Intervals / Evaluation – Terrain based navigation

Robotique/UE 4.1 – Mars 2022

Supports de cours disponibles sur
`www.codac.io`

A. Introduction

An underwater robot $\mathcal{R}$ is evolving above the seabed, without surfacing. It is equipped with sensors for measuring its velocity ϑ and its heading ψ. Without surfacing, the robot is not able to take benefit from GNSS positioning information, thus inducing a significant drift of its localisation. Indeed, the environment (the seabed) is unstructured and there is no objects of interest to rely on, for using state estimation methods such as SLAM or beacon-based localisation.

The objective is to limit the drift of the underwater robot by embedding the map of the environment, prior to performing the mission. The state estimation process consists in estimating the position $\mathbf{x}_{1,2} \in \mathbb{R}^2$ of the robot in the map. This is achievable thanks to a single-beam sonar embedded on the vehicle. We consider that this sensor returns, on a regular basis, scalar values $z_i \in \mathbb{R}$ corresponding to the bathymetry of the environment: the elevation of the seabed, with respect to the sea surface. See Figure 1 for an illustration.

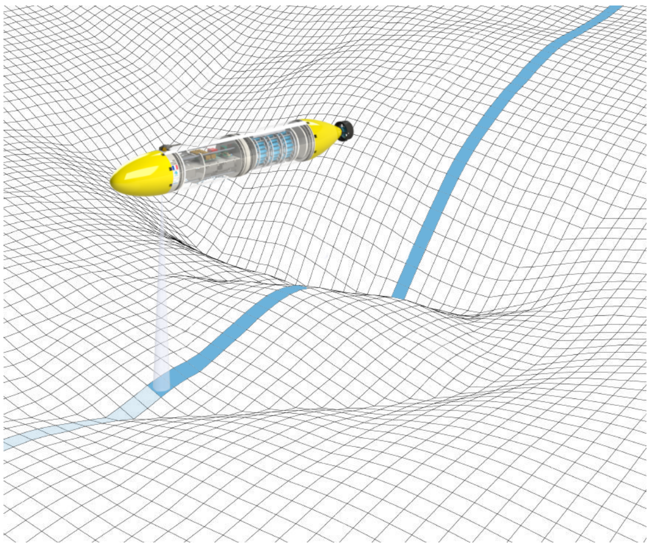
Figure 1: A robot measuring the bathymetry under its trajectory, thanks to a single-beam echosounder.

We will solve this problem using interval methods.
The problem is considered in two dimensions (horizontal positioning), as it is usually the case in underwater robotics.

B. Comments

1. Each question must be appended by comments in your code.

C. Dead reckoning

In this section, we do not sense the environment yet: we only assess the drift of the robot based on proprioceptive data. This is what we call *dead reckoning.*

The unknown trajectory of the robot, denoted by $\mathbf{x}^*(\cdot)$, is given by:

$$\mathbf{x}^*(t) = \begin{pmatrix} x_1^* \\ x_2^* \\ \psi^* \\ \vartheta^* \end{pmatrix} = \begin{pmatrix} 10\cos(t) + t \\ 5\sin(2t) + t \\ \text{atan2}\big((10\cos(2t)+1), (-10\sin(t)+1)\big) \\ \sqrt{(-10\sin(t)+1)^2 + (10\cos(2t)+1)^2} \end{pmatrix}. \quad (1)$$

Furthermore, the robot follows the state equation given by:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) = \begin{pmatrix} \vartheta\cos(\psi) \\ \vartheta\sin(\psi) \\ u_1 \\ u_2 \end{pmatrix}. \tag{2}$$

The input $\mathbf{u}(\cdot)$ is considered unknown here, but the robot directly measures its velocity ϑ and heading ψ, with some error that we consider bounded within $[-0.03, 0.03]$ for ϑ and $[-0.01, 0.01]$ for ψ.

2. Simulate, for $t \in [0, 7]$, the actual but unknown trajectory $\mathbf{x}^*(\cdot)$.
   The obtained heading $\psi(\cdot) = x_3(\cdot)$ is a discontinuous trajectory, so we apply a method to make it continuous:

   ```
   x_truth.sample(dt);
   x_truth[2].make_continuous();
   ```

3. Compute the set of feasible trajectories estimated by the robot, considering:
   - the bounded uncertainties assumed on ψ and ϑ,
   - that the initial position $\mathbf{x}(0)$ is known to be in $[\mathbf{x}_0] = \big([9, 11] \times [-1, 1]\big)$.
4. Display this set together with the actual trajectory on a `VIBesFigMap`.

D. Terrain based navigation (scalar measurement)

The system is now described by:

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}\big(\mathbf{x}, \mathbf{u}\big) & \text{(evolution eq.)} \\ z_i = g_{\mathbb{M}}\big(\mathbf{x}_{1,2}(t_i)\big) & \text{(observation eq. related to the terrain map } \mathbb{M}) \end{cases} \tag{3}$$

Prior to the mission, the environment has been sensed by hydrographers using high resolution sonars. A mathematical model has been computed for approximating the DEM (Digital Elevation Model) of the seabed, and results in an analytical function $g_{\mathbb{M}}$ given by:

$$g_{\mathbb{M}} : \begin{array}{ccc} \mathbb{R}^2 & \to & \mathbb{R} \\ \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} & \mapsto & \sqrt{(p_1/2)^2 + (p_2/2)^2} + \sin\big(p_1/2\big) + \cos\big((p_1 + p_2)/2\big) \cdot \cos\big(p_1/2\big) \end{array} \tag{4}$$

5. Compute the signal (trajectory) $z(\cdot)$ representing the actual but unknown bathymetry that should be sensed by the robot along its path. Display the result in a `VIBesFigTube` view[1].
6. Build a *map contractor* $\mathcal{C}_{\mathbb{M}}\big([z], [\mathbf{p}]\big)$ that contracts the sensed bathymetry $[z] \in \mathbb{IR}$ and the related bounded position $[\mathbf{p}] \in \mathbb{IR}^2$ according to Eq. (4).
7. Perform the state estimation of the robot based on a discrete set of bathymetric measurements z_i given for each $t_i \in \{0, 0.25, 0.5, 0.75, \dots, 7\}$. The errors related to these measurements are assumed to be bounded in $[-0.001, 0.001]$.
8. Display the resulting set in the `VIBesFigMap`, by revealing the contraction effect.
9. In a clearly visible comment in your code, explain why this method is competitive compared with conventional estimation methods such as Kalman.
10. Could the constraint that has been set on the initial position (Question 3) be *relaxed* (removed)? Explain why, in a second comment in your code.

[1] `http://codac.io/manual/07-graphics/02-figtube.html`

E. Multi-beam echosounder measurements

We now assume that at each time t_i, the robot measures more than one bathymetric point, as displayed in Figure 2. This is possible thanks to multibeam echosounders able to perceive the environment on each side of the vehicle.

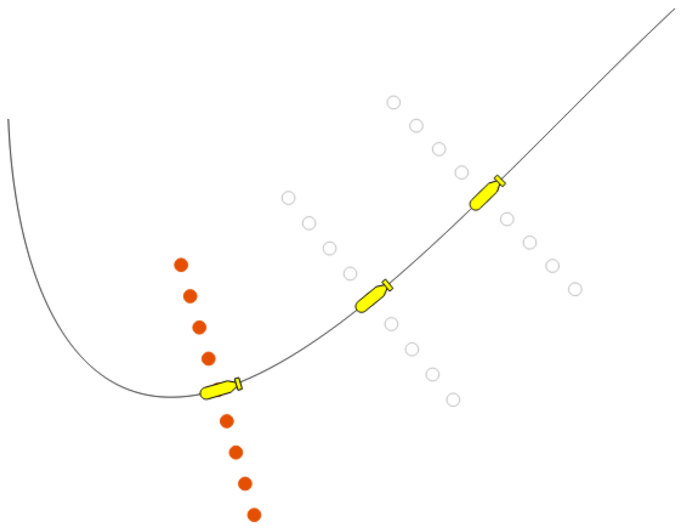

Figure 2: A robot measuring eight bathymetric points (*probes*) under its trajectory, thanks to a multi-beam echosounder.

For simplicity here, we assume that the horizontal positions of the probes $\mathbf{q}^j$, in the robot reference frame $\mathcal{R}$, are given by:

$$\mathbf{q}^j = \begin{pmatrix} 0 \\ j \end{pmatrix} \qquad \text{with } j \in \{-4, -3, \dots, 3, 4\}. \tag{5}$$

11. Update the `VIBesFigTube` of Question 5 for displaying these new multibeam data (which amounts to displaying eight scalar trajectories).

12. Update your state estimation solver for improving the localisation based on these new dataset.
    Display and comment the results.