

Lecture 15: Action and Sensor Models

CS 344R/393R: Robotics
Benjamin Kuipers

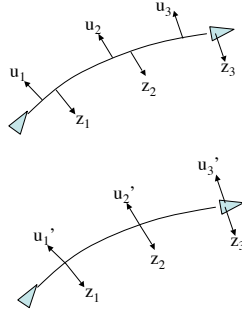
Action and Sensor Models

$$Bel(x_t) = \eta P(z_t | x_t) \int P(x_t | u_{t-1}, x_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

- The Markov localization equation depends on two types of knowledge about the robot.
- The **action model**: $P(x_t | u_{t-1}, x_{t-1})$
 - Given a state x_t and odometry u_t , the distribution over possible next states x_{t+1}
- The **sensor model**: $P(z_t | x_t)$
 - Given a state x_t , the distribution over possible sensor images z_t .

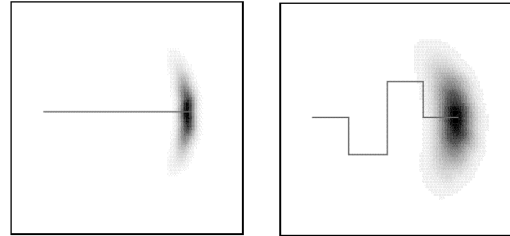
Interpolate Observation Times

- Odometry u_i and laser scans z_i actually arrive at slightly different times.
- Interpolate to give estimated odometry u_i' at the same time as the laser scan z_i .



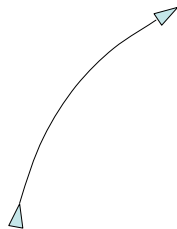
Action Model $P(x_t | u_{t-1}, x_{t-1})$

- Probability density function over poses, after traveling 40m or 80m.



The Action Error Model

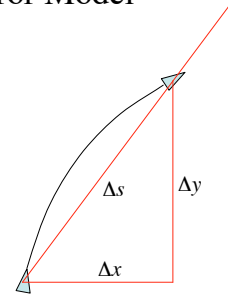
Suppose odometry gives:
 - (x_1, y_1, φ_1)
 - (x_2, y_2, φ_2)
 in a slowly drifting frame of reference.



The Action Error Model

From odometry get:
 - (x_1, y_1, φ_1)
 - (x_2, y_2, φ_2)
 in a slowly drifting frame of reference.

- Then:
- $(\Delta x, \Delta y, \Delta \varphi)$
 - $\Delta x^2 + \Delta y^2 = \Delta s^2$
 - Δs and $\Delta \varphi$ are relatively reliable, and independent of the frame of reference.



The Action Error Model

From odometry get:

- (x_1, y_1, φ_1)
- (x_2, y_2, φ_2)

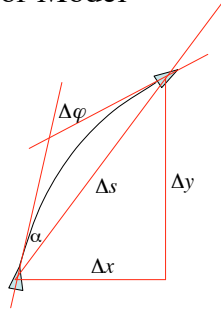
in a slowly drifting frame of reference.

Then:

- $(\Delta x, \Delta y, \Delta \varphi)$
- $\Delta x^2 + \Delta y^2 = \Delta s^2$

and:

- $\alpha = \text{atan2}(\Delta y, \Delta x) - \varphi_1$
- (Measure angle counter-clockwise from x -axis)



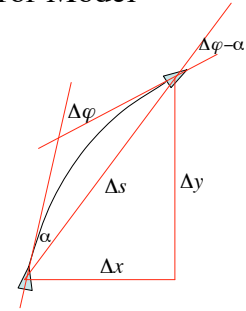
The Action Error Model

From odometry get:

- (x_1, y_1, φ_1)
- (x_2, y_2, φ_2)
- $(\Delta x, \Delta y, \Delta \varphi)$
- $\Delta x^2 + \Delta y^2 = \Delta s^2$
- $\alpha = \text{atan2}(\Delta y, \Delta x) - \varphi_1$

• Model the action as:

- Turn(α)
- Travel(Δs)
- Turn($\Delta \varphi - \alpha$)



The Action Error Model

• Model the action as:

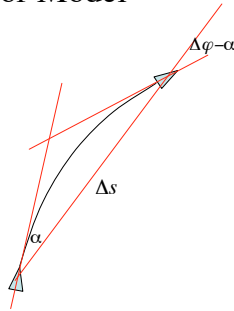
- Turn($\alpha + \varepsilon_1$)
- Travel($\Delta s + \varepsilon_2$)
- Turn($\Delta \varphi - \alpha + \varepsilon_3$)

where

- $\varepsilon_1 \sim N(0, k_1 \alpha)$
- $\varepsilon_2 \sim N(0, k_2 \Delta s)$
- $\varepsilon_3 \sim N(0, k_3 (\Delta \varphi - \alpha))$

• This combines three Gaussian errors.

- Std dev proportional to action magnitude



Tune the Action Error Model

• Model the action as:

- Turn($\alpha + \varepsilon_1$)
- Travel($\Delta s + \varepsilon_2$)
- Turn($\Delta \varphi - \alpha + \varepsilon_3$)

where

- $\varepsilon_1 \sim N(0, k_1 \alpha)$
- $\varepsilon_2 \sim N(0, k_2 \Delta s)$
- $\varepsilon_3 \sim N(0, k_3 (\Delta \varphi - \alpha))$

• Tune the model by finding k_1 and k_2 .

• Compute error between:

- odometry observed
- odometry corrected by localization.

• Divide by turn or travel magnitude.

• Compute standard deviations

- $k_1 = 1.0$
- $k_2 = 0.4$

The Action Model $P(x_t | u_{t-1}, x_{t-1})$

• Given a small motion $u_{t-1} = (\alpha, \Delta s, \Delta \varphi - \alpha)$

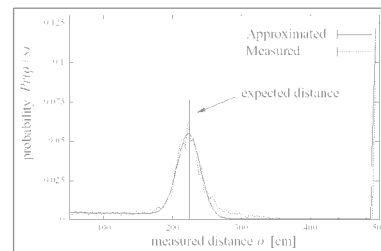
$$\begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} = \begin{pmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \end{pmatrix} + \begin{pmatrix} (\Delta s + \varepsilon_2) \cos(\theta_{t-1} + \alpha + \varepsilon_1) \\ (\Delta s + \varepsilon_2) \sin(\theta_{t-1} + \alpha + \varepsilon_1) \\ \Delta \varphi + \varepsilon_1 + \varepsilon_3 \end{pmatrix}$$

where

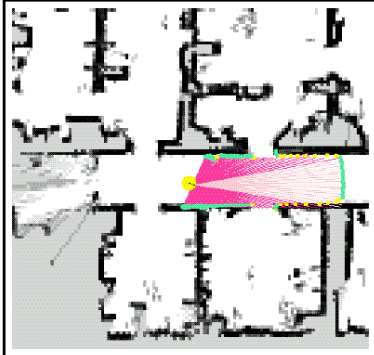
- $\varepsilon_1 \sim N(0, k_1 \alpha)$
- $\varepsilon_2 \sim N(0, k_2 \Delta s)$
- $\varepsilon_3 \sim N(0, k_3 (\Delta \varphi - \alpha))$

Sensor Model $P(z_t | x_t)$

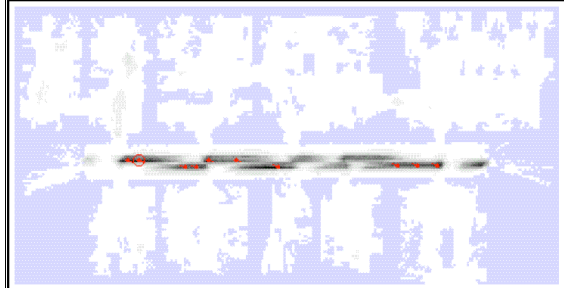
• For a given range measurement at a given location x_t in the map:



Laser Rangefinder Scan z_t



Density $P(z_t | x_t)$ by Location



Computing $P(z_t | x_t)$

- Image probabilities are too small to be meaningful:

$$P(z_t | x_t) = \prod_{i=1}^{180} P(\text{ray}(i) = d_i | x_t)$$

- But log probabilities can be accumulated

$$\log P(z_t | x_t) = \sum_{i=1}^{180} \log P(\text{ray}(i) = d_i | x_t)$$

- without numerical underflow.

Q: Didn't we use log odds before?

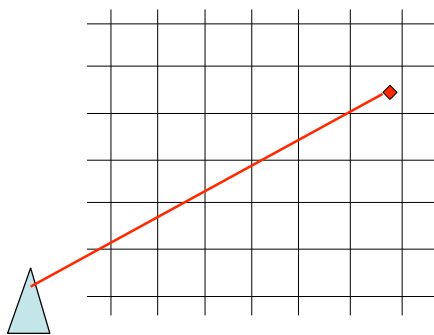
- Yes, but that was in the occupancy grid.
 - It is useful as a way to represent $P(\text{occ}(i,j))$
 - Certainty on either side of ignorance

- Here, we're doing Markov localization.

$$\text{Bel}(x_t) = \eta P(z_t | x_t) \int P(x_t | u_{t-1}, x_{t-1}) \text{Bel}(x_{t-1}) dx_{t-1}$$

- We need to compare values of $P(z_t | x_t)$.
 - Odds would just be a distraction
 - Log helps us avoid numerical underflow

$P(\text{ray}(i) = d_i | x_t)$



Computing $\log P(\text{ray}(i) = d_i | x_t)$

- If $\text{ray}(i)$ terminates at the first obstacle:
 - $\log P(\text{ray}(i)=d_i | x_t) = -4$
- If $\text{ray}(i)$ terminates before an obstacle:
 - $\log P(\text{ray}(i)=d_i | x_t) = -8$
- If $\text{ray}(i)$ terminates after an obstacle:
 - $\log P(\text{ray}(i)=d_i | x_t) = -12$
- Add them up for $i=1$ to 180.

$$\log P(z_t | x_t) = \sum_{i=1}^{180} \log P(\text{ray}(i) = d_i | x_t)$$

- $\log P(z_t | x_t)$ totals between -720 and -2160
- Exponentiating would give zero!

How Can We Normalize?

$$Bel(x_t) = \eta P(z_t | x_t) \int P(x_t | u_{t-1}, x_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

- To compute η , we must *add* un-normalized values, each with log much less than -720.
 - Exponentiating would give zero for every one!
- Solution:
 - Adjust by adding the largest log P to all values.
 - Exponentiate. The largest values remain non-zero.
 - Then normalize, which divides out the adjustment.
 - Underflow only eliminates negligible values.

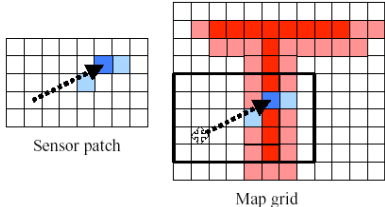
Avoiding Numerical Underflow

- We're summing many negative log values

$$\log P(z_t | x_t) = \sum_{i=1}^{180} \log P(\text{ray}(i) = d_i | x_t)$$
- Exponentiating will just give us zero!
- But we're just going to normalize, next.
 - So multiply by a constant, and normalize it out.
 - I.e., add a constant to each log $P(z_t | x_t)$
- Shift the largest log $P(z_t | x_t)$ value to zero.
 - I.e., add max log $P(z_t | x_t)$ to each value.
 - Then we have values of $P(z_t | x_t)$ we can normalize.

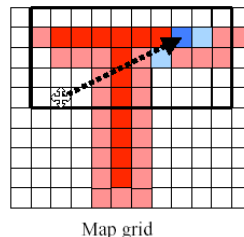
Estimate $P(z_t | x_t)$ with Correlation

- Given a pose hypothesis x_t , build a “sensor patch” (new map) from scan z_t .
- Compute correlation between sensor patch and map grid.



An Approximation to Scan Matching

- Ignores visibility problems.



Fast Approximate Correlation

- Let c_i be a cell in the sensor patch
 - If the endpoint of a laser ray hits, $c_i = 1$.
 - Otherwise, $c_i = 0$.
- The corresponding cell in the map grid is m_i and $p(m_i)$ is its occupancy probability.

$$\text{Corr}(z_t, x_t) = \sum_i c_i p(m_i)$$

$$Bel(x_t) = \eta \text{Corr}(z_t, x_t) Bel^-(x_t)$$

- Orders-of-magnitude speed improvement [Konolige & Chou, IJCAI-99].

Potential Problems with $P(z_t | x_t)$

- Grid cells are only partially occupied.
 - Should we model the *porosity* of each cell?

Future Attractions

- Particle filter algorithm.
- Landmark-based mapping.
- Topological mapping.