Rescue mode engaged

Sven Dierfeld[1], Tolga Dirikgil[1], Asan Adamanov[1], Jonathan Hellwig[2]

TU Hamburg[1]
University of Hamburg[2]

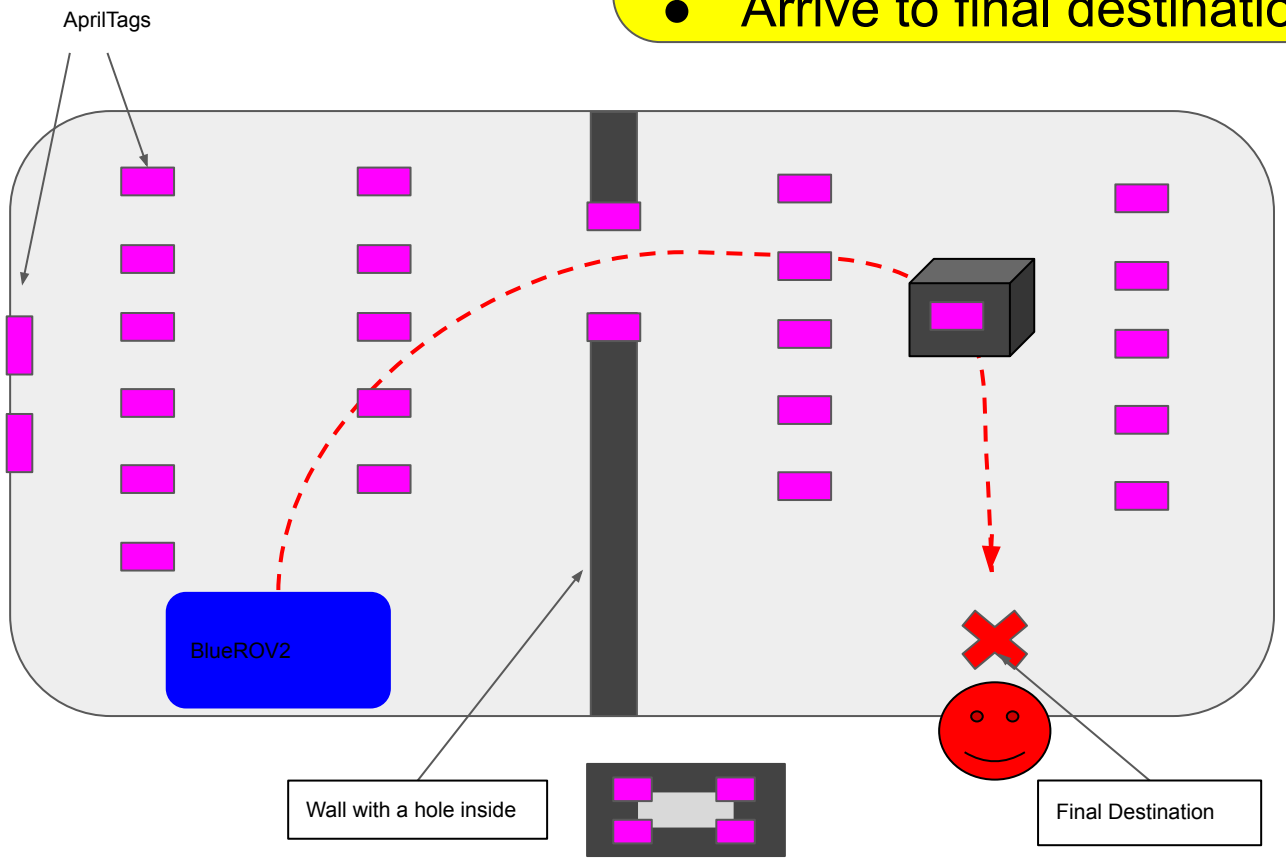
Wet Whale

Formulas and Vehicles Winter 2020/21

Introduction - PROPOSAL

TASKS

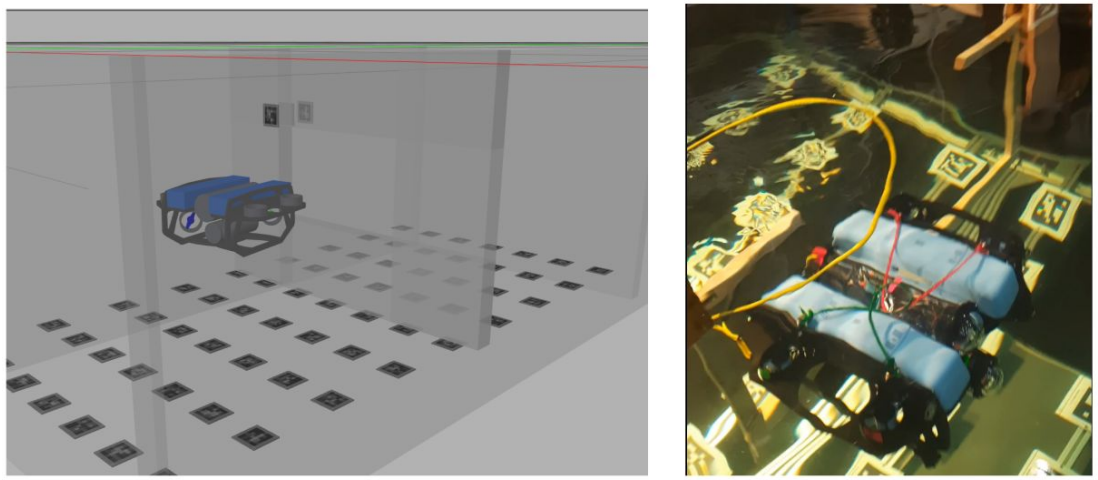
- Given: Static Tank (w/o Wall & Obstacle) and final destination
- **Construct the path**, to pass through the hole and avoid obstacle
- Arrive to final destination, do **Person detection**



Frozen Ammersee in Bavaria
Von I, T. Then, CC BY-SA 3.0,
https://commons.wikimedia.org/w/index.php?curid=2268567

Simulation and experimental design

- Tank: 3,35 m x 1,67 m x 1,4 m
- Bluerov2: IMU, front and down cameras, pressure sensor
- **Simulation:** Gazebo wall with tags, also tags for object
- **Experiment:** wooden wall with same dimensions, no object
- position from given localization



(a) (b)

Fig. 2: Depiction of the (a) simulation and (b) experiment environment.

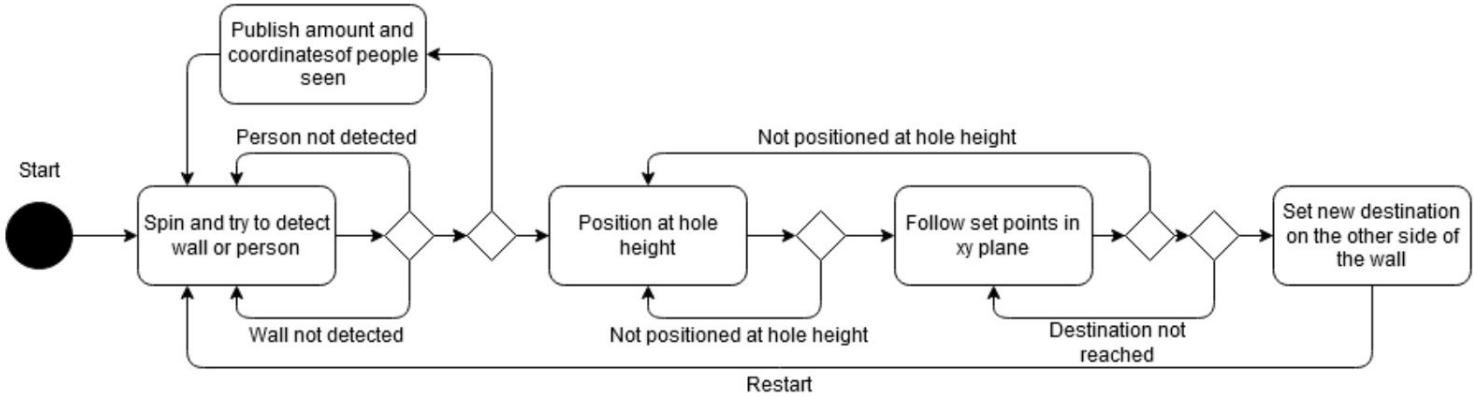


Fig. 1: The ROV operation scheme.

Tested Hypothesis

1. ROV manages to follow a path five consecutive times without hitting a virtual wall inserted into the path planning map.

2. ROV correctly detects the tag on the wall at the position we measured for five different angles.

3. ROV successfully navigates from side of the tank to the other without bumping against the walls on the sides or the wall inserted into the middle of the tank.

4. Detect a person on a printed and laminated picture.

Methods: Object detection

Simulation:

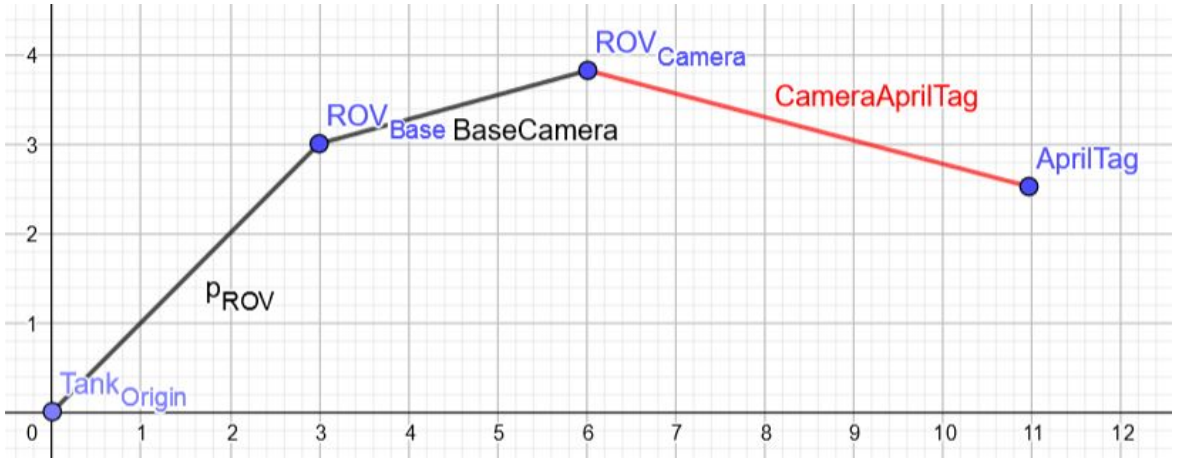
- Gazebo plugin

Experiment:

- Camera sensor
- AprilTag detection package

Measurement received:

- Vector to AprilTag in the system of the Camera
- Position of the wall obtained by



Coordinate frames

$$p_{\text{Wall}} = p_{\text{Base}} + R^{\text{Tank}}_{\text{Base}} v^{\text{Camera}}_{\text{Base}} + R^{\text{Tank}}_{\text{Base}} v^{\text{AprilTag}}_{\text{Camera}}$$

Mapping (of box and wall)

- **Given:** Static map Fig 4-a
- **Inputs**: received coord. of detected obstacle
  - The length of the hole is encoded. Also, the box dimension is encoded
  - Box mapped as a circle, to simply orientation of box
  - Grid size 5 cm
  - Wall Buffer 30cm

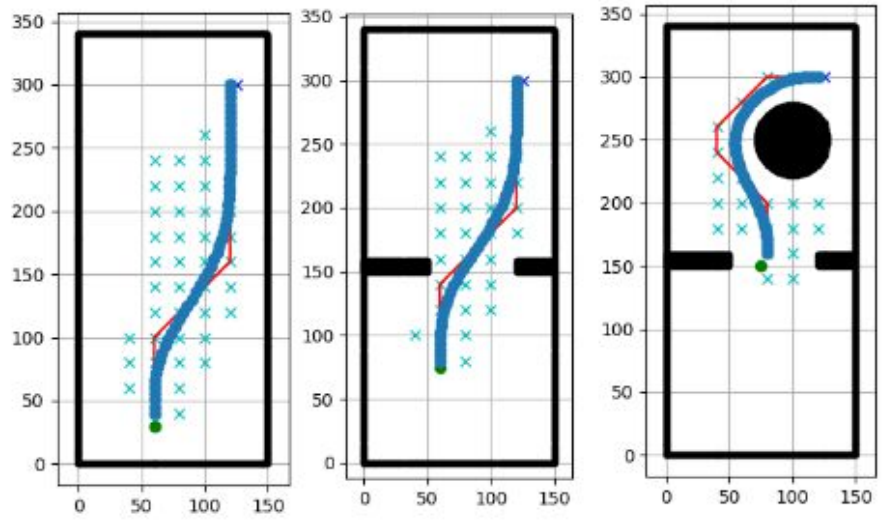


Fig. 4: Three depictions of the path planning in the xy-plane with A-Star (red) with fitted Bezier Curve paths (blue).

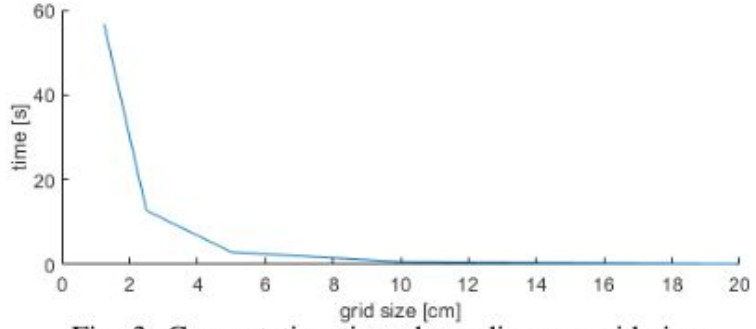


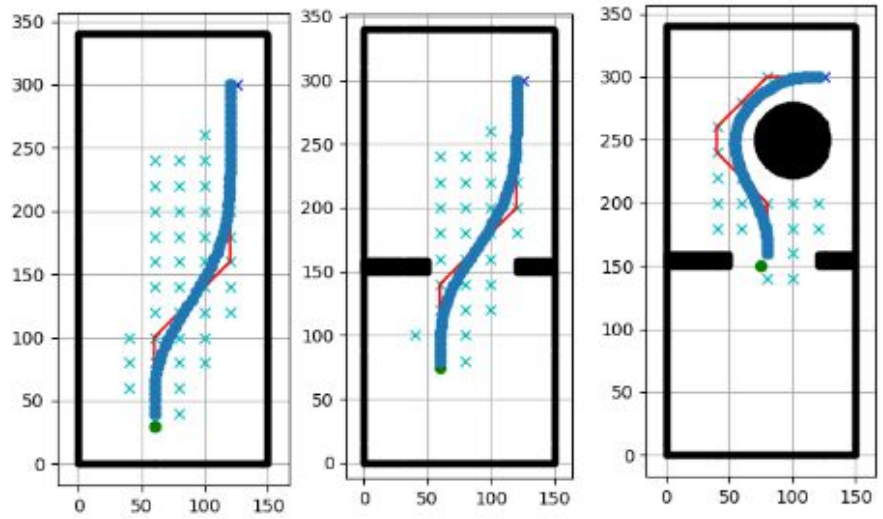
Fig. 3: Computation time depending on grid size

Methods: Path Planning

Global/Local (Dynamical) Path Planning

- Grid Based Approach: A* Algorithm with L2 - Norm Heuristic, Complexity = O(b^2), Optimal, f(v) = g(v) + h(v)
  - Smoothing through: Bezier Curve
- **Inputs**: Current Position
- **Output**: Set of points to destination
- **Trick**: Intermediate node for storing setpoints which is publish to controller.

Both parts are totally independent of all other stacks e.g Obstacle Detection and Controller, hence they are robust.



(a) initial map (b) with wall (c) with box

Fig. 4: Three depictions of the path planning in the xy-plane with A-Star (red) with fitted Bezier Curve paths (blue).

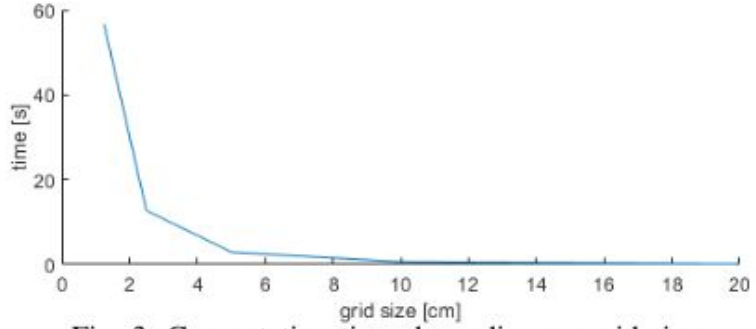


Fig. 3: Computation time depending on grid size

Methods: Controller

- based on multiple SISO PID-Controller
- vertical thrust: standard PID-controller
- yaw rotation: additional ceiling to distance between setpoint and state
- thrust and lateral thrust PID-Controller use the relative distance from ROV to setpoint for PID Controller
- low implementation effort and easy to tune

$$T_{yaw} = \begin{pmatrix} \cos(\gamma_{yaw}) & \sin(\gamma_{yaw}) \\ \sin(\gamma_{yaw}) & \cos(\gamma_{yaw}) \end{pmatrix} \quad (1)$$

$$p_{dis} = T_{yaw} \cdot (p_{set} - p_{ROV}) \quad (2)$$

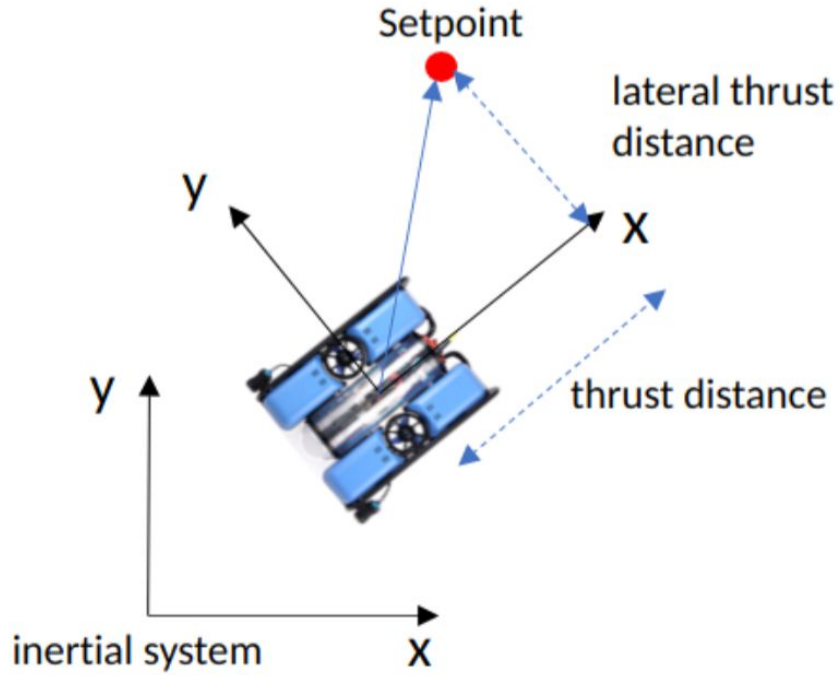
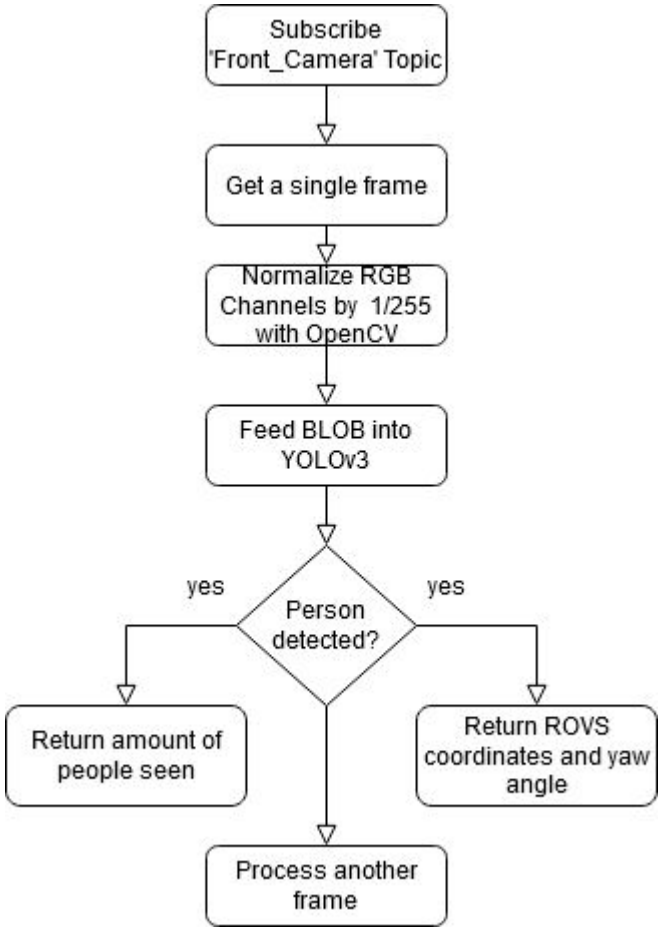


Fig. 3: A Visualization of the controller.

Methods: Person Detection

- Frame processing done by using OpenCV
- Person detection done using YOLOv3 Framework with pretrained YOLOv3 model



Flowchart of person detection

Methods: 3D Person Detection

Camera used

- Intel D435i

Calibration out of lab

- 7x9 checkerboard
- 20 mm squares
- Hand to eye approach
- 5 calibration positions

Calibration in lab

- Checkerboard at fixed position
- ROV goes do different setpoints

Test environment

Calibration using checkerboard

Integration of the components

Algorithm:

1. Calculate path with current map
2. Send path to controller
3. Iterate over path and take set point with sufficient distance
4. Update path if available

- Objects are dynamically detected and path is updated accordingly
- Expensive computation runs continuously in the background

Results

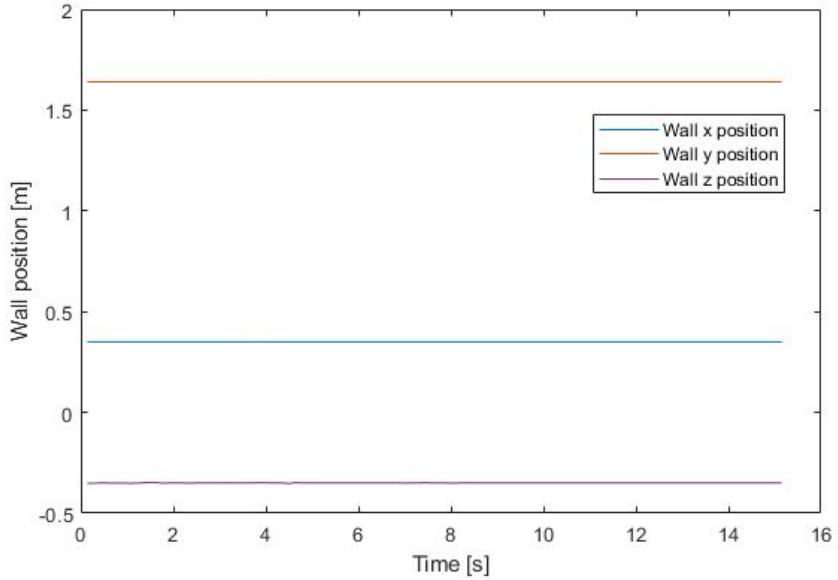
Results: Object detection

Simulation:

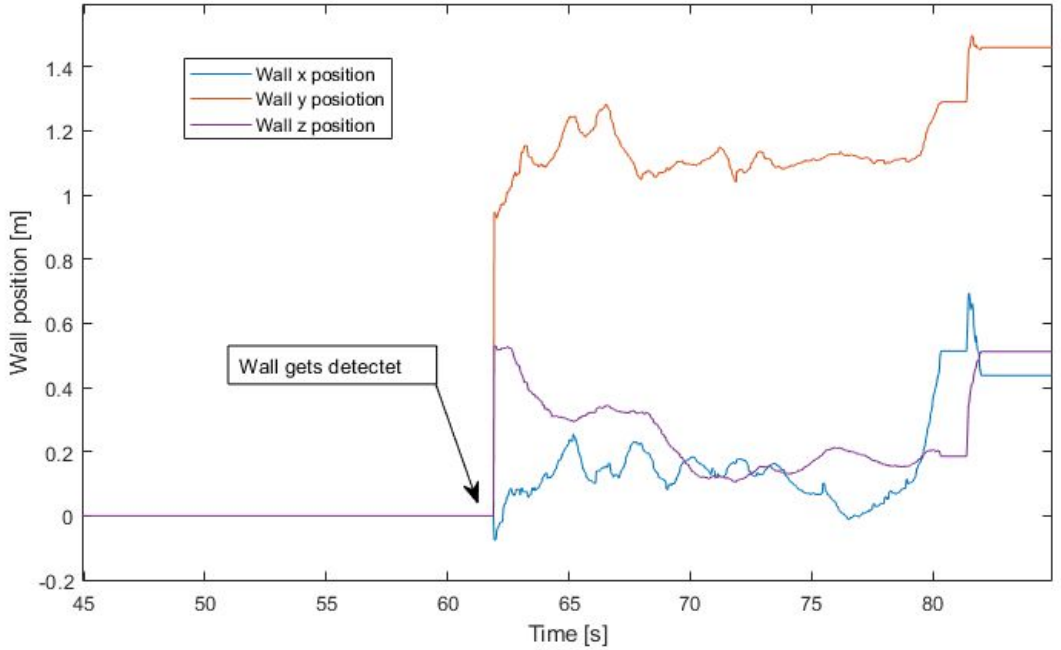
- Stable wall position

Experiment:

- Oscillations in detected wall position



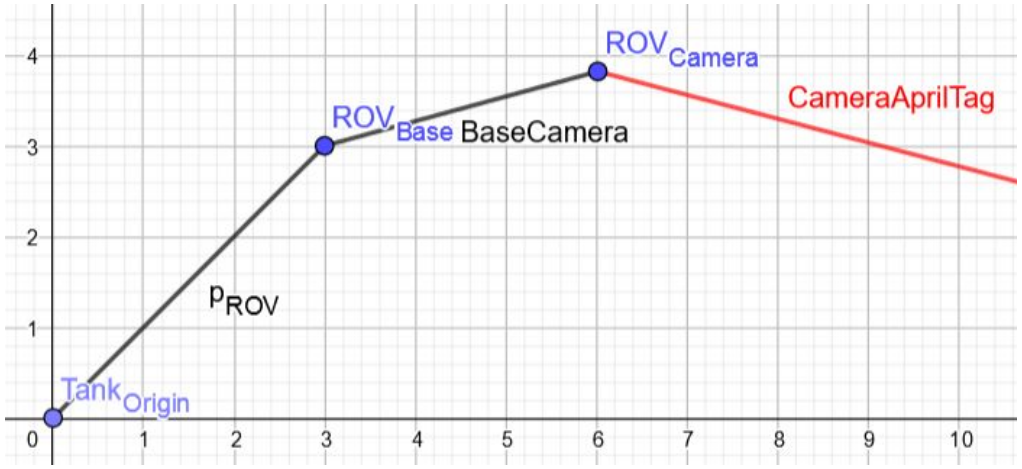
Wall position in simulation



Wall position in experiment

Results: Object detection - error in mapping

- Object detection in simulation works
- Camera frame and base frame are the same



Coordinate frames

Corrected formula:

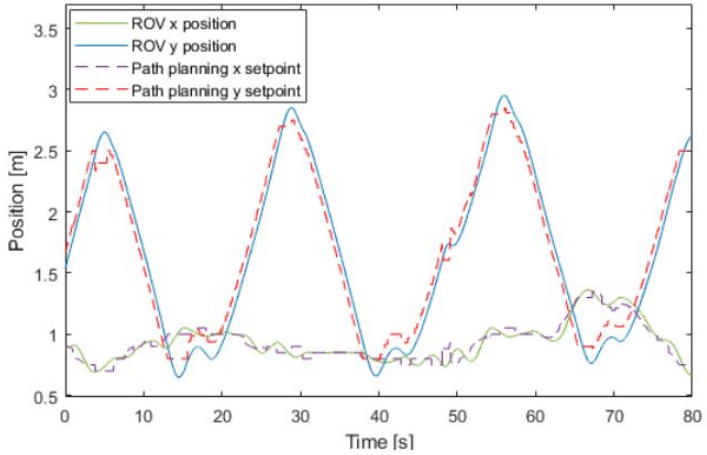
$$p_{\text{Wall}} = p_{\text{Base}} + R^{\text{Tank}}_{\text{Base}} v^{\text{Camera}}_{\text{Base}} + R^{\text{Tank}}_{\text{Base}} R^{\text{Base}}_{\text{Camera}} v^{\text{AprilTag}}_{\text{Camera}}$$

Results: Controller

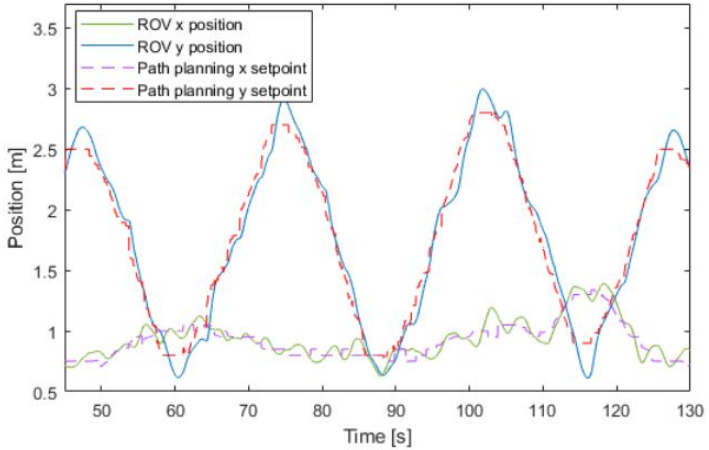
- Threshold for setpoint is at 15 cm in front on the path.
- Ceiling for yaw controller is at 20°.
- Due to time constraints during experiment, PID-controllers are not fully tuned.
- ROV follows setpoint well even though there is some overshoot.

	Vertical thrust		lateral thrust		thrust		yaw	
	sim	exp	sim	exp	sim	exp	sim	exp
k_p	4.0	5.0	2.0	3.0	2.0	3.0	0.3	0.5
k_d	0.018	0.02	0.0	0.007	0.0	0.007	0.0	0.005
k_i	0.006	0.006	0.0	0.002	0.0	0.002	0.0	0.003

TABLE I: Parameters for the PID-Controllers from the experiment and the simulation.



(a) Simulation

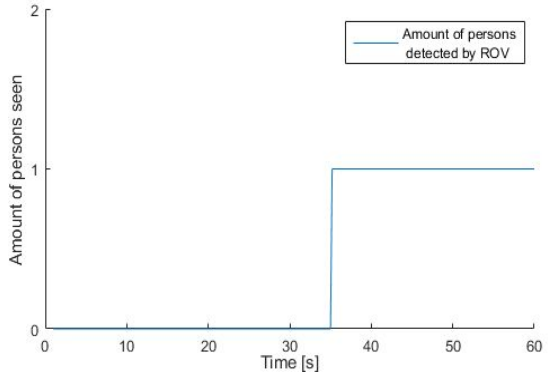


(b) Laboratory

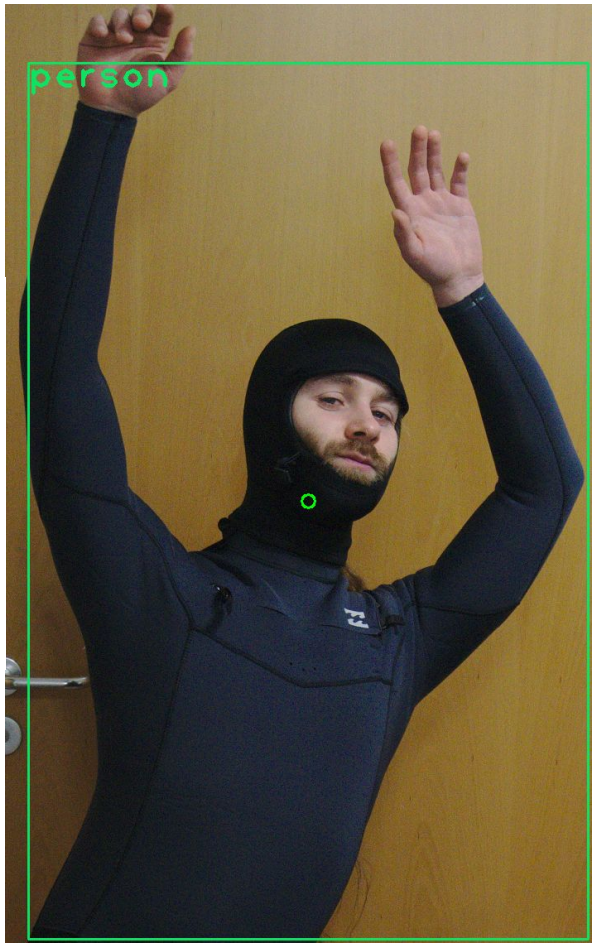
Fig. 8: Setpoint following for the x- and y-direction

Results: Person Detection

- Person detection tested prior to experiment with Rosbag of an underwater recording
- Person detection itself worked
- Interaction by returning ROV coordinates and yaw angle while getting past the person didn’t work
- Possible cause is slow frame processing



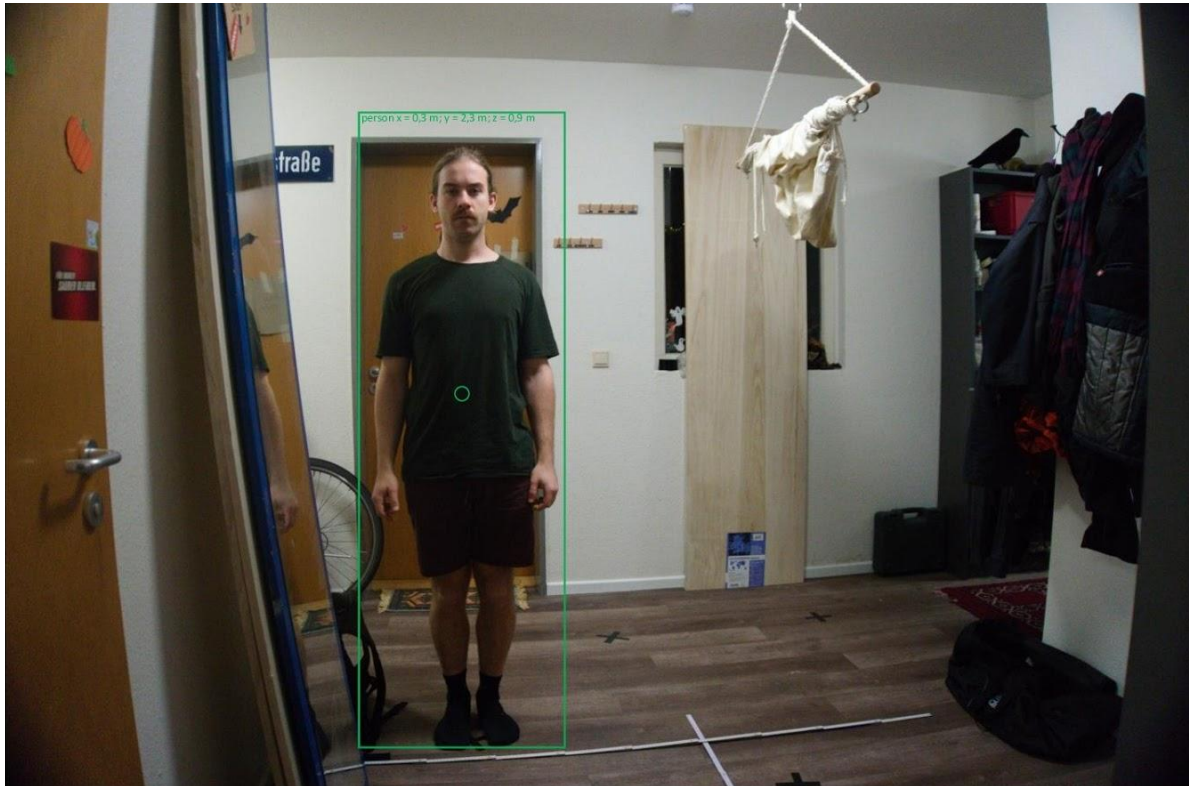
People seen by ROV



Picture used in lab

Results: 3D person detection

- Being able to return 3D coordinates of detected person out of lab
- Determined coordinates slightly inaccurate
- More calibration points might make it more accurate



Person detected, actual coordinates x= 0.43 m, y = 2.23 m, z = 1.07 m, determined coordinates x= 0.3 m, y = 2.3 m, z = 0.9 m

Conclusion

- Desirable goals were almost achieved

	Simulator	Lab
Object Detection	+ Succeeded	- Failed
Mapping / Path Planning	+ Succeeded	+ Succeeded
Controller	+ Succeeded	+ Succeeded
Person Detection	+ Succeeded	- Failed

Wrong coordinate Transformation

Processing of frame containing person not finished once ROV got past the person

- Possible future investigation for improvement
  - Extend to 3D Occupancy Mapping with direct camera and use OctoMap
  - 3D Path Planning with A* or RRT*
  - More advanced controller e.g. model based, LQR/LQG