Test Protocol

December 2021

Version 1.0

Status

Reviewed	Ekström, Viktor	26-11-2021
Approved		

AGV Control Machine Learning
carhe007@student.liu.se

Project Identity

Group E-mail: carhe007@student.liu.se

Homepage: https://tsrt10.gitlab-pages.liu.se/2021/toyota/

Orderer: Anton Kullberg
Phone: Not applicable
E-mail: anton.kullberg@liu.se

Customer: Oskar Bergkvist (Toyota Material Handling)
Phone: Not applicable
E-mail: oskar.bergkvist@toyota-industries.eu

Supervisor: Hamed Haghshenas
Phone: Not applicable
E-mail: hamed.haghshenas@liu.se

Course Responsible: Daniel Axehill
Phone: Not applicable
E-mail: daniel.axehill@liu.se

Project members

Name	Post	E-mail
Carl-Hampus Hedén	Project manager	carhe007@student.liu.se
Mahdi Najafi	-	mahna987@student.liu.se
Alfred Boman	Head of design	alfbo741@student.liu.se
Adam Kagebeck	-	adaka206@student.liu.se
Kalle Blomkvist	Head of software	karla625@student.liu.se
Rasmus Björk	Head of documentation	rasbj268@student.liu.se
Viktor Ekström	Head of testing	vikek514@student.liu.se

DOCUMENT HISTORY

Version	Date	Changes	Made By	Reviewed
0.1	26-11-2021	Draft	All	Viktor Ekström
1.0	14-12-2021	Final version	All	Mahdi Najafi

carhe007@student.liu.se

1 INTRODUCTION

This document is the test protocol for the project *AGV Control Optimization with Machine Learning*. The tests are based on the previously documented test plan [1] and requirements specification [2]. The tests are divided into parts of general tests, simulator tests and auto-tuner tests. For each test the execution, test date, corresponding requirements and criteria for success is described as well the person that executed the test. A requirement denoted with, *, corresponds to it being an extra requirement that did not pass the test. The tests are commented below each table.

2 GENERAL TESTS

Table 1: Tests for the general system

Test name (Executed by)	Execution	Criteria	Status (Date)	Requirement(s)
Performance (Adam Kagebeck)	Run a completed scenario and view the results.	Fully learnt parameters constructed in less than 1 hour, produces a path with less error than the performance of the controller in the thesis [3]	Failed (26-11-2021)	24, 25
Maximum positioning error (Viktor Ekström)	Run the system and save the state and reference data	The maximum error should not exceed 20 mm	Passed (25-11-2021)	26
Maximum orientation error (Viktor Ekström)	Run the system and save the state and reference data	The maximum error should not exceed 2.5°	Passed (25-11-2021)	27
Visualize data (Adam Kagebeck)	View the data, either from the GUI or from a file	If the data can be viewed, the test is complete	Passed (24-11-2021)	5
Number of auto-tuners (Viktor Ekström)	Count the number of auto-tuner implementations	The number of auto-tuners shall exceed or equal to two	Passed (24-11-2021)	2

2.1 Results and conclusions from general tests

2.1.1 *Performance*

The simulation can be ran with both of the different auto-tuners. It is possible to train the agent so that the Maximum positioning error and the Maximum heading error are within accepted range in less than one hour. However, since it is hard to compare the performance of the controller in the thesis with the auto-controller, we can not say that the performance of the auto-tuner is better. The performance can be seen in Figures 1 and 2 below.

2.1.2 *Maximum positioning error*

The test where preformed with the DDPG auto-tuner and a reference velocity of 1 m/s. The positioning error is within 2 cm so the requirements of the test is considered to be a success. Figure 1 presents the results of the test.

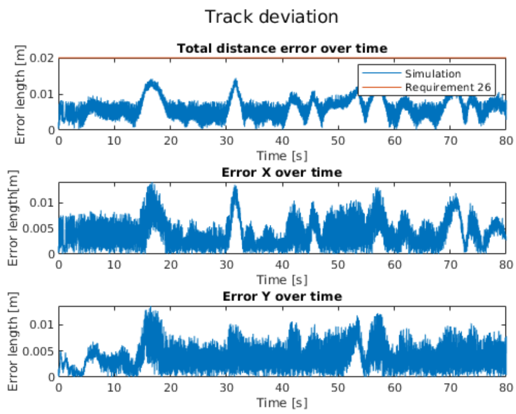

Figure 1: The Figure shows the results from the test.

2.1.3 *Maximum orientation error*

The test where preformed with the DDPG auto-tuner and a reference velocity of 1 m/s. The orientation error is within 2.5 degrees so the requirements of the test are considered as achieved. Since every path is different the test only passes when the initial condition of the heading is set according to the specific path. The left graph in Figure 2 presents the results of the passed test with an initial condition corresponding to the path while the right graph shows the performance when the initial heading of the AGV differs from the path's. From the Figure one can conclude that after the initial error it yields the same result.

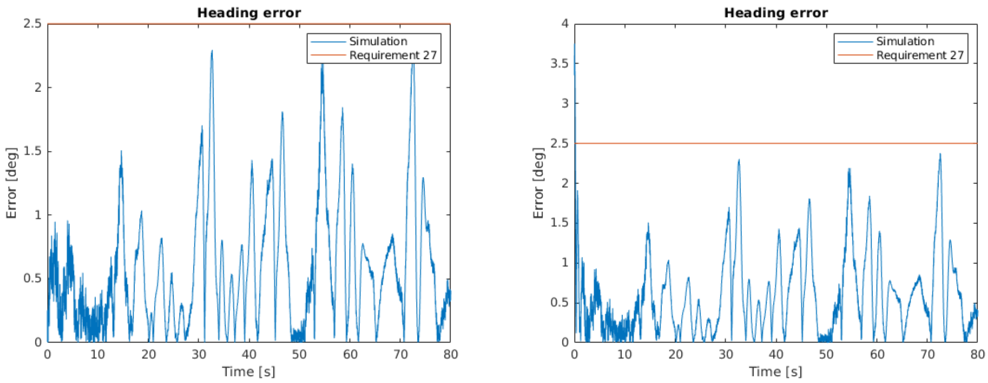

Figure 2: Orientation error with a initial heading according to the path to the left and orientation error with initial heading set to zero.

2.1.4 *Visualize data*

The data are able to be visualize through the GUI. The results can be viewed by plots of the path, heading error, velocity error, position errors (in x and y coordinates) and distance error. Figure 3 show the results from a test where the AGV drives along a path.

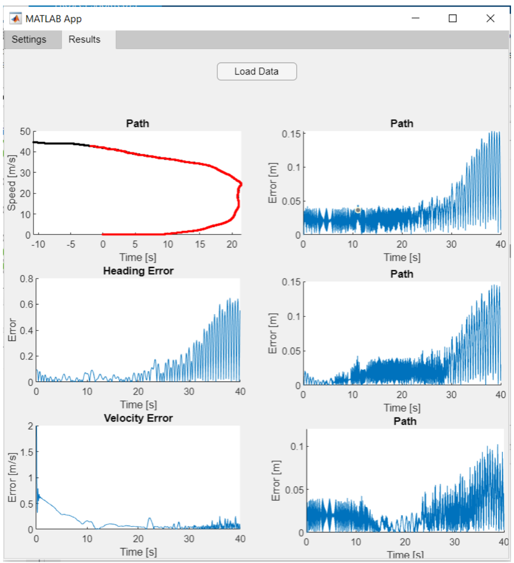

Figure 3: Illustration of how the data from a simulation can be visualized by the GUI.

2.1.5 *Number of auto-tuners*

Since the test is to count the number of auto-tuners, there is no conclusions to make other than that there are two different auto-tuners in the system.

3 SIMULATOR TESTS

Table 2: Tests for the simulator

Test name (Executed by)	Execution	Criteria	Status (Date)	Requirement(s)
Kinematic model test (Adam Kagebeck)	The kinematic model will be simulated with the provided data from Toyota	If the output from the kinematic model is similar to the real output, the model will be deemed satisfactory	Passed (02-11-2021)	11, 12, 19
Disturbances (Kalle Blomkvist)	The various disturbances are plotted in MATLAB against the undisturbed data	The test is OK if the effects on the data from the disturbances can be viewed as a contrast to the undisturbed data	Passed (16-11-2021)	13, 14, 15, 16*, 17*
Simulator outputs (Adam Kagebeck)	If the simulator can supply the user with track deviation and the time required for the ML to learn	The simulator can supply the given data	Passed (26-11-2021)	18, 20
Learning visualization (Adam Kagebeck)	Run a ML-instance and watch if the learning-process is displayed	All the iterative learning processes can be displayed: graphically with the path, the time taken to learn the process, and the changing control parameters	Passed (26-11-2021)	1, 21, 22, 23
Programming language (Adam Kagebeck)	Count the number of occurrences of each programming language	If the majority of the files are written in MATLAB or Python, the test is successful	Passed (24-11-2021)	7

3.1 Results and conclusions from simulator tests

3.1.1 *Kinematic model test*

The kinematic model is able to repeat the behavior of the AGV. The results from the test are presented in Figure 4. As seen in Figure 4, the positions are not exactly as the real AGV but the results are considered to be good enough.

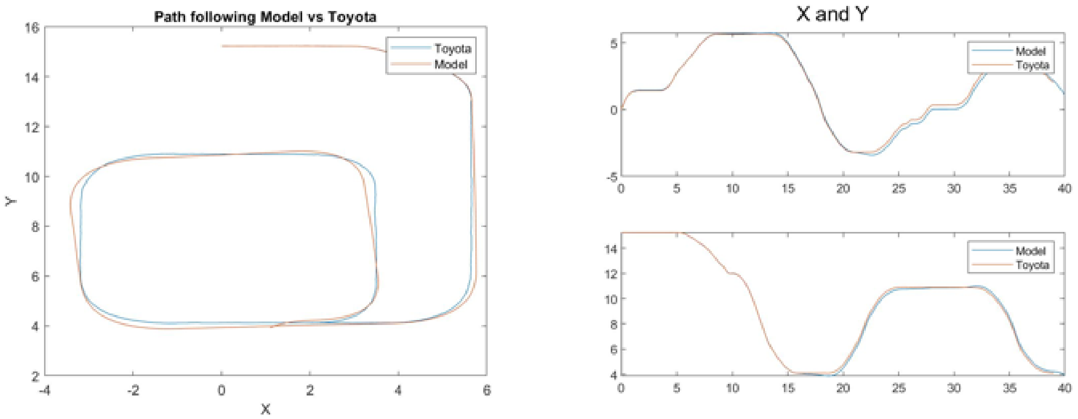

Figure 4: Comparison between the position of the real AGV and the simulated AGV using the same input signals.

3.1.2 *Disturbances*

The disturbances are implemented and able to affect the measurements and dynamic in the system. Figures 5, 6 and 7 show the results of the measurements, with and without the disturbances.

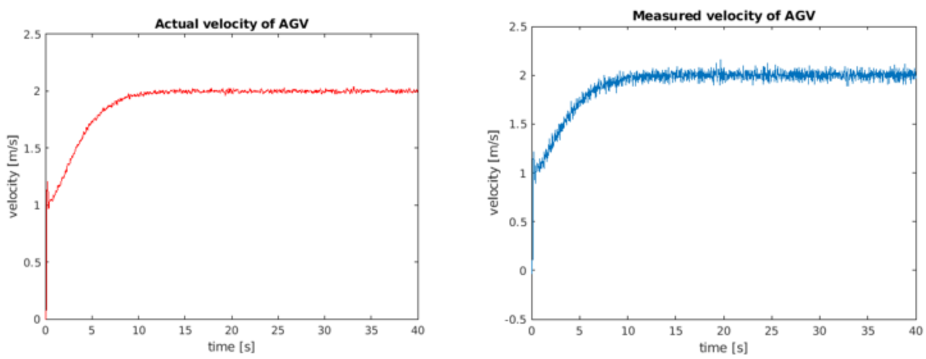

Figure 5: The left plot shows the actual velocity of the AGV and the right plot shows the measured velocity of the AGV during the test.

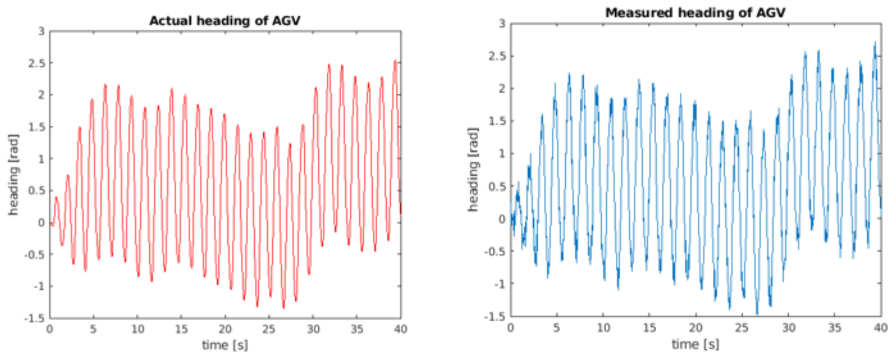

Figure 6: The left plot shows the actual heading of the AGV and the right plot shows the measured heading of the AGV during the test.

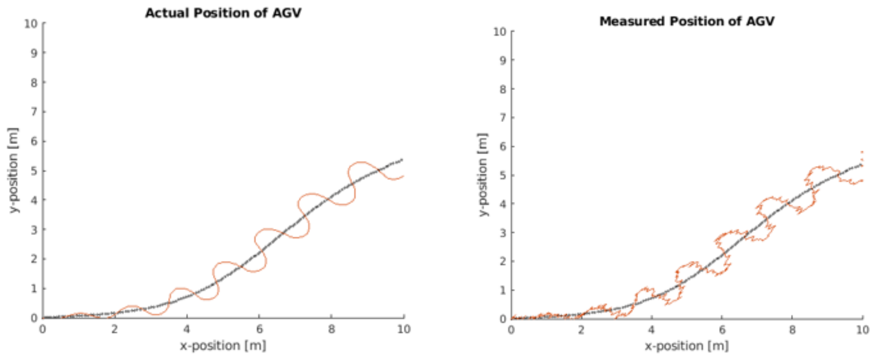

Figure 7: The left plot shows the actual x-position of the AGV against the measured x-position of the AGV during the test. The right plot shows the actual y-position of the AGV against the measured y-position of the AGV during the test.

3.1.3 *Simulator outputs*

The GUI provides an over view of the training process and displays the path graphically and the training time. The results of the test are viewed in Figures 8 and 9.

3.1.4 *Learning visualization*

The GUI provides an over view of the training process and displays the path graphically for the last episode, and the training time. The GIU also displays the largest position- and heading error for every iteration during the training. The reward for every episode is presented in a graph. The provided data can be seen in Figures 8 and 9.

3.1.5 *Programming language*

All the files are written in either Matlab or Python so the test is considered to be successful.

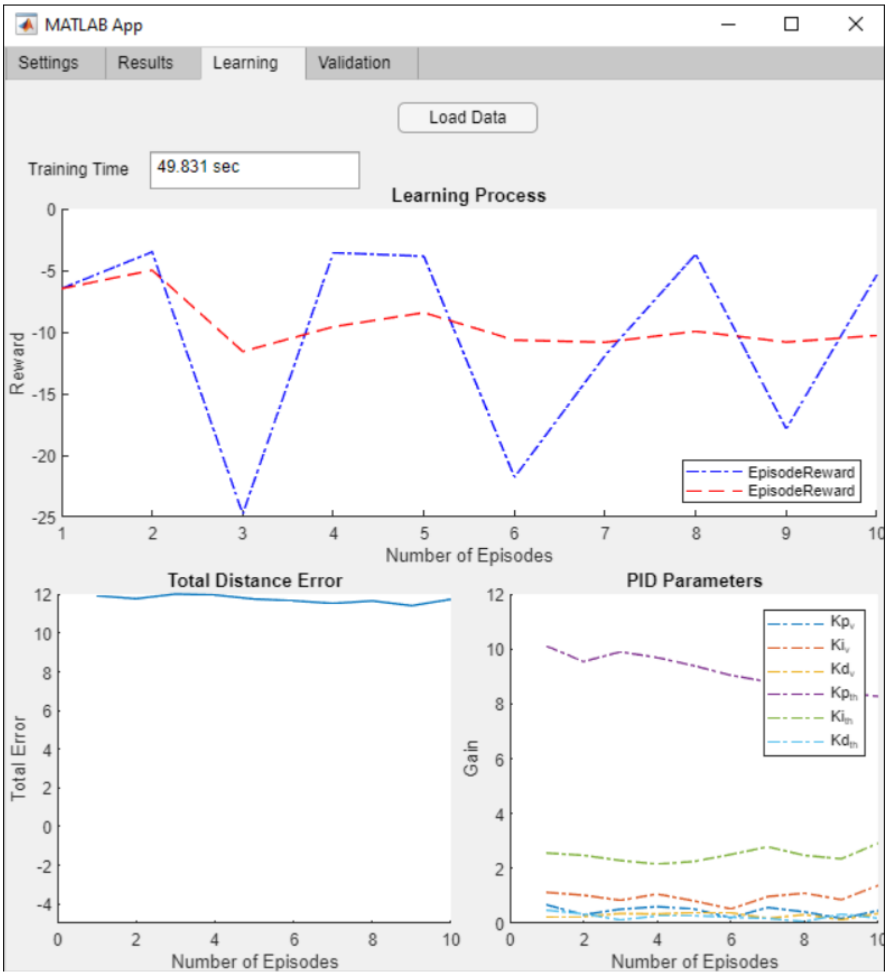

Figure 8: Shows the reward, total distance error and the PID parameters during the training process. The training time can be viewed as well.

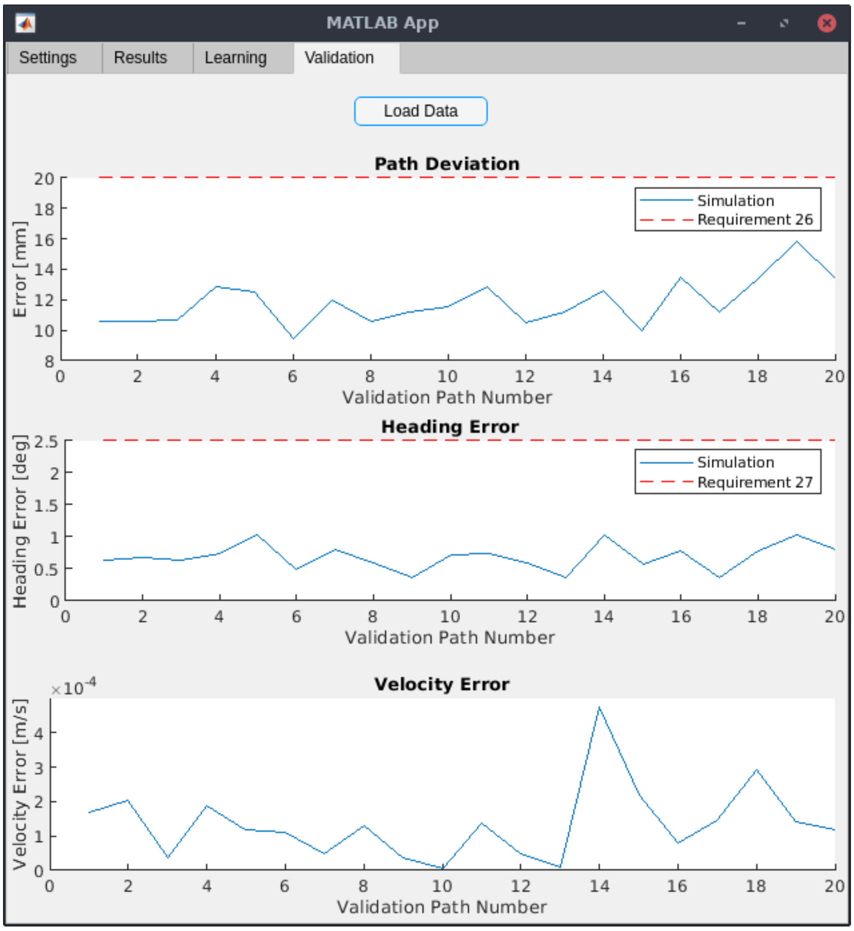

Figure 9: Shows the path deviation, the heading error and the velocity error during the training process

4 AUTO TUNER TESTS

Table 3: Tests for the auto tuner

Test name (Executed by)	Execution	Criteria	Status (Date)	Requirement(s)
Compatibility Test (Viktor Ekström)	Run the complete system with both the auto tuner and the simulator	The test is OK if the system does not crash	Passed (15-11-2021)	6
Programming language (Viktor Ekström)	Locate the files for the auto tuner and count the number of occurrences of each programming language	The test is OK if the majority of the files are written in either MATLAB or Python	Passed (15-11-2021)	7
RL (Viktor Ekström)	Review the implementation of the auto tuner and check if it implements one or more reinforcement learning algorithm	The test is OK if the auto tuner is based on one or more reinforcement learning algorithms	Passed (15-11-2021)	8
ML auto tuner test (Viktor Ekström)	The auto tuner with the ML method is implemented in a simulation environment (not necessarily the AGV environment) and simulated	The test is OK if the auto tuner is able to change the tuning parameters in the controller	Passed (15-11-2021)	3, 7, 8, 9
Different environments (Viktor Ekström)	Run the auto tuner with the simulator using different paths and disturbances	The test is OK if the system runs in the various environments	Passed (24-11-2021)	10

4.1 Results and conclusions from auto tuner tests

4.1.1 *Compatibility Test*

The system can be run with the auto tuner and the simulator without crashing so test is considered to be passed.

4.1.2 *Programming language*

All of the files for the auto tuner were written in MATLAB hence the test is passed.

4.1.3 *RL*

Two methods were implemented for the auto tuner and both of these are based on reinforcement learning algorithms therefore the test passes.

4.1.4 *ML auto tuner test*

Both of the different auto-tuners are able to change the control parameters in the controller during the training process which provides different performances and lead to different rewards. Figures 10 and 11 present the results when both the auto-tuners train the agent and provides different control parameters.

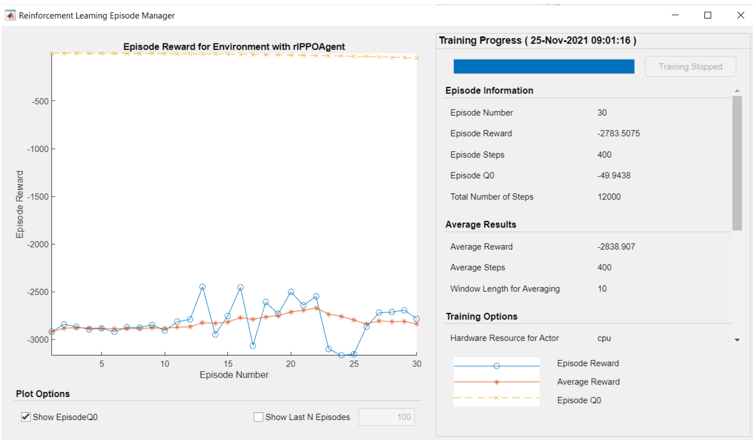

Figure 10: Illustration of the training of the PPO auto-tuner. The Figure shows the different rewards for the episodes during the training process which indicates different control parameters in the different episodes.

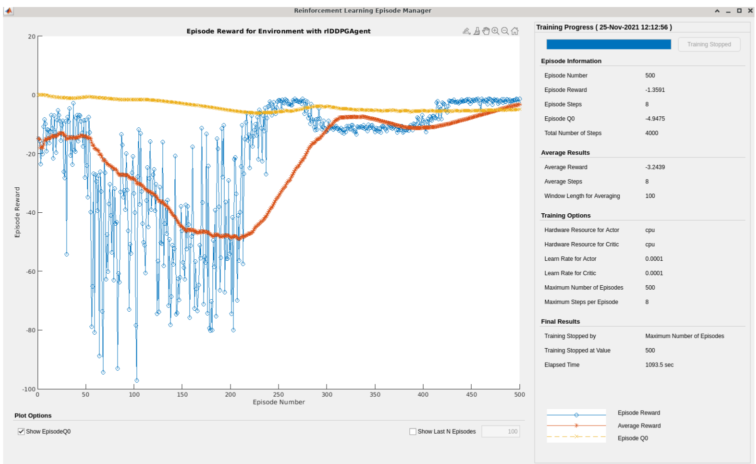

Figure 11: Illustration of the training of the DDPG auto-tuner. The Figure shows the different rewards for the episodes during the training process which indicates different control parameters in the different episodes.

4.1.5 *Different environments*

The system can be run using different environments in the shape of disturbances and paths. For this test 1000 randomly generated paths were used for the training. A randomly selected path is used for each episode and the power of the measurement noise is randomly selected from a uniform distribution hence receiving different power for each episode. The Figures 12 and 13 shows four of the paths that could be randomly selected.

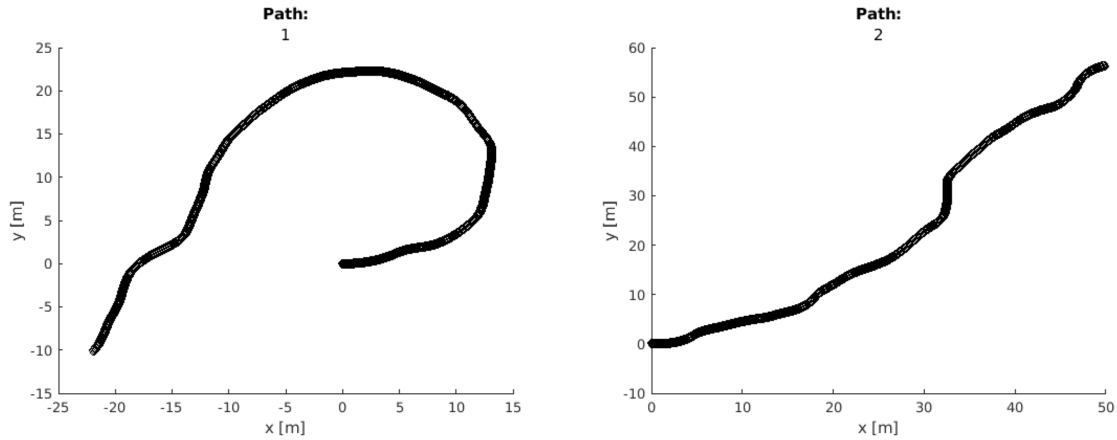

Figure 12: Path 1 and 2 used for training.

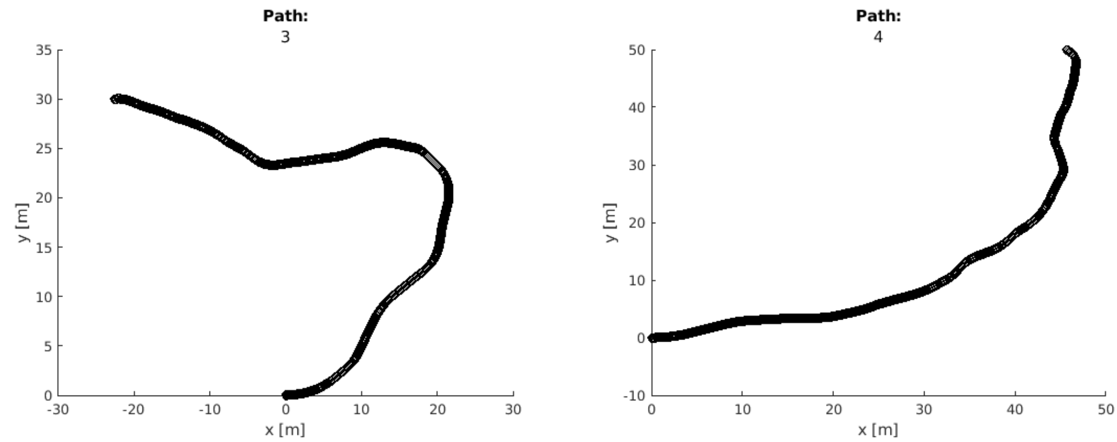

Figure 13: Path 3 and 4 used for training.

REFERENCES

[1] C. H. Heden et al, “Test plan,” Oct 2021.

[2] ——, “Requirement specification,” Sep 2021.

[3] A. Holgersson and J. Gustafsson, “Trajectory tracking for automated guided vehicle,” 2021.