Feature-Based and Featureless Localization

Michael Dimitrov
Computer Science
CS 458 - Intro to Robotics
Purdue University - West Lafayette
dimitrm@purdue.edu

I. Introduction

This report implements and compares two localization pipelines: feature-based (matching detected landmarks to a known map) and featureless (scan/shape matching without explicit landmarks). Due to accumulation of small errors like slip, the motion model drifts over time from the ground truth robot position. Both methods perform pose correction: feature-based estimates a least-squares similarity transform from landmark pairs, while featureless uses the geometry boundaries.

II. Feature-Based Localization

In feature-based localization, matched landmark pairs provide a direct geometric constraint to correct the motion model via a least-squares similarity transform. After outlining the transform derivation, we show how the motion model is updated each step.

A. Derivation of Least-Squares Transform

Pairs are partitioned into two sets, the ground truth landmark positions r and the observed landmark positions l, where r_i is matched to l_i. We can describe a transformation from l_i to r_i as

$$\lambda R l_i + t = r_i$$

where $R = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}$.

We can solve for λ, $\cos\alpha$, $\sin\alpha$, t_x, t_y by minimizing:

$$\min_{\lambda,\alpha,t_x,t_y} \sum_i ||\lambda R l_i + t - r_i||^2$$

.

1) Compute the center of mass of all m points:

$$\bar{l} = \frac{1}{m}\sum_i^m l_i, \quad \bar{r} = \frac{1}{m}\sum_i^m r_i$$

2) Center all the points:

$$l'_i = l_i - \bar{l}, \quad r'_i = r_i - \bar{r}$$

3) Solve for t and λ:

$$\begin{aligned}
&\lambda R l_i + t - r_i \\
&= \lambda R(l'_i + \bar{l}) + t - r'_i + \bar{r} \\
&= \lambda R l'_i - r'_i + (\lambda - \bar{r} + t) \\
&= \lambda R l'_i - r'_i + t' \\
&\to \min \sum_i ||\lambda R l'_i - r'_i + t'||^2 \\
&= \min \sum_i ||\lambda R l'_i - r'_i||^2 + ||m t'||^2 \\
&\to t' = 0 \\
&\to \lambda R \bar{l} - \bar{r} + t = 0
\end{aligned}$$

$$\boxed{t = \bar{r} - \lambda R \bar{l}}$$

$$\begin{aligned}
&\to \sum_i ||\lambda R l'_i - r'_i||^2 = 0 \\
&\to \sum_i ||\sqrt{\lambda} R l'_i - \frac{r_i}{\sqrt{\lambda}}'||^2 = 0 \\
&= \lambda \sum_i ||R l'_i||^2 - 2\sum_i r'_i R l'_i + \frac{1}{\lambda}\sum_i ||r'_i||^2 \\
&\lambda a + b + \frac{1}{\lambda} c = 0 \quad \to \lambda^2 = \frac{c}{a}
\end{aligned}$$

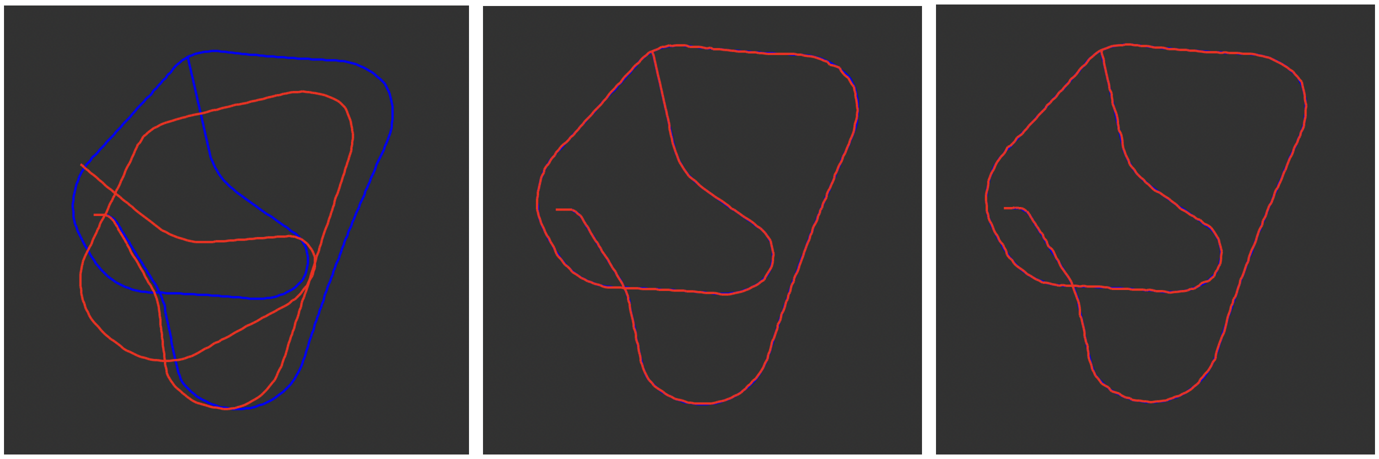
$$\boxed{\lambda = \sqrt{\frac{\sum_i ||r'_i||^2}{\sum_i ||R l'_i||^2}} = \sqrt{\frac{\sum_i ||r'_i||^2}{\sum_i ||l'_i||^2}}}$$

4) If we rotate l_i by an unknown amount α, then the dot product of r_i with the rotated l_i is greatest when they are parallel. Use this fact to find R:

$$\begin{aligned}
&\max_\alpha \Big(\sum_i \begin{bmatrix} r'_x & r'_y \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} l'_x \\ l'_y \end{bmatrix}\Big) \\
&= \max_\alpha \Big(\sum_i \begin{bmatrix} r'_x & r'_y \end{bmatrix} \begin{bmatrix} l'_x \cos\alpha & -l'_y \sin\alpha \\ l'_x \sin\alpha & l'_y \cos\alpha \end{bmatrix} \\
&= \max_\alpha \Big(\sum_i \begin{bmatrix} \cos\alpha & \sin\alpha \end{bmatrix} \begin{bmatrix} \sum_i r'_x l'_x + r'_y l'_y \\ \sum_i -r'_x l'_y + r'_y l'_x \end{bmatrix}
\end{aligned}$$

The max of the above dot product happens when the vectors are equal:

$$\boxed{\begin{bmatrix} \cos\alpha \\ \sin\alpha \end{bmatrix} = \frac{1}{\sqrt{C^2 + S^2}} \begin{bmatrix} C = \sum_i r'_x l'_x + r'_y l'_y \\ S = \sum_i -r'_x l'_y + r'_y l'_x \end{bmatrix}}$$



(a) No localization (b) Feature-Based Localization (c) Featureless Localization

Fig. 1: Robot trajectories (red) compared to ground truth (blue) under three settings.

B. Robot Pose Update

We next use the above similarity transform to update the robots position at each time step:

$$p_x' = \lambda \cos(\alpha) p_x - \lambda \sin(\alpha) p_y + t_x$$

$$p_y' = \lambda \sin(\alpha) p_x + \lambda \cos(\alpha) p_y + t_y$$

To update, we first predict the current pose using the robot's motion (v, v_{alpha}), then re-calculate and pair the landmarks, and use that to estimate the transform and update the position.

C. Results

Feature-based localization pulls the estimate onto the ground-truth line. Fitting a least-squares transform from matched landmarks counteracts drift, so the path in Fig. 1b hugs the line far better than Fig. 1a. As you can see in Fig. 1a with no localization the robot estimated position over steers, and overtime drifts significantly from the ground truth path. However, the red path in Fig. 1a is generally smoother than Fig. 1b since localization has to snap the position back at regular intervals using the similarity transform, which can leave the path very slightly rough.

III. Featureless Localization

Featureless localization uses the geometry of the room as landmarks for the similarity transform. At each time step it sub samples the lidar scans, and matches them to the nearest points on the boundary. It then uses these matchings to compute the transform using the IPC algorithm, which iteratively gets closer to a local minimum optimal transform. This transform is then used to correct the robot position.

Just like in Feature-based Localization, the corrected robot position from the previous timestep, plus the robot motion from the previous to the current time step, is used to predict the robot's position at the current time step, which then performs all the localization steps described above.

A. Results

Just like Feature-based Localization, Featureless localization vastly improves on the estimated robot position when compared to no localization. This can be seen by comparing Fig. 1a to Fig. 1c, which clearly shows that Featureless Localization is much tighter to the line.

The difference between Feature-based (Fig. 1b) and Featureless (Fig. 1c) localization is very minor - both methods very closely follow the ground truth path, but the Featureless path is very slightly more bumpy. This could be due to IPC using local minimum, subsampling at x0.1, and using a threshold of 0.1 to match lidar scans to their points on the wall. It could also be that when the robot is near an obstacle, many of its lidar scans are occluded and don't reach the boundaries, so we have fewer constraints to do IPC with.