Assignment 2

Problem Statement

Consider the scenario of an underwater robot autonomously navigating in a confined volume, as depicted in Figure 1. Navigating refers in this context to way-point tracking and path following. In order to design a path following controller for this task, accurate online information about the robot's absolute position is required first. Hence, a pipeline covering both aspects, namely *localization* and *tracking control* has to be developed in order to fulfill the autonomous navigation task.

Background

The robot's absolute position can be obtained based on range measurements d_i between the robot and reference points at known positions, referred to as anchors. For the sake of simplicity, we assume the availability of an on-board ranging-sensor which directly provides data on the distance towards the individual anchors. Note that due to the physical limitations of this sensor, we cannot assume that all anchors are detected at each measurement cycle. On-board the real robot we use a vision-based ranging system based on the BlueROV2's forward-facing camera. Hence, it has to be ensured that the anchor points lie within the camera's field of view (FOV) in order to receive range measurements.

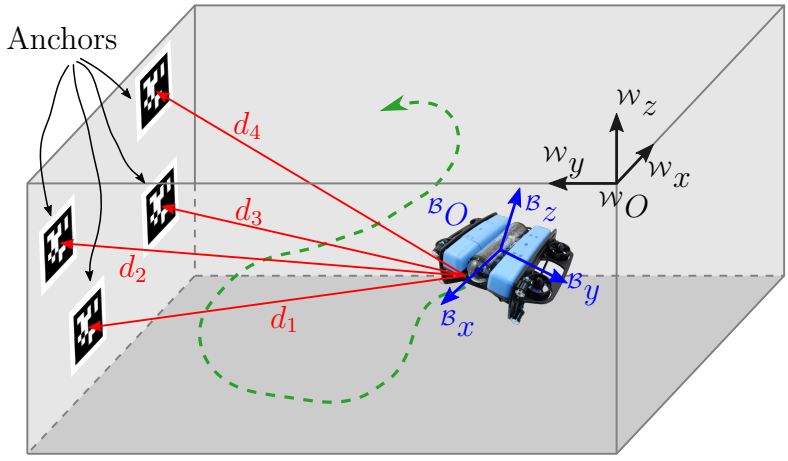

Figure 1: Underwater vehicle autonomously navigating in a confined volume using an absolute localization system based on visual range measurements towards anchors placed at known positions.

Tasks

The following steps are intended to guide you through the development of a) the Kalman Filter algorithm used for estimate the position of your robot followed by b) the extension of your controller (Assignment 1) towards 3D position control. See also the Hint Section below.

Position Estimation via Kalman Filtering

Goal and Starting Point:

- We want to estimate the 3D-position $\boldsymbol{x}$ of the underwater robot with respect to landmarks (4 anchors $\mathcal{A}_{0:3}$ at known positions represented by AprilTag markers, see Figure 1).
- The robot carries a range sensor that computes the distances $d_{0:3}$ between the robot and the corresponding Anchors $\mathcal{A}_{0:3}$ if the Anchors are visible from the robot.
- The template already provides you with structure for the Kalman Filter algorithm that mainly consists of three steps: 1) initialization/set-up, 2) the time prediction step, and 3) the measurement update step.

Let's get started:

1. First, we need to decide on a state $\boldsymbol{x}$. A simple choice is $\boldsymbol{x} = \begin{bmatrix} x, y, z \end{bmatrix}^\top$. You could also include velocities, as done in the example for the depth Kalman Filter. However, start with the simple version (position only) in any case. Extending this later is quite straight forward.
2. For the *time prediction step*, we start with the most simple approach of assuming our system *does not* possess any dynamic, i. e. is assumed to be *static* between two time step, i. e. $\boldsymbol{x}_k = \boldsymbol{A}\,\boldsymbol{x}_{k-1}$ with $\boldsymbol{A} = \boldsymbol{I}$ where $\boldsymbol{x}$ is the robot state in x,y,z-coordinates and $\boldsymbol{I}$ is the identity matrix of corresponding size. For this case, $\boldsymbol{I}$ is a 3-by-3 matrix.

   While we have now considered the temporal evolution of the state mean, we should not forget to consider the temporal evolution of the state uncertainty $\boldsymbol{P}$. We remember from the lecture (and any textbook on this topic), this evolves reading $\boldsymbol{P}_k = \boldsymbol{A}\,\boldsymbol{P}_{k-1}\,\boldsymbol{A}^\top + \boldsymbol{Q}$.

Dynamic models, such as ours above, never match reality exactly. We account for this by increasing the state uncertainty $\boldsymbol{P}$ by $\boldsymbol{Q}$ to which we refer to as *process noise*. Roughly speaking, Q describes the inaccuracy of our dynamic (process) model. As we suffer from this inaccuracy each time we predict the next time step, it is convenient to increase the state uncertainty each time we predict the next time step – you may remember the example on close-your-eyes at road crosswalks from lecture, "uncertainty increases if we do not receive measurements - solely predict what is going on".

- Let us assume we run the prediction step at higher frequency, e. g. at 100 Hz ($\Delta t = 0.01\,\mathrm{s}$) instead of 5 Hz ($\Delta t = 0.20\,\mathrm{s}$). While the miss-match between the real-world and our simplified model stays the same, how can we consider this within our process noise $\boldsymbol{Q}$?

3. Revisit the measurement update step in the lecture. As mentioned before, we will use a range measurement sensor that computes distances to point of interest, e. g. Anchors/Tags, down to an accuracy of R to which we refer as measurement uncertainty. Without going in detail on the theory we require expressions for the measurement function $\boldsymbol{h}$ that models the present sensor, here range measurements between anchor and robot. Let $\boldsymbol{z}$ be the vector of distance measurements d_i between the robot position $\boldsymbol{x}$ and the anchor position $\boldsymbol{p}_i$ of Anchor $\mathcal{A}_i$. As the anchors are placed at static fixed positions we can write this as $\boldsymbol{z} = \boldsymbol{h}(\boldsymbol{x})$ where $\boldsymbol{h}(\boldsymbol{x}) = \begin{bmatrix} h_1(\boldsymbol{x}) & \dots & h_4(\boldsymbol{x}) \end{bmatrix}^\top$.

   - Derive the mathematical expression for $z_i = d_i = h_i(\boldsymbol{x})$ for Anchor $\mathcal{A}_i$ at position $\boldsymbol{p}_i$.

   We also need their corresponding Jacobian matrices. Derive the Jacobians $H_i = \frac{\partial h_i(\boldsymbol{x})}{\partial \boldsymbol{x}}$.

   - What are the dimensions of H_i and $\boldsymbol{H}$?
   - Revisit the Kalman Filter equations, what is the dimension of the Kalman gain $\boldsymbol{K}$?

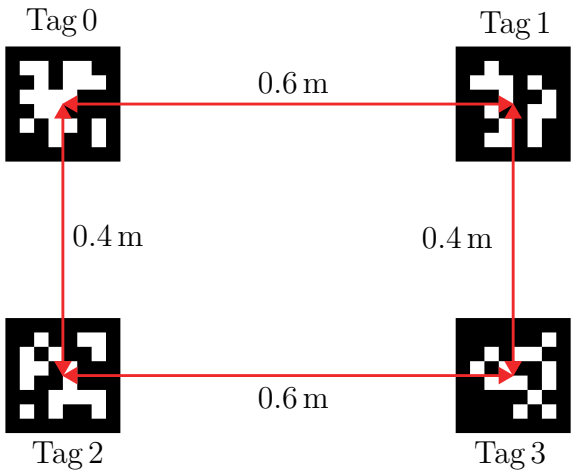

Figure 2: AprilTag positions on the test tank wall.

3D-Position Control

Extend the controller from assignment 1 to control all 3 positional degrees of freedom of the robot. Additionally, you can control the yaw angle. We provide you with a simple P-controller for keeping the robot facing the wall as a starting point. Your controller should be able follow a sequence of pre-defined way points, e. g. a rectangle or circle.

If your localization is not working yet, use the pose from `/bluerov00/vision_pose_cov` to help you develop and evaluate your controller.

Hints/Challenges you might face

- In the simulation, the position of Tag 0 is $[0.7,\ 3.8,\ -0.5]^\top$ m. Make sure you can adjust the absolute position of the tags for the experiment, since we cannot exactly tell you where they will be. We will keep the general shape the same.
- The distance is measured from tag to the camera, not the robot's body frame. You can use the robot's orientation to consider this.
- Single/multiple distance measurements might get lost.
- The distance measurements' range is limited.
- The update rate of the ranging sensor might be low and is varying over time.
- ...

Procedure

At Home:

1. **Read the Assignment 2 section of our docs thoroughly** and follow the steps to update your packages for this assignment. Additionally, you can find **more hints in the docs**.
2. The two separate parts of this assignment can be done in any order:
   - Implement a Kalman Filter which solves the localization problem described above, taking into account possible challenges.
   - Analogue to Assignment 1, implement a controller for the robot's x and y position. If your localization is not working yet, use the pose from `/bluerov00/vision_pose_cov`.
3. Extensively test your algorithm in Gazebo!

Experiment

1. You will have 2×120 min in the lab.
2. Make sure to thoroughly plan your experiments.
3. *Optionally*, you can ask us for a Zoom meeting for advice or feedback.
4. Test and evaluate the performance of your localization algorithm and controller. You decide how you use your (limited) time.

Submission:

1. Summarize your approach and your algorithm. Describe, analyze, and critically discuss simulation and experimental results. The paper must not exceed 4 pages using the template.
2. Submit your paper via your group's slack channel. The name of your submission should follow the format: `assignment2_groupX.pdf`.

Deadline Paper: 23.12.2024, 23:59 CET