired communication	Hub-less daisy-chained Ethernet 802.3 @ 100Mbps
Wireless communication1	WiFi 802.11b/g @ 54Mbps
Position data noise level	+/- 1 sub-pixels
Maximum frame buffer size	1152 MB
Power	Daisy-chained, 36-72 VDC @ 30 W maximum
Operating temperature	5+ 0-35 °C, 6+ and 7+: 0-30 °C, contact us for wider temperature ranges
Camera body, 5+/6+	185 × 110 × 124 mm (7.3 × 4.3 × 4.9 inches) Weight: 1.9 kg (4.2 lbs)
Camera body, 7+	200 × 145 × 155 mm (7.9 × 5.7 × 6.1 inches) Weight: 2.1 kg (4.6 lbs)
Available camera housing	Standard / Water resistant IP67 / MRI / Underwater

		5+	**6+**	**7+**
Normal mode (full FOV)	**Pixels**	4 MP	6 MP	12 MP
	Resolution	2048 × 2048	3072 × 1984	4096 × 3072
	Frame rate	180 fps	450 fps	300 fps
High-speed mode (full FOV)	**Pixels**	1 MP	1.5 MP	3 MP
	Resolution	1024 × 1024	1536 × 992	2048 × 1536
	Frame rate	355 fps	1660 fps	1100 fps
Standard lens (hFOV)		49°	56°	54°
Lens options (hFOV)		25°, 49°, 70°	37°, 56°	31°, 54°, 70°
Max frame rate (reduced FOV)		10,000 fps	10,000 fps	10,000 fps
Measurement distances with 16 mm markers		25 m	28 m	35+
Active filtering (improved outdoor support)		Yes	Yes	Yes
Sun filter[1]		Yes	Yes	Yes
High-speed video option[1]		Yes	No	No
Motorized lens		No	No	Yes
Lens mount		C	C	SLR

[1] Optional accessory/feature, not available for all camera models.

B.3 Life Fitness 95T



KEY | STANDARD | ○ OPTIONAL | — UNAVAILABLE

SPECIFICATIONS

		95T ENGAGE
HEART RATE MONITORING		
Polar® Telemetry (optional chest strap required)		
Lifepulse™ Digital Heart Rate Monitoring with DSP (Digital Signal Processing)		
WORKOUTS		
Quick Start		
Classic Workouts	Manual, Random, Hill	
Heart Rate+ Workouts	Cardio , Fat Burn	
	Heart Rate Hill, Heart Rate Interval, Extreme Heart Rate	
Hill+ Workouts	Around the World, Cascades, Foothills, Kilimanjaro	
Advanced Workouts	Fit Tests: Army PFT, Navy PRT, Air Force PRT, Marine PFT	
	Fit Test	
	Gerkin Protocol	
	Physical Efficiency Battery (PEB)	
	Create Your Own: 8 Customized Workouts , 2 Create Your Own™ Workouts	
	Sport Training: 5k, 10k, Speed Training, Speed Interval Training	
Goal Workouts	Time, Calories, Distance, Distance Climbed, Time in Zone, Pace	
Customized Cool Down		
PRODUCT FEATURES		
DX3™ Belt and Deck System		
FlexDeck® Shock Absorption System: 8 Lifespring™ shock absorbers		
Speed Range: 0.5–14 mph (0.8–23 kph)		
Rollers: 3.5" (9 cm) precision crowned steel rollers, front and back		
Elevation		0%–15%
Motor System: 4.0 HP AC motor with MagnaDrive™ motor controller		
Ergo™ Bar		
Ergo™ Side Handrails		26"x 5" (66 cm x 13 cm) flared
Activity Zone: Most often used buttons located on Ergo bar		
Walk, Jog, Run		
Stride Sensor		
Integrated Reading Rack		
Integrated iPod® /Accessory Tray		
2 Removable Cup Holders		
Welded Steel Frame, Front Roller Lift Wheels and Rear Levelers		
Networking Capabilities: CSAFE-Ready, FitLinxx™ Certified		
SERVICE ENHANCEMENTS		
Manager's Menu Options		
Flash Programmable via USB Stick		
Proactive Belt Wear Notification		
TECHNICAL SPECIFICATIONS		
Maximum User Weight		400 lbs (181 kg)
Power Requirements: Dedicated 10 amp circuit		
Running Surface: 22" x 60" (55 cm x 152 cm)		
Length		80" (203 cm)
Width		37" (94 cm)
Height		62.25" (158 cm)
Unit Weight		444lbs (201 kg)
Step-up Height		10" (25 cm)
Warranty†	7-year on Lifespring shock absorbers; 2-year on all electrical components; 1-year on all mechanical components and labor	
	7-year on motor and frame	

†Warranties outside the U.S. may vary.

Specifications subject to change.

ELEVATION™ SERIES

LifeFitness WHAT WE LIVE FOR

KEY	● STANDARD	○ OPTIONAL	– UNAVAILABLE

DISPLAY TECHNOLOGY

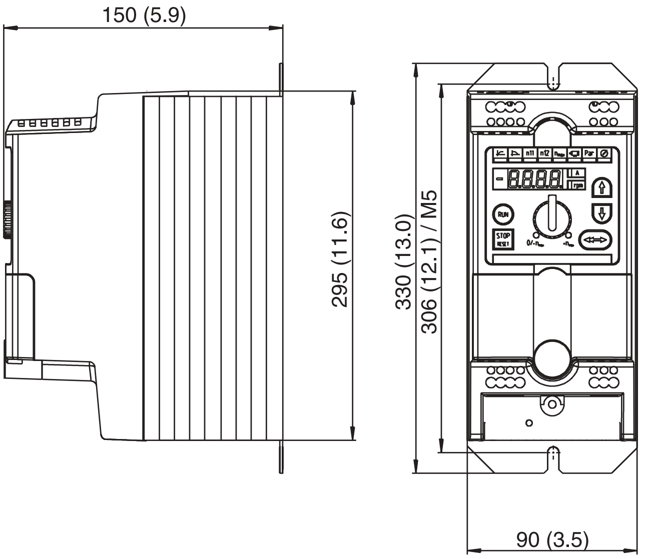
		95T ENGAGE
CONSOLE OPTIONS		
15" Integrated LCD Screen with Touch Screen Technology		●
VIDEO VIEWING OPTIONS		
Integrated 15" Diagonal Video Screen	Video content and workout data display	●
	Three different TV viewing sizes	●
DISPLAY READOUT		
Workout Feedback	Speed, Incline, Heart Rate, Pace	●
	Elapsed Time, Time Remaining, Time in Zone, Time of Day	●
	Distance, Distance Climbed, Distance Remaining	●
	Calories, Calories per Hour, Watts, METs	●
Custom Messaging		●
Profile Display		●
Language Choices:		13 languages
-English, Spanish, Portuguese, Italian, French, German, Dutch and Turkish		●
-Chinese (traditional and simplified), Japanese, Korean and Russian		●
SPECIAL FEATURES		
Intel® Microprocessor		●
FM Radio-ready		●
iPod and iPhone Compatibility	Video capability on LCD screen	●
	Playlist management on LCD screen	●
	Charging	●
Virtual Trainer		●
Workout Tracking: UBS connectivity and Virtual Trainer Web Site		●
Workout Landscape™ Perspectives		●
Programmable Go System™ (Walk-Jog-Run)		●
Manager-Defined Marathon Mode (unlimited workout time)		●
Manager-Defined/ User-Selected Languages	English, Spanish, Portuguese, Italian, French, German, Dutch, Russian, Chinese (traditional and simplified), Japanese, Turkish and Korean	●
User Units Selection (mph/kph and lbs/kg)		●
Zoom Feature		●
TV Controls: Touch Screen		●
Channel Memory	180 available channels	●
Favorite Channels		●
Previous Channel Viewed		●
Channel Renumbering		●
Secondary Audio Programs (SAP) - SAP TV broadcast required		●
Closed Captioning		●
Promo Channel		●
Secure Channel		●
Mute Feature		●
Asset Management and Advanced Diagnostics		●
Screen Protection: Protective top layer, internal shock mounts, gasket		●
TECHNICAL SPECIFICATIONS		
Available Tuner Systems: NTSC/ATSC/QAM, NTSC, PAL/SECAM		●
Channel Coverage: VHF 2-13, UHF 14-69, CATV 1-125		●
Manual Fine Tuning		●
Headphone Jack: 3.5 mm Stereo		●
Easy-to-Remove Headphone Jack		●

Specifications subject to change.

ELEVATION™ SERIES

Life Fitness
WHAT WE LIVE FOR

MOVITRAC® MC07A (1-phase supply system)		011-2B1-4-..	015-2B1-4-..	022-2B1-4-..
GENERAL				
Power loss at I_N	P_V	75 W	100 W	125 W
Current limitation		125 % I_N continuous duty (fan/pump operation) 150 % I_N for maximum 60 seconds		
PWM frequency	f_{PWM}	4 / 8 / 12 / 16 kHz		
Speed range Resolution	n_A Δn_A	0 ... 5500 rpm 1 rpm		
Connections		Terminals 4 mm^2		
Dimensions	W x H x D	90 x 295 x 150 mm 3.5 x 9.5 x 5.9 in		
Weight	m	2.5 kg 5.5 lb		



05805AXX

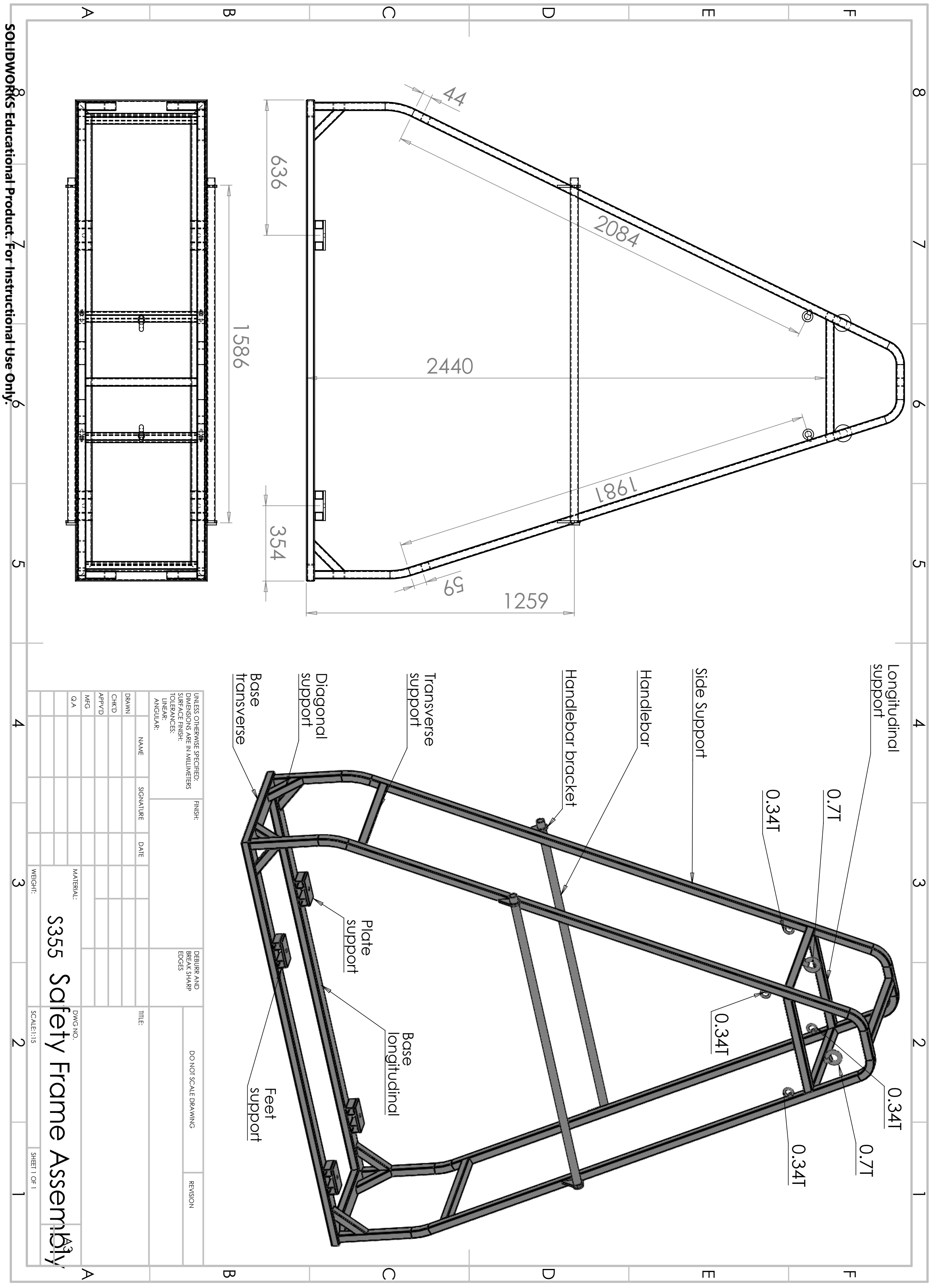
Figure 7: Dimensions, MOVITRAC® 07 size 0L

Provide 100 mm (4 in) clearance above and below the unit to ensure adequate cooling! There is no need for clearance at the sides. You can line up the units directly next to one another. Make sure that the circulation of air is not disrupted by cables or other installation materials. Prevent the heated exhaust air from other units from blowing onto this unit.

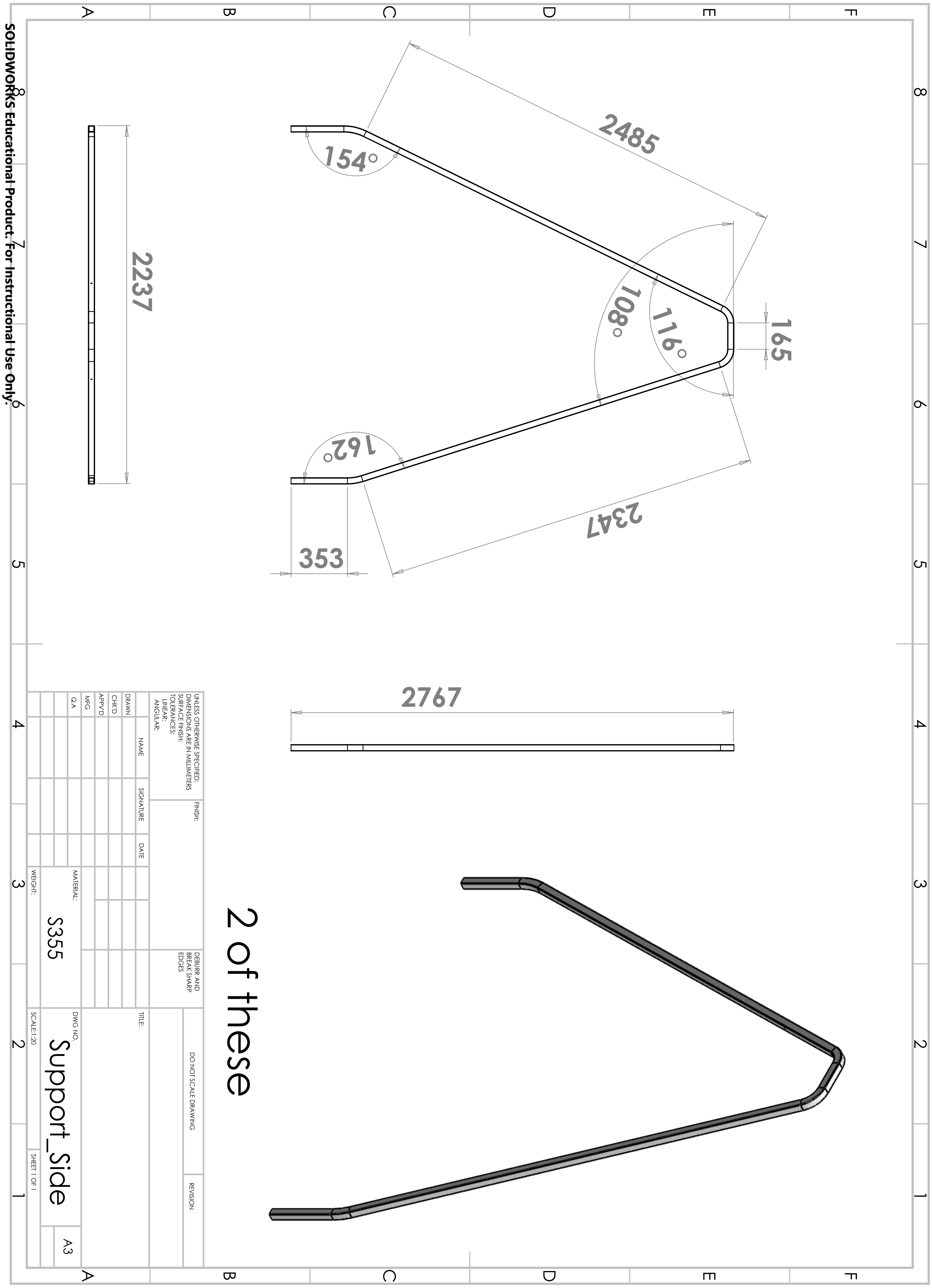
C. Blueprints

C.1 Safety Frame

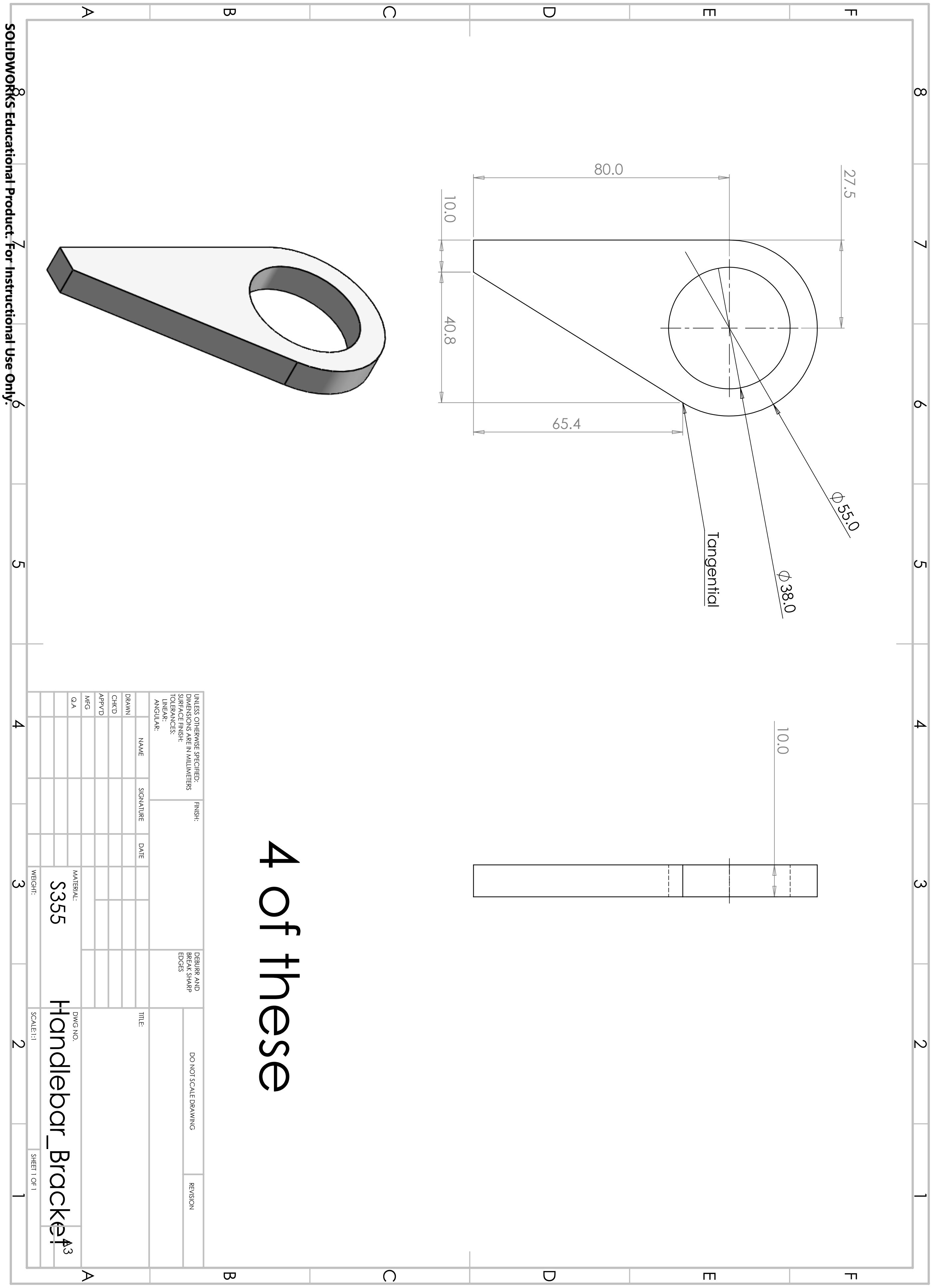
Assembly



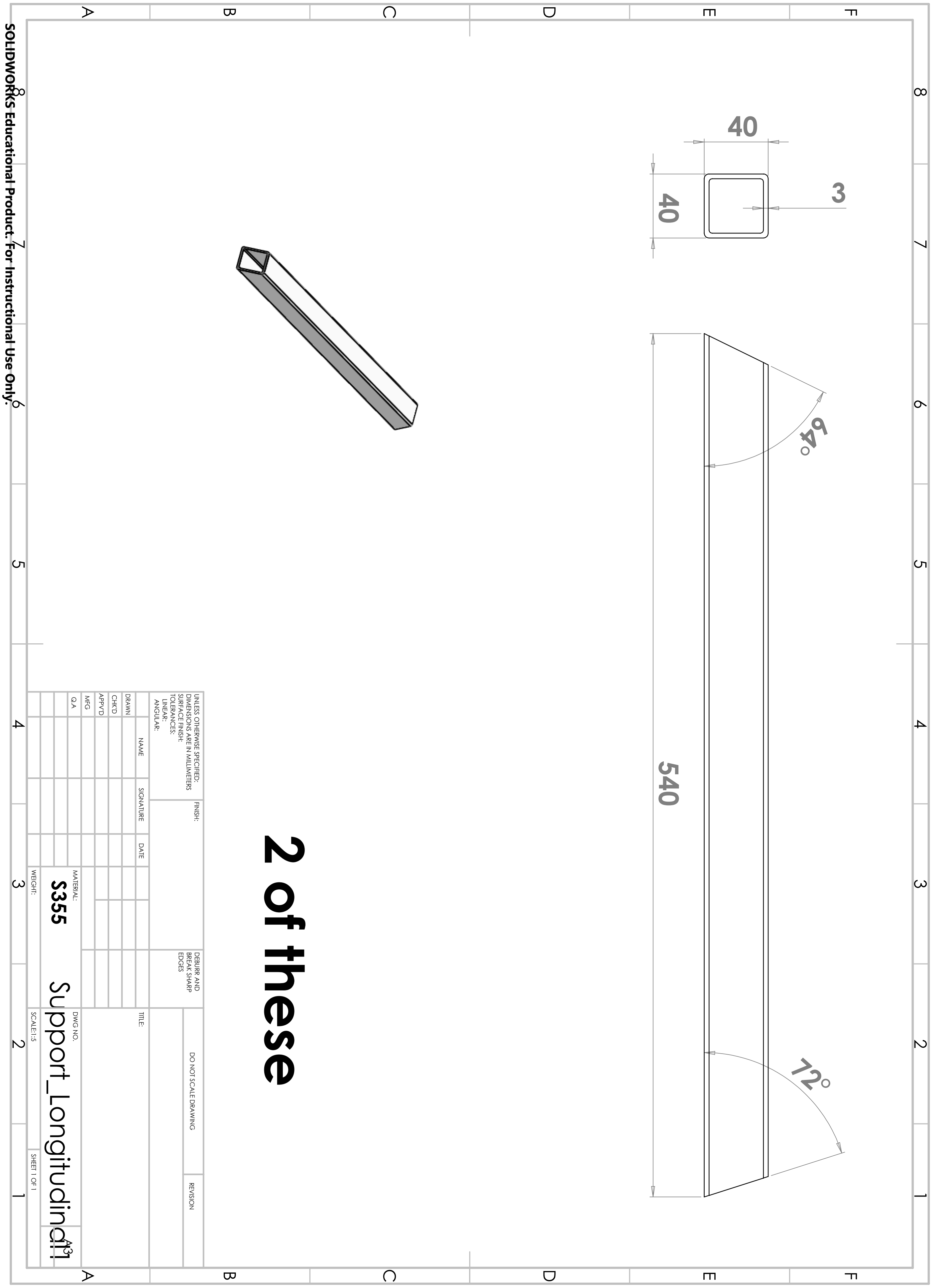
Side Supports



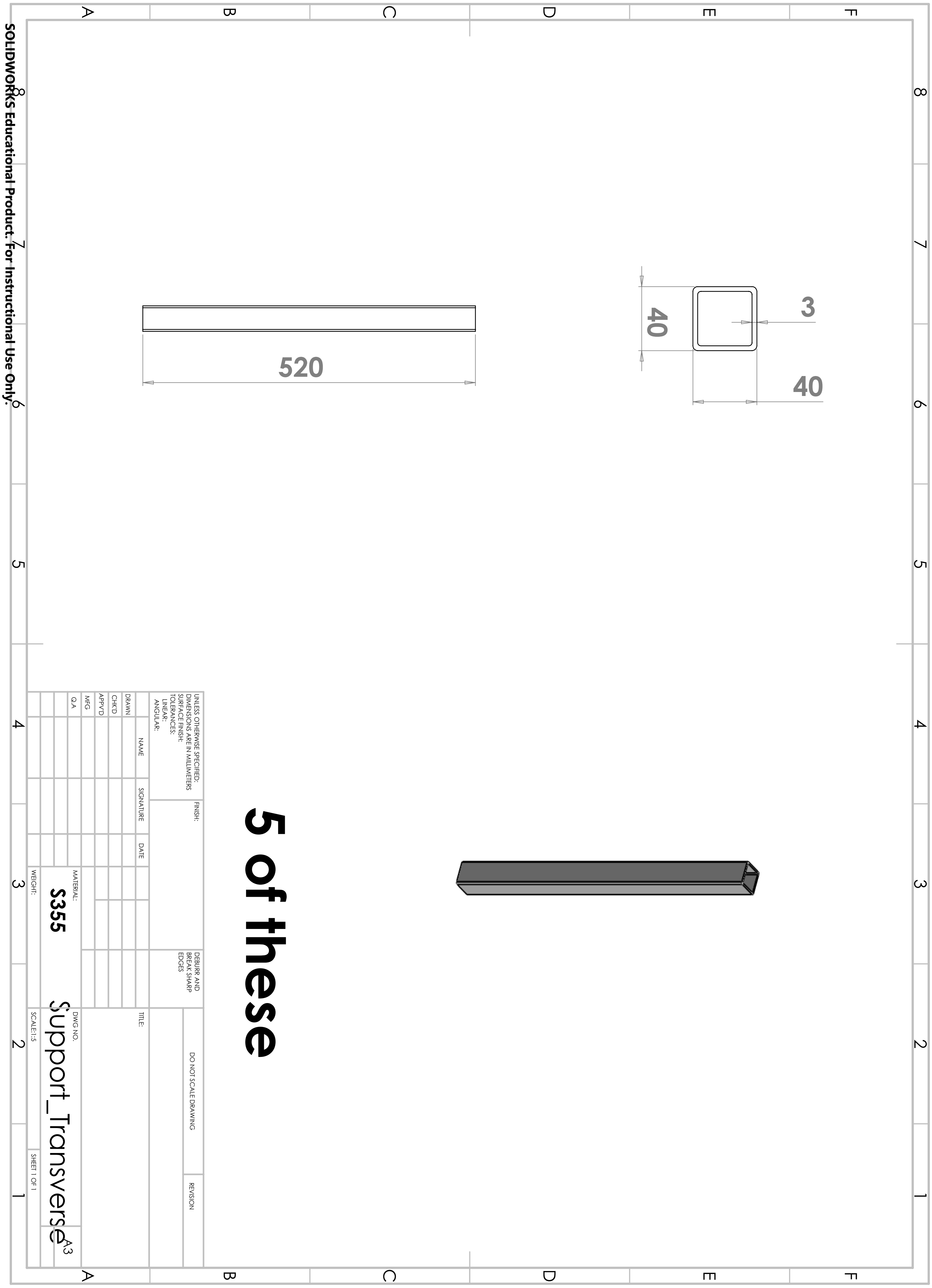
Handlebar Bracket



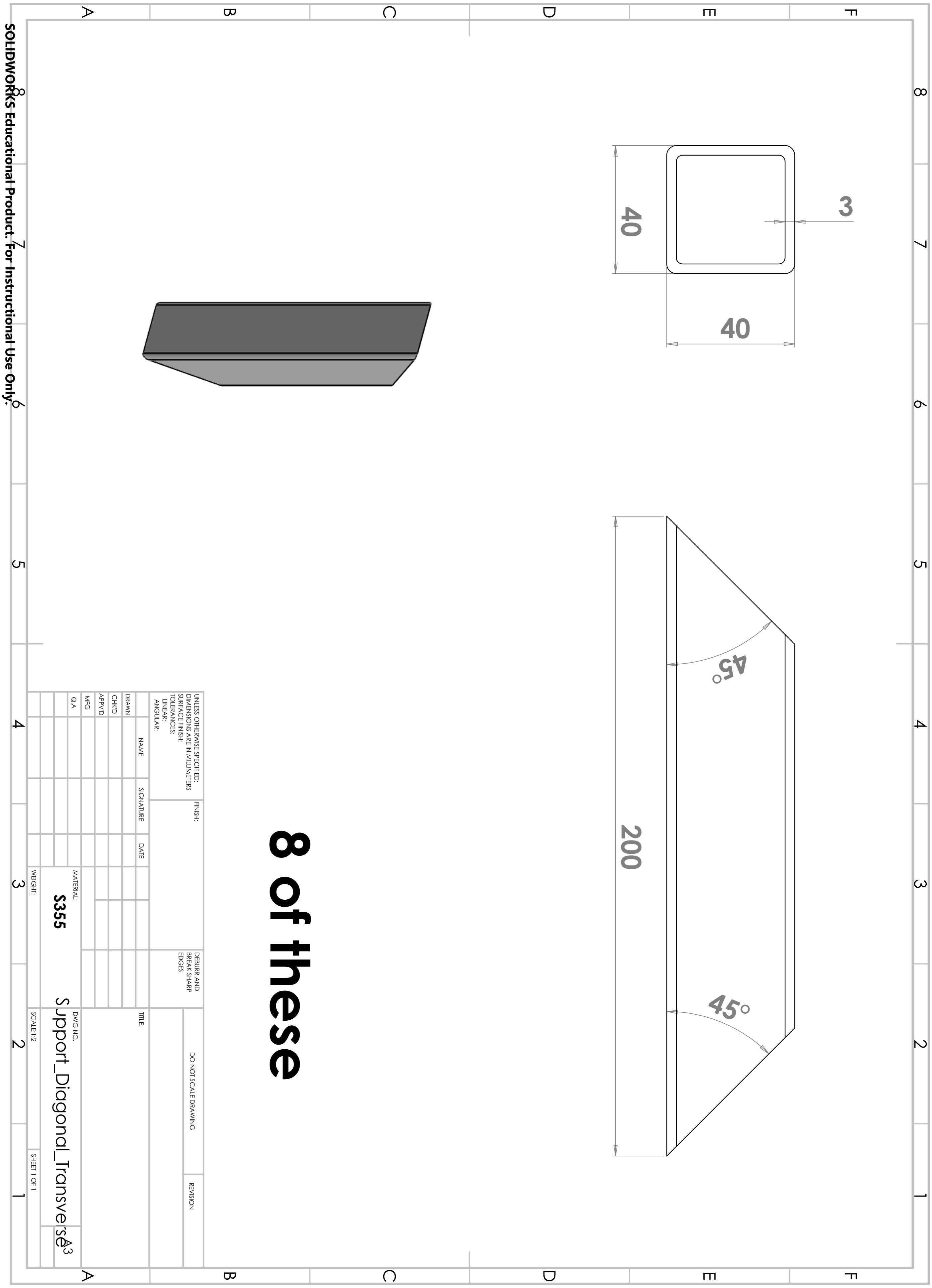
Longitudinal Supports



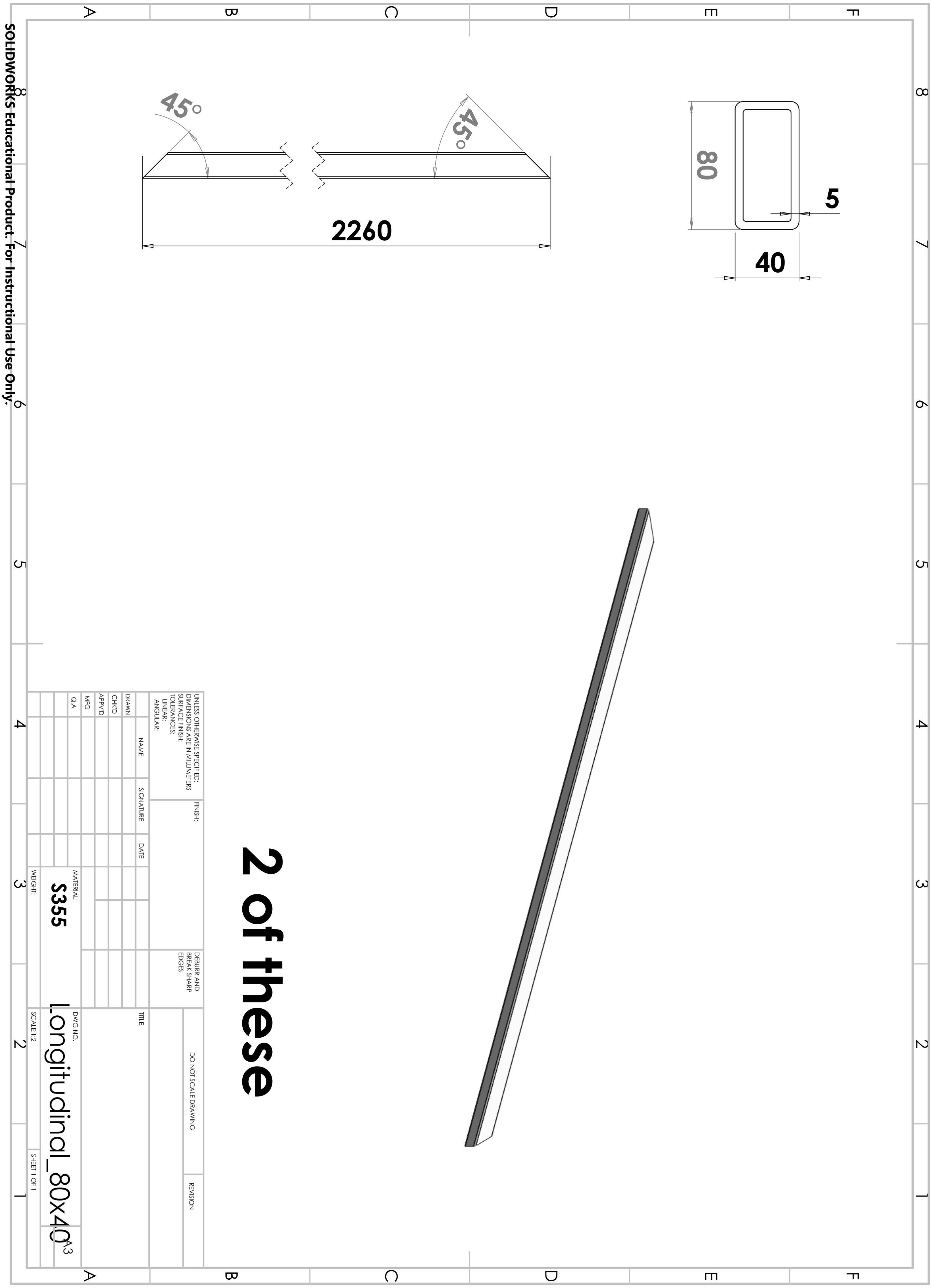
Transverse Supports



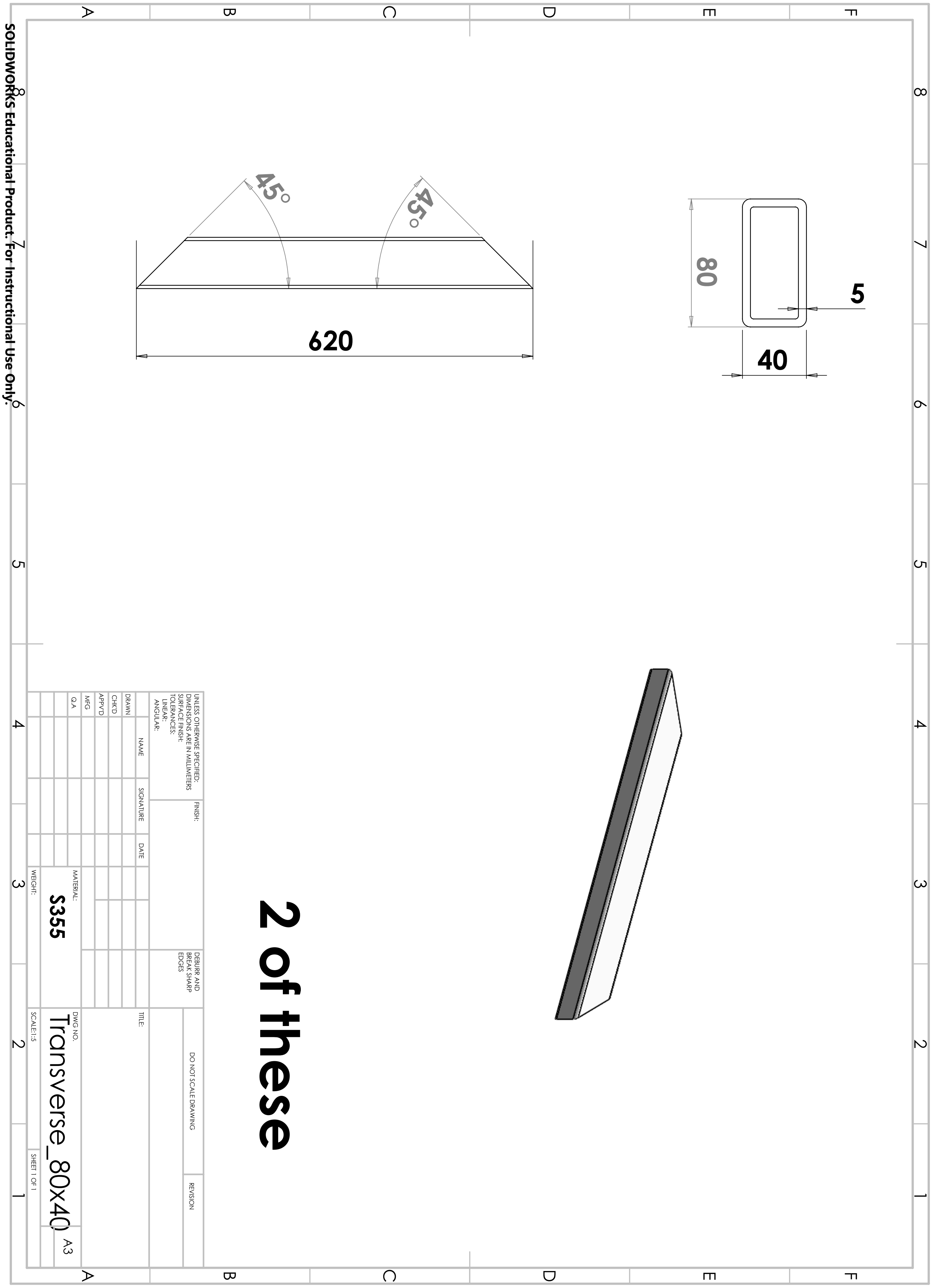
Diagonal Supports



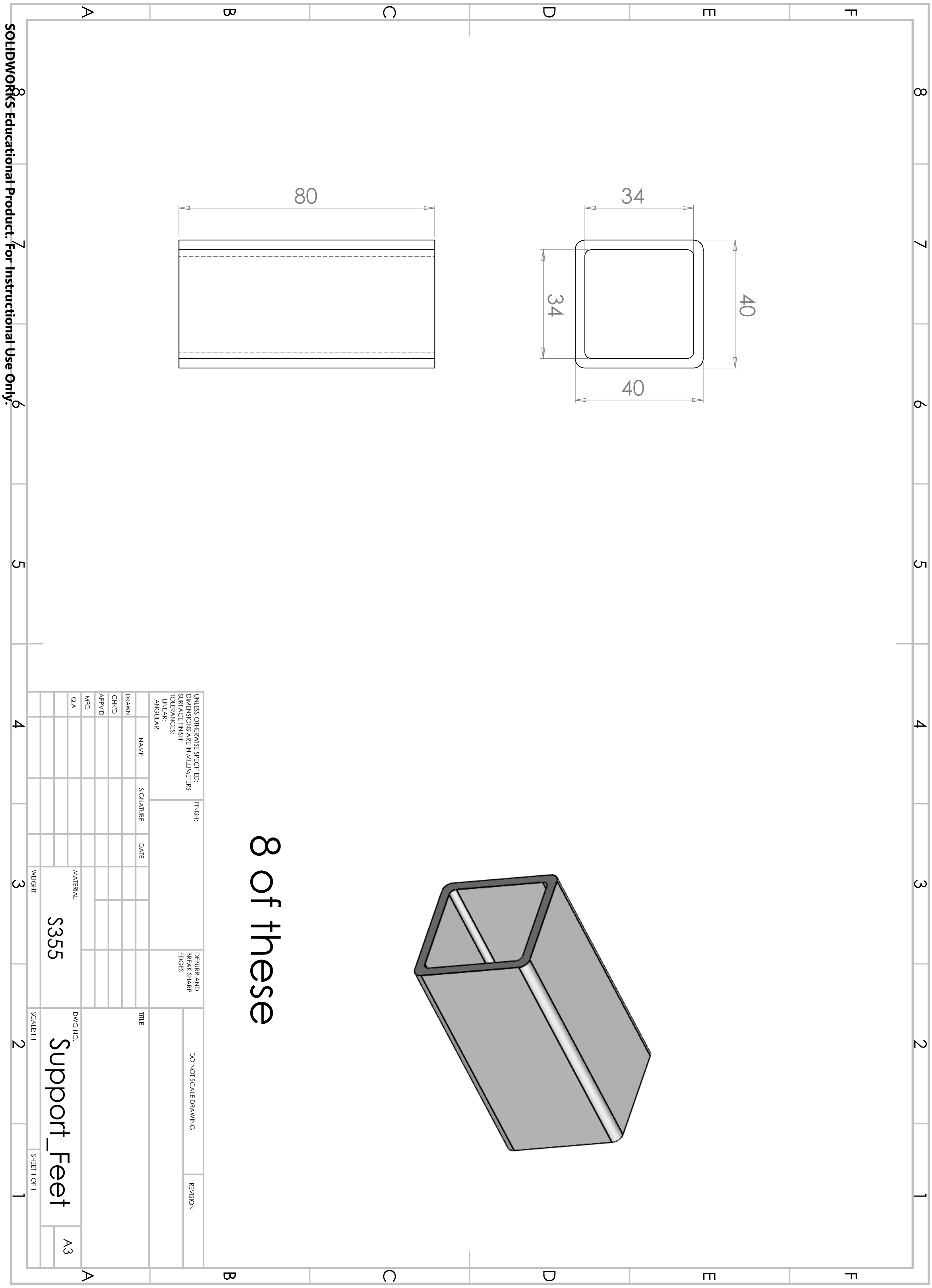
Base Longitudinal



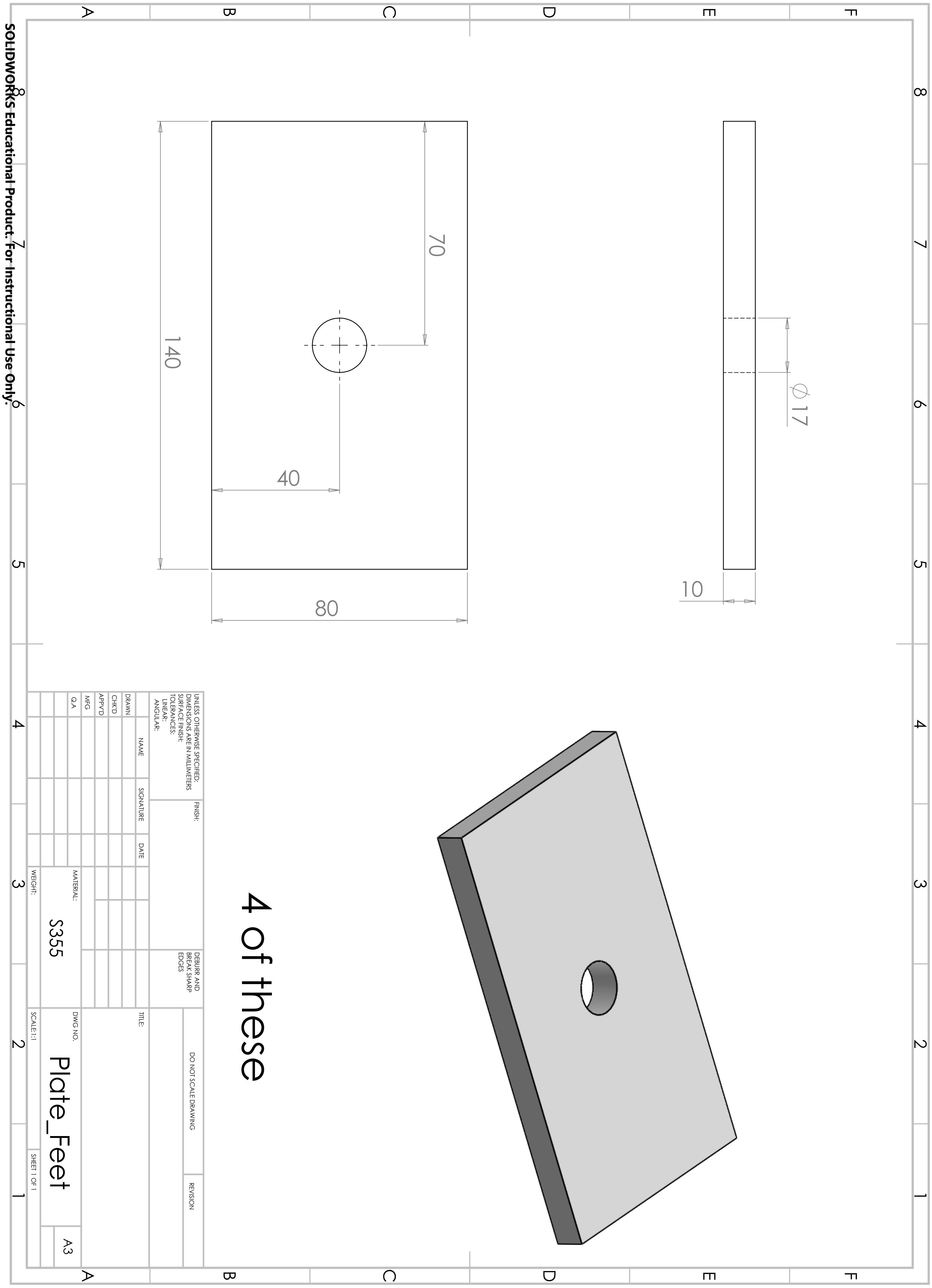
Base Transverse



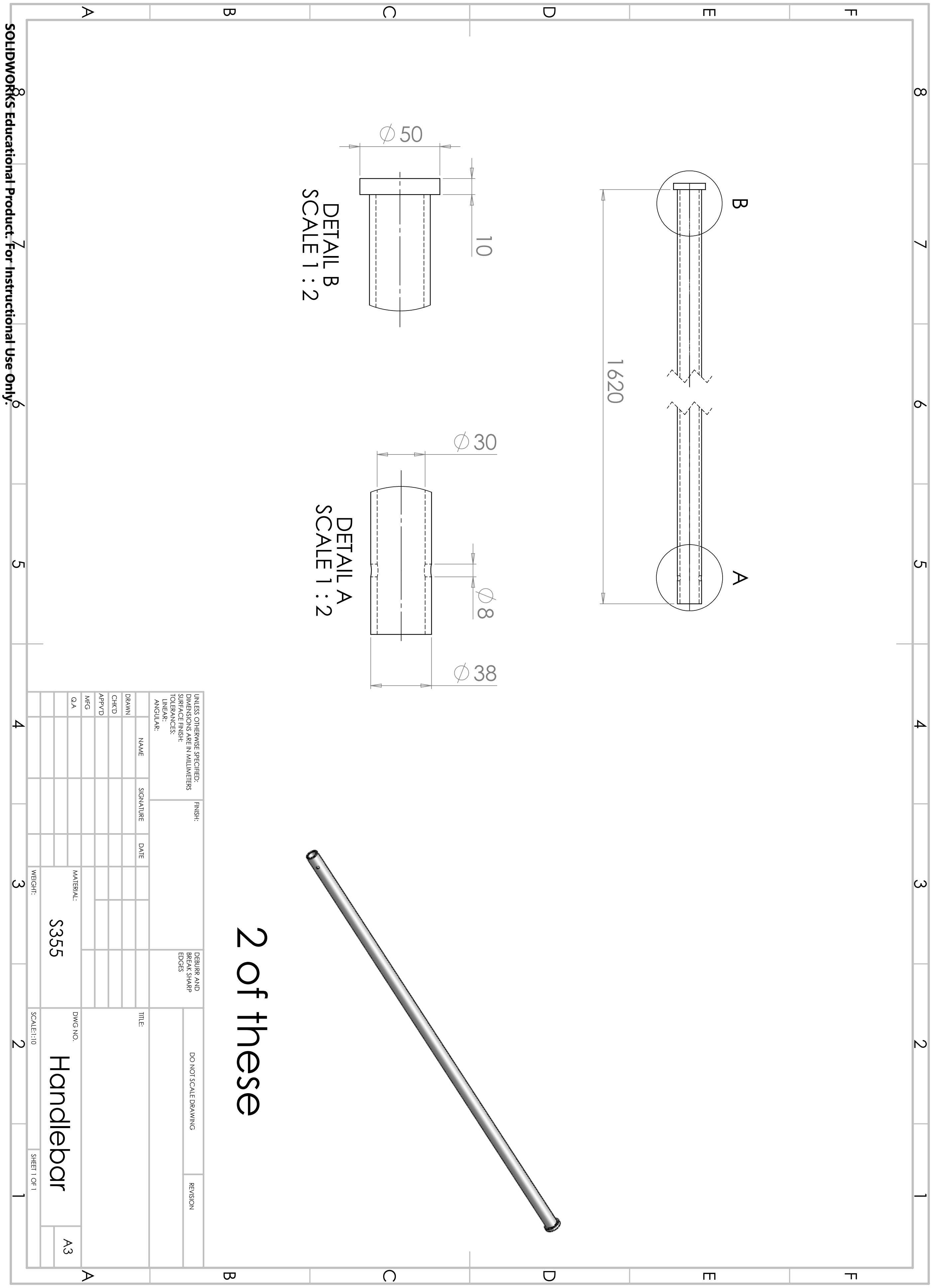
Feet Supports



Feet Plate


SOLIDWORKS Educational Product. For Instructional Use Only.
140
70
40
80
⌀17
10
4 of these
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH:
TOLERANCES:
LINEAR:
ANGULAR:
FINISH:
DEBURR AND BREAK SHARP EDGES
DO NOT SCALE DRAWING
REVISION
NAME
SIGNATURE
DATE
DRAWN
CHK'D
APPV'D
MFG
Q.A
MATERIAL:
S355
WEIGHT:
TITLE:
DWG NO.
Plate_Feet
A3
SCALE:1:1
SHEET 1 OF 1
A
B
C
D
E
F
1
2
3
4
5
6
7
8

Handlebars



SOLIDWORKS Educational Product. For Instructional Use Only.
∅50
10
DETAIL B
SCALE 1 : 2
∅30
∅8
∅38
DETAIL A
SCALE 1 : 2
1620
A
B
2 of these
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH:
TOLERANCES:
LINEAR:
ANGULAR:
FINISH:
DEBURR AND BREAK SHARP EDGES
DO NOT SCALE DRAWING
REVISION
NAME
SIGNATURE
DATE
DRAWN
CHK'D
APPV'D
MFG
Q.A
TITLE:
MATERIAL:
S355
DWG NO.
Handlebar
A3
WEIGHT:
SCALE:1:10
SHEET 1 OF 1

D. Math

D.1 Transformation Matrices

1. Transformation Matrices

$$\mathbf{TR_{X,p,b}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(R_{X,N,p}) & -sin(R_{X,N,p}) \\ 0 & sin(R_{X,N,p}) & cos(R_{X,N,p}) \end{bmatrix}$$

$$\mathbf{TR_{Y,p,b}} = \begin{bmatrix} cos(R_{Y,N,p}) & 0 & sin(R_{Y,N,p}) \\ 0 & 1 & 0 \\ -sin(R_{Y,N,p}) & 0 & cos(R_{Y,N,p}) \end{bmatrix}$$

Simplifying terms to shorten further expressions:

$$\begin{aligned} cos() &= c() \\ sin() &= s() \\ R_{X,N,p} &= R_X \\ R_{Y,N,p} &= R_Y \\ \mathbf{TR_{XY,p,b}} &= \begin{bmatrix} c(R_Y) & 0 & s(R_Y) \\ s(R_X)s(R_Y) & c(R_X) & -s(R_X)c(R_Y) \\ -c(R_X)s(R_Y) & s(R_X) & c(R_X)c(R_Y) \end{bmatrix} \end{aligned}$$

$$\dot{\mathbf{TR}}_{\mathbf{XY,p,b}} = \begin{bmatrix} -\dot{R}_Y s(R_Y) & 0 & \dot{R}_Y c(R_Y) \\ \dot{R}_X c(R_X)s(R_Y) + \dot{R}_Y s(R_X)c(R_Y) & -\dot{R}_X s(R_X) & -\dot{R}_X c(R_X)c(R_Y) + \dot{R}_Y s(R_X)s(R_Y) \\ \dot{R}_X s(R_X)s(R_Y) - \dot{R}_Y c(R_X)c(R_Y) & \dot{R}_X c(R_X) & -\dot{R}_X s(R_X)c(R_Y) - \dot{R}_Y c(R_X)s(R_Y) \end{bmatrix}$$

E. Matlab Scripts and Functions

E.1 Simulation of Haptic Models

```
close all;
clear;
clc;

% This script simulates either general board model or
   skateboard
% model depending on what's chosen.
% Model parameters and time-dependant input signals must be
   specified.

% General Board model   -> ModelType = 0;
% Skateboard model      -> ModelType = 1;

ModelType = 0;

%% Parameters
switch ModelType
    case 0
L_B = 1.5;      % Length of board
W_B = 0.4;      % Width of board
m_p = 75;       % Mass of person
g = 9.81;       % Gravitational acceleration
k = 4000;        % Stiffness of ground material
d = 1000;        % Damping of ground material

m = m_p/4;

    case 1
L_B = 0.8;      % Length of board
W_B = 0.25;     % Width of board
m_p = 75;       % Mass of person
g = 9.81;       % Gravitational acceleration
k = 4000;        % Stiffness of trucker bushing
d = 1000;        % Damping of trucker bushing

m = m_p;
k = k*2;
d = d*2;
    otherwise
disp("Not a valid ModelType")
end

% Define time vector for simulation
```

```
N = 1000;
tEnd = 13;
t = linspace(0,tEnd,N);

%% Developing state-space model of mass-spring-damper systems
% Here it is possible to set different stiffnesses and damping
   for each
% system.

A = [0 1; -k/m -d/m];
B = [0 0 0; 1/m k/m d/m];
C = [1 0; 0 1];
D = [0 0 0; 0 0 0];
%u = [Fpg; zsDot; zs];

NorthModel = ss(A,B,C,D);
SouthModel = ss(A,B,C,D);
EastModel = ss(A,B,C,D);
WestModel = ss(A,B,C,D);

%% Input signals
omega = 1;

% Force plate
F_v(1,1:N) = 0.6+0.4*sin(1.5*t);
F_p = m_p*F_v*g;
x_bFp(1,1:N) = L_B/2*sin(t);
y_bFp(1,1:N) = W_B/2*cos(t);

% Simulation
T_ZNsim(1,1:N) = 1;%0.2*cos(omega*t);
R_YNsim(1,1:N) = 0;%0.2*sin(omega*t);
R_XNsim(1,1:N) = 0;%0.3*cos(omega*t);
T_ZNsimDot(1,1:N) = 0;%-0.2*omega*sin(omega*t);
R_YNsimDot(1,1:N) = 0;%0.3*omega*cos(omega*t);
R_XNsimDot(1,1:N) = 0;%-0.3*omega*sin(omega*t);


%% Splitting signals into signals suitable for each model

% Input positions
z_simn = T_ZNsim - L_B/2*sin(R_YNsim);
z_sims = T_ZNsim + L_B/2*sin(R_YNsim);
z_sime = T_ZNsim + W_B/2*sin(R_XNsim);
z_simw = T_ZNsim - W_B/2*sin(R_XNsim);

% Input velocities
z_simnDot(1,1:N) = T_ZNsimDot - R_YNsimDot*L_B/2.*cos(R_YNsim);
z_simsDot(1,1:N) = T_ZNsimDot + R_YNsimDot*L_B/2.*cos(R_YNsim);
z_simeDot(1,1:N) = T_ZNsimDot + R_XNsimDot*W_B/2.*cos(R_XNsim);
z_simwDot(1,1:N) = T_ZNsimDot - R_XNsimDot*W_B/2.*cos(R_XNsim);
```

```
switch ModelType
    case 0
% Input forces
F_pgn(1,1:N) = (1/2+x_bFp/L_B).*F_p;
F_pgs(1,1:N) = (1/2-x_bFp/L_B).*F_p;
F_pge(1,1:N) = (1/2+y_bFp/W_B).*F_p;
F_pgw(1,1:N) = (1/2-y_bFp/W_B).*F_p;
    case 1
F_pgn(1,1:N) = (1/2+x_bFp/L_B).*F_p;
F_pgs(1,1:N) = (1/2-x_bFp/L_B).*F_p;
F_pge(1,1:N) = (1/2+y_bFp/W_B).*F_p;

F_pgw(1,1:N) = -y_bFp/W_B.*F_p;
end
%% Model simulation
% Initial conditions
initNorth = [0; 0];
initSouth = [0; 0];
initEast = [0; 0];
initWest = [0; 0];

% Input vectors
uNorth = [F_pgn;z_simn;z_simnDot];
uSouth = [F_pgs;z_sims;z_simsDot];
uEast = [F_pge;z_sime;z_simeDot];
uWest = [F_pgw;z_simw;z_simwDot];


[yNorth,t,xNorth] = lsim(NorthModel,uNorth,t,initNorth);
[ySouth,t,xSouth] = lsim(SouthModel,uSouth,t,initSouth);
[yEast,t,xEast] = lsim(EastModel,uEast,t,initEast);
[yWest,t,xWest] = lsim(WestModel,uWest,t,initWest);


% Position outputs
z_bn = yNorth(:,1)';
z_bs = ySouth(:,1)';
z_be = yEast(:,1)';
z_bw = yWest(:,1)';



ax = axes();
axis equal
xlim(ax, [-L_B/2 L_B/2])
ylim(ax, [-W_B/2, W_B/2])
zlim(ax, [-1.5,1.5])
xlabel('x')
ylabel('y')
zlabel('z')
```

```
grid on
view(ax, 3)
hold(ax, 'on')

switch ModelType
    case 0

% Force vector illustration
% Location in z
z_F1(1,1:N) = (z_bn+z_bs)/2 + x_bFp.*(z_bn-z_bs)/L_B...
      + y_bFp.*(z_be-z_bw)/W_B;
z_F2(1,1:N) = z_F1-F_v;

% Reducing processing required in the for-loop:
SNRightSimZ1 = -z_sims-(z_simw-z_sime)/2;
SNRightSimZ2 = -z_simn-(z_simw-z_sime)/2;
SNLeftSimZ1 = -z_sims+(z_simw-z_sime)/2;
SNLeftSimZ2 = -z_simn+(z_simw-z_sime)/2;
SNRightbZ1 = -z_bs-(z_bw-z_be)/2;
SNRightbZ2 = -z_bn-(z_bw-z_be)/2;
SNLeftbZ1 = -z_bs+(z_bw-z_be)/2;
SNLeftbZ2 = -z_bn+(z_bw-z_be)/2;
WERightSimZ1 = -z_simw-(z_sims-z_simn)/2;
WERightSimZ2 = -z_sime-(z_sims-z_simn)/2;
WELeftSimZ1 = -z_simw+(z_sims-z_simn)/2;
WELeftSimZ2 = -z_sime+(z_sims-z_simn)/2;
WERightbZ1 = -z_bw-(z_bs-z_bn)/2;
WERightbZ2 = -z_be-(z_bs-z_bn)/2;
WELeftbZ1 = -z_bw+(z_bs-z_bn)/2;
WELeftbZ2 = -z_be+(z_bs-z_bn)/2;

% Initialize plot
plotSN = plot3([-L_B/2 L_B/2], [0,0], [-z_sims(1) -z_simn(1)],'
   k',...
    [-L_B/2 L_B/2], [0,0], [-z_bs(1),-z_bn(1)],'b',...
    [-L_B/2 L_B/2], [-W_B/2,-W_B/2], [SNRightSimZ1(1)
       SNRightSimZ2(1)],'k',...
    [-L_B/2 L_B/2], [W_B/2,W_B/2], [SNLeftSimZ1(1) SNLeftSimZ2
       (1)],'k',...
    [-L_B/2 L_B/2], [-W_B/2,-W_B/2], [SNRightbZ1(1) SNRightbZ2
       (1)],'b',...
    [-L_B/2 L_B/2], [W_B/2,W_B/2], [SNLeftbZ1(1) SNLeftbZ2(1)],
       'b',...
    [0 0], [-W_B/2 W_B/2],[-z_simw(1) -z_sime(1)],'k',...
    [0 0], [-W_B/2 W_B/2],[-z_bw(1) -z_be(1)],'b',...
    [-L_B/2 -L_B/2], [-W_B/2 W_B/2],[WERightSimZ1(1)
       WERightSimZ2(1)],'k',...
    [L_B/2 L_B/2], [-W_B/2 W_B/2],[WELeftSimZ1(1) WELeftSimZ2
       (1)],'k',...
    [-L_B/2 -L_B/2], [-W_B/2 W_B/2],[WERightbZ1(1) WERightbZ2
       (1)],'b',...
    [L_B/2 L_B/2], [-W_B/2 W_B/2],[WELeftbZ1(1) WELeftbZ2(1)],'
       b');
pause(1)
for i = 1:N
delete(plotSN)

plotSN = plot3([-L_B/2 L_B/2], [0,0], [-z_sims(i) -z_simn(i)],'
   k',...
    [-L_B/2 L_B/2], [0,0], [-z_bs(i),-z_bn(i)],'b',...
    [-L_B/2 L_B/2], [-W_B/2,-W_B/2], [SNRightSimZ1(i)
       SNRightSimZ2(i)],'k',...
    [-L_B/2 L_B/2], [W_B/2,W_B/2], [SNLeftSimZ1(i) SNLeftSimZ2(
       i)],'k',...
    [-L_B/2 L_B/2], [-W_B/2,-W_B/2], [SNRightbZ1(i) SNRightbZ2(
       i)],'b',...
    [-L_B/2 L_B/2], [W_B/2,W_B/2], [SNLeftbZ1(i) SNLeftbZ2(i)],
       'b',...
    [x_bFp(i), x_bFp(i)], [y_bFp(i), y_bFp(i)], [-z_F1(i),-z_F2
       (i)], 'r',...
    [0 0], [-W_B/2 W_B/2],[-z_simw(i) -z_sime(i)],'k',...
    [0 0], [-W_B/2 W_B/2],[-z_bw(i) -z_be(i)],'b',...
    [-L_B/2 -L_B/2], [-W_B/2 W_B/2],[WERightSimZ1(i)
       WERightSimZ2(i)],'k',...
    [L_B/2 L_B/2], [-W_B/2 W_B/2],[WELeftSimZ1(i) WELeftSimZ2(i
       )],'k',...
    [-L_B/2 -L_B/2], [-W_B/2 W_B/2],[WERightbZ1(i) WERightbZ2(i
       )],'b',...
    [L_B/2 L_B/2], [-W_B/2 W_B/2],[WELeftbZ1(i) WELeftbZ2(i)],'
       b');


pause(0.001)
end

    case 1

% Force vector illustration
% Location in z
z_F1(1,1:N) = T_ZNsim + x_bFp.*(z_simn-z_sims)/L_B...
      + y_bFp.*(T_ZNsim-z_bw)/W_B;
z_F2(1,1:N) = z_F1-F_v;

% Reducing processing required in the for-loop:
SNRightSimZ1 = -z_sims-(z_simw-z_sime)/2;
SNRightSimZ2 = -z_simn-(z_simw-z_sime)/2;
SNLeftSimZ1 = -z_sims+(z_simw-z_sime)/2;
SNLeftSimZ2 = -z_simn+(z_simw-z_sime)/2;
SNRightbZ1 = -z_sims-(z_bw-T_ZNsim);
SNRightbZ2 = -z_simn-(z_bw-T_ZNsim);
SNLeftbZ1 = -z_sims+(z_bw-T_ZNsim);
SNLeftbZ2 = -z_simn+(z_bw-T_ZNsim);
```

```
WERightSimZ1 = -z_simw-(z_sims-z_simn)/2;
WERightSimZ2 = -z_sime-(z_sims-z_simn)/2;
WELeftSimZ1 = -z_simw+(z_sims-z_simn)/2;
WELeftSimZ2 = -z_sime+(z_sims-z_simn)/2;
WERightbZ1 = -z_bw-(z_sims-z_simn)/2;
WERightbZ2 = -(2*T_ZNsim-z_bw)-(z_sims-z_simn)/2;
WELeftbZ1 = -z_bw+(z_sims-z_simn)/2;
WELeftbZ2 = -(2*T_ZNsim-z_bw)+(z_sims-z_simn)/2;

% Initialize plot
plotSN = plot3([-L_B/2 L_B/2], [0,0], [-z_sims(1) -z_simn(1)],'
   k',...
    [-L_B/2 L_B/2], [0,0], [-z_sims(1),-z_simn(1)],'b',...
    [-L_B/2 L_B/2], [-W_B/2,-W_B/2], [SNRightSimZ1(1)
       SNRightSimZ2(1)],'k',...
    [-L_B/2 L_B/2], [W_B/2,W_B/2], [SNLeftSimZ1(1) SNLeftSimZ2
       (1)],'k',...
    [-L_B/2 L_B/2], [-W_B/2,-W_B/2], [SNRightbZ1(1) SNRightbZ2
       (1)],'b',...
    [-L_B/2 L_B/2], [W_B/2,W_B/2], [SNLeftbZ1(1) SNLeftbZ2(1)],
       'b',...
    [0 0], [-W_B/2 W_B/2],[-z_simw(1) -z_sime(1)],'k',...
    [0 0], [-W_B/2 W_B/2],[-z_bw(1) -(2*T_ZNsim(1)-z_bw(1))],'b
       ',...
    [-L_B/2 -L_B/2], [-W_B/2 W_B/2],[WERightSimZ1(1)
       WERightSimZ2(1)],'k',...
    [L_B/2 L_B/2], [-W_B/2 W_B/2],[WELeftSimZ1(1) WELeftSimZ2
       (1)],'k',...
    [-L_B/2 -L_B/2], [-W_B/2 W_B/2],[WERightbZ1(1) WERightbZ2
       (1)],'b',...
    [L_B/2 L_B/2], [-W_B/2 W_B/2],[WELeftbZ1(1) WELeftbZ2(1)],'
       b');
pause(1)
for i = 1:N
delete(plotSN)

plotSN = plot3([-L_B/2 L_B/2], [0,0], [-z_sims(i) -z_simn(i)],'
   k',...
    [-L_B/2 L_B/2], [0,0], [-z_sims(i),-z_simn(i)],'b',...
    [-L_B/2 L_B/2], [-W_B/2,-W_B/2], [SNRightSimZ1(i)
       SNRightSimZ2(i)],'k',...
    [-L_B/2 L_B/2], [W_B/2,W_B/2], [SNLeftSimZ1(i) SNLeftSimZ2(
       i)],'k',...
    [-L_B/2 L_B/2], [-W_B/2,-W_B/2], [SNRightbZ1(i) SNRightbZ2(
       i)],'b',...
    [-L_B/2 L_B/2], [W_B/2,W_B/2], [SNLeftbZ1(i) SNLeftbZ2(i)],
       'b',...
    [x_bFp(i), x_bFp(i)], [y_bFp(i), y_bFp(i)], [-z_F1(i),-z_F2
       (i)], 'r',...
    [0 0], [-W_B/2 W_B/2],[-z_simw(i) -z_sime(i)],'k',...
    [0 0], [-W_B/2 W_B/2],[-z_bw(i) -(2*T_ZNsim(i)-z_bw(i))],'b
```

```
       ',...
    [-L_B/2 -L_B/2], [-W_B/2 W_B/2],[WERightSimZ1(i)
       WERightSimZ2(i)],'k',...
    [L_B/2 L_B/2], [-W_B/2 W_B/2],[WELeftSimZ1(i) WELeftSimZ2(i
       )],'k',...
    [-L_B/2 -L_B/2], [-W_B/2 W_B/2],[WERightbZ1(i) WERightbZ2(i
       )],'b',...
    [L_B/2 L_B/2], [-W_B/2 W_B/2],[WELeftbZ1(i) WELeftbZ2(i)],'
       b');


pause(0.001)
end


end
```

E.2 Linear Data Filler Function

```
function [filledtime,filledmeas,gapInd] = LinearDataFiller(time
   ,meas,maxtimegap)
%LinearDataFiller
%   This function fills the missing parts of a dataset with a
   linear
%   function (linspace). Inputs are
%   time: elapsed time (Must start at 0)
%   meas: measurements logged during time
%   maxtimegap: maximum allowed gap without filling it


%% Find where the missind data is [at what itteration, at what
   time,
%% what is the time gap to next itteration, what is the
   measurement,
%% what is the measurement gap to next itteration]
j = 1;
for i = 1:size(time,1)-1
    timegap = time(i+1)-time(i);
    measgap = meas(i+1,:)-meas(i,:);
    if timegap > maxtimegap
        gapInd(j+1,:) = [i, time(i), timegap, meas(i,:),
           measgap];
        j = j + 1;
    end
end

%% Find the longest contiuous range of data to approximate
%% how many point should be created per second
%% cont = [how many points, how much time]

o = 1;
```

```
for n = 1:size(gapInd,1)-1
    cont(n,:) = [gapInd(n+1,1)-(gapInd(n,1)+1)+1, gapInd(n+1,2)
       -(gapInd(n,2)+gapInd(n,3))];

    if cont(n,1)/cont(n,2) == inf

    else
            nMeasPerSec1(o,1) = cont(n,1)/cont(n,2);
            o = o+1;
    end
end
nMeasPerSec = mean(nMeasPerSec1);

%% Create new arrays with filled gaps

filledtime = time(1,1);
filledmeas = meas(1,:);
for k = 1:size((gapInd),1)-1

    % Determine how many points are needed to fill gap #k
    nPoints(k) = round(gapInd(k+1,3)*nMeasPerSec);

    % Fill time array
    filledtime = [filledtime; time(gapInd(k,1)+1:gapInd(k+1,1)
       ,1)...
        ;linspace(gapInd(k+1,2),gapInd(k+1,2)+gapInd(k+1,3),
           nPoints(k))'];

    % Create an array filler for m measurements (2 dim)
    for m = 1:size(filledmeas,2)
        filler(:,m) = linspace(gapInd(k+1,m+3),gapInd(k+1,m+3)+
           gapInd(k+1,m+3+size(filledmeas,2)),nPoints(k))';
    end
    % Fill measurement array
    filledmeas = [filledmeas; meas(gapInd(k,1)+1:gapInd(k+1,1)
       ,:)...
        ;filler];

clear filler
end
% Add the data after last gap
filledtime = [filledtime; time(gapInd(k+1,1)+1:end,1)];
filledmeas = [filledmeas; meas(gapInd(k+1,1)+1:end,:)];

end
```

E.3 Data Merger

```
close all;
clear;
```

```
% This script merges two data sets with similar time stamps.
   One data set
% is used to patch missing parts of the other in order to
   obtain one
% complete data set.

%% Load data sets to merge

%load('Dry_Run_State_Machine_Test')
%load('Dry_Run_Simulation_Test')
%load('Conditioning_Test')
%load('AOA_Test')
%load('OA_Test')
%load('Dyn_Behaviour_Test')
%load('Dyn_Beh_Test')
load('Filtered_Signals_Test')

%% Gather time and data from the loaded matrix
time1 = signalData1(:,1)-signalData1(1,1);
trans1 = signalData1(:,2:end);

time2 = signalData2(:,1)-signalData2(1,1);
trans2 = signalData2(:,2:end);


time = time1;
meas = trans1;
time2 = time2;
meas2 = trans2;
maxtimegap = 0.1;

%% Fill the missing parts with correctly spaced linear data
   point
[filledtime1, filledmeas1, gapInd1] = LinearDataFiller(time,
   meas,maxtimegap);
[filledtime2, filledmeas2, gapInd2] = LinearDataFiller(time2,
   meas2,maxtimegap);

%% Plot the two data sets to manually analyse if they can be
   can be
% completely merged
figure
plot(time1,trans1(:,1:5))
grid on
figure
plot(time2,trans2(:,1:5))
grid on
figure
plot(filledtime1,filledmeas1(:,1:5),filledtime2,filledmeas2
   (:,1:5))
grid on
```

```
%% Select which gap to patch by setting value 1 (default is all gaps.
% however, some gaps is best to leave unpatched. If so set them to 0)
replacegaps = ones(size(gapInd1,1),1);
replacegaps(9:10,1) = 0;
%replacegaps(12:14,1) = [0;0;0];

%% Gaps are located and patched
for i = 1:size(gapInd1,1)
    tempstart1arr = find(round(filledtime1(:,1),1)==round(gapInd1(i,2),1));
    tempend1arr = find(round(filledtime1(:,1),1)==round(gapInd1(i,2)+gapInd1(i,3),1));
    tempstart2arr = find(round(filledtime2(:,1),1)==round(gapInd1(i,2),1));
    tempend2arr = find(round(filledtime2(:,1),1)==round(gapInd1(i,2)+gapInd1(i,3),1));
    tempstart1 = round((tempstart1arr(1,1)+tempstart1arr(end,1))/2);
    tempend1 = round((tempend1arr(1,1)+tempend1arr(end,1))/2);
    tempstart2 = round((tempstart2arr(1,1)+tempstart2arr(end,1))/2);
    tempend2 = round((tempend2arr(1,1)+tempend2arr(end,1))/2);
    tempstartend(i,:) = [tempstart1, tempend1, tempstart2, tempend2];
    if replacegaps(i) == 1
    filledmeas1(tempstart1:tempend1,:) = ...
    filledmeas2(tempstart2:tempstart2+tempend1-tempstart1,:);
    else
    end
end
%% Plot the repaired data set

%% Transmitted signal transition
% figure
% plot([0 60], [0.2 0.2],'r','linewidth',1)
% hold on
% plot([0 60], [0 0],'b','linewidth',1)
% hold on
%
% plot(filledtime1,filledmeas1(:,3),'k','linewidth',2)
% legend('Idle','Init','Z_b')
%
% set(gca, 'YDir', 'reverse')
% xlabel('Time [s]')
% ylabel('Z_N [m]')
% ylim([-0.3, 0.3])
% xlim([0, 50])
% grid on
```

```
%% Transmitted signal simulation
% figure
% subplot(2,1,1)
% plot(filledtime1,filledmeas1(:,1),'Color','#A2142F','linewidth',2)
% hold on
% plot(filledtime1,filledmeas1(:,2),'Color','#0072BD','linewidth',2)
% hold on
% plot(filledtime1,filledmeas1(:,3),'Color','#77AC30','linewidth',2)
% legend('X_b','Y_b','Z_b')
%
% xlabel('Time [s]')
% ylabel('Translation [m]')
% ylim([-0.07, 0.07])
% xlim([0, 50])
% grid on
%
% subplot(2,1,2)
% plot(filledtime1,filledmeas1(:,4),'Color','#A2142F','linewidth',2)
% hold on
% plot(filledtime1,filledmeas1(:,5),'Color','#0072BD','linewidth',2)
% legend('R_{X,N,b}','R_{Y,N,b}')
%
% xlabel('Time [s]')
% ylabel('Rotation [rad]')
% ylim([-0.1, 0.1])
% xlim([0, 50])
% grid on


%% Signal conditioning
% figure
%
% subplot(1,2,1)
% plot(filledtime1,filledmeas1(:,1:4),'linewidth',2)
% legend('NW','NE','SW','SE')
% xlabel('Time [s]')
% ylabel('Sensor values [-]')
% ylim([-100, 20000])
% xlim([0, 37])
% grid on
%
% subplot(1,2,2)
% plot(filledtime1,filledmeas1(:,1:4),'linewidth',2)
% xlabel('Time [s]')
% ylabel('Sensor values [-]')
% ylim([-100, 3000])
% xlim([0, 37])
% grid on

%% AOA test
% figure
%
% subplot(3,1,1)
% plot(filledtime1,filledmeas1(:,1:4),'linewidth',2)
% legend('NW','NE','SW','SE')
% xlabel('Time [s]')
% ylabel('Sensor values [-]')
% ylim([-100, 12000])
% xlim([0, 38])
% grid on
%
% subplot(3,1,2)
% plot(filledtime1,filledmeas1(:,5:6),'linewidth',2)
% legend('L_{X,aoa,p}','L_{Y,aoa,p}')
% xlabel('Time [s]')
% ylabel('Displacement [-]')
% ylim([-1.5, 1.5])
% xlim([0, 38])
% grid on
%
% subplot(3,1,3)
% plot(filledtime1,filledmeas1(:,7),'linewidth',2)
% legend('y_{lc}')
% xlabel('Time [s]')
% ylabel('Load magnitude [-]')
% ylim([-100, 15000])
% xlim([0, 38])
% grid on

%% OA Test
% figure
%
% subplot(3,1,1)
% plot(filledtime1,filledmeas1(:,1:4),'linewidth',2)
% lgd1 = legend('NW','NE','SW','SE');
% lgd1.NumColumns = 2;
% xlabel('Time [s]')
% ylabel('Sensor values [-]')
% ylim([-100, 12000])
% xlim([0, 29])
% grid on
%
% subplot(3,1,2)
% plot(filledtime1,filledmeas1(:,5:8),'linewidth',2)
% lgd2 = legend('L_{X,oa,p}','L_{Y,oa,p}','L_{X,aoa,p}','L_{Y,aoa,p}');
% lgd2.NumColumns = 2;
% xlabel('Time [s]')
% ylabel('Displacement [-]')
% ylim([-1.5, 2])
% xlim([0, 29])
% grid on
%
% subplot(3,1,3)
% plot(filledtime1,filledmeas1(:,9),'linewidth',2)
% legend('y_{lc}')
% xlabel('Time [s]')
% ylabel('Load magnitude [-]')
% ylim([-100, 12000])
% xlim([0, 29])
% grid on

%% Direction of Motion Test
% figure
%
% subplot(3,1,1)
% plot(filledtime1,filledmeas1(:,1:4),'linewidth',2)
% lgd1 = legend('NW','NE','SW','SE');
% lgd1.NumColumns = 2;
% xlabel('Time [s]')
% ylabel('Sensor values [-]')
% ylim([-100, 8000])
% xlim([0, 50])
% grid on
%
% subplot(3,1,2)
% plot(filledtime1,filledmeas1(:,5),'Color','#A2142F','
   linewidth',2)
% hold on
% plot(filledtime1,filledmeas1(:,6),'Color','#0072BD','
   linewidth',2)
% hold on
% plot(filledtime1,filledmeas1(:,7),'Color','#77AC30','
   linewidth',2)
% lgd2 = legend('X_b','Y_b','Z_b');
% lgd2.NumColumns = 3;
% xlabel('Time [s]')
% ylabel('Translation [m]')
% ylim([-0.04, 0.04])
% xlim([0, 50])
% grid on
%
% subplot(3,1,3)
% plot(filledtime1,filledmeas1(:,8),'Color','#A2142F','
   linewidth',2)
% hold on
% plot(filledtime1,filledmeas1(:,9),'Color','#0072BD','
   linewidth',2)
% legend('R_{X,N,b}','R_{Y,N,b}','Location','SouthEast')
% xlabel('Time [s]')
% ylabel('Rotation [rad]')
% ylim([-0.1, 0.1])
% xlim([0, 50])
% grid on

%% Dynamic Behaviour Test
% figure
%
% subplot(3,1,1)
% plot(filledtime1,filledmeas1(:,1:4),'linewidth',2)
% lgd1 = legend('NW','NE','SW','SE');
% lgd1.NumColumns = 2;
% xlabel('Time [s]')
% ylabel('Sensor values [-]')
% ylim([-100, 5000])
% xlim([0, 29.5])
% grid on
%
% subplot(3,1,2)
% plot(filledtime1,filledmeas1(:,5),'Color','#A2142F','
   linewidth',2)
% hold on
% plot(filledtime1,filledmeas1(:,6),'Color','#0072BD','
   linewidth',2)
% hold on
% plot(filledtime1,filledmeas1(:,7),'Color','#77AC30','
   linewidth',2)
% lgd2 = legend('X_b','Y_b','Z_b');
% lgd2.NumColumns = 3;
% xlabel('Time [s]')
% ylabel('Translation [m]')
% ylim([-0.07, 0.07])
% xlim([0, 29.5])
% grid on
%
% subplot(3,1,3)
% plot(filledtime1,filledmeas1(:,8),'Color','#A2142F','
   linewidth',2)
% hold on
% plot(filledtime1,filledmeas1(:,9),'Color','#0072BD','
   linewidth',2)
% lgd3 = legend('R_{X,N,b}','R_{Y,N,b}');
% lgd3.NumColumns = 3;
% xlabel('Time [s]')
% ylabel('Rotation [rad]')
% ylim([-0.15, 0.15])
% xlim([0, 29.5])
% grid on
```

```
%% Filtered Signals Test
% figure
%
% subplot(3,1,1)
% plot(filledtime1(288:end,1),filledmeas1(288:end,2),
   filledtime1(288:end,1),filledmeas1(288:end,1),'linewidth',1)
% lgd1 = legend('Unfiltered LC','Filtered LC');
% lgd1.NumColumns = 2;
% xlabel('Time [s]')
% ylabel('Sensor values [-]')
% ylim([5000, 12000])
% xlim([3, 29.5])
% grid on
%
% subplot(3,1,2)
% plot(filledtime1(288:end,1),filledmeas1(288:end,5),
   filledtime1(288:end,1),filledmeas1(288:end,3),'linewidth',1)
% lgd2 = legend('Unfiltered pos','Filtered pos','Location','
   SouthEast');
% lgd2.NumColumns = 2;
% xlabel('Time [s]')
% ylabel('Position [m]')
% ylim([1, 2])
% xlim([3, 29.5])
% grid on
%
% subplot(3,1,3)
% plot(filledtime1(288:end,1),filledmeas1(288:end,6),
   filledtime1(288:end,1),filledmeas1(288:end,4),'linewidth',1)
% lgd3 = legend('Unfiltered vel','Filtered vel');
% lgd3.NumColumns = 2;
% xlabel('Time [s]')
% ylabel('Velocity [m/s]')
% ylim([-1, 2])
% xlim([3, 29.5])
% grid on

%% With VR Test
clear
load('BaardPaaData_ForAnimated','time','FL_Xoap','FL_Yoap')
load('BaardPaaData')

[filledtime1, filledmeas1, gapInd1] = LinearDataFiller(numData
   (:,1)-numData(1,1), numData(:,2:end),0.1);
filledtime1 = filledtime1(19340:end,1);
filledmeas1 = filledmeas1(19340:end,:);
time = time(18971:end,1);
FL_Xoap = FL_Xoap(18971:end);
FL_Yoap = FL_Yoap(18971:end);
figure
```

```
subplot(3,1,1)
plot(time,FL_Xoap,time,FL_Yoap,'linewidth',2)
lgd1 = legend('L_{X,oa,p}','L_{Y,oa,p}');
lgd1.NumColumns = 2;
xlabel('Time [s]')
ylabel('Location within oa [-]')
ylim([-1, 1])
xlim([233.5, 242])
grid on

subplot(3,1,2)
plot(filledtime1,filledmeas1(:,5),'Color','#A2142F','linewidth'
   ,2)
hold on
plot(filledtime1,filledmeas1(:,6),'Color','#0072BD','linewidth'
   ,2)
hold on
plot(filledtime1,filledmeas1(:,7),'Color','#77AC30','linewidth'
   ,2)
hold on
plot(filledtime1,filledmeas1(:,35)*(-1),'--','Color','#77AC30',
   'linewidth',2)
lgd2 = legend('X_b','Y_b','Z_b','Z_{sim}');
lgd2.NumColumns = 4;
xlabel('Time [s]')
ylabel('Translation [m]')
ylim([-0.2, 0.2])
xlim([233.5, 242])
grid on

subplot(3,1,3)
plot(filledtime1,filledmeas1(:,8),'Color','#A2142F','linewidth'
   ,2)
hold on
plot(filledtime1,filledmeas1(:,9),'Color','#0072BD','linewidth'
   ,2)
hold on
plot(filledtime1,filledmeas1(:,37),'--','Color','#A2142F','
   linewidth',2)
hold on
plot(filledtime1,filledmeas1(:,36)*(-1),'--','Color','#0072BD',
   'linewidth',2)
lgd3 = legend('R_{X,N,b}','R_{Y,N,b}','R_{X,N,sim}','R_{Y,N,sim
   }');
lgd3.NumColumns = 4;
xlabel('Time [s]')
ylabel('Rotation [rad]')
ylim([-0.1, 0.2])
xlim([233.5, 242])
grid on
```

E.4 Signal conditioning and oa establishment

```
close all;
clear;

% This script conditions the load cell measurements and
   calculates
% the location and magnitude of the force applid by the user.
% This script prepares data for animating the real-time
   location
% of the force within operating area (Used for demo video).


%% Load data set which has been filled with linear data points
load('BaardPaaData_Fixed','temptime','templc'); % Run WITH VR
%load('BaardPaaData_woVR_Fixed','temptime','templc'); % Run
   WITHOUT VR 1
%load('BaardPaaData_woVR2_Fixed','temptime','templc'); % Run
   WITHOUT VR 2

%% Gather data
time = temptime-temptime(1);   % Gather time and zero the
   starting value
load_cell_data = templc;       % Gather LC data
videotime = time-time(378)+47; % videotime for run WITH VR
%videotime = time; % videotime for run WITHOUT VR 1
%videotime = time; % videotime for run WITHOUT VR 2

%% Plot raw load cell data (time stamp suits recorded video)
figure(1)
plot(videotime,load_cell_data)
title('Video fitted')
legend('NW','NE','SW','SE')

%% Determine where and duration of identification processes
% WITH VR
Zerolow = 1;
Zerohigh = 378;
Gainlow = 3712;
Gainhigh = 3904;
OAlow = 5396;
OAhigh = 5500;

% % WITHOUT VR 1
% Zerolow = 1;
% Zerohigh = 378;
% Gainlow = 2297;
% Gainhigh = 2431;
% OAlow = 3203;
% OAhigh = 3429;
% % WITHOUT VR 2
% Zerolow = 1;
% Zerohigh = 378;
% Gainlow = 2304;
% Gainhigh = 2411;
% OAlow = 3333;
% OAhigh = 3433;



%% Idently zeroes and plot 1st conditioned LC signals
zeros = mean(load_cell_data(Zerolow:Zerohigh,1:4));

Zeroed_LC_Data = load_cell_data(:,1:4) - zeros; % Subtract zero
    values

figure(2)
plot(videotime,Zeroed_LC_Data)
title('Zeroed')
legend('NW','NE','SW','SE')

%% Two-point calibration (identify gains), plot 2nd conditioned
    LC signals

for j = 1:Gainhigh-Gainlow+1
    s_ref(j) = sum(Zeroed_LC_Data(Gainlow-1+j,1:4))/4;
    G(j,1:4) = [s_ref(j)/Zeroed_LC_Data(Gainlow-1+j,1)...
        ,s_ref(j)/Zeroed_LC_Data(Gainlow-1+j,2)...
        ,s_ref(j)/Zeroed_LC_Data(Gainlow-1+j,3)...
        ,s_ref(j)/Zeroed_LC_Data(Gainlow-1+j,4)];
end
G_mean = mean(G);

ZeroedGained_LC_Data = Zeroed_LC_Data.*G_mean; % Subtract zero
   values

figure(3)
plot(videotime,ZeroedGained_LC_Data)
title('Zeroed & gained')
legend('NW','NE','SW','SE')

%% Establish operating area

for k = 1:length(ZeroedGained_LC_Data)
L_Xaoap(k) = (sum(ZeroedGained_LC_Data(k, 1:2))...
    -sum(ZeroedGained_LC_Data(k, 3:4)))...
    /sum(ZeroedGained_LC_Data(k, 1:4));
L_Yaoap(k) = (sum(ZeroedGained_LC_Data(k, [2 4]))...
    -sum(ZeroedGained_LC_Data(k, [1 3])))...
    /sum(ZeroedGained_LC_Data(k, 1:4));
end
```

```
L_Xaoaoa = mean(L_Xaoap(1,OAlow:OAhigh));
L_Yaoaoa = mean(L_Yaoap(1,OAlow:OAhigh));

L_Xoap = L_Xaoap - L_Xaoaoa;
L_Yoap = L_Yaoap - L_Yaoaoa;



%% Add LP-filter (SS model)
m = 3;
k = 50;
d = 20;

A = [0 1; -k/m -d/m];
B = [0; 1/m];
C = [1 0];
D = [0];

Filter = ss(A,B,C,D);


% Filtering load cell measurements (using raw signals)
timet = linspace(videotime(1,1),videotime(end),length(time));

x0 = [templc(1,1)/k,0];
[yNW,t,xNW] = lsim(Filter,templc(:,1),timet,x0);
[yNE,t,xNE] = lsim(Filter,templc(:,2),timet,x0);
[ySW,t,xSW] = lsim(Filter,templc(:,3),timet,x0);
[ySE,t,xSE] = lsim(Filter,templc(:,4),timet,x0);
figure(7)
plot(t,yNW,t,yNE,t,ySW,t,ySE)

FZeroed_LC_Data = k*[yNW yNE ySW ySE] - zeros; % subtracting
   zeroes
FZeroedGained_LC_Data = FZeroed_LC_Data.*G_mean; % multiplying
   with gain

%% Location within available operating area
for m = 1:length(FZeroedGained_LC_Data)
FL_Xaoap(m) = (sum(FZeroedGained_LC_Data(m, 1:2))...
    -sum(FZeroedGained_LC_Data(m, 3:4)))...
    /sum(FZeroedGained_LC_Data(m, 1:4));
FL_Yaoap(m) = (sum(FZeroedGained_LC_Data(m, [2 4]))...
    -sum(FZeroedGained_LC_Data(m, [1 3])))...
    /sum(FZeroedGained_LC_Data(m, 1:4));
end

%% Location within operating area
FL_Xoap = FL_Xaoap - L_Xaoaoa;
FL_Yoap = FL_Yaoap - L_Yaoaoa;
```

```
%% Saturate (limits at -1 & 1)
for n = 1:length(videotime)
    if FL_Xoap(n) >= 1
        FL_Xoap(n) = 1;
    elseif FL_Xoap(n)<=-1
        FL_Xoap(n) = -1;
    else
        FL_Xoap(n) = FL_Xoap(n);
    end
    if FL_Yoap(n) >= 1
        FL_Yoap(n) = 1;
    elseif FL_Yoap(n)<=-1
        FL_Yoap(n) = -1;
    else
        FL_Yoap(n) = FL_Yoap(n);
    end
end

%% Dimensions of board
W_sb = 0.4;
L_sb = 0.6;

%% Location within board dimensions
T_Xoap = FL_Xoap*L_sb/2;
T_Yoap = FL_Yoap*W_sb/2;


%% Plot to verify the conditioning and oa establishment
figure(5)
plot(videotime,FL_Xoap,videotime,FL_Yoap)
legend('FL_Xoap','FL_Yoap')
```

E.5 Animate location of applied force

```
close all;
clear;

% This script animates the location of applied force.

load('BaardPaaData_ForAnimated','T_Xoap','T_Yoap','videotime','
   W_sb','L_sb')
%Run WITH VR
% load('BaardPaaData_woVR_ForAnimated','T_Xoap','T_Yoap','
   videotime','W_sb','L_sb')
% %Run WITHOUT VR 1
%load('BaardPaaData_woVR2_ForAnimated','T_Xoap','T_Yoap','
   videotime','W_sb','L_sb')
%Run WITHOUT VR 2

%% Initialize plot
```

```
oaPlot = plot(T_Xoap(1),T_Yoap(1),'or','linewidth',8);
axis equal
set(gca, 'YDir', 'reverse')
xlim([-0.4 0.4])
ylim([-0.3, 0.3])
xlabel('X [m]')
ylabel('Y [m]')
grid on
hold on
%% Draw board circumference
SBPlot = plot([-L_sb/2 -L_sb/2],[-W_sb/2 W_sb/2], '-k',...
        [L_sb/2 L_sb/2],[-W_sb/2 W_sb/2], '-k',...
        [-L_sb/2 L_sb/2],[-W_sb/2 -W_sb/2], '-k',...
        [-L_sb/2 L_sb/2],[W_sb/2 W_sb/2], '-k','linewidth',3);
legend('Pressure Location','Board','AutoUpdate','off')
N = length(videotime);
currTime = round(videotime);

%% Animate
for n = 1:N
    tic;
    delete(oaPlot)
    title(currTime(n))
    oaPlot = plot(T_Xoap(n), T_Yoap(n),'or','linewidth',8);
    executiontime = toc;
    deltatime = 0.01-executiontime;
    pause(deltatime)
end
```

Bibliography

[1] M. Slimani, R. Ramirez-Campillo, A. Paravlic, L. D. Hayes, N. L. Bragazzi, and M. Sellami, “The effects of physical training on quality of life, aerobic capacity, and cardiac function in older patients with heart failure: A meta-analysis,” *Frontiers in physiology*, vol. 9, p. 1564, 2018.

[2] N. Kasven-Gonzalez, R. Souverain, and S. Miale, “Improving quality of life through rehabilitation in palliative care: case report,” *Palliative & supportive care*, vol. 8, no. 3, pp. 359–369, 2010.

[3] S. Paolucci, A. V. Di, R. Massicci, M. Traballesi, I. Bureca, A. Matano, M. Iosa, and C. Guariglia, “Impact of participation on rehabilitation results: a multivariate study.” *European journal of physical and rehabilitation medicine*, vol. 48, no. 3, pp. 455–466, 2012.

[4] S. Ogus. Zwift raises $120m to expand its virtual cycling technology to esports. [Online]. Available: https://www.forbes.com/sites/simonogus/2018/12/21/zwift-raises-120m-to-expand-its-virtual-cycling-app-to-esports/#2589a6285f42

[5] TechnoGym. Zwiftillustration. [Online]. Available: https://www.technogym.com/int/news-events-blog/mycycling-is-compatible-with-zwift/

[6] H. I. Krebs, J. J. Palazzolo, L. Dipietro, M. Ferraro, J. Krol, K. Rannekleiv, B. T. Volpe, and N. Hogan, “Rehabilitation robotics: Performance-based progressive robot-assisted therapy,” *Autonomous robots*, vol. 15, no. 1, pp. 7–20, 2003.

[7] Motek. Caren. [Online]. Available: https://www.motekmedical.com/solution/caren/

[8] summitmedsci. Caren extended. [Online]. Available: https://summitmedsci.co.uk/products/motek-caren/

[9] NAV, *Assistive technology in Norway*, 02 2017, pp. 3–5.

[10] GamanRehabilitation. Rehabillustration. [Online]. Available: https://gamanrehabilitation.com/stroke-rehabilitation/

[11] MayoClinic. Mayoclinicrehabilitation. [Online]. Available: https://www.mayoclinic.org/diseases-conditions/stroke/in-depth/stroke-rehabilitation/art-20045172

[12] AugmentedRealityInMedicine. Augmentedrealityinmedicine. [Online]. Available: http://arinmed.com/gait-training-with-augmented-reality-the-gaspar-trial/

[13] E. Biffi, E. Beretta, E. Diella, D. Panzeri, C. Maghini, A. C. Turconi, S. Strazzer, and G. Reni, “Gait rehabilitation with a high tech platform based on virtual reality conveys improvements in walking ability of children suffering from acquired brain injury,” in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2015, pp. 7406–7409.

[14] PlayStation. Dualshock 4 traadløs kontroller. [Online]. Available: https://www.playstation.com/no-no/explore/accessories/dualshock-4-wireless-controller/

[15] A. Q. Keemink, H. van der Kooij, and A. H. Stienen, "Admittance control for physical human–robot interaction," *The International Journal of Robotics Research*, vol. 37, no. 11, pp. 1421–1444, 2018.

[16] M. C. Lin, "Recent advances in haptic rendering and applications," pp. A1 – A11, 2005. [Online]. Available: https://cgl.ethz.ch/Downloads/Publications/Tutorials/2005/Lin05/SIG05Course.pdf

[17] B. Hannaford and A. Okamura, *Haptics*, 01 2008, pp. 719–739.

[18] M. Otaduy and M. Lin, "Introduction to haptic rendering," p. 3, 07 2005.

[19] V. Davis. Mapbox introduces martini, a client-side terrain mesh generation code. [Online]. Available: https://hub.packtpub.com/mapbox-introduces-martini-a-client-side-terrain-mesh-generation-code/

[20] N. Lazzaro. What makes a game fun? [Online]. Available: https://www.youtube.com/watch?v=qtLUJJSw8H8

[21] TINOUCLT. Ssx 3, gamelles et courses folles sur xbox one x au programme. [Online]. Available: https://xboxsquad.fr/news/2018/04/ssx-3-gamelles-et-courses-folles-sur-xbox-one-x-au-programme/

[22] J. Hindy. 10 best skateboarding games for android! [Online]. Available: https://www.androidauthority.com/best-skateboarding-games-android-883916/

[23] Rexroth. emotion 1500. [Online]. Available: https://www.boschrexroth.com/en/us/industries/machinery-applications-and-engineering/motion-simulation-technology/products-and-solutions/6dof-motion-platforms/emotion-1500/index

[24] Qualisys. Qualisys 7+. [Online]. Available: https://www.qualisys.com/hardware/5-6-7/#!#tech-specs

[25] LifeFitness. Lifefitness 95t. [Online]. Available: https://www.turbosquid.com/3d-models/life-fitness-95t-treadmill-max/659335

[26] N. Instruments. Ni myrio-1900. [Online]. Available: https://www.ni.com/en-no/shop/select/myrio-student-embedded-device

[27] N. Instrument, "Ni myrio- 1900 user guide and specifications," *National instruments Corporation*, 2013.

[28] Boquite. Dylf-102. [Online]. Available: https://www.amazon.ca/DYLF-102-Weighing-Pressure-Tension-0-1000kg/dp/B082R8X1P3

[29] Nintendo. Wii balance board launch. [Online]. Available: https://en.wikipedia.org/wiki/Wii_Balance_Board#cite_note-5

[30] S. Arendt. Miyamoto: Wii fit's purpose isn't to make you fit. [Online]. Available: https://www.wired.com/2008/02/miyamoto-wii-fi-2/

[31] Nintendo. Wii balance board. [Online]. Available: http://www.nintendolife.com/news/2019/08/random_you_can_always_rely_on_your_parents_to_find_a_clever_use_for_the_wii_fit_balance_board

[32] SEW. Sew eurodrice movitrac. [Online]. Available: https://www.ebay.de/itm/SEW-Eurodrive-Frequenzumrichter-MC07A022-2B1-4-10-/111641415734

[33] W. Central. Do you need a pc to set up an oculus quest? [Online]. Available: https://www.windowscentral.com/do-you-need-pc-set-oculus-quest

[34] Unknown. Wii balance board. [Online]. Available: https://wiibrew.org/wiki/Wii_Balance_Board

[35] RovingNetworks. Bluetooth hid profile. [Online]. Available: https://cdn.sparkfun.com/datasheets/Wireless/Bluetooth/RN-HID-User-Guide-v1.0r.pdf

[36] craya. Use wii balance board in labview. [Online]. Available: https://forums.ni.com/t5/LabVIEW/Use-Wii-Balance-Board-in-LabVIEW/td-p/710740?profile.language=en

[37] JohannS. Using a wiimote with labview. [Online]. Available: https://forums.ni.com/t5/LabVIEW/Using-a-wiimote-with-LabVIEW/m-p/526676?profile.language=en

[38] P. Electronics. The fundamentals of 4...20 ma current loops. [Online]. Available: https://www.prelectronics.com/the-fundamentals-of-420-ma-current-loops

[39] Qualisys. Qualisys track manager. [Online]. Available: https://www.qualisys.com/software/qualisys-track-manager/

[40] K. Academy. What is centripetal acceleration? [Online]. Available: https://www.khanacademy.org/science/physics/centripetal-force-and-gravitation/centripetal-acceleration-tutoria/a/what-is-centripetal-acceleration

[41] L. Learning. Fictitious forces and non-inertial frames: The coriolis force. [Online]. Available: https://courses.lumenlearning.com/physics/chapter/6-4-fictitious-forces-and-non-inertial-frames-the-coriolis-force/

[42] O. Languages. Holistic definition. [Online]. Available: https://www.lexico.com/en/definition/holistic

[43] Silvalea. Silvalea walking harness. [Online]. Available: http://www.silvalea.com/walking-harness.html