Marker-based FastSLAM on the AlphaBot2

Autonomous Systems Project 2023/2024

J. Pessoa
ist198915

J. Gonçalves
ist199995

M. Ribeiro
ist196446

T. Nogueira
ist1100029

Group 27

Instituto Superior Técnico

Presentation Overview

The FastSLAM Algorithm

The FastSLAM Algorithm

Core Idea — Rao-Blackwellization

$$p(x_{0:t}, m_{1:M} \,|\, z_{1:t}, u_{1:t}) = p(x_{0:t} \,|\, z_{1:t}, u_{1:t}) \prod_{j=1}^{M} p(m_j \,|\, x_{0:t}, z_{1:t})$$

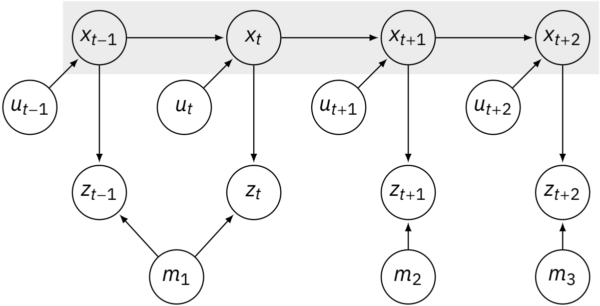

Figure: FastSLAM graph representation

1. **Do, for all particles** $(k = 1, \dots, N)$:
   - **Retrieval:** retrieve a pose $x_{t-1}^{[k]}$ from the particle set X_{t-1}.
   - **Prediction:** sample a new pose $x_t^{[k]} \sim p(x_t \mid x_{t-1}^{[k]}, u_t)$.
   - **Measurement Update:** for each observed feature identify the correspondence j for the measurement z_t^i and incorporate the measurement z_t^i into the corresponding EKF, by updating the mean $\mu_{j,t}^{[k]}$ and covariance $\Sigma_{j,t}^{[k]}$ of the j-th landmark.
   - **Importance Weight:** calculate the importance weight $w_t^{[k]}$ for the new particle.
2. **Resampling:** sample, with replacement, N particles. Each particle is sampled with a probability proportional to $w_t^{[k]}$.

Motion Model

Given the absence of wheel odometry, we make use of the **velocity motion model**:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} -\frac{\hat{v}}{\hat{\omega}}\sin\theta + \frac{\hat{v}}{\hat{\omega}}\sin(\theta + \hat{\omega}\Delta t) \\ \frac{\hat{v}}{\hat{\omega}}\cos\theta - \frac{\hat{v}}{\hat{\omega}}\cos(\theta + \hat{\omega}\Delta t) \\ \hat{\omega}\Delta t + \hat{\gamma}\Delta t \end{bmatrix}$$

Where $\hat{v}$ and $\hat{\omega}$ model **real motion!** (noisy values).

Exception: For **linear movement**, the relation becomes

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \hat{v}\cos(\theta)\Delta t \\ \hat{v}\sin(\theta)\Delta t \\ \hat{\gamma}\Delta t \end{bmatrix}$$

Here, $\hat{v}$ and $\hat{\omega}$ represent the velocities corrupted by noise:

$$\begin{cases} \hat{v} = v + \varepsilon_v \\ \hat{\omega} = \omega + \varepsilon_\omega \end{cases}$$

- v and ω: commanded velocities of the robot.
- ε_v and ε_ω: zero mean random variables modeling control noise with variance δ_v and δ_ω respectively.
- $\hat{\gamma}$: models degeneracy in the robot's rotation when halted.

Measurement Model

Measurement Model

Kinematics model

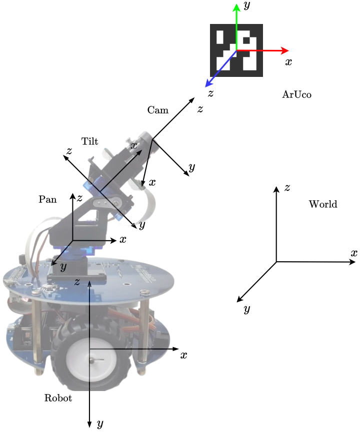

Figure: AlphaBot2’s kinematic model

- Cameras provide a rich source of environmental data.
- We've decided to represent landmarks with a 7D state:

$$m = \begin{bmatrix} \mathbf{p} \\ \mathbf{q} \end{bmatrix}$$

  with coordinate $\mathbf{p} = [x\ y\ z]^T$ and orientation given by a unit quaternion $\mathbf{q} = [q_w\ q_x\ q_y\ q_z]^T$.
- Although the extra information does not aid in solving the localization problem, it proves useful for addressing the data association problem.

The transformation from the camera's frame to the world frame involves the following chain operation

$$\begin{bmatrix} \mathbf{p}^w \\ \mathbf{q}^w \end{bmatrix} = {}^w\mathbf{H}_c \begin{bmatrix} \mathbf{p}^c \\ \mathbf{q}^c \end{bmatrix} + \begin{bmatrix} \mathbf{h}_c \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{h}_r \\ \mathbf{0} \end{bmatrix}$$

where we define

- ${}^w\mathbf{H}_c = {}^w\mathbf{T}_r\,{}^r\mathbf{T}_p\,{}^p\mathbf{T}_t\,{}^t\mathbf{T}_c$, final camera to world transformation;
- $\mathbf{h}_c = [0\;0\;0.12]^T$, camera's height offset (in meters);
- $\mathbf{h}_r$ the robot's coordinates in the world frame.

- Diagonal Block Matrices:

$$\mathbf{T} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_n \end{bmatrix}$$

We will denote this idea by the $\mathrm{blkdiag}(\mathbf{A}_1, \ldots, \mathbf{A}_\mathrm{n})$ operator.

We employ a conversion from quaternion to its corresponding matrix representation, given by:

$$\mathbf{Mq} = \begin{bmatrix} q_w & -q_x & -q_y & -q_z \\ q_x & q_w & -q_z & q_y \\ q_y & q_z & q_w & -q_x \\ q_z & -q_y & q_x & q_w \end{bmatrix},$$

where $\mathbf{q} = [q_w \; q_x \; q_y \; q_z]^T$ is the quaternion obtained from the rotation matrix. This matrix form enables the quaternion product to replicate matrix algebra.

Finally, the measurement model is obtained by reversing the order of operations, i.e.,

$$\begin{bmatrix} \mathbf{p}^c \\ \mathbf{q}^c \end{bmatrix} = {}^w\mathbf{H}_c{}^T \left(\begin{bmatrix} \mathbf{p}^w \\ \mathbf{q}^w \end{bmatrix} - \begin{bmatrix} \mathbf{h}_r \\ \mathbf{0} \end{bmatrix} - \begin{bmatrix} \mathbf{h}_c \\ \mathbf{0} \end{bmatrix} \right).$$

Rotation matrices and quaternion matrices embedded in the diagonal block matrix are orthogonal (${}^w\mathbf{H}_c{}^{-1} = {}^w\mathbf{H}_c{}^T$).

Resampling

- Particle filters are approximate and as such subject to approximation errors, namely:
  - Variance of the estimator.
  - Particle depletion.
- Some strategies to reduce the sampling error:
  - Reduce the frequency at which resampling takes place.
  - Low variance sampling.

To prevent sample **impoverishment and degeneration** we resample only when the particle variance is low.

$$\text{ESS} = \frac{1}{\sum_{k=1}^{N} w_k^2}$$

- Resampling is performed if the Effective Sample Size (ESS) is below the threshold "number of particles $\times$ 0.85".

Maximum Likelihood Data Association

Maximum Likelihood Data Association

Challenges with ArUco Markers

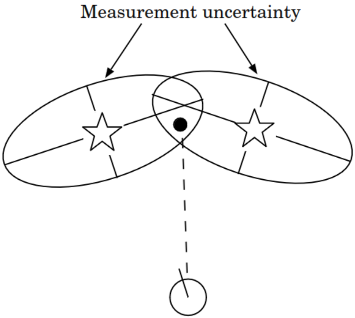

Figure: Measurement Ambiguity.

- **ArUco:** Known and predetermined.
- **Real-World:** Unknown number and identity of landmarks.

Objective: Match observed features with landmarks
Selection Criterion: Landmark with the smallest Mahalanobis distance is chosen, maximizing the likelihood of correct association

$$D^2 = (z_t^i - \hat{z}_j)^T Z_j^{-1} (z_t^i - \hat{z}_j)$$

- z_t^i: Observed feature
- $\hat{z}_j$: Predicted measurement of landmark j
- Z_j: Predicted covariance matrix

For each observation z_t^i, identify the landmark j maximizing $p(z_t^i \mid x_t, m_j)$

Implementation

Motion Model Noise Estimation

Control noise is **Gaussian** with zero mean and variance

$$\delta_u = \begin{bmatrix} \delta_v \\ \delta_\omega \end{bmatrix}$$

Heuristic Determination:

- Over 30 trials conducted to ensure application of the Central Limit Theorem
- **Evaluation:** Time to traverse 48 cm and Time to perform a full rotation (2π rad)

Mean Squared Error (MSE):

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2, \quad N = 30,$$

where $\overline{x}$ is the average controlled velocity.

Motion Model Noise Estimation
Modeling Control Noise

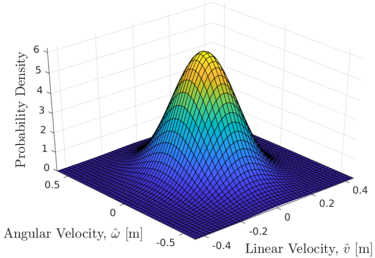

Figure: Bivariate density function.

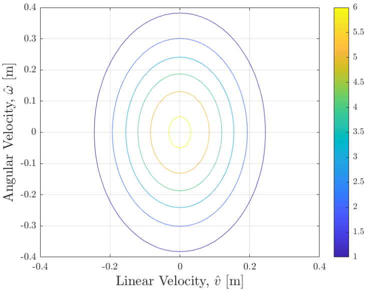

Figure: top view.

Measurement Model Covariance Matrix

- **Method:** Using an 8×6 checkboard with 0.035 m squares
- **Resolution:** 400×304 pixel^2
- **Frame Rate:** 30 FPS for real-time operation
- **Purpose:** Ensures accurate estimation of ArUco marker poses by `aruco_detect` and noise covariance estimation

Measurement Model Covariance Matrix

Calibration

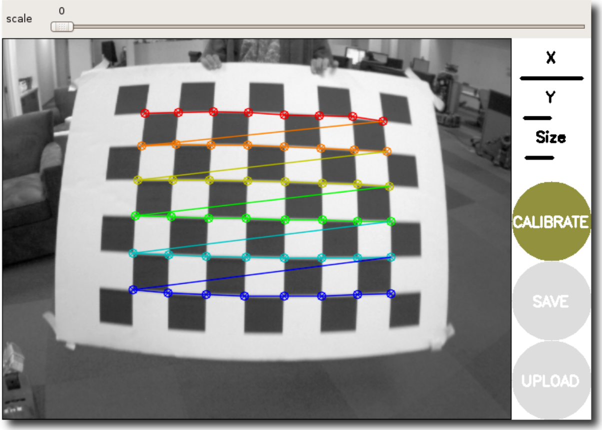

Figure: Camera calibration procedure (taken from the ROS wiki)

Steps:

- Samples taken at: 60 cm, 120 cm, 180 cm, 240 cm
- Samples taken at three angles: -45º, 0º, 45º
- Variables under study: $x, y, z, q_x, q_y, q_z, q_w$
- Variance calculated for each variable

Fit: Second-degree polynomial for each variable

$$f(r,\theta) = \alpha_0 + \alpha_1 \cdot \theta + \alpha_2 \cdot r + \alpha_3 \cdot \theta^2 + \alpha_4 \cdot r \cdot \theta + \alpha_5 \cdot r^2$$

And

$$R = \mathrm{Diag}\left(f(r,\theta)_x^2, f(r,\theta)_y^2, f(r,\theta)_z^2, f(r,\theta)_{q_w}^2, f(r,\theta)_{q_x}^2, f(r,\theta)_{q_y}^2, f(r,\theta)_{q_z}^2\right)$$

Measurement Model Covariance Matrix

Estimation — Example

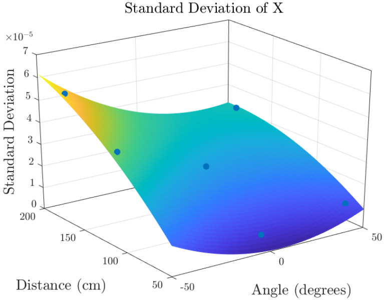

Figure: Fit for variable *x*.

```
Linear model Poly22:
fitResult(x,y) = p00 + p10*x + p01*y + ...
                 p20*x^2 + p11*x*y + p02*y^2
Coefficients (with 95% confidence bounds):
  p00 =  -2.199e-05  (-4.995e-05, 5.973e-06)
  p10 =   3.054e-08  (-2.316e-07, 2.927e-07)
  p01 =    5.02e-07  (-1.812e-08, 1.022e-06)
  p20 =   4.289e-09  (4.761e-10, 8.103e-09)
  p11 =  -1.658e-09  (-3.68e-09, 3.641e-10)
  p02 =  -1.064e-09  (-3.209e-09, 1.08e-09)

  R-squared: 0.9819
  Adjusted R-squared: 0.9518
  RMSE: 0.0000
```

Heuristic for Data Association
&
Outlier Rejection

Mahalanobis Distance Squared (D^2):

- Follows a chi-squared distribution [Cooper2005]
- Degrees of Freedom: 7 (equal to feature vector dimension)
- Confidence level: 95%

Observations exceeding threshold are **non-matches**:

- Sums individual compatibilities of each observation with all landmarks
  1. **Null Sum:** Marks the observation as a new landmark
  2. **Non Null Sum:** discards measurements

Reference: A. Cooper, "A Comparison of Data Association Techniques for Simultaneous Localization and Mapping," S.M. Thesis, MIT, 2005.

Measurement	Landmark 1	Landmark 2
1	a	b

Measurement 1 is compared to threshold $\chi^{-2}_{95\%}(7)$:

- **Case 1:** $a < b$ and $a <$ thresh. Measurement → Landmark 1.
- **Case 2:** $b < a$ and $b <$ thresh. Measurement → Landmark 2.
- **Case 3:** $(a \;\&\; b) >$ thresh. Measurement → new landmark.
- **Case 4:** $(a \;\&\; b) <$ thresh. Measurement → Discarded.

We can repurpose the algorithm to discard bad measurements in known data association:

Method:

- Checks Mahalanobis distance between observation and matched landmark
- If distance exceeds threshold, observation is discarded
- Ensures only reliable data is used for updating

Results

Results in the Micro-Simulator

Results in the Micro-Simulator
ESS

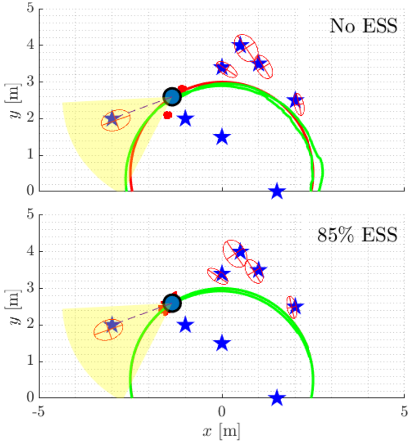

Figure: Known data association. Demonstration of resampler w/ ESS impact

Table: RMSE as a Function of Particle Number.

N	10	20	50	100	200	500
RMSE	0.350	0.280	0.200	0.150	0.149	0.140

Results in the Micro-Simulator
3D Simulation

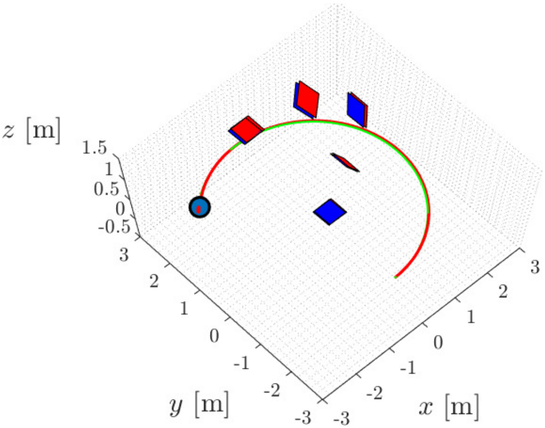

Figure: 3D results of the final FastSLAM algorithm.

Results on Real-World Data

Results on Real-World Data

Pan and Tilt

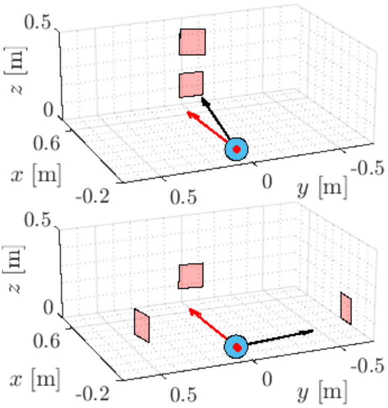

Figure: Use of pan and tilt during test procedure. The first plot showcases the use of tilt, the second one the use of pan, in real-world data.

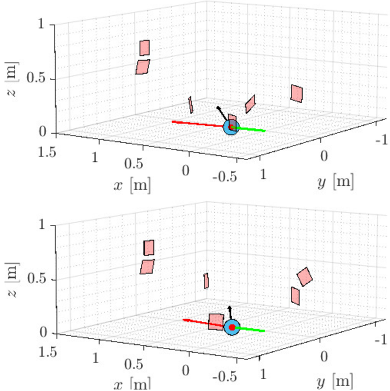

Figure: Effect of outlier rejection. The top map displays results without the implementation of outlier rejection, while the bottom map incorporates it, demonstrating improved accuracy.

Results on Real-World Data

Unknow Data Association

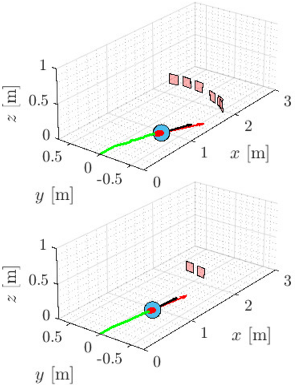

Figure: Landmark creation problem. The top image illustrates the scenario without any adjustments, while the bottom image shows the corrected setup. The real environment contains only two markers.

Results on Real-World Data

Adjustments to Help Reduce Wobbliness

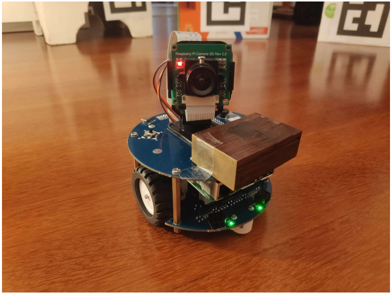

Figure: Added weight

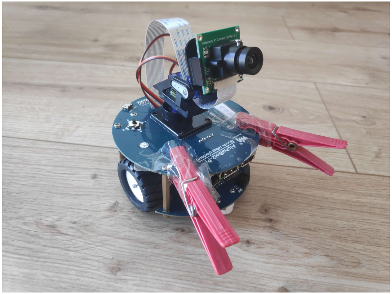

Figure: Added clothespins

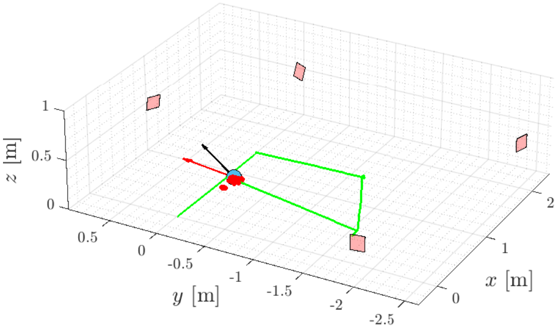

Figure: Correct data association with threshold of $\chi^{-2}_{95\%}(7)$ for a balanced robot. The map has four landmarks in four different corners.

Encountered Problems

We came across several difficult challenges:

- The robot was **unable to move in a straight line** when given only a linear velocity;
- The robot was **unable to spin on itself**, turn left and right;
- The robot's pan and tilt was **jittery and of difficult control**;
- The provided twist message contained velocities that did not correspond to reality due to **wrong processing in the driver's code**;

Solution WIP: https://youtu.be/F2Mxs-8-AKQ

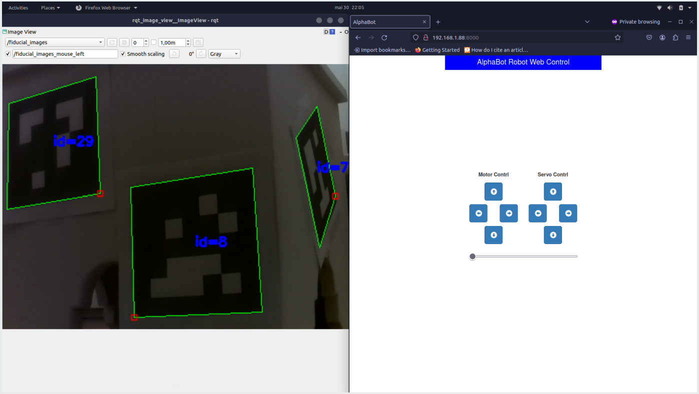

Figure: Developed motion driver’s control interface

Questions

Thank You!