COS 495 - Lecture 5
Autonomous Robot Navigation

Instructor: Chris Clark
Semester: Fall 2011

Figures courtesy of Siegwart & Nourbakhsh
Control Structure

Prior Knowledge  →  Localization  →  Perception  →  Motion Control

Operator Commands  →  Cognition

Operator Commands  →  Prior Knowledge
Motion Uncertainty

1. Odometry & Dead Reckoning
2. Modeling motion
3. Odometry on the X80
Odometry & Dead Reckoning

- Odometry
  - Use wheel sensors to update position
- Dead Reckoning
  - Use wheel sensors and heading sensor to update position
- Straight forward to implement
- Errors are integrated, unbounded
Odometry & Dead Reckoning

- Odometry Error Sources
  - Limited resolution during integration (time increments, measurement resolution).
  - Unequal wheel diameter (deterministic)
  - Variation in the contact point of the wheel (deterministic)
  - Unequal floor contact and variable friction can lead to slipping (non deterministic)
Odometry & Dead Reckoning

- Odometry Errors
  - Deterministic errors can be eliminated through proper calibration
  - Non-deterministic errors have to be described by error models and will always lead to uncertain position estimate.
Motion Uncertainty

1. Odometry & Dead Reckoning
2. Modeling motion
3. Odometry on the X80
Modeling Motion

- If a robot starts from a position $p$, and the right and left wheels move respective distances $\Delta s_r$ and $\Delta s_l$, what is the resulting new position $p'$?
Modeling Motion

- To start, let’s model the change in angle $\Delta \theta$ and distance travelled $\Delta s$ by the robot.
  - Assume the robot is travelling on a circular arc of constant radius.
Modeling Motion

- Begin by noting the following holds for circular arcs:

\[
\Delta s_l = R\alpha \\
\Delta s_r = (R+2L)\alpha \\
\Delta s = (R+L)\alpha
\]
Modeling Motion

- Now manipulate first two equations:

\[ \Delta s_l = R\alpha \quad \Delta s_r = (R+2L)\alpha \]

To:

\[ R\alpha = \Delta s_l \]
\[ L\alpha = (\Delta s_r - R\alpha)/2 \]
\[ = \Delta s_r/2 - \Delta s_l/2 \]
Modeling Motion

- Substitute this into last equation for $\Delta s$:

$$
\Delta s = (R+L)\alpha \\
= R\alpha + L\alpha \\
= \Delta s_l + \Delta s_r / 2 - \Delta s_l / 2 \\
= \Delta s_l / 2 + \Delta s_r / 2 \\
= \frac{\Delta s_l + \Delta s_r}{2}
$$
Modeling Motion

- Or, note the distance the center travelled is simply the average distance of each wheel:

\[ \Delta s = \frac{\Delta s_r + \Delta s_l}{2} \]
Modeling Motion

- To calculate the change in angle $\Delta \theta$, observe that it equals the rotation about the circular arc’s center point

$$
\Delta \theta = \alpha
$$
So we solve for $\alpha$ by equating $\alpha$ from the first two equations:

\[
\Delta s_l = R\alpha \quad \Delta s_r = (R+2L)\alpha
\]

This results in:

\[
\frac{\Delta s_l}{R} = \frac{\Delta s_r}{(R+2L)}
\]

\[
(R+2L) \Delta s_l = R \Delta s_r
\]

\[
2L \Delta s_l = R (\Delta s_r - \Delta s_l)
\]

\[
2L \Delta s_l = R
\]

\[
(R \Delta s_r - \Delta s_l)
\]
Modeling Motion

- Substitute $R$ into

\[
\alpha = \frac{\Delta s_I}{R} = \frac{\Delta s_I (\Delta s_r - \Delta s_l)}{(2L \Delta s_I)} = \frac{(\Delta s_r - \Delta s_l)}{2L}
\]

So...

\[
\Delta \theta = \frac{(\Delta s_r - \Delta s_l)}{2L}
\]
Modeling Motion

- Now that we have $\Delta \theta$ and $\Delta s$ we can calculate the position change in global coordinates.
  - We use a new segment of length $\Delta d$. 

![Diagram showing motion with coordinates and angles labeled]
Modeling Motion

- Now calculate the change in position as a function of $\Delta d$. 

\[\Delta d \quad \theta + \Delta \theta/2 \quad \Delta x \quad \Delta y\]

\[\theta + \Delta \theta\]
Modeling Motion

- Using Trig:
  \[ \Delta x = \Delta d \cos(\theta + \Delta \theta/2) \]
  \[ \Delta y = \Delta d \sin(\theta + \Delta \theta/2) \]
Now if we assume that the motion is small, then we can assume that $\Delta d \approx \Delta s$:

- $\Delta x = \Delta s \cos(\theta + \Delta \theta/2)$
- $\Delta y = \Delta s \sin(\theta + \Delta \theta/2)$
Modeling Motion

- Summary:

\[
\begin{align*}
\Delta x &= \Delta s \cos(\theta + \Delta \theta / 2) \\
\Delta y &= \Delta s \sin(\theta + \Delta \theta / 2) \\
\Delta \theta &= \frac{\Delta s_r - \Delta s_l}{b} \\
\Delta s &= \frac{\Delta s_r + \Delta s_l}{2} \\
p' &= f(x, y, \theta, \Delta s_r, \Delta s_l) = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \\
&\quad \begin{bmatrix} \frac{\Delta s_r + \Delta s_l}{2} \cos\left(\theta + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\
\frac{\Delta s_r + \Delta s_l}{2} \sin\left(\theta + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\
\frac{\Delta s_r - \Delta s_l}{b} \end{bmatrix}
\end{align*}
\]
Modeling Uncertainty in Motion

- Let’s look at delta terms as errors in wheel motion, and see how they propagate into positioning errors.
  - Example: the robot is trying to move forward 1 m on the x axis.

If:

\[ \Delta s = 1 + e_s \]
\[ \Delta \theta = 0 + e_\theta \]

where \( e_s \) and \( e_\theta \) are error terms.
Modeling Uncertainty in Motion

- According to the following equations, the error $e_s = 0.001\text{m}$ produces errors in the direction of motion.

\[
\begin{align*}
\Delta x &= \Delta s \cos(\theta + \Delta \theta/2) \\
\Delta y &= \Delta s \sin(\theta + \Delta \theta/2)
\end{align*}
\]

- However, the $\Delta \theta$ term affects each direction differently.

If $e_\theta = 2\text{ deg}$ and $e_s = 0\text{ meters}$, then:

\[
\begin{align*}
\cos(\theta + \Delta \theta/2) &= 0.9998 \\
\sin(\theta + \Delta \theta/2) &= 0.0175
\end{align*}
\]
Modeling Uncertainty in Motion

- So
  \[ \Delta x = 0.9998 \]
  \[ \Delta y = 0.0175 \]
- But the robot is supposed to go to \( x=1, y=0 \), so the errors in each direction are
  \[ \Delta x = +0.0002 \]
  \[ \Delta y = -0.0175 \]
- THE ERROR IS BIGGER IN THE “Y” DIRECTION!
Modeling Uncertainty in Motion

- Errors perpendicular to the direction grow much larger.
Modeling Uncertainty in Motion

- Error ellipse does not remain perpendicular to direction.
Motion Uncertainty

1. Odometry & Dead Reckoning
2. Modeling Uncertainty in motion
3. Odometry on the X80
Odometry on the X80

- Goals:
  - Calculate the resulting robot position and orientation from wheel encoder measurements.
  - Display them on the GUI.

- Method:
  - Use the forward kinematics equation from last lecture.
Odometry on the X80

- Method cont’:
  - Make use of the fact that your encoder has resolution of 1200 counts per revolution. Be able to convert this to a distance travelled by the wheel.

\[
\tau \varphi_r = \Delta s_r
\]

- Given the distance travelled by each wheel, we can calculate the change in the robot’s distance and orientation.

\[
\Delta s = \frac{\Delta s_r + \Delta s_l}{2} \quad \Delta \theta = \frac{\Delta s_r - \Delta s_l}{2L}
\]
Odometry on the X80

- Method cont’:
  - Now you should be able to update the position/orientation in global coordinates.

\[
\Delta x = \Delta s \cos(\theta + \Delta \theta/2) \\
\Delta y = \Delta s \sin(\theta + \Delta \theta/2)
\]