Using proximity sensors for AGV docking – side site case approach

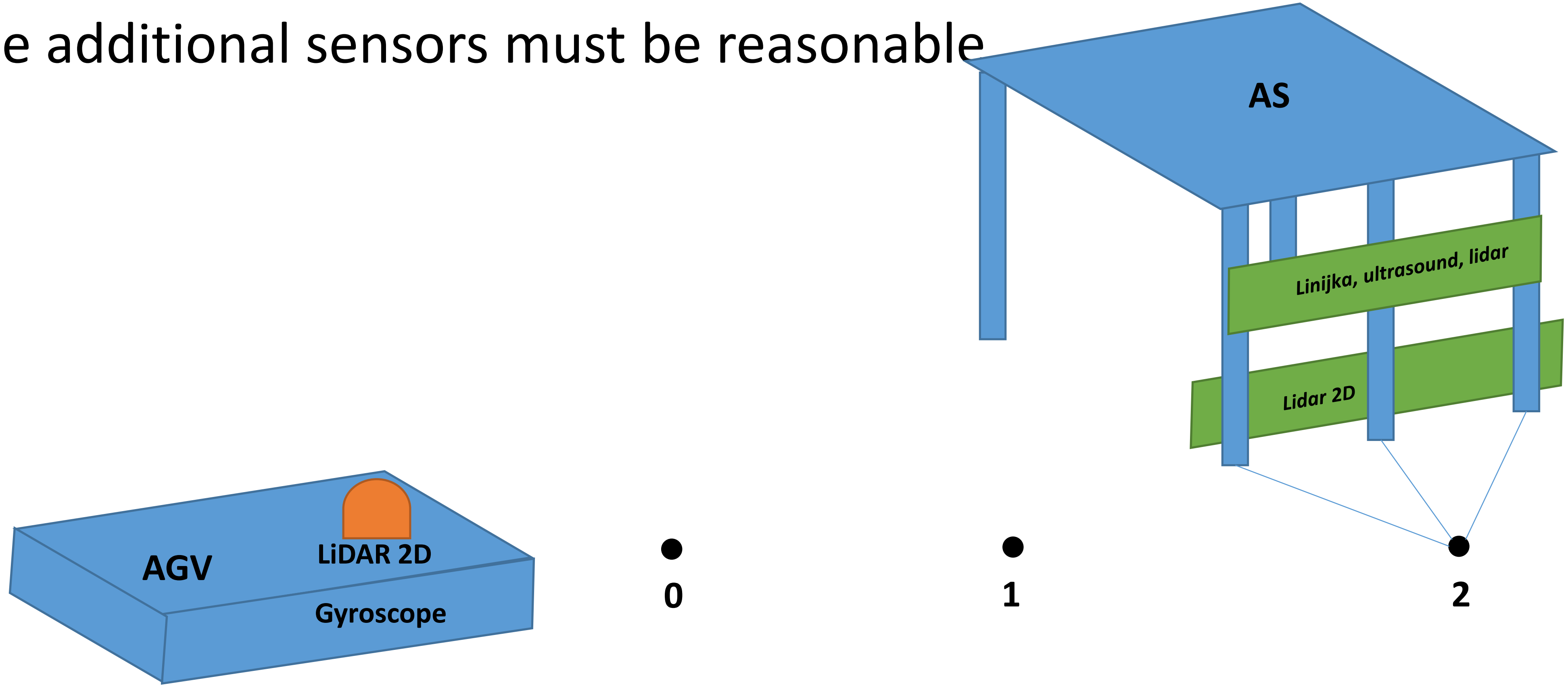
Damian Grzechca

Silesian University of Technology, Gliwice, Poland

Silesian University of Technology

Goals

- Use AGV on the assembly station and collaborate with other autonomous systems like robotic arm
- cost of the additional sensors must be reasonable



Preliminary problems found (may reflect on practical implementation)

Main goal is to get high enough docking accuracy of the AGV

- Communication time delay is crucial and introduce low accuracy on the hardware level.
- Acceptable delay should be less then 20ms. If greater than 100ms than the speed of the AGV must be reduced to 5cm/s.
- Twist and motion accuracy depends on BLDC type, encoders and delay(!): required prediction algorithm or open loop control scheme with calibration.
- Platform cannot use “standard” navigation algorithms due to problem with cost map.
- Conclusion. There are two reasonable solutions:
  - Speed reduction to its very low value and control remotely
  - Pass the control procedure to AGV and keep the constant speed

Docking station with AGV

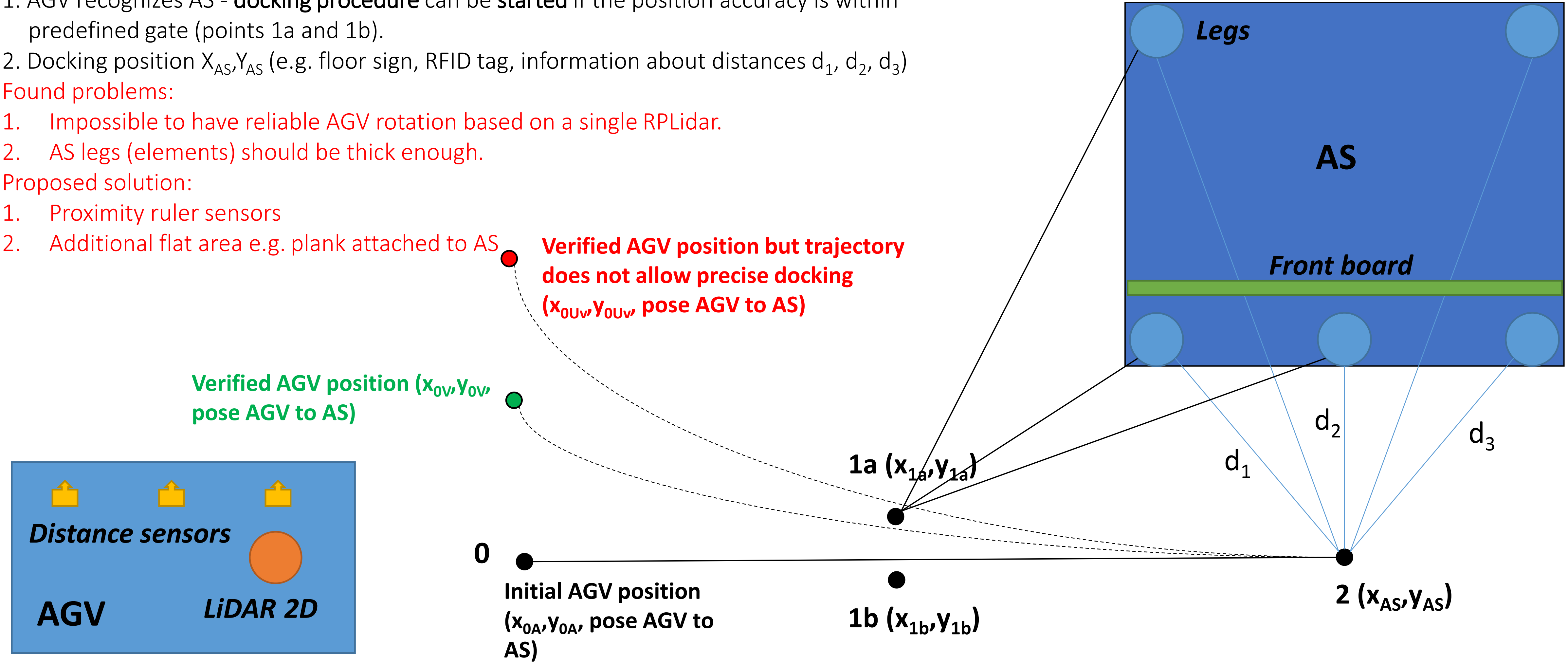
0. AGV ground truth assumption – can be verified by 3rd party devices with limited accuracy! **AS is not visible by AGV by itself.**
1. AGV recognizes AS - **docking procedure** can be **started** if the position accuracy is within predefined gate (points 1a and 1b).
2. Docking position X_{AS},Y_{AS} (e.g. floor sign, RFID tag, information about distances d_1, d_2, d_3)

Found problems:

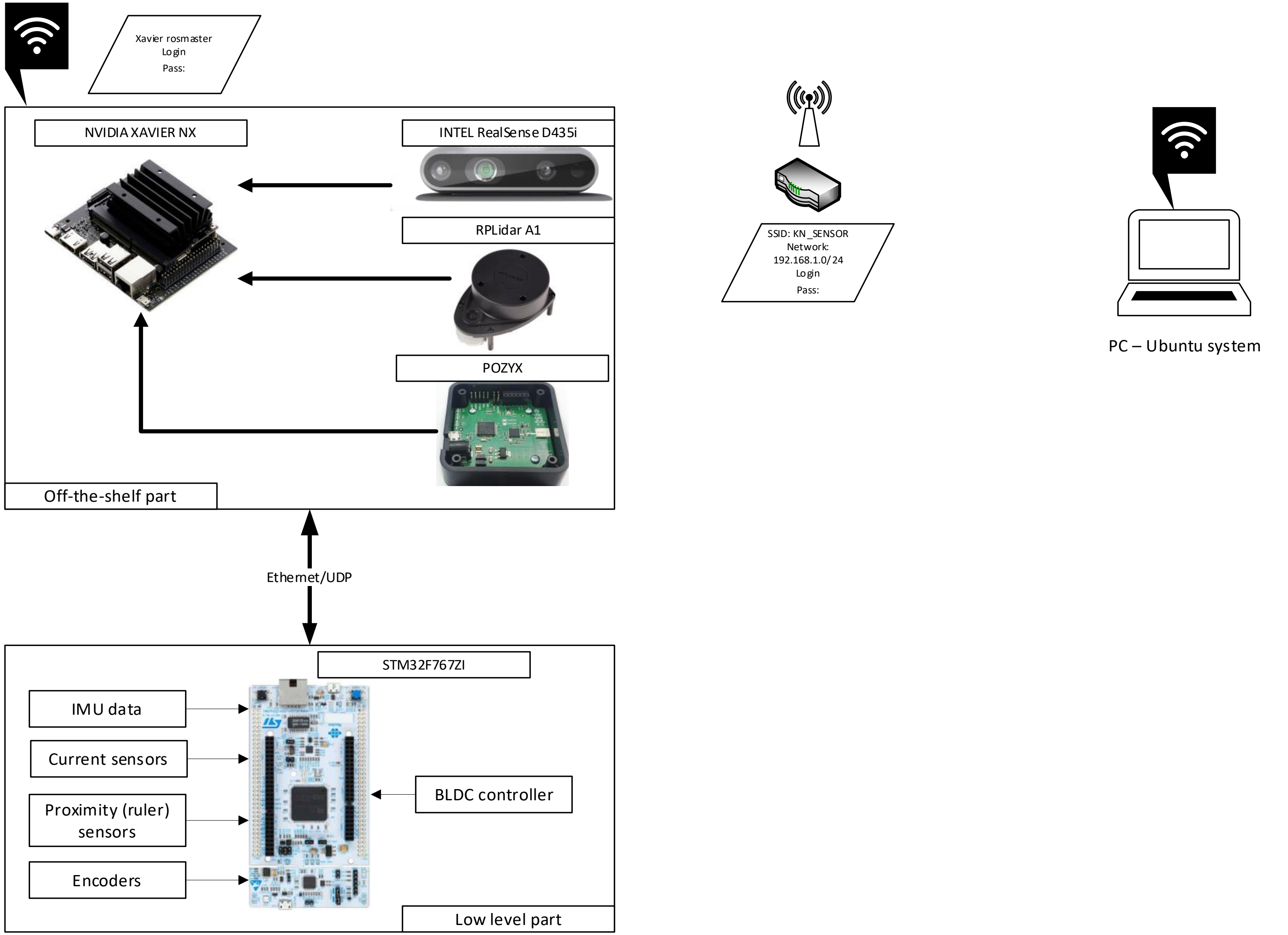
1. Impossible to have reliable AGV rotation based on a single RPLidar.
2. AS legs (elements) should be thick enough.

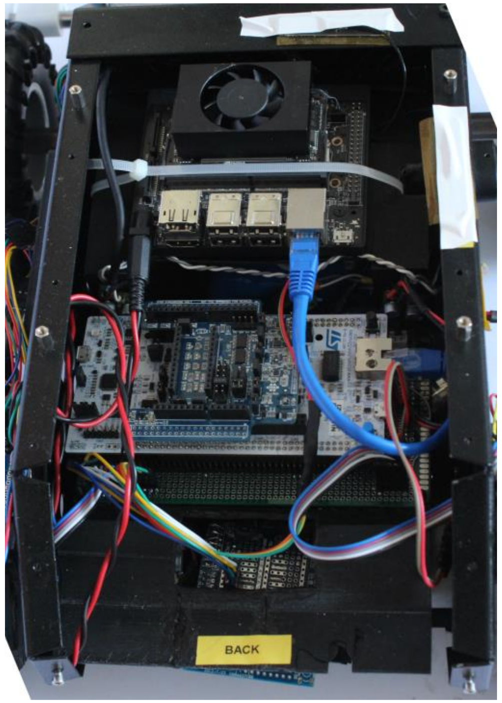
Proposed solution:

1. Proximity ruler sensors
2. Additional flat area e.g. plank attached to AS

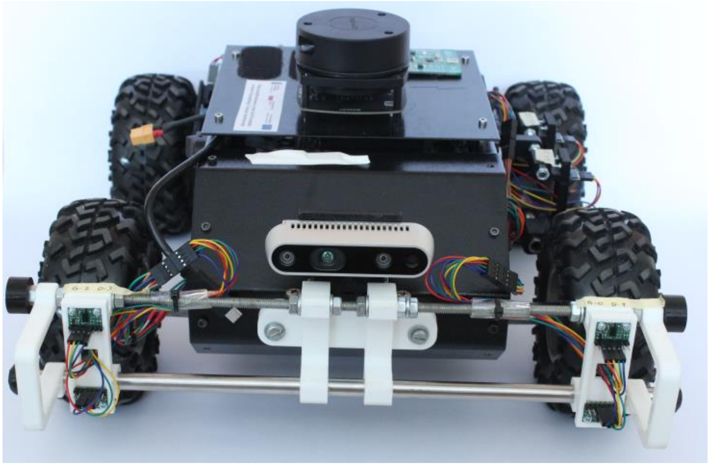


General structure of the AGV test platform v.1



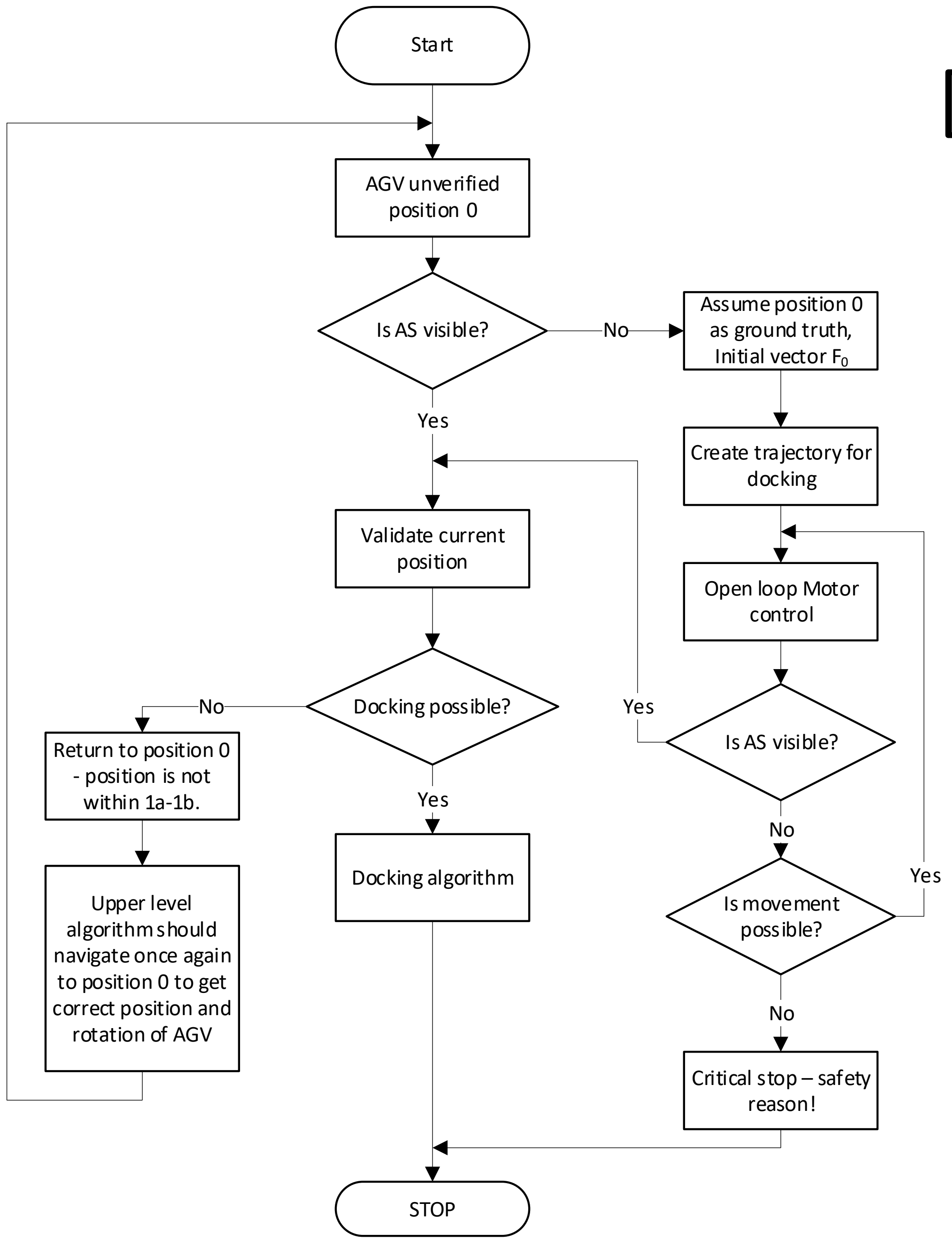


The AGV test platform v.1



- The following sensors have been introduced:
  - 2 BLDC motors
  - Proximity senors (distance rulers: front and side)
  - RPLIDAR A1 – short range lidar – it is the main sensor for navigation purpose
  - Camera Intel Realsense D435i – depth sensor, vision camera, IMU
  - Encoders (odometry)
  - IMU - X-Nucleo-IKS01A3 (yaw, roll and pitch estimation)
  - Current measurement modules – control, safety and reliability of the system

Docking procedure



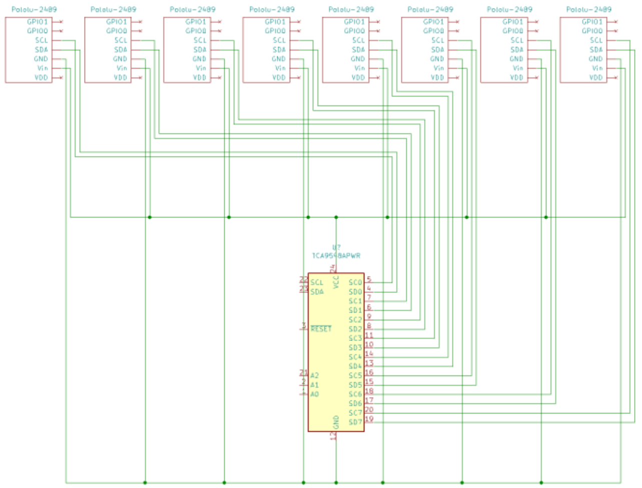
Docking procedure

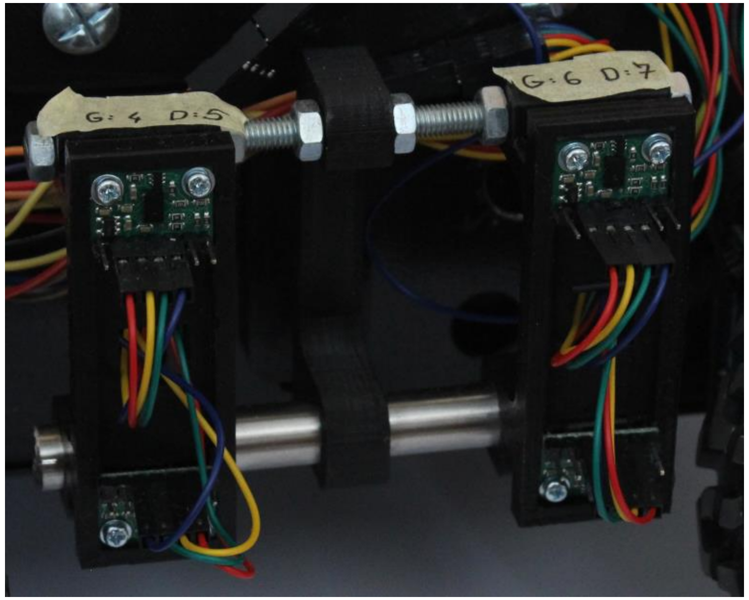
1. AGV position should be verified (usually based on the AS identification)
2. Virtual gate as a decision point: position of the AGV (AS must be visible) – passing the control to AGV
3. Start docking algorithm

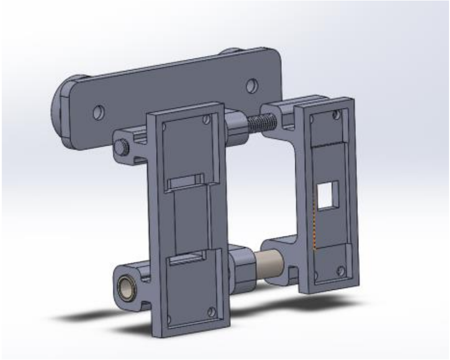
Proximity ruler –

PCB design, mounting rack design and final implementation

Calibration process is mandatory!



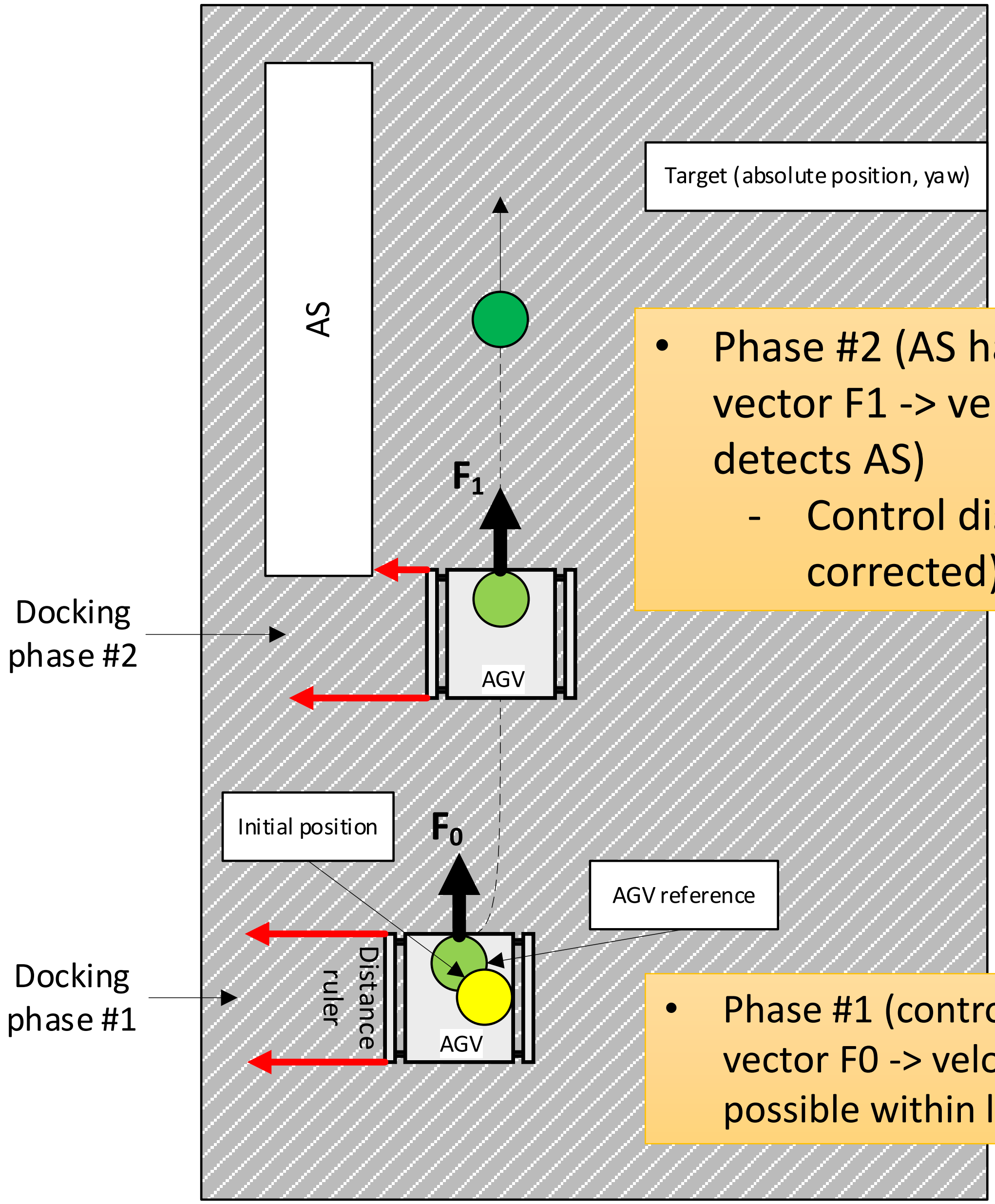




Proximity sensor (side sensor)

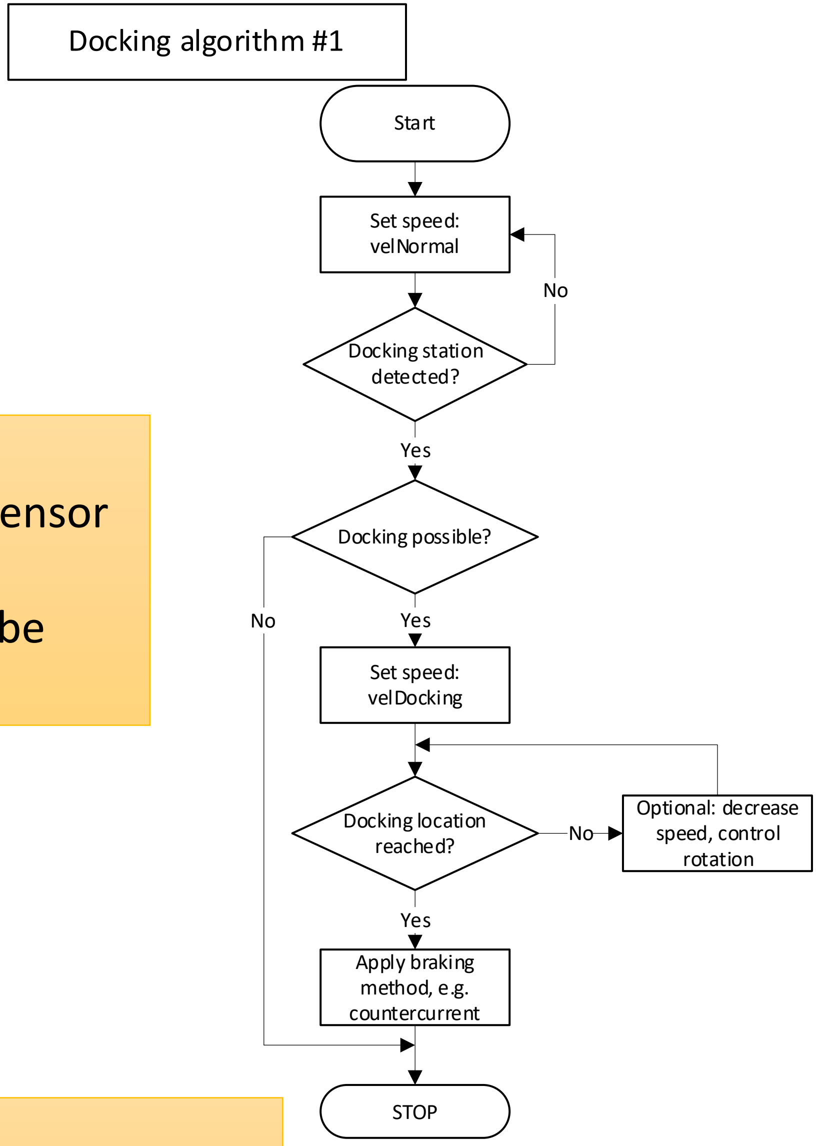
Measured distance [mm]				Reference distance [mm]				Absolute error [mm]			
PS_1	PS_2	PS_3	PS_4	PS_1	PS_2	PS_3	PS_4	PS_1	PS_2	PS_3	PS_4
137	28	185	57	131	27	181	54	-6	-1	-4	-3
135	29	185	56	132	28	183	53	-3	-1	-2	-3
162	97	143	122	158	96	140	119	-5	-1	-3	-3
163	98	145	121	162	97	141	118	-1	-1	-4	-3
164	172	94	186	163	172	93	180	-1	0	-1	-6
165	172	93	187	164	173	93	181	-1	1	0	-6
83	103	54	165	81	102	51	164	-2	-1	-3	-1
80	104	53	164	81	102	51	163	-1	-2	-2	-1
28	53	28	63	27	51	25	59	-1	-2	-3	-4
27	52	29	64	26	51	27	60	-1	-1	-2	-4
							Average error	-1.9	-0.9	-2.4	-3.4

Side Docking Action Server



- Phase #2 (AS has been detected):
  vector F1 -> velocity “Docking” (first proximity sensor detects AS)
  - Control distance using odometry (yaw can be corrected)

- Phase #1 (control is passing to the AGV);
  vector F0 -> velocity “Normal”, yaw and position correction are possible within limited range depending on e.g. Lidar quality



Ground truth reference: Leica Disto X4

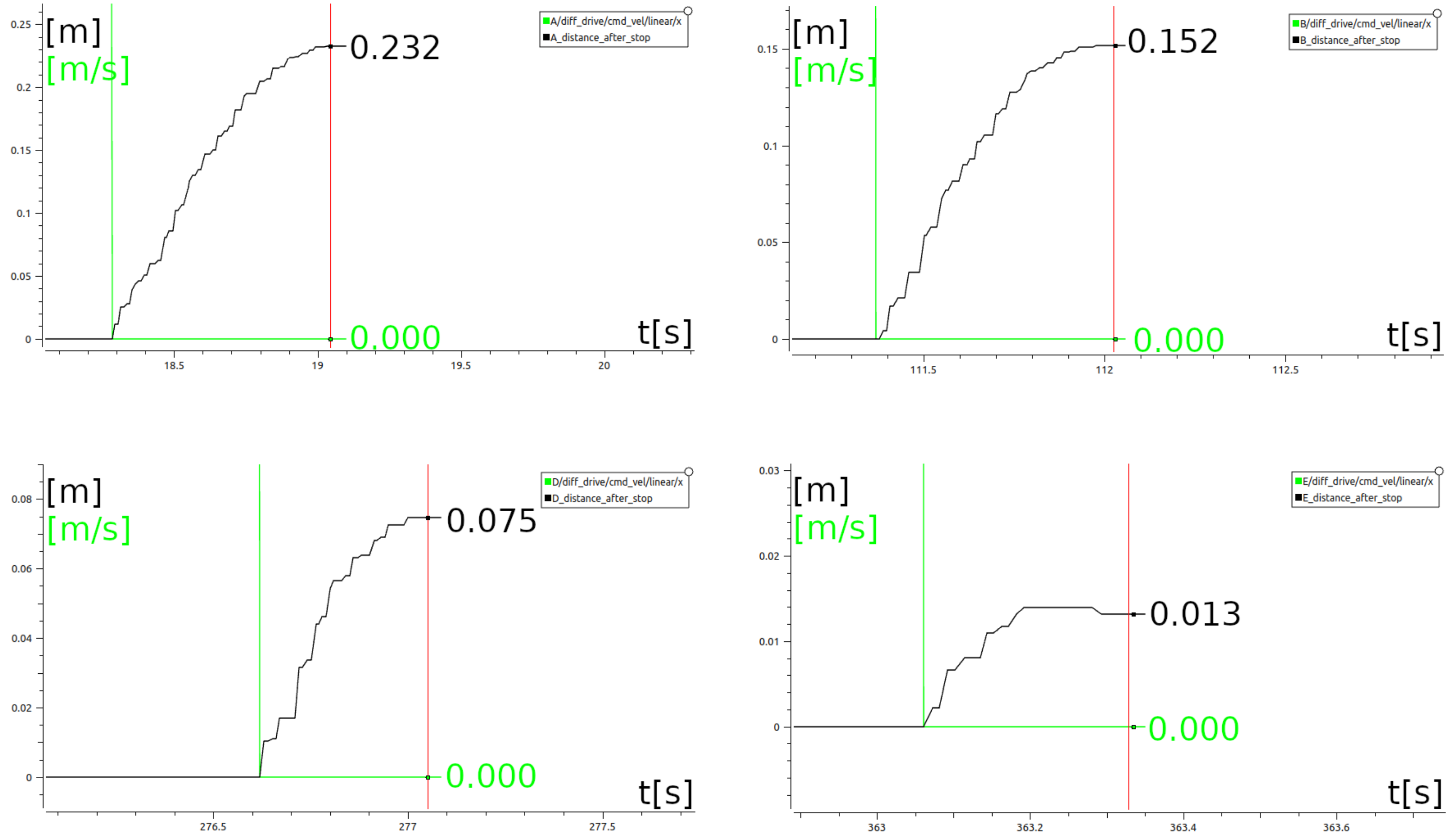
Research – docking scenario #1 (without tracing the length of AS)

- Four measurement series

No	Velocity "normal" [m/s]	Velocity "docking" [m/s]	Braking method
1	0.6	0.6	-
2	0.3	0.3	-
3	0.6	0.6	Countercurrent
4	0.8	0.2	Countercurrent

Initial assumptions: the AGV is paralel to the AS, so we don't need to check the alignment.

No	Velocity “normal” [m/s]	Velocity “docking” [m/s]	Braking method
1	0.6	0.6	-

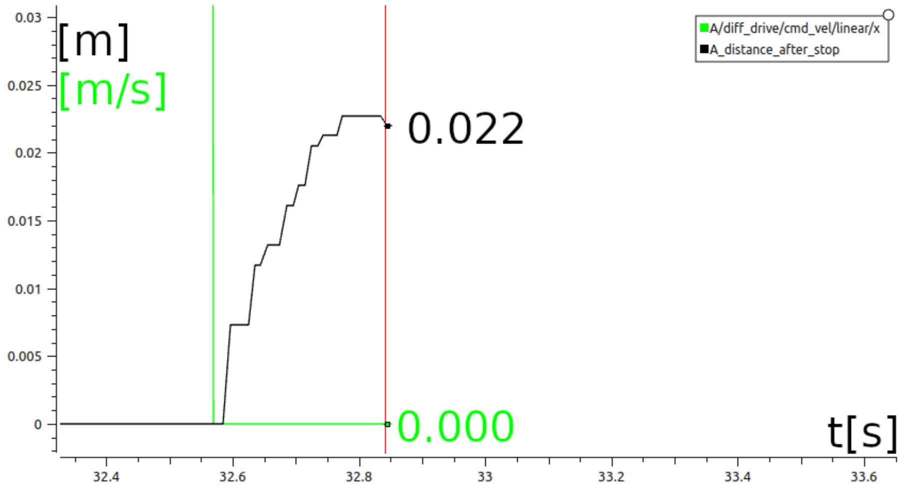


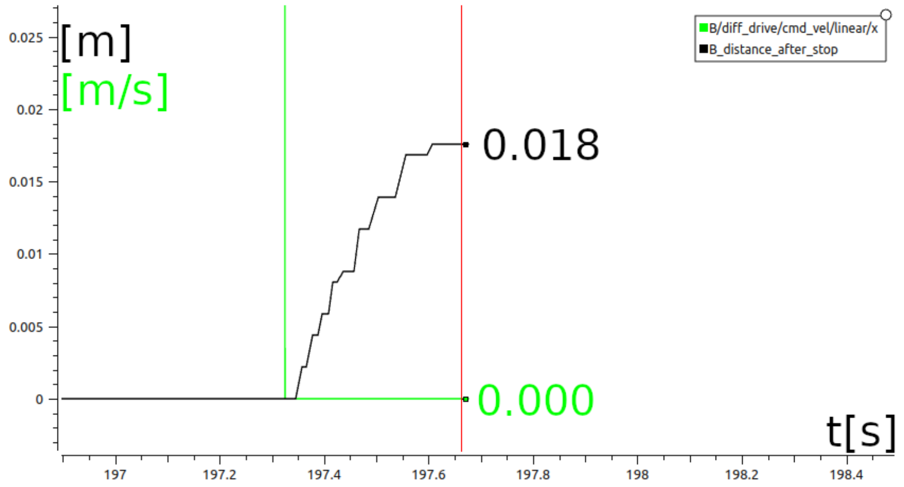
Scenario 1, series 1 – results

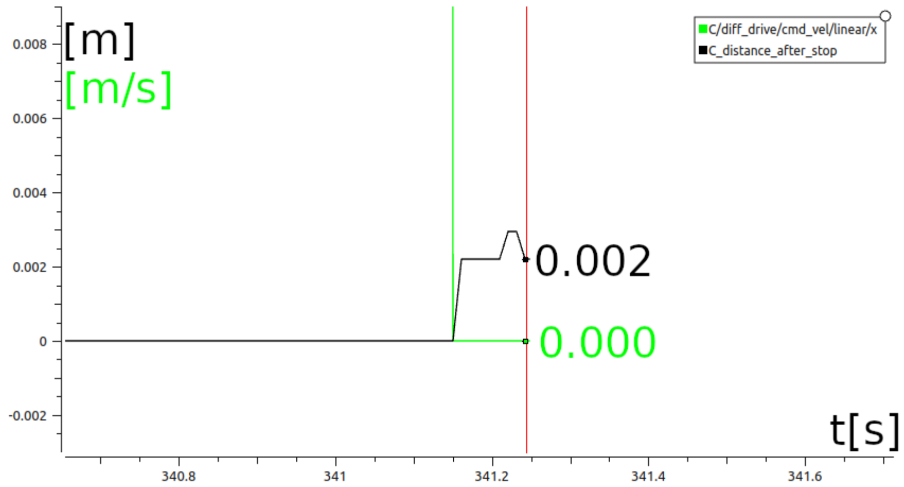
Expected distance is 2.1058m

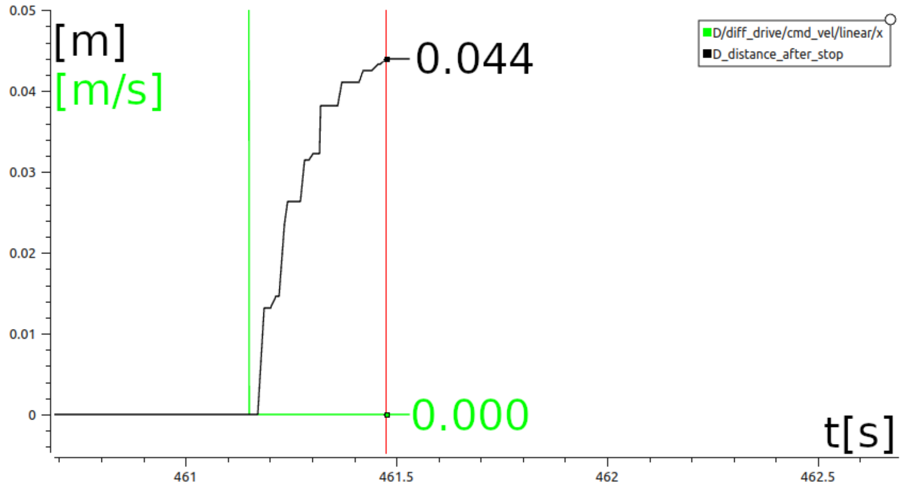
Reference distance (DR) [m]	Travelled distance (DO) [m]	Total absolute error [m]	Compensated absolute error [m]
2.3723	0.232	0.2665	0.0345
2.2911	0.152	0.1853	0.0333
2.3286	0.202	0.2228	0.0208
2.1997	0.075	0.0939	0.0189
2.1414	0.013	0.0356	0.0266
Averaged values			
2.26662	0.01348	0.16082	0.02602

No	Velocity “normal” [m/s]	Velocity “docking” [m/s]	Braking method
2	0.3	0.3	-







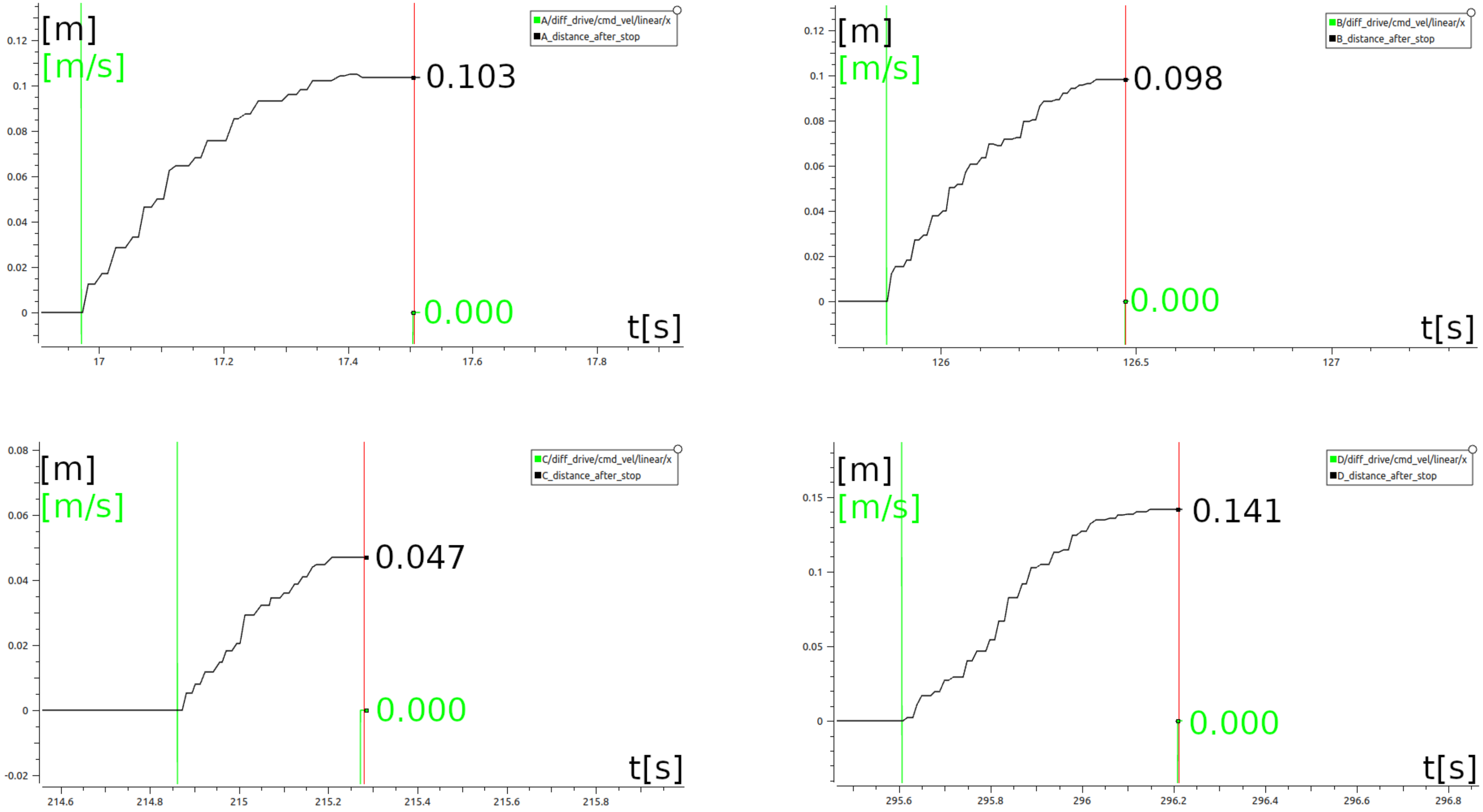


Scenario 1, series 2 – results

Expected distance is 2.1058m

Reference distance (DR) [m]	Travelled distance (DO) [m]	Total absolute error [m]	Compensated absolute error [m]
2.1447	0.220	0.0389	-0.1811
2.1453	0.018	0.0395	0.0215
2.1134	0.002	0.0076	0.0056
2.1678	0.044	0.0620	0.0180
2.1510	0.032	0.0452	0.0132
Averaged values			
2.14444	0.0632	0.03864	-0.02456

No	Velocity "normal" [m/s]	Velocity "docking" [m/s]	Braking method
3	0.6	0.6	Countercurrent

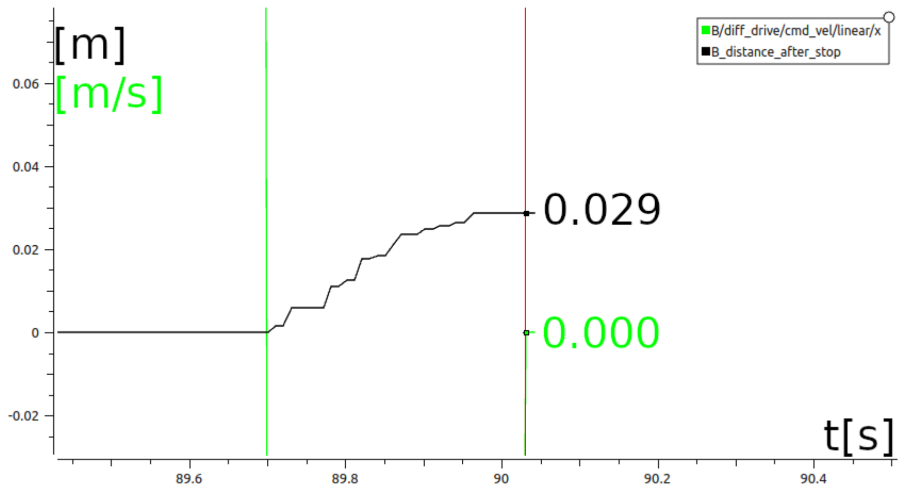


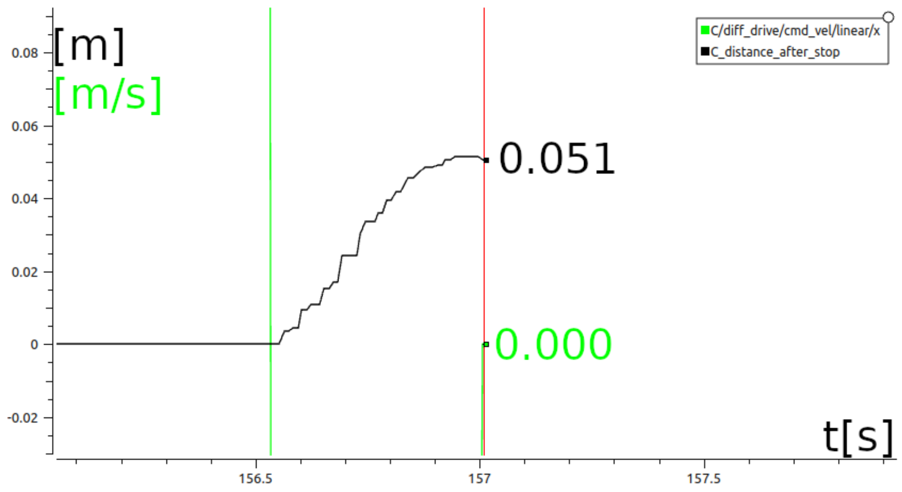
Scenario 1, series 3 – results

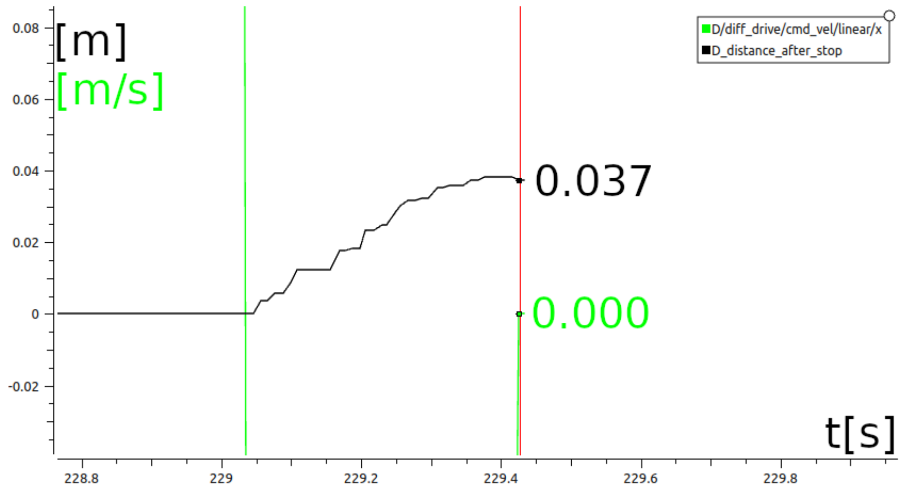
Expected distance is 2.1058m

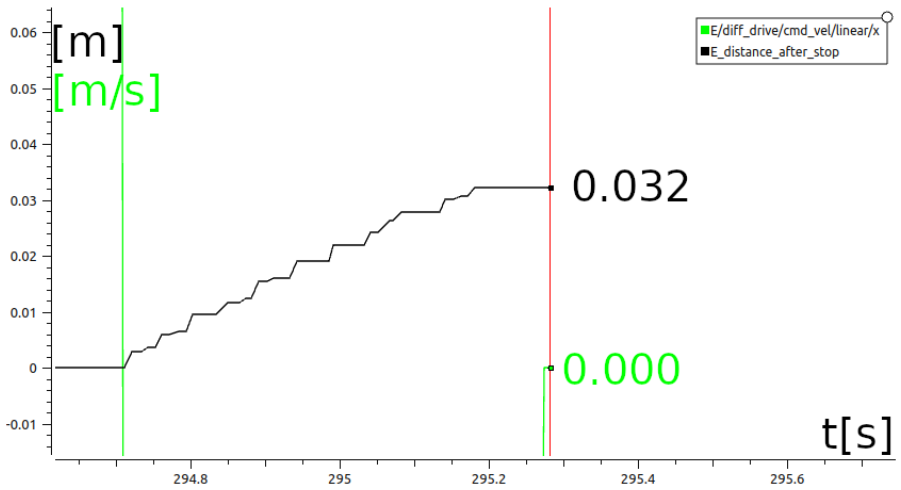
Reference distance (DR) [m]	Travelled distance (DO) [m]	Total absolute error [m]	Compensated absolute error [m]
2.2301	0.103	0.1243	0.0213
2.2168	0.098	0.1110	0.0130
2.1570	0.047	0.0512	0.0042
2.2890	0.141	0.1832	0.0422
2.2589	0.131	0.1531	0.0221
Averaged values			
2.23036	0.1040	0.12456	0.02056

No	Velocity “normal” [m/s]	Velocity “docking” [m/s]	Braking method
4	0.8	0.2	Countercurrent









Scenario 1, series 4 – results

Expected distance is 2.1058m

Reference distance (DR) [m]	Travelled distance (DO) [m]	Total absolute error [m]	Compensated absolute error [m]
2.1844	0.049	0.0786	0.0296
2.1553	0.029	0.0495	0.0205
2.1773	0.051	0.0715	0.0205
2.1651	0.037	0.0593	0.0223
2.1583	0.032	0.0525	0.0205
Averaged values			
2.16808	0.0396	0.06228	0.02268

Scenario 1 – summary

Series No	Total absolute error [m] (based on odometry)	Distance traveled - odometry [m]
1	0.16082	0.1348
2	0.03864	0.0632
3	0.12456	0.1040
4	0.06228	0.0396

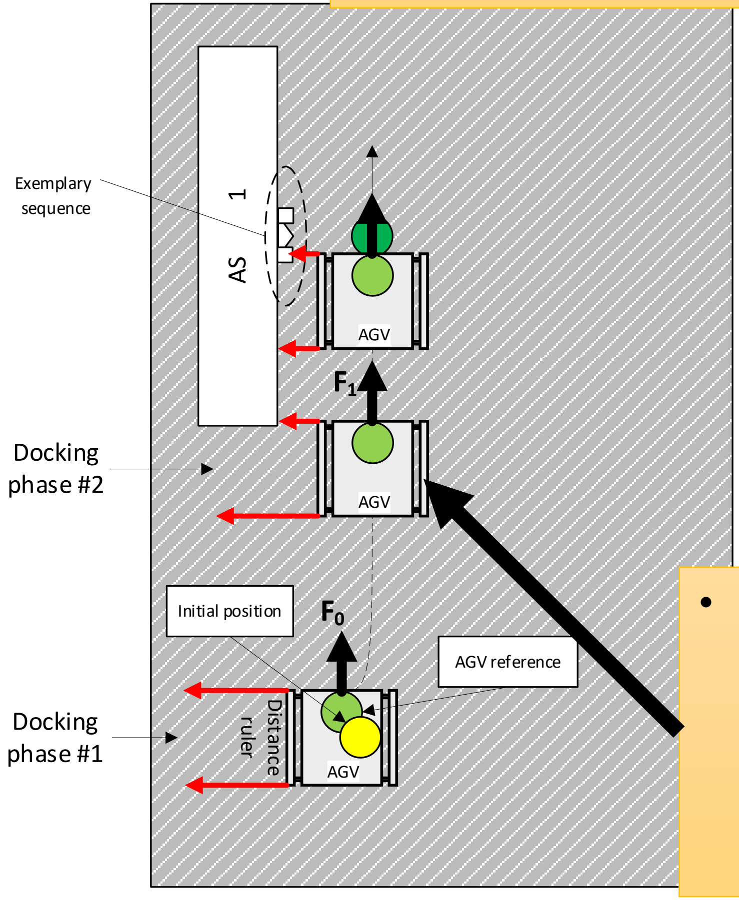
Series #1: the average distance after sending stop command: 0.1348m
Series #2: the average distance for reduced velocity: 0.0632m
Series #4: the average distance traveled for counter current braking method: 0.0396m

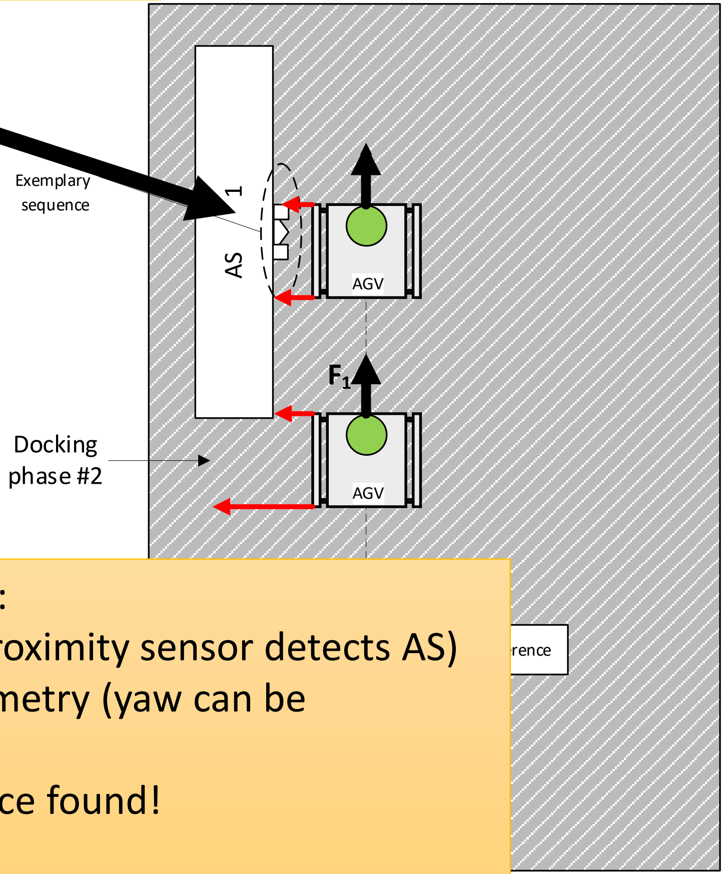
Scenario 1 - summary

- The system latency is the most important element with respect to accuracy.
- Countercurrent braking method is preferable.
- The docking velocity must be adjusted to system latency in order to fulfill accuracy requirements.
- **Caution!** Minimum AGV velocity depends on a number of elements and it is often unpredictable e.g. motor characteristics, weigh of the package, floor condition, wheels condition, etc.
- In presented scenario reduced velocity and countercurrent braking gives accuracy 3.96cm.

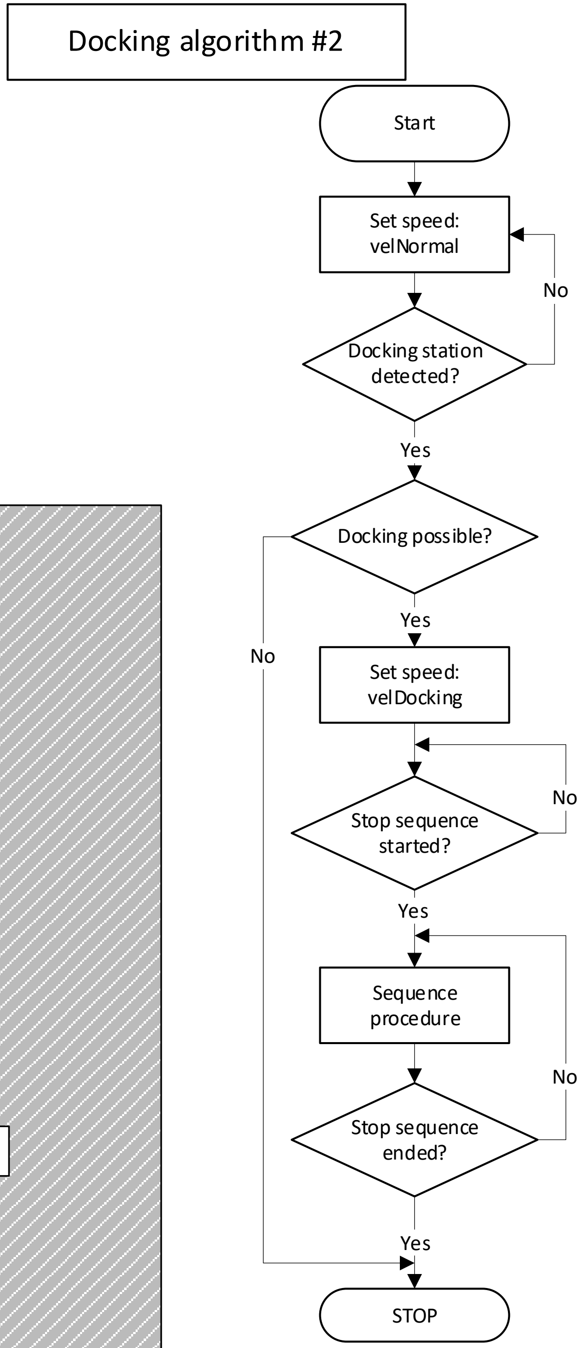
Side Docking Action Server

- Phase #3 (sequence has been detected):
  vector F2 -> velocity “docking” (first proximity sensor detects sequence)
  - Control distance using odometry (yaw cannot be corrected)
  - Stop if characteristic sequence completed





- Phase #2 (AS has been detected):
  vector F1 -> velocity “AS” (first proximity sensor detects AS)
  - Control distance using odometry (yaw can be corrected)
  - Move forward until sequence found!



Research – docking scenario #2 (with tracing the length of AS)

- six measurement series (expected distance is 2.2523m)

No	Velocity "normal" [m/s]	Velocity "docking" [m/s]	Velocity "sequence" [m/s]	Braking method
1	0.8	0.6	0.6	-
2	0.8	0.6	0.4	-
3	0.8	0.4	0.2	-
4	0.8	0.6	0.6	Countercurre
5	0.8	0.6	0.4	Countercurre
6	0.8	0.4	0.2	Countercurre

Results

No	Total error [m] (based on odometry)	Average error [m]
1	0.10954	0.02714
2	0.11328	0.02988
3	0.05514	0.02334
4	0.08326	0.03506
5	0.09510	0.03534
6	0.05726	0.01766

Initial assumptions: the AGV is paralel to the AS, so we don't need to check the alignment.

Comparison of two approches

Scenario – serie	Total Error [m]	Average error [m]	distance traveled [m] after command stop
1 - 1	0.16082	0.02602	0.1348
1 - 2	0.03864	-0.02456	0.0632
1 - 3	0.12456	0.02056	0.1040
1 - 4	0.06228	0.02268	0.0396
2 - 1	0.10954	0.02714	0.0824
2 - 2	0.11328	0.02988	0.0834
2 - 3	0.05514	0.02334	0.0318
2 - 4	0.08326	0.03506	0.0482
2 - 5	0.09510	0.03534	0.0782
2 - 6	0.05726	0.01766	0.0396

Results for scenario #2

- Introducing the AS speed reduces an average error
- Having sequence as docking point minimizes the AGV displacement (after command stop)
- Another issues: both scenarios do not take into account alignement to the AS. Unfotunately, the wheels’ rotation velocity varies!

Docking algorithm - introducing PID controller

- Algorithm is divided into three blocks:
  - Wheel control
  - Alignment
  - Distance control

All blocks are based on PID controlers for different actions.

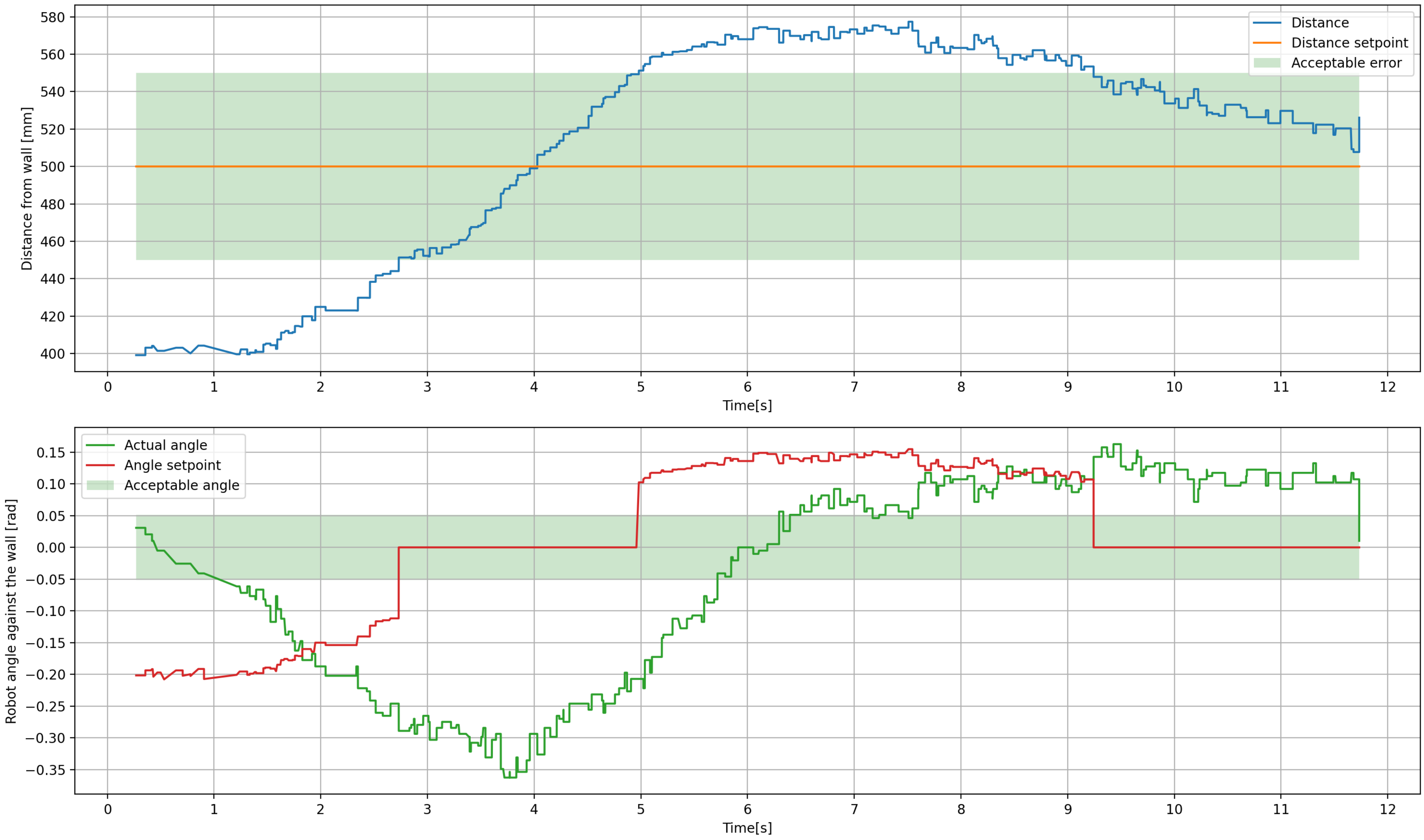
What does the model do?
The model estimates distance required for docking, based on initial distance to the AS, rotation and distance setpoint (docking coordinates and distance to the AS)

PID parameters

Wheels RPM Control			
Settings	Kp	Ki	Kd
Right wheel	6	5	0.1
Left wheel	5	5	0.1

Angle Control (Alignment)			
Settings	Kp	Ki	Kd
Wheel RPM	3	0.5	0.008

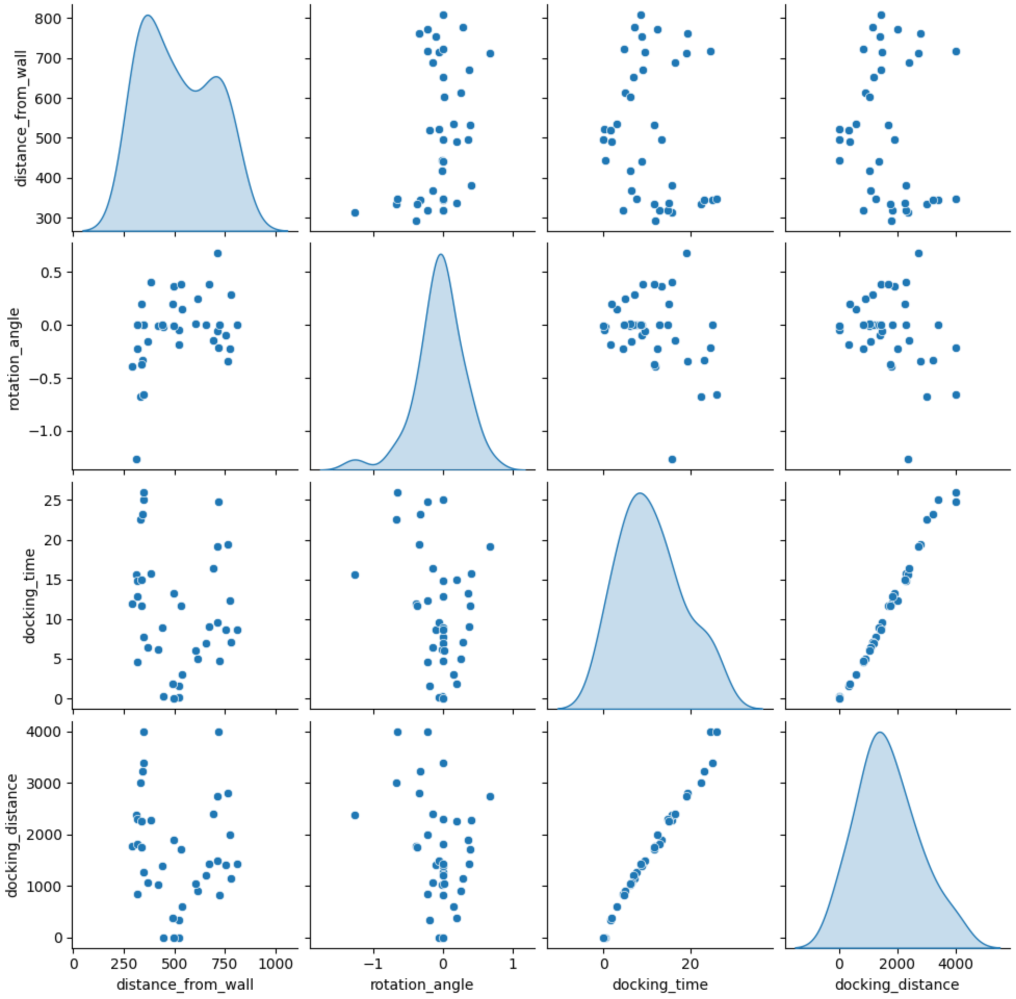
Distance Control			
Settings	Kp	Ki	Kd
Angle	0.002	0.00002	0.00002

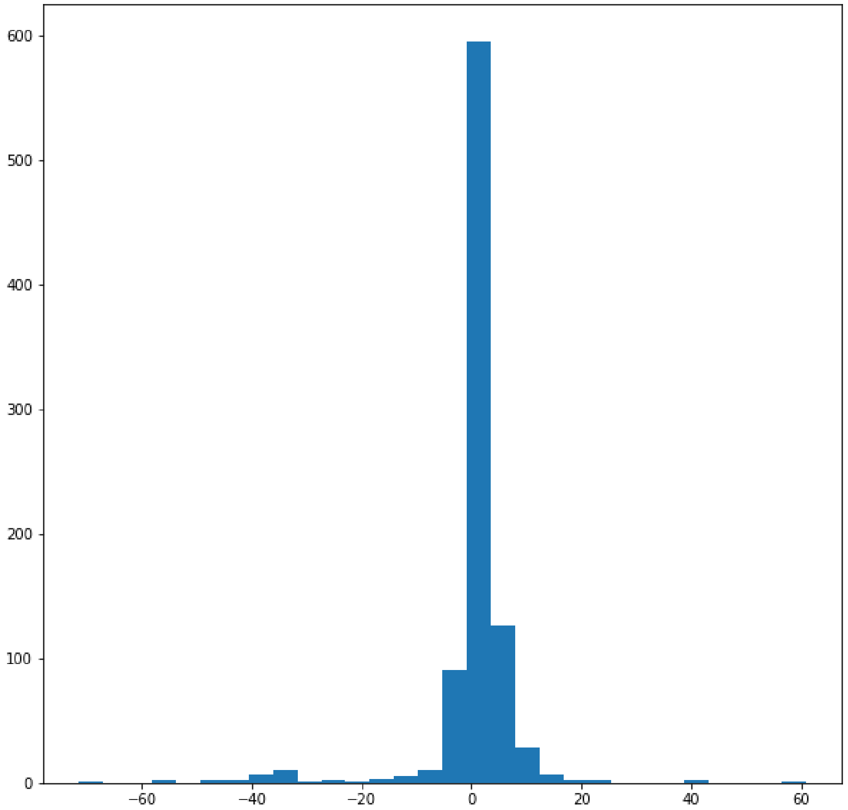


Distance
Distance setpoint
Acceptable error
Distance from wall [mm]
Time[s]
Actual angle
Angle setpoint
Acceptable angle
Robot angle against the wall [rad]
Time[s]

Distance
Distance setpoint
Acceptable error
Distance from wall [mm]
Time[s]
Actual angle
Angle setpoint
Acceptable angle
Robot angle against the wall [rad]
Time[s]

Feature matrix & distance prediction





- Machine learning model (Deep Neural Network – Tensorflow Keras). It calculates the required distance the robot should drive in order to dock.

Conclusion

- Why ruler?
  - It gives much more reliable rotation with respect to RPLidar,
  - it is cheaper than RPLidar,
  - gives less data and reduces computational resources for PID controller
- Why countercurrent?
  - reduces dead area (if any),
  - reduces AGV inertia (especially if heavily loaded)
- Why passing the control to the AGV?
  - reduces time delay of the communication system
- If the independenc of AGV is not possible, the speed reduction is mandatory.

Thank you for attention!