User Manual

Autonomous Truck with Trailer

December 15, 2022

Version 0.1

ADAPT™

Status

Reviewed	Alfred Sundstedt	2022-12-12
Approved	Shamisa Shoja	2022-12-12

Project Identity

Group E-mail:	alfsu259@liu.se
Homepage:	https://www.control.isy.liu.se/student/tsrt10/
Orderer:	Shamisa Shoja, Reglerteknik, ISY E-mail: shamisa.shoja@liu.se
Customer:	Daniel Axehill, Reglerteknik, ISY E-mail: daniel.axehill@liu.se
Supervisor:	Carl Hynén Ulfsjöö, Reglerteknik, ISY E-mail: carl.hynen@liu.se
Course Responsible:	Daniel Axehill, Reglerteknik, ISY E-mail: daniel.axehill@liu.se

Participants of the group

Name	**Responsibility**	**E-mail**
Martin Axelsson		marax633@student.liu.se
Jesper Barreng	Test Manager	jesba281@student.liu.se
Isak Bokne	Design Manager	isabo438@student.liu.se
Charlie Elf		chael086@student.liu.se
Terese Johansson	Document Manager	terjo233@student.liu.se
Alfred Sundstedt	Project Leader	alfsu259@student.liu.se
Emil Wiman	Software Architect	emiwi425@student.liu.se

CONTENTS

DOCUMENT HISTORY

Version	Date	Changes made	Sign	Reviewer
0.1	2022-12-05	First draft.	J.Barreng, T.Johansson	A.Sundstedt

1 INTRODUCTION

This manual aims to specify how to run the developed system. The manual will be divided into the different sections listed below:

- **Hardware** - describes the hardware needed to run the system and how to operate them.
- **Setup and Software** - describes the setup and installations of the software needed to run the project.
- **Usage** - describes in detail the steps needed to be executed in order to run the system in simulation.

1.1 Definition of terms

Terms found in the document are described below.

- **Git** - Software used for version control.
- **MPC** - **M**odel **P**redictive **C**ontrol, a method for process control.
- **Qualisys** - Motion capture and 3D positioning tracking system.
- **ROS** - **R**obot **O**perating **S**ystem, a set of software libraries and tools used for robot apllications.
- **RPi** - Raspberry Pi, a single board computer.
- **Visionen** - An arena for research and education at Linköping University.
- **EV3** - Unit used to control and power the actuators and sensors on the truck.
- **RViz** - A visualization tool used in ROS.
- **POMDP** - **P**artially **O**bservable **M**arkov **D**ecision **P**rocesses. A mathematical framework for decision making with uncertainty. The agent can not observe the full underlying state, hence it is partially observable
- **IMM filter** - **I**nteractive **M**otion **M**odel filter is a filter designed to track several objects that are highly maneuverable.
- **CMake** - An open source tool designed to build, test and package software.

2 HARDWARE

This section describes the hardware that is being used in the project. The hardware for the system consists of the Lego Truck, an RPi, an EV3 and a project computer. To be able to make all these different hardware communicate with each other it is required that they are all connected to the same network.

2.1 Lego truck

The Lego truck is built up by the components listed below:

- **EV3** - is used to control the truck and providing the sensor measurements.
- **RPi** - is used for the communication between the systems and sending steering commands to the EV3.
- **Powerbank** - works as powersupply for the RPi.
- **Chassis** - Holds the above mentioned components along with sensors, motors and markers for the positioning system.

2.1.1 *EV3*

The EV3 is used to control the motors and the sensors of the Lego truck. The EV3 needs 6 AA-batteries to be powered.

The motors and sensors need to be connected correctly on the EV3, this can be checked by making sure that:

- The driving motor is connected to port C.
- The steering motor is connected to port B.
- The β_2 sensor is connected to port 1.
- The β_3 sensor is connected to port 2.

2.1.2 *RPi*

The RPi takes care of all the main computations.

2.1.3 *Chassis*

The chassis used for the Lego truck is shown in Figure 1.

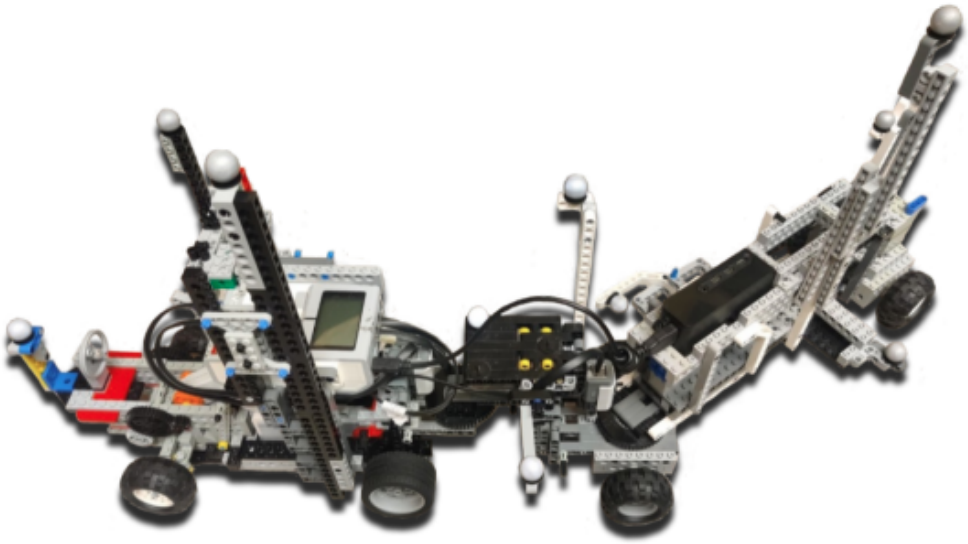

Figure 1: The truck and trailer used in the project

2.2 Project Computer

The computer used in this project is a HP laptop that has the following specifications:

- Intel Core i5-2000 CPU GHzx4
- Mesa Intel HD Graphics 5500 (BDW GT2)
- 8GB Ram
- 250 GB hard disk

The project computer is running Ubuntu 20.04.3 LTS.

2.3 Visionen hardware

In visionen there are two computers used to run the system, SHOWN IN FIGURE.....

One of the computers is responsible for running and operating the projectors in Visionen and is shown to the left in the figure. This computer runs Windows 10 and is accessed through a Liu-account. It needs to have Matlab 2019b and the Projector Toolbox installed.

The other computer is responsible for running the positioning system called Qualisys, this computer is shown to the right in the figure. As the previous computer this also uses Windows 10 and is accessed through a Liu-account.

3 SETUP AND SOFTWARE

This section describes the required software needed to run the system. A short description of the setup process for each software follows.

3.1 Project computer

To build and run the project, the Ubuntu software is required. The project computer is already using Ubuntu 20.04 so an insallation is not needed. In order to build and run the project, make sure that Git and ROS are installed on the computer.

3.2 Installing Git

To access the Git repository of the project, Git is needed and the installation of this is done through the following command lines:

```
$ sudo apt update
$ sudo apt install git-all
```

3.3 Installing ROS

For installation of ROS on an external computer, see [1]. ROS should already be installed on the project computer.

3.4 Git Repository

The project is located at a GitLab-repository. Create a folder with the name *rev_truck* and open a terminal in this folder. The project is downloaded via the terminal by typing the following command:

```
$ git clone https://gitlab.liu.se/tsrt10/2022/rev_truck.git
```

The project folder will now be found in:

```
~\rev_truck
```

3.5 ROS

ROS is a set of software libraries to develop robot applications and is used as a middleware for the system. The general documentation for ROS can be found in [1].

3.6 Acados

Acados is a software package for solutions of optimal control and estimation problems. The complete installation and documentation guide for the Acados software can be found in [2]. In order to run Acados in Matlab, type the following command in the Matlab command window:

```
mex -setup
```

After executing this command, choose the appropriate compiler to generate C-code.

3.7 Building the project

Before building the project, make sure that all necessary folders and files are existing in the folder structure of the project.
When building the project, navigate to the folder named *catkin_ws* by typing in the following command in the terminal:

```
$ cd rev_truck/catkin_ws
```

Inside the *catkin_ws* folder, the project is built by running the command below:

```
$ catkin_make
```

Catkin is included by default when using the ROS package.

4 USAGE

In this section the setup for the system and how to run it in the simulation environment will be explained in detail. Two versions exist for running the system in the simulation environment. The version described in section 4.1 is based on 2021 year's project and the version in section 4.2 is based on this year's project.

4.1 Running the implementation from 2021

The system can be tested in the simulation environment using the project computer. After the project is built using *Catkin*, open a separate terminal. In the new terminal, the ROS master node is executed by typing the following command:

```
$ roscore
```

If any problems occur when starting the node, open the bash file and type:

```
$ export ROS_MASTER_URI=http://localhost:11311
```

Comment out all the lines containing ROS_MASTER_URI and ROS_IP. Now start the simulation, service caller and RViz. RViz is supposed to be started with the file *conf_2021.rviz*. If this does not work, the launch file named *simulation_2021_.launch* needs to be changed to match the folder structure. In the simulation node launch file there is a parameter called *fake_ekf* which is used to determine if the simulations should send messages to the topic named *state_observer/estimated_state*. By setting this parameter to *true* the system can be simulated without using the state observer. If set to false, the state observer has to be started. These steps are executed by typing the following commands into the terminal.

```
$ cd /path/to/catkin_ws/src/launch
$ roslaunch simulation_2021.launch
$ roslaunch lego_truck_state_observer node.launch
```

Start the planner and MPC-controller:

```
$ roslaunch lego_truck_planner node.launch
$ roslaunch lego_truck_mpc_controller_trajectory node.launch
```

To start the obstacle simulator, use the following commands:

```
$ cd /path/to/catkin_ws/src/launch
$ roslaunch obs_predictor.launch
```

The dynamic obstacle simulator cannot only alternate between stochastic and determenistic behaviour, but also between the types "ground vehicle" or "pedestrian" which is set in the file predictor.py

```
self.determenistic = False
self.type = 0
```

When the nodes are up and running in the simulation a mission can be executed which is done in the service caller GUI where a start and goal position is specified.

4.2 Running the implementation from 2022

This section aims to describe how to run the implementation of the different modules that was built during the fall of 2022. Three different modules was developed, the IMM-filter, the POMPD Motion Planner and the MPC-controller using Acados. The following subsections describe how to start and test each module separately.

4.2.1 *IMM-filter*

This section describes how to run and visualize the IMM-filter in RViz. Make sure that you have installed everything appropriately as described in section 3.

Open a terminal and go to the project directory. Start roscore in this terminal.

```
$ cd /path/to/rev_truck
$ roscore
```

Open a new terminal (On Ubuntu, use *Ctrl+Alt+T*). From this terminal head to the launch folder. Here we start the the simulator and RViz.

```
$ cd catkin_ws/src/launch
$ roslaunch simulation_2022.launch
```

Now the simulation should start and RViz should pop up. Open a new terminal (in the same directory) and execute.

```
$ roslaunch obs_imm.launch
```

This will start the obstacle simulator, which will publish measurements on a topic. It will also start the IMM-filter which will subscribe on this topic and publish relevant information to RViz. If you switch over to RViz there should now be a dynamic obstacle that is moving (represented by a cube or a sphere). The tracking of the obstacle is the arrow. The ground truth trajectory is the green line and the predicted trajectory is the blue line.

To change the type of the dynamic obstacle, head to where the obstacle simulator is defined. In a new terminal starting in the root of the repo, execute.

```
$ cd catkin_ws/src/lego_truck_obstacle_simulator/src/lego_truck_obstacle_simulator
```

Open the python file "obstacle_simulator.py" using your favorite editor and head to line 47.

```
47 self.type = 0 # 0 = pedestrian, 1 = vehicle
```

Simply edit this line to change the type of the dynamic obstacle. After this, rerun the "roslaunch simulation_2022.launch" command in the same terminal (Stop it by *Ctrl+C*) and the simulation will restart in RViz.

4.2.2 *POMDP Motion planner*

4.2.3 *MPC using Acados*

4.3 Visionen setup

The setup of QualiSys in visionen is to be done at the Windows computer connected to the Visionen positioning system. The setup is done in steps and needs to be done correctly for the system to work as good as possible. The following steps needs to be done:

1. If the cameras in *Visionen* are off, turn them on with the controller. A green light on the cameras indicate that they are on.
2. If not already downloaded, download the *QualiSys Track Manager* folder from the Git repository.
3. Unzip the *Qualisys.zip* in the *QualiSys Track Manager* folder or desired derictory.
4. Start QualiSys Track Manager and select, in the window shown in Figure 2, *QualiSys Track Manages/QualiSys*, or the path where */Qualisys* was unzipped, as the project folder. The user should now be in the start window, as seen in Figure 3. Click "New Project" and choose "Base the new project on: Settings imported from another project".
5. Press the *Start Capture* button and then press *OK* and then *Cancel* in the capture window as seen in Figure 4 Now the user should be in the measurement window as seen in Figure 5. Make sure that the message *Realtime started* is given in the buttom of the window.
6. Perform the calibration setup that is described in Section 4.3.2.
7. Redefine the bodies according to Section 4.3.3.
8. Open the 3D-view.

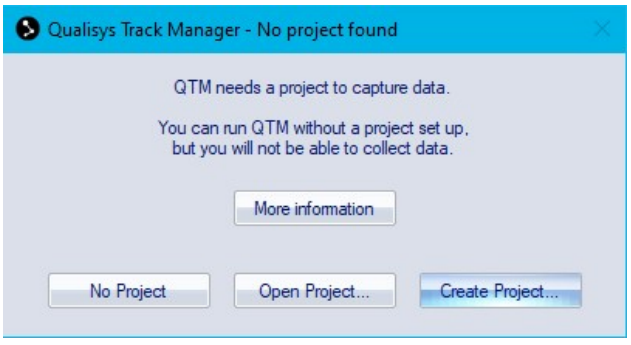

Figure 2: Starting Qualisys

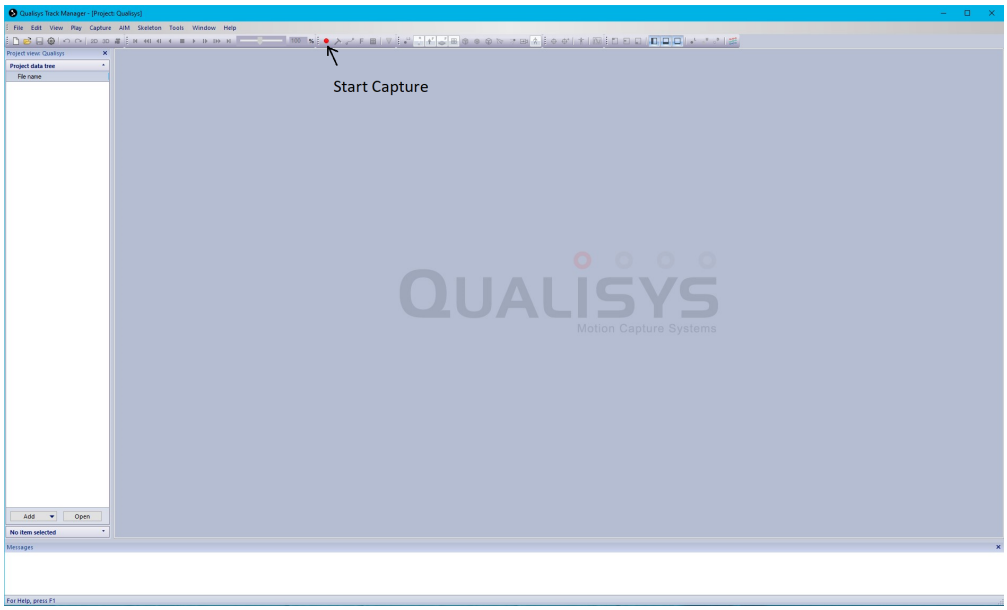

Figure 3: Qualisys Track Manager start menu

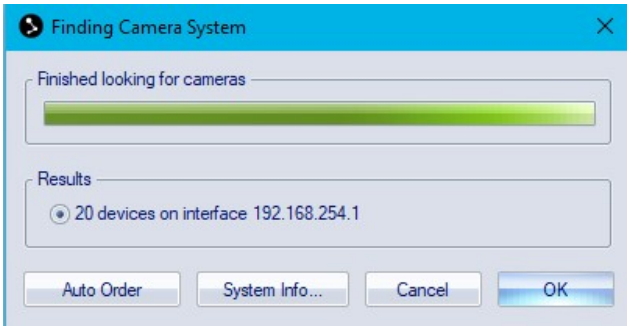

Figure 4: Finding the cameras

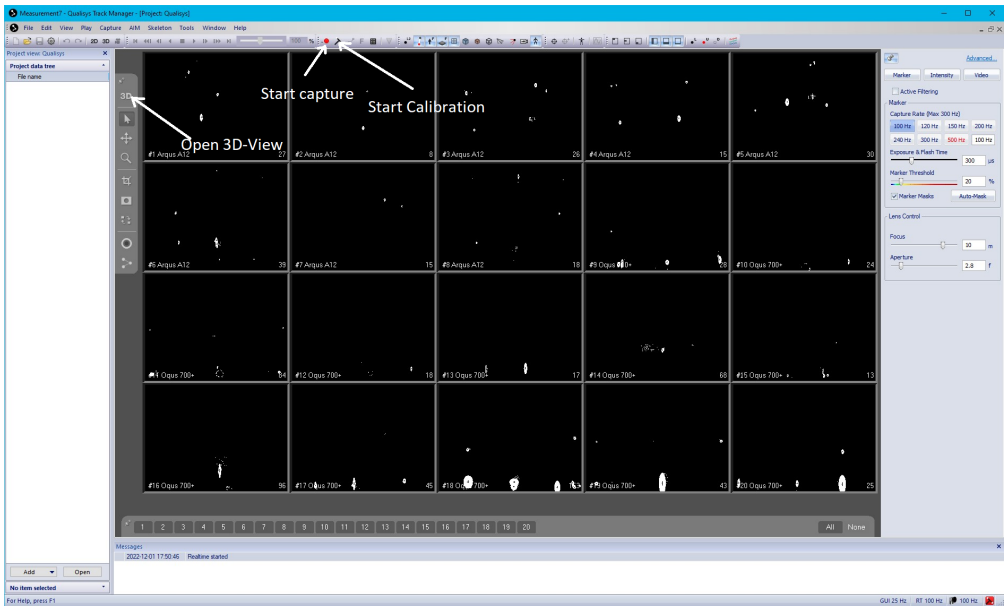

Figure 5: Qualisys Track Manager measurement menu

4.3.1 *Configure cameras*

1. In the *Camera View (Measurement window)*, make sure that no markers appear in the *Visionen Arena*.
2. Click on the *2D-view* and then the *Auto Mask* button on the right side of the view and allow to remove all previous masks if desirable.
3. Starting from *Camera 1* in the top left corner, double click on it to open the *Single camera view* and if needed, add masks manually where it is necessary. There should be none or very few detections after the masking is done. Since only five masks per camera can be used, it is of importance to distribute them effectively. Use the short commando *hold shift + m* for a faster masking process. Double click on the screen to switch to the *All camera view*. Apply the same process for the remaining cameras. The goal of the camera configuration is that no other detections than the truck's reflectors should be visible.
4. If to many detections appear in the camera view, adjust the *Exposure Time* which is found on the right side bar in the *2D-view*.

4.3.2 *Calibration of the positioning system in Visionen*

In this section the guide on how to perform the calibration of the positioning system i Visionen is presented. To be able to do this a L-shaped tool with reflective balls attached are used to define the x-, y- and z-axis. The longer arm of the L-shaped tool is the x-axis. A calibration wand will be used as well to be able to perform the calibration. The calibration needs to be performed each time a new project is created. It might be good to redo these steps from time to time, especially if the accuracy is decreasing.

1. Start by placing the L-shaped tool, shown in Figure 7 where the origin should be defined and with the desired orientation of the coordinate system.

2. Make sure that the wand is assembled as seen in Figure 6.
3. Press thw *Start Calibration* button in the measurement window.
4. The user should now be in the calibration window according to Figure 8. Set the *Calibration time* to 240 seconds.
5. Select *Options*, now the user should be in the Calibration Option Window. Then select *Wand calibration* and set the exact wand length to 601.9mm and set the *Maximum number of frames used as calibration input* to 6000 as shown in Figure 9.
6. Select *3D Tracking* in the left-hand menu.
7. In the 3D-tracking window, redefine the upper limit of the bounding box's z-direction to 1000mm for the 3D data as seen in Figure 10.
8. Start the calibration by pressing the calibration button, choosing a calibration time and press *OK*.
9. During the calibration make sure to walk slowly around the area with the wand. Slowly make large movements with the wand or rotate it on its own axis. Make sure that all possible combinations of the pose are covered. When the calibration is done a dashed line will be shown instead of these numerical values, thus one person can perform the calibration without someone needing to inform when the calibration is done from the QualiSys computer.
10. When the calibration is done it will either pass or fail. If the calibration passes, continue to Section 4.3.3 and redifine the bodies of the Lego truck. If the calibration fails then redo steps 1-9 and make sure that everything is done right, most likely there was a wand movement that was insufficient and not all possible pose combinations were covered.

Figure 6: The calibration wand.

Figure 7: The L-Shaped tool used for calibration.

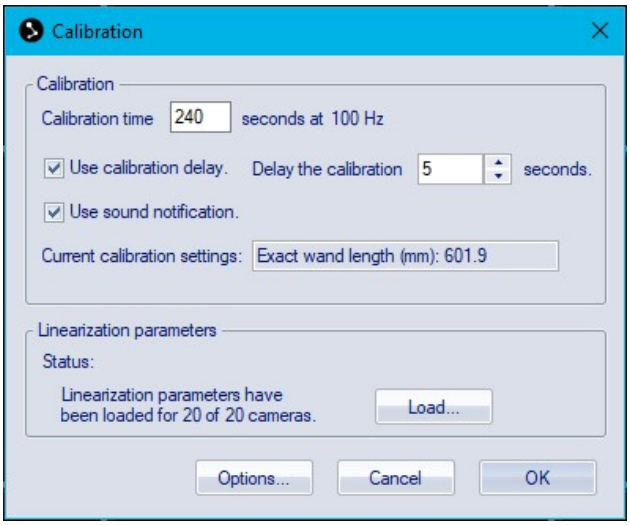

Figure 8: Calibration menu

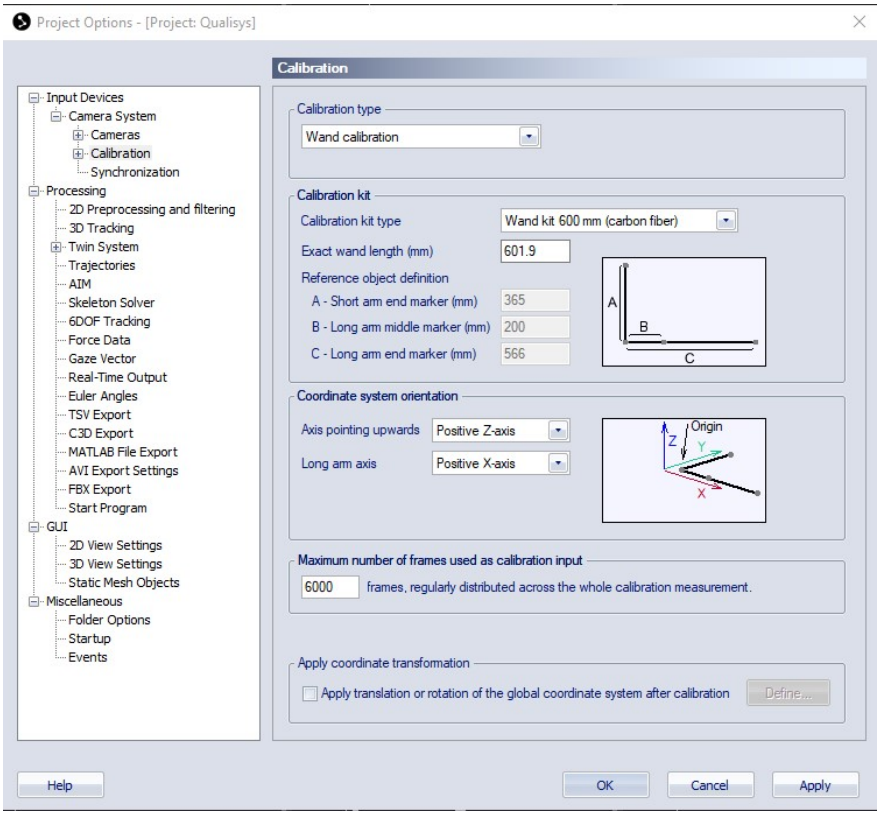

Figure 9: Calibration options

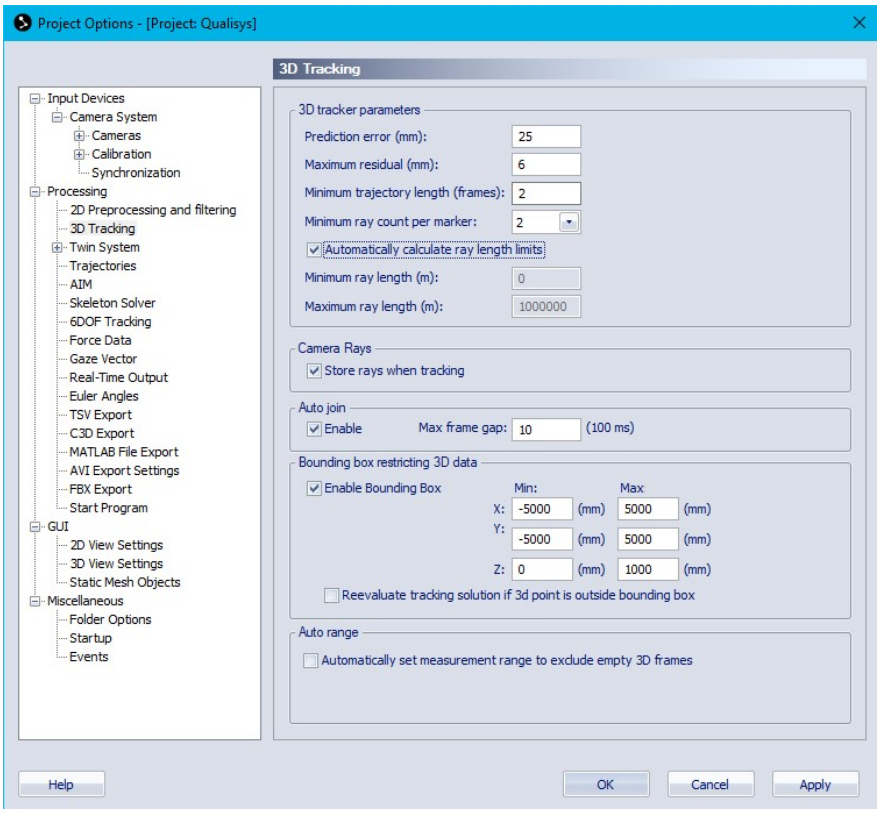

Figure 10: 3D Tracking options

4.3.3 *Redefining the bodies*

This section describes how the Lego trucks rigid bodies, consisting of truck, dolly and trailer, are defined. This part must be done every time QualiSys is setup since the positioning system is millimeter precise and are therefor very sensitive to any potential changes of the reflective balls position on the rigid bodies. Below are the steps needed to redefine the rigid bodies. This can be found in previous user manuals seen in [3].

1. Start in the measuring window seen in Figure 11, open Project option using the shortcut *ctrl* + ,.
2. In the left-hand menu select the *6DOF Tracking*.
3. The user should now be in the 6DOF tracking window as seen in Figure 11. Remove all the predefined bodies.
4. Place the first body that is to be defined, e.g. the truck, with the rear axle placed directly over the origin with the orientation along the x-axis.
5. Switch to *3D-View* from the measurement window. The user should now be able to see the reflective balls of the Lego truck. Mark all the points corresponding to the body defined, pressing shift + dragging the left mouse button will then select multiple points at the same time. Then right click on one of the points and select *Define a rigid body* → *Current frame* and name the body "truck", "dolly" or "trailer" depending on the body being defined.

6. Open the *6DOF tracking* window again by pressing *ctrl* + *,*.Mark the newly defined body, e.g. "truck", and press *Add point*. The added point will now represent the body's origin.

7. Check the *Virtual* box for the new point.

8. While observing the body in *3D-View* gradually change the coordinates for the virtual point until it is right above the global origin, that is pierced by the z-axis for the global coordinate system. The virtual point of the trailer will be marked (white). When the placement of the virtual point is satisfactory the coordinates could be something similarly to Figure 12.

9. Select the defined body and press *Translate*.

10. In the translation window that can be seen in Figure 13 select *To point **n** in the body*, where **n** should be chosen as the number corresponding to the virtual point of the body, press *OK*. Now the origin of the defined body is placed above its rear axle.

11. Back at the 6DOF tracking window select the defined body and press *Reset Rotation* if it is possible.

12. Repeat step 1-11 for the remaining bodies.

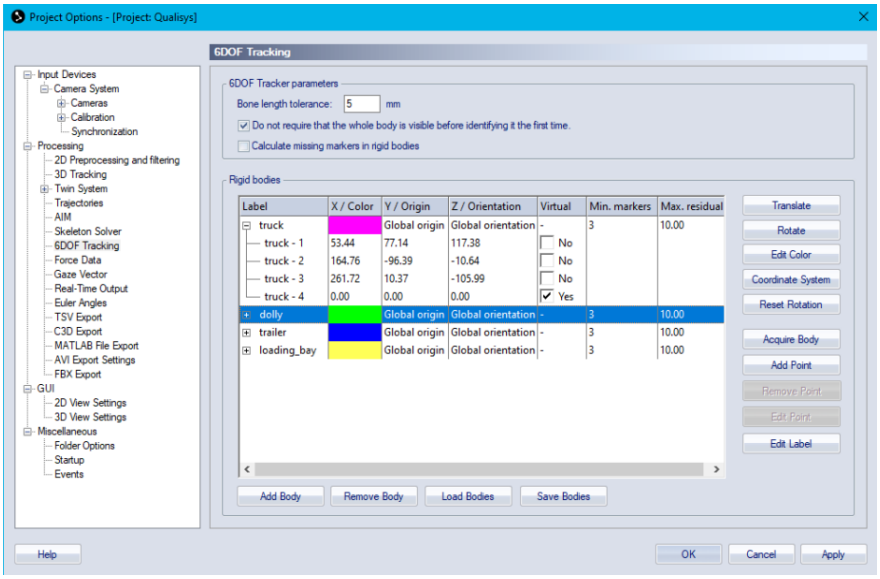

Figure 11: 6DOF Tracking menu

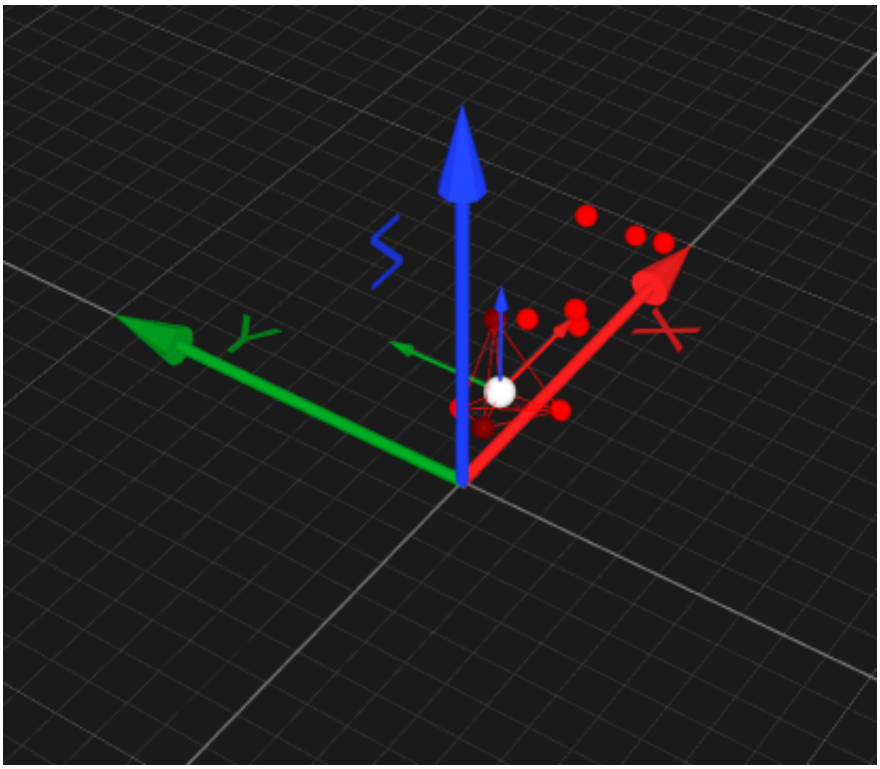

Figure 12: How the defined bodies may look like i 3D-View, with the virtual point marked.

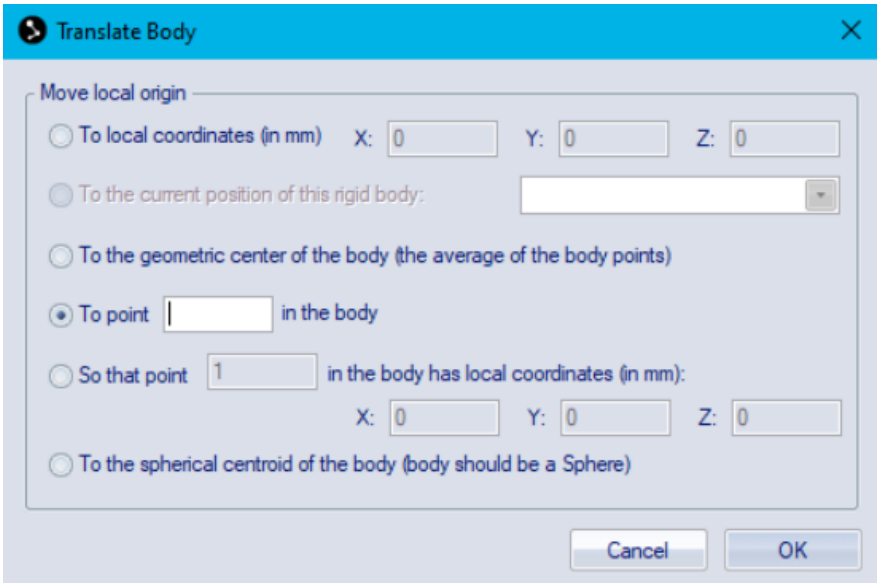

Figure 13: Translate window.

4.4 Trouble Shooting

4.4.1 *No cameras or only a few are showing*

Shut down the *Qualisys Track Manager*, wait for a couple of minutes and restart the program, 20 cameras should be visible after restart. If the problem remains and there are any of the cameras that for some reason are not working, e.g. if there are too high light exposure, exclude them from the performed linearization. This can be done using the following steps:

1. Open the *Project Options* through the shortcut
2. The user should now be in the *Project Options* window. In the left-handed-menu select: *Input Devices* → *Camera system* → *Cameras* → *Linearization*.
3. Deselect the cameras that are not functioning and press *Apply*.

4.4.2 *Error(s) when starting realtime*

Exit Qualisys Track Manager and then restart the cameras using the camera controller.

4.4.3 *Calibration fail*

The L-frame should be visible for all cameras which is obtained by adjusting the exposure time until the reflectors on the L-frame are visible. Make sure that the person using the calibration wand is not using any reflectors, dark clothes are recommended.

4.5 Setup of the Linux Computer

The following steps has to be done to setup the Linux computer:

1. Make sure that all the software has been installed from Section 3.
2. Connect to the network called *Visionen*.
3. Check the IP adress by running the command "hostname -I" in the terminal, it should be 192.168.50.xxx. The last three digits vary, but make sure that the other digits should match.

4.6 Setup of the truck

To setup the truck the follwing steps needs to be done

1. The EV3-unit needs to be equipped with fully charged batteries.
2. There needs to be weights placed over the rear axle of the truck and front axle of the trailer to avoid slip.
3. The driving motor should be connected to port C on the EV3-unit.
4. Make sure that the steering motor is connected to port B on the EV3-unit.

5. Make sure that the angle sensor measuring the angle between the truck and the dolly is connected to port 1 on the EV3-unit.
6. Make sure that the angle sensor measuring the angle between the dolly and trailer is connected to port 2 on the EV3-unit.
7. To start the EV3, press the middle button on the unit. The system is ready when the light glows green.

4.7 Starting the system

The next step is to start the Lego truck and the trailer tracking system. To start the system in Visionen, follow the User manual written in 2019 that can be found in [4].

REFERENCES

[1] ROS. (2022) Ros. [Online]. Available: http://wiki.ros.org/Documentation

[2] acados. (2022) acados. [Online]. Available: https://docs.acados.org/

[3] P. Antonsson, E. Bourelius, G. Erbing, J. Gustafsson, A. Holgersson, O. Ismail, F. Jussila, P. Liljeström, K. Rajala, D. Salomonsson, and V. Uvesten. (2020) User manual autonomous truck with a trailer. [Online]. Available: http://www.isy.liu.se/edu/projekt/tsrt10/2020/rev_truck/images/documents/user_manual.pdf

[4] T. Fridén, L. Junler, A. Källström, O. L. Jonsson, T. Nyberg, and T. Westny. (2019) User manual autonomous reversing truck. [Online]. Available: http://www.isy.liu.se/edu/projekt/tsrt10/2019/rev_truck/documents/usermanual.pdf