EECE 5550 Mobile Robotics Lab #2

David M. Rosen

Due: Oct 22, 2021

Problem 1: Differential drive kinematics on $\mathrm{SE}(2)$

Consider a differential-drive robot with track width w and wheel radius r (Fig. 1).

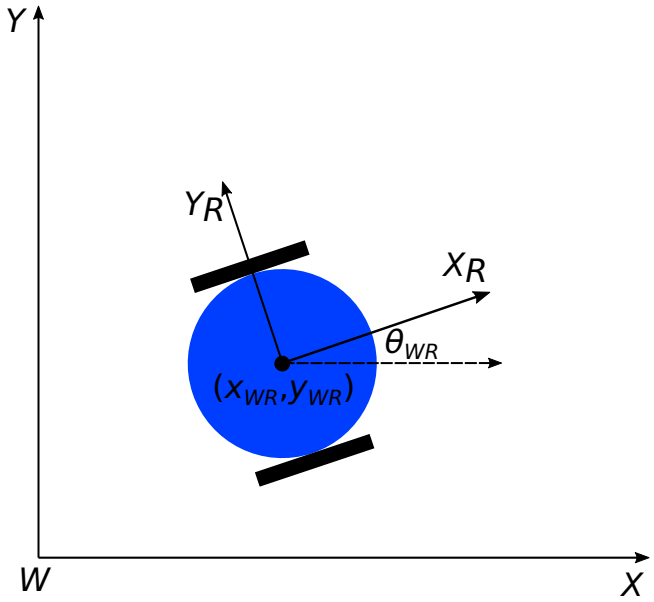

Figure 1: Schematic of a differential drive robot

Let $t_{WR} \triangleq (x_{WR}, y_{WR}) \in \mathbb{R}^2$ and $\theta_{WR} \in \mathbb{R}$ denote the position and orientation angle of the robot in the world coordinate frame W, as shown in Fig. 1. We showed in class that the equations of motion for this vehicle are:

$$\begin{pmatrix} \dot{x}_{WR} \\ \dot{y}_{WR} \\ \dot{\theta}_{WR} \end{pmatrix} = \begin{pmatrix} R(\theta_{WR}) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{r}{2}(\dot{\varphi}_r + \dot{\varphi}_l) \\ 0 \\ \frac{r}{w}(\dot{\varphi}_r - \dot{\varphi}_l) \end{pmatrix}, \tag{1}$$

where $\dot{\varphi}_l$ and $\dot{\varphi}_r$ are the angular speeds of the left and right wheels, respectively, with positive values corresponding to forward motion (that is, motion along the robot's body-centric positive x-axis, $+x_R$), and

$$\begin{aligned} R&\colon \mathbb{R} \to \mathrm{SO}(2) \\ R(\theta) &\triangleq \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \end{aligned} \tag{2}$$

is the mapping that assigns to each *angle* θ the corresponding *rotation matrix* $R(\theta)$.

In this exercise, we will reformulate the equations of motion (1) on the Lie group SE(2):

$$\mathrm{SE}(2) \cong \left\{ \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \in \mathbb{R}^{3\times 3} \mid R \in \mathrm{SO}(2),\, t \in \mathbb{R}^2 \right\}, \tag{3}$$

and then apply Lie group theory to perform *forward simulation* of the vehicle's trajectory.

(a) Let's begin by defining the mapping:

$$\begin{aligned} &\Psi \colon \mathbb{R}^2 \times \mathbb{R} \to \mathrm{SE}(2) \\ &\Psi(t,\theta) \triangleq \begin{pmatrix} R(\theta) & t \\ 0 & 1 \end{pmatrix} \end{aligned} \tag{4}$$

that sends each pair (t, θ) to the element of SE(2) obtained by replacing the *angle* θ by its corresponding *rotation matrix* $R(\theta)$.

Notice that

$$\Psi(0,0) = \begin{pmatrix} R(0) & 0 \\ 0 & 1 \end{pmatrix} = I; \tag{5}$$

that is, the map Ψ sends the vector $(0,0) \in \mathbb{R}^2 \times \mathbb{R}$ to the identity $I \in \mathrm{SE}(2)$. This means that the *derivative map*:

$$d\Psi_{(0,0)} \colon T_{(0,0)}(\mathbb{R}^2 \times \mathbb{R}) \to \mathrm{Lie}(\mathrm{SE}(2)) \tag{6}$$

sends each tangent vector $(\dot{t}, \dot{\theta}) \in T_{(0,0)}(\mathbb{R}^2 \times \mathbb{R}) \cong \mathbb{R}^3$ to a tangent vector in $T_I(\mathrm{SE}(2)) \cong \mathrm{Lie}(\mathrm{SE}(2))$, the *Lie algebra* of SE(2). Derive a closed-form expression for the derivative map $d\Psi_{(0,0)}$ in (6).

(b) The differential drive kinematic equation (1) describes how the left and right wheel speeds $(\dot{\varphi}_l, \dot{\varphi}_r) \in \mathbb{R}^2$ determine the robot's velocity $(\dot{t}, \dot{\theta}) \in T_{(t,\theta)}(\mathbb{R}^2 \times \mathbb{R})$. Suppose that the robot is at the origin $(0,0) \in \mathbb{R}^2 \times \mathbb{R}$. Using the result of part (a), derive the corresponding mapping:

$$\dot{\Omega} \colon \mathbb{R}^2 \to \mathrm{Lie}(\mathrm{SE}(2)) \tag{7}$$

that sends the pair of wheel speeds $(\dot{\varphi}_l, \dot{\varphi}_r)$ to the robot's velocity $\dot{\Omega}(\dot{\varphi}_l, \dot{\varphi}_r)$ in the Lie group SE(2) at I.

(c) The mapping $\dot{\Omega}$ from wheel speeds to velocities that you derived in part (b) was obtained under the assumption that the robot's pose in the world frame is $T_{WR} = I$.

Suppose now that the robot's pose in the world frame is $T_{WR} = X \in \mathrm{SE}(2)$. Derive a closed-form expression for the map:

$$V \colon \mathrm{SE}(2) \times \mathbb{R}^2 \to T(\mathrm{SE}(2)) \tag{8}$$

that accepts as input the robot's pose $X \in \mathrm{SE}(2)$ and wheel speeds $(\dot{\varphi}_l, \dot{\varphi}_r) \in \mathbb{R}^2$ and returns its velocity $V(X, \dot{\varphi}_l, \dot{\varphi}_r) \in T_X(\mathrm{SE}(2))$. You may write your answer in terms of the Lie algebra element $\dot{\Omega}(\dot{\varphi}_l, \dot{\varphi}_r)$.

[Hint: you may find it convenient to introduce an auxiliary coordinate frame F that is *fixed* with respect to the world frame W, and is aligned with the robot's body-centric frame – i.e., frame F is *defined* so that the pose of the robot in frame F is $T_{FR} = I$. What is the velocity of the robot with respect to frame F? How are the velocities of the robot in frames F and W related?]

(d) Suppose that we *fix a specific choice* of wheel speeds $(\dot{\varphi}_l, \dot{\varphi}_r) \in \mathbb{R}^2$. Given this choice, the map V that you derived in part (c) simplifies to a function $V_{(\dot{\varphi}_l, \dot{\varphi}_r)}(X)$ that assigns to each pose $X \in \mathrm{SE}(2)$ the velocity:

$$V_{(\dot{\varphi}_l, \dot{\varphi}_r)}(X) \triangleq V(X, \dot{\varphi}_l, \dot{\varphi}_r) \in T_X(\mathrm{SE}(2)). \tag{9}$$

In other words: each choice of wheel speeds $(\dot{\varphi}_l, \dot{\varphi}_r) \in \mathbb{R}^2$ determines a *vector field* $V_{(\dot{\varphi}_l, \dot{\varphi}_r)}$ on $\mathrm{SE}(2)$ that accepts as input a robot pose $X \in \mathrm{SE}(2)$, and returns the robot velocity $V_{(\dot{\varphi}_l, \dot{\varphi}_r)}(X)$ determined by the selected wheel speeds at X.

Prove that $V_{(\dot{\varphi}_l, \dot{\varphi}_r)}$ is in fact a *left-invariant* vector field. [Hint: You may find Problem 2 from Lab #1 useful here.]

(e) **Forward kinematics:** Suppose that the robot is at pose $X_0 \in \mathrm{SE}(2)$ at time $t = 0$, and that we drive its wheels at constant velocity $(\dot{\varphi}_l, \dot{\varphi}_r) \in \mathbb{R}^2$. Given this initial data, write down a closed-form formula for the curve:

$$\gamma \colon \mathbb{R} \to \mathrm{SE}(2) \tag{10}$$

that reports the robot's pose $\gamma(t)$ at time t. You may express your answer in terms of $\dot{\Omega}(\dot{\varphi}_l, \dot{\varphi}_r)$.

(f) **Inverse kinematics:** Now suppose that the robot is at pose $X \in \mathrm{SE}(2)$, and we would like to drive it to pose $Y \in \mathrm{SE}(2)$. Using the result of part (e), determine what (constant) wheel speeds $(\dot{\varphi}_l, \dot{\varphi}_r) \in \mathbb{R}^2$ should be commanded so that the robot leaves pose X at time $t = 0$ and arrives at pose Y at time $t = T$.

Problem 2: Turtlebot3 kinematics

Let's get some robots driving around! Go to this repository, and follow the instructions in the "Simulation setup" section to create a Gazebo simulation instance with a Turtlebot3.

(a) **Kinematic parameters and control limits:** In order to control our robots using the differential drive kinematic model, we must first know the parameters w and r that describe the robot's geometry. The ROBOTIS manual indicates that the nominal track width and wheel radius for the Turtlebot3 Burger are $w = .16$ m and $r = .033$ m, respectively.

Furthermore, in order to plan trajectories that are *feasible* (that is, that our robots can actually physically *follow*), we must also know the *control limits*: in this case, the maximum wheel speeds that we can actually command for the Turtlebot 3. While this is not directly stated in the linked manual, it does indicate that the Turtlebot3 Burger has a maximum (linear) speed of $v_{\max} = .22$ m/s. What is the maximum wheel speed $\dot{\varphi}_{\max}$ that would correspond to this maximum linear velocity?

(b) Using ROS, we can directly specify our robot's desired velocity by sending commands to the `/cmd_vel` topic. The message type for this topic is `geometry_msgs/Twist` (see here for the documentation), which specifies the desired translational and rotational velocities *in the robot's body-centric coordinate frame*. That is, the desired velocity is specified using an element of the Lie algebra SE(3)!

Now since our Turtlebots move only on the ground plane (i.e. on SE(2)), not all 6 degrees of freedom in SE(3) are feasible for our robots. Specifically, our vehicle's kinematics do not permit *lateral* motion (so $\dot{y}_R \equiv 0$), and similarly they have no means of moving in the

vertical direction (so $\dot{z}_R \equiv 0$). Similarly, the flat-floor assumption means that the vehicle's *pitch* and *roll* are always 0, and therefore the only change in the vehicle's orientation is due to changes in *yaw* (that is, *heading*), which corresponds to rotation about the vehicle's $+Z_R$ axis. Thus, the velocity messages for our Turtlebots will take the form:

$$\dot{\mu} = \begin{pmatrix} \dot{x}_R \\ 0 \\ 0 \\ 0 \\ 0 \\ \dot{\theta}_R \end{pmatrix}. \tag{11}$$

Suppose that we would like our robot to follow a circular path of radius $R = 1.5$ m counterclockwise at a (linear) velocity of $v = .2$ m/s. Calculate the message $\dot{\mu}$ in (11) that determines this trajectory, and verify that it is dynamically feasible.

(c) In simulation, send the velocity command $\dot{\mu}$ that you computed in part (b) to the Turtlebot3, and verify that it follows the intended trajectory (i.e. a circle of radius $R = 1.5$ m, traversed in a counterclockwise direction).

[Hint: For this exercise, you may find it convenient to use the `rospub` command. This is a handy utility that lets you construct and publish simple ROS messages to a specified topic directly from the command line. In this case, the command would take the form:

```
rostopic pub /cmd_vel geometry_msgs/Twist -r 10 '[dx, 0, 0]' '[0, 0, dtheta]'
```

to publish the velocity message $(\dot{x}, 0, 0, 0, 0, \dot{\theta})$ to the topic `/cmd_vel` at a rate of 10 Hz.]

Problem 3: Visual servoing

In this exercise you will apply your knowledge of frame transformations and differential drive kinematics to implement a simple visual servoing behavior on the Turtlebot3.

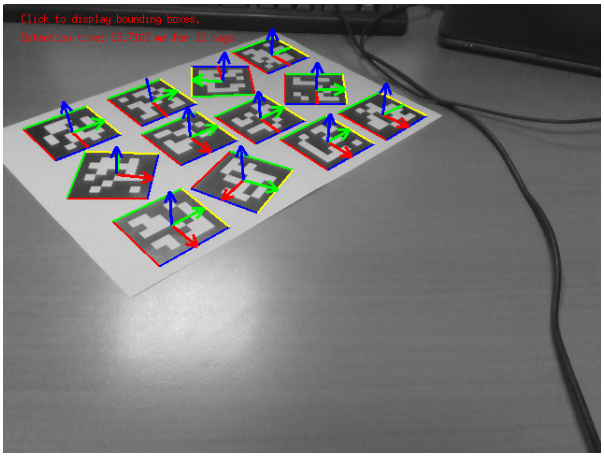

Figure 2: AprilTags detected in an image

AprilTags are simple 2D barcodes (similar to QR codes) that are often used as visual fiducials in robotics applications. Because the corners of these tags are easy to detect in images, and the

tags themselves have a *known geometry*, it is possible to estimate their pose relative to a calibrated camera from a *single image*.

The goal of this exercise will be to drive a Turtlebot3 to a target pose that is specified with respect to an AprilTag – this is the same basic behavior that industrial and household robots (e.g. Roombas) use when "auto-docking" themselves to a base station.

Setup: Go to this repository, and follow the instructions in the "Lab 2" section to install the ROS AprilTag package and set up a Gazebo simulation environment containing an AprilTag.

Task: Your task in this exercise will be to write a simple ROS node that performs the following functions:

(i) At startup, it subscribes to the `/tag_detections` topic. This topic's messages report any AprilTags detected in the Turtlebot's RGB camera, together with their pose in the camera's body-centric coordinate frame.

(ii) Once it receives an initial AprilTag detection, your node must calculate a *dynamically feasible* trajectory to drive your Turtlebot3 to the unique pose $X \in \mathrm{SE}(2)$ satisfying the following conditions:

   (a) The projection of the AprilTag's centerpoint onto the ground plane is at position (.12, 0, 0) in the robot's `base_footprint` frame.

   (b) The AprilTag's body-centric $+Z$ axis is anti-parallel to the `base_footprint` frame's $+X$ axis.

   In words, conditions (a) and (b) above correspond to the configuration in which the tag is 12cm in front of the robot, and the robot is directly facing the "front" of the tag.

   [Note: You may find the results of Problem 1 useful for computing the trajectory here.]

(iii) Once it has computed a dynamically feasible trajectory, your robot must publish velocity commands to the robot on the `/cmd_vel` topic to follow the trajectory to the target pose, and then halt the robot when it arrives.