Lecture 15: Differential Drive Robots: Kinematics
Rounding up Logistics:

Recap:
• 2D continuous space
• Omnidirectional motion
• Gaussian motion model
• Complex sensors
• Sophisticated perception

One approach to path planning:
• Place high reward at the goal configuration.
• Place negative reward along obstacle edges.
• Compute the value function.
• Follow the gradient of the value function from the current configuration to the goal.
Reminder about Value Iteration

Recall the Value Iteration Equation:

\[ V^{k+1}(x) \leftarrow \max_a \left\{ R(x, a) + \gamma \sum_{x'} P(x'|x, a) V^k(x') \right\} \]

- \( R(x, a) \) is the expected reward for applying action \( a \) in state \( x \).
- \( P(x'|x, a) \) is the motion model.
- \( V^k(x') \) is the approximation to the optimal value function at iteration \( k \).

- For classical path planning, we ignore uncertainty, so that \( P(x'|x, a) = 1 \) for the deterministic outcome, and \( P(x'|x, a) = 0 \) for all other outcomes.

- If we add a bit of uncertainty, we increase robustness of the plan (e.g., decrease the probability that the robot might accidentally collide with an obstacle).
Value Iteration in the Warehouse

• In our warehouse example, we set the reward to 100 for the goal, and to -50 for obstacle edges, and used value iteration to compute the value function:

• To move from *any* point in the warehouse toward the goal, simply follow the gradient of the value function!
Value Iteration at Work

- *Value iteration is expensive.*
- At each iteration, we must compute $N$ updates (one for each possible state).
- For the warehouse, we have $N = 50 \times 100 = 5000$.
- Many problems have much larger state spaces, which makes value iteration impractical (and possibly intractable).

➢ At each iteration, every grid cell updates its estimate of the optimal value function.
Our logistics robot had super simple kinematics:
• Thanks to omni-wheels, the logistics robot could roll in any direction at any time.
• Because of this, there was no need to pay attention to the orientation of the robot.
• We didn’t really worry about a body-attached coordinate frame, since the robot frame was always parallel to the world frame.

Differential Drive Robots don’t have omni-wheels...
• The kinematics (relationship between input commands and robot motion) are more interesting.
• We need to explicitly pay attention to the orientation.
Mobile Robots

• There are many kinds of wheeled mobile robots.
• We have seen omni-directional robots.
• Now we’ll study *differential drive robots*.

Mobile Robot Kinematics

• Relationship between input commands (e.g., wheel velocity) and pose of the robot, not considering forces. *If the wheels turn at a certain rate, what is the resulting robot motion?*
• No direct way to measure pose (unless we sensorize the environment), but we can integrate velocity (odometry) to obtain a good estimate.
More Modern AGVs
Differential Drive Robots

Two wheels with a common axis, and that can spin independently
The Duckiebot Platform

A typical DDR, with two actuated wheels in front, and a passive castor wheel in the back.
Differential Drive Robots

Differential drive robots (aka DDRs):

- Two actuated wheels that share an axis
- A castor wheel that rotates freely, mainly to stabilize the robot (three points define a plane – castor wheel keeps the robot from tipping over).
Differential Drive Robots

To specify the position and orientation of the DDR, we attach a coordinate frame to the robot.

- This frame is called the **body-attached frame**, or the robot frame.
- The body attached frame is **rigidly** attached to the robot: it translates and rotates with the robot.
- The origin of the body-attached frame is located at the midpoint between the two wheels along their axis of rotation.
- The $x$-axis is the steering (or driving) direction of the robot.
- The $y$-axis is coincident with the common axis.
Configuration Space

- A **configuration** is a complete specification of the position of every point in a robot system.
- The **configuration space** is the set of all configurations.
- We use $q$ to denote a point in a configuration space $Q$.

Example:
- Our logistics robot was able to translate in the plane.
- It’s orientation never changed (i.e., it could not rotate).
- We can attach a coordinate frame with origin at the center of the robot, and axes parallel to the world $x$- and $y$-axes.
- If we know the $x, y$ coordinates of this coordinate frame, we can easily determine the location of any desired point on the robot w.r.t. the world coordinate frame (in the $x$-$y$ plane).
Configuration Space

• A **configuration** is a complete specification of the position of every point in a robot system.
• The **configuration space** is the set of all configurations.
• We use $q$ to denote a point in a configuration space $Q$.

Example:
• Our logistics robot was able to translate in the plane.
• It’s orientation never changed (i.e., it could not rotate).
• We can attach a coordinate frame with origin at the center of the robot, and axes parallel to the world $x$- and $y$-axes.
• If we know the $x$, $y$ coordinates of this coordinate frame, we can easily determine the location of any desired point on the robot w.r.t. the world coordinate frame (in the $x$-$y$ plane).
• For example, if the robot has radius $R$, then the center of wheel 2 is:

$$
\begin{bmatrix}
  x_2 \\
  y_2
\end{bmatrix} = \begin{bmatrix}
  x_{robot} + R \cos \theta_2 \\
  y_{robot} + R \sin \theta_2
\end{bmatrix}
$$
Configuration Space

• A **configuration** is a complete specification of the position of every point in a robot system.

• The **configuration space** is the set of all configurations.

• We use $q$ to denote a point in a configuration space $Q$.

In this example, the configuration space is easy to characterize:

$$Q = \mathcal{D} \subset \mathbb{R}^2$$
$$q = (x, y) \in \mathcal{D}$$

where $\mathcal{D}$ is the floor space of the warehouse.

- Given $q = (x, y)$, we can calculate the position of any point on the robot.
- Note: This assumes, of course, that we have a model of our robot, which we do.
Because our DDR can rotate in the plane, it is necessary to know both the position and the orientation of the body-attached frame to specify a configuration:

\[ Q = \mathbb{R}^2 \times [0,2\pi) \]

\[ q = (x, y, \theta) \in Q \]

If we know the configuration, \( q = (x, y, \theta) \), we can compute the location of any point on the robot.

Let’s start with the wheel centers.
If the robot is in configuration $q = (x, y, \theta)$, the left and right wheel centers are located at:

$$
\begin{bmatrix}
x_{\text{left}} \\
y_{\text{left}}
\end{bmatrix} = \begin{bmatrix}
x - \frac{L}{2} \sin \theta \\
y + \frac{L}{2} \cos \theta
\end{bmatrix}
$$

and

$$
\begin{bmatrix}
x_{\text{right}} \\
y_{\text{right}}
\end{bmatrix} = \begin{bmatrix}
x + \frac{L}{2} \sin \theta \\
y - \frac{L}{2} \cos \theta
\end{bmatrix}
$$
Kinematics of DDRs

- We can generalize this to any point \( p \) on the DDR.
- Suppose the coordinates of \( p \) in the body frame are given by
  \[
  p_{\text{body}} = [p_x, p_y]
  \]

- If the robot is in configuration \( q = (x, y, \theta) \), the coordinates of \( p \) in the world frame are given by:
  \[
  p_{\text{world}} = \begin{bmatrix}
  x + p_x \cos \theta - p_y \sin \theta \\
  y + p_x \sin \theta + p_y \cos \theta
  \end{bmatrix}
  \]

Soon, we will generalize this technique using homogeneous coordinate transformations... but not today.
Differential Drive Robots are very different from robots with omni-wheels:

- The wheels roll without slipping – *no sideways motion*.
- The instantaneous velocity of the robot is always in the steering direction.
- The velocity perpendicular to the steering direction is always zero.

If the robot follows the curve $\gamma(s)$, the instantaneous velocity $\mathbf{v}$ is tangent to $\gamma$. 
Velocity of the DDR

- Since the robot cannot move in the direction of the body-attached \( y \)-axis, its linear velocity, when expressed with respect to the body frame is:
  \[
  \mathbf{v}_{\text{body, linear}} = \begin{bmatrix} v_x \\ 0 \end{bmatrix}
  \]

- The steering direction, expressed w.r.t. the world frame, is given by:
  \[
  \begin{bmatrix} \cos \theta \\ \sin \theta \end{bmatrix}
  \]

- Therefore, when the robot is in configuration \( q(s) = (x, y, \theta) \), its linear velocity is expressed with respect to the world frame by:
  \[
  \mathbf{v}_{\text{world, linear}} = \begin{bmatrix} v_x \cos \theta \\ v_x \sin \theta \end{bmatrix}
  \]
Angular Velocity of the DDR

- The orientation of the robot is given by the angle $\theta$.
- Since this robot is able to rotate, $\theta$ can be considered as a function of time.
- We define the robot’s angular velocity as
  \[ \omega = \frac{d}{dt} \theta = \dot{\theta} \]
- Note that the positive sense for $\omega$ is defined using the right-hand rule: point the thumb of your right hand in the direction of the world $z$-axis, and your fingers will curl in the positive $\theta$ direction.
Total Velocity of the DDR

• The velocity of the DDR includes both the linear and angular velocities.
• We stack these into a single vector to describe the robot’s instantaneous velocity w.r.t. the body frame of the world frame:

\[
\begin{align*}
\mathbf{v}_{body} &= \begin{bmatrix} v_x \\ 0 \\ \dot{\theta} \end{bmatrix}, \\
\mathbf{v}_{world} &= \begin{bmatrix} v_x \cos \theta \\ v_x \sin \theta \\ \dot{\theta} \end{bmatrix}
\end{align*}
\]

➢ Note that the \( z \)-axis of the body-attached frame \textbf{is the same as} the \( z \)-axis of the world frame, so that the angular velocity is given by \( \dot{\theta} \) for both of these coordinate frames.
Wheel Actuation and DDR Velocity

• The two wheels of the DDR are independently actuated, and able to spin in both directions.
• Let $\phi_R$ and $\phi_L$ denote the instantaneous orientation of the right and left wheels (e.g., the angle from the world $z$-axis to some identifiable mark on the wheel).
• The angular speeds of the wheels are therefore given by $\dot{\phi}_R$ and $\dot{\phi}_L$.

As we saw with the omni-directional robot, the relationship between forward speed of the wheel and its angular speed is given by

$$\frac{\text{d}}{\text{d}t}x = r\dot{\phi}$$

and therefore, since the wheel rolls without slipping, and $v_y = 0$, we have

$$\dot{\phi} = \frac{v_x}{r}$$
Wheel Actuation and DDR Velocity

When the two wheels turn with the same angular speed, $\dot{\phi}_R = \dot{\phi}_L$, the robot moves with pure translation.

In this case,
- $v_{\text{left}} = v_{\text{right}}$
- Both $v_{\text{left}}$ and $v_{\text{right}}$ are parallel to the steering direction.
- The robot’s angular velocity is zero (i.e., $\omega = 0$).
- The angular wheel speed is related to the robot’s linear velocity by

$$\dot{\phi}_R = \dot{\phi}_L = \frac{v_x}{r}$$
Total DDR Velocity

When the two wheels turn with the opposite angular velocity, \( \dot{\phi}_R = -\dot{\phi}_L \), the robot moves with pure rotation.

In this case,
- \( v_{\text{left}} = -r \dot{\phi}_L \) and \( v_{\text{right}} = r \dot{\phi}_R \)
- Both \( v_{\text{left}} \) and \( v_{\text{right}} \) are tangent to the circle centered at the origin of the body-attached frame w/radius 0.5\( L \).
- The robot’s angular velocity satisfies the eqn for circular motion:
  \[
  \frac{L}{2} \omega = v_{\text{right}} = r \dot{\phi}_R
  \]
  and
  \[
  \frac{L}{2} \omega = -v_{\text{left}} = -r \dot{\phi}_L
  \]
- Rearranging terms, we obtain:
  \[
  \dot{\phi}_R = \frac{L}{2} \frac{\omega}{r} \quad \text{and} \quad \dot{\phi}_L = -\frac{L}{2} \frac{\omega}{r}
  \]
Wheel Actuation and DDR Velocity

All other velocities are linear combinations of these two cases, and therefore we can apply superposition.

- For pure translation we have:
  \[ \dot{\varphi}_0 = \frac{v_x}{r} \quad \text{and} \quad \dot{\varphi}_1 = \frac{v_x}{r} \]

- For pure rotation we have:
  \[ \dot{\varphi}_0 = \frac{L \omega}{2r} \quad \text{and} \quad \dot{\varphi}_1 = -\frac{L \omega}{2r} \]

- Combining (adding) the two equations for \( \dot{\varphi}_0 \) and \( \dot{\varphi}_1 \) we obtain:
  \[ \dot{\varphi}_0 = \frac{L \omega}{2r} + \frac{v_x}{r} \quad \text{and} \quad \dot{\varphi}_1 = -\frac{L \omega}{2r} + \frac{v_x}{r} \]

\[ \text{These two equations tell us how to choose } \dot{\varphi}_0 \text{ and } \dot{\varphi}_1 \text{ to achieve a desired velocity } v, \omega. \]
Wheel Actuation and DDR Velocity

All other velocities are linear combinations of these two cases, and therefore we can apply superposition.

• The equations

\[ \dot{\phi}_0 = \frac{L \omega}{2r} + \frac{v_x}{r} \quad \text{and} \quad \dot{\phi}_1 = -\frac{L \omega}{2r} + \frac{v_x}{r} \]

are sometimes called inverse equations, since they solve the inverse problem: “what wheel speed (input) to I need to achieve a desired robot behavior (output)?”

• We can easily compute the forward mapping, from \( \dot{\phi}_R \) and \( \dot{\phi}_L \) to \( v, \omega \) using simple algebra:

\[ \dot{\phi}_R + \dot{\phi}_L = 2 \frac{v_x}{r} \Rightarrow v_x = \frac{r}{2}(\dot{\phi}_R + \dot{\phi}_L) \]

\[ \dot{\phi}_R - \dot{\phi}_L = \frac{L \omega}{r} \Rightarrow \omega = \frac{r}{L}(\dot{\phi}_R - \dot{\phi}_L) \]
We can express these equations relative to the body-attached frame or the world frame:

\[
\begin{align*}
&v_{\text{body}} = \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{r}{2} (\dot{\phi}_R + \dot{\phi}_L) \\ 0 \\ \frac{r}{L} (\dot{\phi}_R - \dot{\phi}_L) \end{bmatrix} \\
&v_{\text{world}} = \begin{bmatrix} v_x \cos \theta \\ v_x \sin \theta \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r}{2} (\dot{\phi}_R + \dot{\phi}_L) \cos \theta \\ \frac{r}{2} (\dot{\phi}_R + \dot{\phi}_L) \sin \theta \\ \frac{r}{L} (\dot{\phi}_R - \dot{\phi}_L) \end{bmatrix}
\end{align*}
\]