Robotics Programming Laboratory

Bertrand Meyer
Jiwon Shin

Lecture 4: Robot Control and Obstacle Avoidance
Go forward, go right

Holonomic  
DDOF = DOF

Robot1

Nonholonomic  
DDOF < DOF

Robot2

DOF: Ability to achieve various poses
DDOF: Ability to achieve various velocities
Differential drive

Forward: \( \varphi_L = \varphi_R > 0 \)

Backward: \( \varphi_L = \varphi_R < 0 \)

Right turn: \( \varphi_L > \varphi_R \)

Left turn: \( \varphi_L < \varphi_R \)
Differential drive

Input: \((v, \omega)\)

\[
\begin{align*}
\dot{x} &= v \cos \theta \\
\dot{y} &= v \sin \theta \\
\dot{\theta} &= \omega
\end{align*}
\]
Differential drive

\[
\begin{align*}
\dot{x} &= R \frac{\phi_L + \phi_R}{2} \cos \theta \\
\dot{y} &= R \frac{\phi_L + \phi_R}{2} \sin \theta \\
\dot{\theta} &= \frac{R}{B} (\phi_R - \phi_L)
\end{align*}
\]
Odometry for small t

\[ x(t) = x(t - 1) + d_c \cos \theta(t) \]
\[ y(t) = y(t - 1) + d_c \sin \theta(t) \]
\[ \theta(t) = \theta(t - 1) + \theta_c \]

\[ d_c = \frac{1}{2} (d_L + d_R) \]
\[ \theta_c = \frac{d_R - d_L}{B} \]
More accurate odometry for small $t$

\[
\begin{align*}
\theta_c &= \arctan\left( \frac{d_R - d_L}{B} \right) \\
d_c &= \frac{1}{2} (d_L + d_R) \\
x(t) &= x(t-1) + d_c \cos(\theta(t-1) + \frac{1}{2} \theta_c) \\
y(t) &= y(t-1) + d_c \sin(\theta(t-1) + \frac{1}{2} \theta_c) \\
\theta(t) &= \theta(t-1) + \theta_c
\end{align*}
\]
Wheel encoder

How do we get the distance each wheel has moved?

- If the wheel has $N$ ticks per revolution:

$$\Delta n_{\text{tick}} = n_{\text{tick}}(t) - n_{\text{tick}}(t - 1)$$

$$d = 2\pi R \frac{\Delta n_{\text{tick}}}{N}.$$ 

- Thymio: $d = d \Delta t$

**DRIFT**
Go to goal

Goal

Control
A collection of two or more dynamical systems, in which each system influences the other, resulting in strongly-coupled dynamics.

- **Open loop**: the systems are not interconnected (no feedback)

- **Closed loop**: the systems are interconnected (with feedback)
Control

The use of algorithms and feedback in engineered systems

Robot speed control

- **Actuator**: set the robot’s speed
- **Sensor**: sense the robot’s actual speed
- **Control goals**: set the robot’s speed such that:
  - **Stability**: the robot maintains the desired speed
  - **Performance**: the robot responds quickly to changes
  - **Robustness**: the robot tolerates perturbation in dynamics
On-off controller

\[ u = \begin{cases} 
    u_{\text{max}} & \text{if } e > 0 \\
    u_{\text{min}} & \text{if } e < 0 
\end{cases} \]
Proportional controller

\[ u(t) = k_p e(t) \]

More about control gains
Proportional derivative controller

\[ u(t) = k_p e(t) + k_d \frac{de(t)}{dt} \]

More about control gains
Proportional integral derivative controller

\[ u(t) = k_p e(t) + k_i \int_0^t e(\tau) d\tau + k_d \frac{de(t)}{dt} \]

More about control gains
Control gains

Ziegler-Nicols method

- Set $K_i$ and $K_d$ to 0.
- Increase $K_p$ until $K_u$ at which point the output starts to oscillate.
- Use $K_u$ and the oscillation period $T_u$ to set the control gains.

<table>
<thead>
<tr>
<th>Control Type</th>
<th>$K_p$</th>
<th>$K_i$</th>
<th>$K_d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>$0.50K_u$</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PI</td>
<td>$0.45K_u$</td>
<td>$1.2K_p/T_u$</td>
<td>-</td>
</tr>
<tr>
<td>PID</td>
<td>$0.60K_u$</td>
<td>$2K_p/T_u$</td>
<td>$K_pT_u/8$</td>
</tr>
</tbody>
</table>

Manual tuning!
P, PI, PID, .....?

\[ u(t) = k_p e(t) + k_i \int_0^t e(\tau) d\tau + k_d \frac{de(t)}{dt} \]

\( k_p, k_i, k_d \neq 0 \)

\( k_i = 0 \)

\( k_d = 0 \)

\( k_p = 0 \)
Obstacle avoidance
Obstacle avoidance
Bug 1

1. Move toward the goal:
   1. If the goal is reached: Stop
   2. If an obstacle is in the way: Go to step 2

2. Follow the obstacle boundary:
   1. Mark the closest
   2. After a complete loop: Go to the closest point to the goal then go back to step 1.

Lumelsky, V. & Stepanov, A. “Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape,”. Algorithmica 2:403-430. 1987
1. **Move toward the goal:**
   1. If the goal is reached: Stop
   2. If an obstacle is in the way: Go to step 2

2. **Follow the obstacle boundary:**
   1. If the goal line is crossed: Go to step 1.

Is Bug 2 always better than Bug 1?

Bug 1
- Exhaustive search: analyze all choices before committing

Bug 2
- Greedy search: take the first viable choice

Goal
1. **Move toward the goal:**
   1. If the goal is reached: Stop
   2. If a local minimum is detected: Go to step 2

2. **Move along the boundary marking** $d_{\text{min}}$:
   1. If the goal is reached: Stop
   2. If $d(V_{\text{leave}, \text{goal}}) < d_{\text{min}}$: Go to step 3

3. **Perform the transition phase:**
   1. Move directly towards $V_{\text{leave}}$ until $Z$, where $d(Z, \text{goal}) < d_{\text{min}}$: Go to step 1

---

Visibility graph & tangent graph

Visibility graph

Tangent graph
Local tangent graph
Local minimum detection

\[ d(V, \text{goal}) < d(x, \text{goal}) \text{ for all } V \]
Wall Following

\[ v_{\text{wall}} = p_2 - p_1 \]

\[ v_{\text{distance}} = (d_{\text{current}} - d_{\text{desired}}) \cdot v_{\text{perpendicular}} \]

\[ v_{\text{robot}} = d_{\text{desired}} \cdot v_{\text{wall}} + v_{\text{distance}} \]
Leave condition detection

\[ d(V_{\text{leave}}, \text{goal}) < d_{\text{min}} \]
Unreachable goal
Loop closure

Challenging!

- Drift
- Limited sensor information