Offline Simultaneous Localization and Mapping (SLAM) using Miniature Robots

• Objectives
• SLAM approaches
• SLAM for ALICE
  – EKF for Navigation
  – Mapping and Network Modeling
• Test results

Philipp Schaer and Adrian Waegli
June 29, 2007
ALICE - ROBOT

- Alice → As cheap and small as possible
  - 2 SWATCH motors with wheels and tires (max speed: 40mm/s)
  - 4 active IR proximity sensors (reflection measurement)
  - Microcontroller with 8Kwords Flash program memory
  - Radio communication module
Test Environment - Centimeter range labyrinth
General SLAM Approaches

Localization
- Kalman filter (EKF, UKF)
- Particle filter
- IA methods (Fuzzy logic, neural networks…)

Mapping
- Grid maps
- Feature recognition / scan matches
- Tree-based network optimizers
SLAM for ALICE

- EKF for navigation
- 3 approaches for mapping:
  - Grid mapping
  - Boundary detection based on range measurements
  - Network reconstruction
EKF for Navigation

**Prediction**

1) Predict state
\[
\hat{x}_k^- = f(\hat{x}_{k-1}^+, u_{k-1})
\]

2) Predict the error covariance
\[
P_k^- = \Phi_{k-1} P_{k-1}^+ \Phi_{k-1}^T + \Gamma_{k-1} Q_{k-1} \Gamma_{k-1}^T
\]

**Update**

1) Compute Kalman filter gain
\[
K_k = P_k^+ H_k^T [H_k P_k^+ H_k^T + R_k]^{-1}
\]

2) Update the state vector with measurement \( z_k \)
\[
\hat{x}_k^+ = \hat{x}_k^- + K_k [z_k - h(\hat{x}_k^-)]
\]

3) Update the error covariance
\[
P_k^+ = [I - K_k H_k] P_k^- [I - K_k H_k]^T + K_k R_k K_k^T
\]

**Initial estimations**

1) State: \( \hat{x}_0 \)
2) Error covariance: \( P_0 \)

**Linear approximation equations**

System process model: \( \Phi_{k-1} \approx \frac{\partial f(x, u)}{\partial x} \bigg|_{x=x_{k-1}, u=u_{k-1}} \)

Measurement model: \( H_k \approx \frac{\partial h(x)}{\partial x} \bigg|_{x=x_k} \)
System Process Model

\[ x_{k+1} = x_k + \frac{d_\ell + d_r}{2} \sin\theta_k \]
\[ y_{k+1} = y_k + \frac{d_\ell + d_r}{2} \cos\theta_k \]
\[ \theta_{k+1} = \theta_k + \frac{d_\ell - d_r}{D} \]

Functional model:
\[
F = \begin{bmatrix}
1 & 0 & \frac{d_\ell + d_r}{2} \cos\theta \\
0 & 1 & -\frac{d_\ell + d_r}{2} \sin\theta \\
0 & 0 & 1
\end{bmatrix}
\]

Stochastic model:
\[
P = \begin{bmatrix}
\sigma_x^2 & \sigma_y^2 \\
\sigma_y^2 & \sigma_\theta^2
\end{bmatrix}
\]
\[
Q_{ww} = \begin{bmatrix}
\sigma_{d_\ell}^2 \\
\sigma_{d_r}^2
\end{bmatrix}
\]
\[
\Gamma = \begin{bmatrix}
\frac{\sin\theta}{2} & \frac{\sin\theta}{2} \\
\frac{\cos\theta}{2} & \frac{\cos\theta}{2} \\
\frac{1}{D} & -\frac{1}{D}
\end{bmatrix}
\]
Measurement Model

\[ h(\hat{x}_k^-) = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \]_{SLAM} \]

\[
\begin{align*}
H_{pos} & = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \\
H_{dir} & = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}
\end{align*}
\]

\[
R_{pos} = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix} \\
R_{dir} = \begin{bmatrix} \sigma_\theta^2 \end{bmatrix}
\]

\[
\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \theta^\circ \end{bmatrix}
\]

\[
\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 32 \\ 0.5 \end{bmatrix} \]

\[
\theta_{SLAM} = 90^\circ \\
\theta_k^- = 110^\circ
\]
Grid Modeling

- Shape recognition (PCA)
Environment Modeling

- Robots programmed with “follow & avoidance” behavior
- Online calibration of the wheel data not feasible
Network Modeling

- Estimate network nodes and segments
- Appropriate for Labyrinth mapping
- Poor IR sensor performance (unusable for Navigation)
- Map matching for navigation
Test Setup

- Construction of “Labyrinth” environment
- 3 Runs of ca 2min with 3 different robots
- Robots programmed with “follow & avoidance” behavior
- Odometry and range data transmission every 15 sec by wireless communication
- Decoding of HEX-data
- SLAM implemented in MATLAB
Raw odometry Data / Calibration issues

- Odometry data:
  - Registered ticks on left and right wheel → conversion of ticks into distance
  - Calibration: tick conversion values are not exactly known
  - Transmission errors in data packages

![Graphs showing uncorrected and corrected data](image_url)
Raw sensor Data

- Raw IR sensor data:
  - Conversion of ADC value into range value
  - Computation of target coordinates (georeferencing)
  - Feature detection (edges, walls)

\[
\begin{align*}
\mathbf{x}_i & = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \\
\text{for sensor} & = \text{left, front, right, back}
\end{align*}
\]

Test Results

Estimated track and range measurements
Grid Modelling (Occupancy Grid Map)

- Map robot position and georeferenced target directly on grid
- Labyrinth reconstruction requires feature recognition
- Feature recognition difficult because of the quality of the
  - Odometer data
  - Range measurements
- Scan matching impossible
  - “Random” measurements in corners
  - Data gaps
Boundary Detection

- Edges are not directly observable
- Fit lines by least squares while robot in “follow wall” behavior
- Evaluate turn rate and identify type of turn:
  - Left turn (90°)
  - Right turn (90°)
  - U-turn (180°)
- Attitude update after each turn

→ Problem: position uncertainty still growing
Network Modelling Approach

- Localization: It is sufficient to know in which corridor robot is
- Mapping: Construct topological network of the environment
- Identify known network points → position update possible
- Attitude update after each turn

→ Position uncertainty reduced
Conclusion

• Grid estimation approach not applicable:
  – Feature recognition and scan matching almost impossible

• Boundary detection possible, but cumbersome:
  – Implicit boundary detection by “follow wall” behavior (robots runs on straight line)
  – Poor quality of georeferencing in turns
  – Difficult to implement position updates

• Network estimation seems to be the most appropriate method for labyrinth mapping for the given sensors:
  – More robust in turns
  – Easier topology recognition
  – Loop closure at every matched network point
  – Map matching along line segments
Outlook

- After several passes through the same network, fewer assumptions might be required:
  - Store and verify topological beliefs
  - Direction and position updates later, but no assumption with respect to the network angles necessary

\[ p(LR) = 45\% \quad \text{Trajectory} \quad p(LS) = 35\% \quad p(LL) = 15\% \quad p(LB) = 5\% \]