Autonomous Robot Navigation

Localization and Mapping Techniques for Mobile Robots

Panos Trahanias
Professor of Computer Science, University of Crete
and
Head, Computational Vision and Robotics Lab
Foundation for Research and Technology – Hellas (FORTH)
trahania@ics.forth.gr

July 2006

Robots Intelligently Interacting With People
2006 ONASSIS LECTURE SERIES in COMPUTER SCIENCE
Autonomous Navigation
Problem Statement

The ability of robots to navigate safely and reliably within their environments

- Operation in industrial environments
- Tour-guiding visitors in museums/exhibition sites
- Helping in household tasks
- Exploring unfriendly environments (volcanoes, sewer systems, underwater)
- Space applications
- ... (the list goes on)
Historical Walkthrough

... in the beginning: Robotics in ancient times - Talos

... and then: a multitude of robotic systems; Industrial robots
More Recently

Trend towards intelligent systems

Basic keyword: autonomous systems
Existing, non-autonomous systems...
... and non-existent, fully autonomous robots
General research goal

Development of robotic systems able to exhibit autonomous behavior in complex and dynamic environments.
An Interesting Research Area

Theoretical interest...
- Mathematical and computational modeling of perception and action

... with important applications
- “Intelligent” robotic wheelchairs
- Robotic tour-guides in museums and exhibition sites
- Exploration of unknown and, possibly hostile, environments
- Routine tasks (surveillance, cleaning, etc)
The above are based to a great extent on the ability of autonomous navigation
Autonomous Navigation
Research Directions

- Given
- An environment representation - Map
- Knowledge of current position C
- Target position G

➢ A path has to be planned and tracked that will take the robot from C to G
Autonomous Navigation Research Directions

- During execution (runtime)
- Objects / Obstacles O may block the robot
- The planned path is no-longer valid
- The obstacle needs to be avoided and the path may need to be re-planned
Navigation Issues

Important questions (Levitt et al '91)

- Where am I
- Where are other places relative to me
- How do I get to other places from here

Important navigation issues

- Robot localization
- Map building
- Path/motion planning

Wednesday
Navigation Issues – Underlying HW

Sensors
Stereo vision

Communications

Sonars

Bump sensors
Infrared sensors
Laser scanner

Bump sensors
Sonars
Odometry

Interaction
Processing
Power

Motors

Laser Scanner
Range Sensor Model

- Laser Rangefinder
- Model range and angle errors.

\[
\begin{bmatrix} x, y \end{bmatrix}^T = \text{Exp}(R(r, \phi)) = [r \cos(\phi), r \sin(\phi)]^T
\]

\[
\Sigma_{\text{polar}} = \begin{bmatrix} k_\phi \phi & 0 \\ 0 & k_{\rho_0} + k_{\rho_1} r \end{bmatrix}
\]

\[
\Sigma_p = \nabla R \Sigma_{\text{polar}} \nabla R^T
\]
Need for Modeling

Robot + Environment

Extremely Complex Dynamical System

Need for Appropriate Modeling
Markov Assumption

- State depends only on previous state and observations
- Static world assumption
- Hidden Markov Model (HMM)

\[ P(x_t | y_1, y_2, \ldots, y_T) \xrightarrow{\text{Bayesian estimation}} P(x_1)P(y_1 | x_1) \prod_{t=2}^{T} P(x_t | x_{t-1})P(y_t | x_t) \]

Bayesian estimation: Attempt to construct the posterior distribution of the state given all measurements
A Dynamic System

- Most commonly - Available:
  - Initial State
  - Observations
  - System (motion) Model
  - Measurement (observation) Model

\[
\begin{align*}
&x_1 \leftrightarrow P(x_1) \\
&y_1 \cdots y_T
\end{align*}
\]

\[
\begin{align*}
x_k &= f_k(x_{k-1}) \quad \leftrightarrow \quad p(x_k \mid x_{k-1}) \\
y_k &= h_k(x_k) \quad \leftrightarrow \quad p(y_k \mid x_k)
\end{align*}
\]
Inference - Learning

- Localization (inference task)
  \( P(x_t = z | y_1, y_2, \ldots, y_t) \)

- Map building (learning task)
  \( m^* = \arg \max_m P(m | y_1, y_2, \ldots, y_T) \)
Belief State

\[ P(x_t \mid y_1, y_2, ..., y_t) = \frac{1}{c_t} P(y_t \mid x_t) \int P(x_t \mid x_{t-1} = z) P(x_{t-1} = z \mid y_1, y_2, ..., y_{t-1}) dz \]

How is the posterior distribution calculated? How is the prior distribution represented?

- **Discrete representation**
  - Grid (Dynamic)  (Dynamic) Markov localization (Burgard98)
  - Samples  Monte Carlo localization (Fox99)

- **Continuous representation**
  - Gaussian distributions  Kalman filters (Kalman60)
Example: State Representations for Robot Localization

Discrete Representations

Continuous Representations

Grid Based approaches (Markov localization)

Particle Filters (Monte Carlo localization)

Kalman Tracking
Kalman Filters - Equations

\[ P(x_t|x_{t-1}) \approx N(Ax_{t-1}, \Gamma) \]
\[ P(y_t|x_t) \approx N(Cx_t, \Sigma) \]

\[ x_t = A_t x_{t-1} + w_t \]
\[ y_t = C_t x_t + v_t \]
\[ w_t \approx N(0, \Gamma) \]
\[ v_t \approx N(0, \Sigma) \]

Where: \[ N(x; m, V) = \frac{1}{|2\pi V|^{1/2}} \exp\left(-\frac{1}{2} (x - m)^T V^{-1} (x - m)\right) \]
Kalman Filters - Update

\[ x_t = A_t x_{t-1} + w_t \]
\[ y_t = C_t x_t + v_t \]
\[ w_t \approx N(0, \Gamma) \]
\[ v_t \approx N(0, \Sigma) \]  

Predict
\[ \hat{x}_t^- = A \hat{x}_{t-1} \]
\[ P_t^- = A P_{t-1} A^T + \Gamma \]

Compute Gain
\[ K_t = P_t^- C^T (C P_t^- C^T + \Sigma)^{-1} \]

Compute Innovation
\[ J_t = \hat{y}_t - C \hat{x}_t^- \]

Update
\[ \hat{x}_t = \hat{x}_t^- - K_t J_t \]
\[ P_t = (I - K_t C) P_t^- \]
Kalman Filter - Example

\[ x_t = A_t x_{t-1} + B + w_t \quad A_t = [1] \]
\[ y_t = C_t x_t + D + v_t \quad B_t = [u_t] \]
\[ w_t \approx N(0, \Gamma) \quad C_t = [-1] \]
\[ v_t \approx N(0, \Sigma) \quad D_t = [1] \]

\[ x_t = x_{t-1} + u_t + w_t \]
\[ y_t = d - x_t + v_t \]
\[ w_t \approx N(0, \Gamma) \]
\[ v_t \approx N(0, \Sigma) \]
Kalman Filter - Example

\[ \hat{x}_i^- = A\hat{x}_{i-1} + B \]
\[ P_i^- = AP_{i-1}A^T + \Gamma \]
Kalman Filter - Example

\[ \hat{x}_i^- = A\hat{x}_{i-1} + B \]
\[ P_i^- = A P_{i-1} A^T + \Gamma \]
Kalman Filter - Example

Predict
\[ \hat{x}_t^- = A\hat{x}_{t-1} + B \]
\[ P_t^- = A P_{t-1} A^T + \Gamma \]

Compute Innovation
\[ J_t = \hat{y}_t - C \hat{x}_t^- \]

Compute Gain
\[ K_t = P_t^- C^T (CP_t^- C^T + \Sigma)^{-1} \]
Kalman Filter – Example

**Predict**

\[ \hat{x}_t = A\hat{x}_{t-1} + B \]
\[ P_t^- = AP_{t-1}A^T + \Gamma \]

**Compute Innovation**

\[ J_t = \hat{y}_t - C\hat{x}_t^- \]

**Compute Gain**

\[ K_t = P_t^-C^T(CP_t^-C^T + \Sigma)^{-1} \]

**Update**

\[ \hat{x}_t = \hat{x}_t^- - K_t J_t \]
\[ P_t = (I - K_tC)P_t^- \]
Kalman Filter – Example

\[ \hat{x}_i^- = A\hat{x}_{i-1} + B \]
\[ P_i^- = AP_{i-1}A^T + \Gamma \]
Non-Linear Case

- Kalman Filter assumes that system and measurement processes are linear
- Extended Kalman Filter -> linearized Case

\[
\begin{align*}
    x_t &= A_t x_{t-1} + w_t \\
    y_t &= C_t x_t + v_t \\
    w_t &\approx N(0, \Gamma) \\
    v_t &\approx N(0, \Sigma)
\end{align*}
\]

\[
\begin{align*}
    x_t &= f(x_{t-1}) + w_t \\
    y_t &= g(x_t) + v_t \\
    w_t &\approx N(0, \Gamma) \\
    v_t &\approx N(0, \Sigma)
\end{align*}
\]
Example: Localization – EKF

- Initialize State
  - Gaussian distribution centered according to prior knowledge
    - large variance

- At each time step:
  - Use previous state and motion model to predict new state
    (mean of Gaussian changes - variance grows)
  - Compare observations with what you expected to see from
    the predicted state – Compute Kalman Innovation/Gain
  - Use Kalman Gain to update prediction
Extended Kalman Filter

Project State estimates forward (prediction step)

\[ \mu_{x_{t+1}} = \text{Exp}(F(\mu_{x_t}, \alpha_t)) \]
\[ \Sigma_{x_{t+1}} = \nabla F_{x} \Sigma_{x_t} \nabla F_{x}^T + \nabla F_{\alpha} \Sigma_{\alpha_t} \nabla F_{\alpha}^T \]

Predict measurements

\[ l_{t+1}^- = H(\mu_{x_{t+1}}) \]
\[ r_{t+1} = l_{t+1} - l_{t+1}^- \]
\[ \Sigma_{r_{t+1}} = \nabla F_{x_{t+1}} \Sigma_{x_{t+1}} \nabla F_{x_{t+1}}^T + \Sigma_{l_{t+1}} \]

Compute Kalman Innovation

\[ K_{t+1} = \Sigma_{x_{t+1}} \nabla F_{x_{t+1}} \Sigma_{r_{t+1}}^{-1} \]
\[ \mu_{x_{t+1}} = \mu_{x_{t+1}}^- + K_{t+1} r_{t+1} \]
\[ \Sigma_{x_{t+1}} = \Sigma_{x_{t+1}}^- - K_{t+1} \Sigma_{r_{t+1}} K_{t+1}^T \]

Compute Kalman Gain

Update Initial Prediction
EKF – Example
motion model for mobile robot

- Synchro-drive robot
- Model range, drift and turn errors

\[
\Sigma_{a_t} = \begin{bmatrix} k_r d_t & 0 \\ 0 & k_t f_t + k_d d_t \end{bmatrix}
\]

\[
\mu_{x_{t+1}} = \text{Exp}(F(\mu_{x_t}, a_t)) = \begin{bmatrix} x_t - d_t \sin(\theta_t) \\ y_t - d_t \cos(\theta_t) \\ \theta_t + d_t \end{bmatrix}
\]

\[
\Sigma_{x_{t+1}} = \nabla F_x \Sigma_{x_t} \nabla F_x^T + \nabla F_a \Sigma_{a_t} \nabla F_a^T
\]
EKF – Example
simulated run
Bayesian Methods
Discrete Representation

Probabilistic localization – the case of global localization
Bayesian Methods
Discrete Approaches

Grid-based representation of the state-space

\[ P((x,y,\theta)(k) \mid u(0 : k-1), y(1:k)) \]
Bayesian Methods
Discrete Approaches

Posterior

\[ P(x(k) \mid u(0 : k-1), y(1 : k)) = n(k) \]

\[ \sum_{x(k-1) \in X} [P(x(k) \mid u(k-1), x(k-1))] \]

\[ P(x(k-1) \mid u(0 : k-2), y(1 : k-1)) \]

Sensor model

Motion model

Past history
Example: Localization – Grid Based

- Initialize Grid
  (Uniformly or according to prior knowledge)

- At each time step:
  - For each grid cell
    - Use observation model to compute \( P(y(k) \mid x(k)) \)
    - Use motion model and probabilities to compute
      \[
      \sum_{x(k-1) \in X} \left[ P(x(k) \mid u(k-1), x(k-1)) \cdot P(x(k-1) \mid u(0:k-2), y(1:k-1)) \right]
      \]
  - Normalize
Bayesian Methods
Discrete Approaches

Density plots of the robot state

... beginning  small No of cycles  sufficient No of cycles
Often models are non-linear and noise in non gaussian.

Use particles to represent the distribution

“Survival of the fittest”

\[
P(x_t | y_{1:t}) = \frac{1}{c_t} P(y_t | x_t) \int P(x_t | x_{t-1} = z) P(x_{t-1} = z | y_{1:t-1}) dz
\]

Motion model
Observation model
Proposal distribution
(=weight)
Particle Filters
SIS-R algorithm

- Initialize particles randomly
  (Uniformly or according to prior knowledge)
- At each time step:
  - For each particle:
    - Use motion model to predict new pose (sample from transition priors)
    - Use observation model to assign a weight to each particle (posterior/proposal)
  - Create a new set of equally weighted particles by sampling the distribution of the weighted particles produced in the previous step.
Particle Filters – Example 1
Particle Filters – Example 1

Use motion model to predict new pose
(move each particle by sampling from the transition prior)
Particle Filters – Example 1

Use measurement model to compute weights
(weight:observation probability)
Particle Filters – Example 1

Resample
Particle Filters – Example 2

Initialize particles uniformly
Particle Filters – Example 2
Particle Filters – Example 2
Particle Filters – Example 2
Particle Filters – Example 2
Particle Filters – Example 2
Discrete State Approaches

- Ability (to some degree) to localize the robot even when its initial pose is unknown.
- Ability to deal with noisy measurements, such as from ultrasonic sensors.
- Ability to represent ambiguities.
- Computational time scales heavily with the number of possible states (dimensionality of the grid, size of the cells, size of the map).
- Localization accuracy is limited by the size of the grid cells.
Continuous State Approaches

- Perform very accurately if the inputs are precise (performance is optimal in the linear case).
- Computational efficiency.
- Requirement that the initial state of the robot is known.
- Inability to recover from catastrophic failures caused by erroneous matches or incorrect error models.
- Inability to track Multiple Hypotheses about the location of the robot.
Hybrid Approaches

- Combination of characteristics from both methods
- Hybrid methods very popular in many scientific areas
  - Control theory
  - Economics
Proposed Model

switching state-space model (SSSM)

The Switching State-Space model
- M continuous State Vectors
- One discrete “switch” variable

Example Belief State

Switching state-space Model

Combines both models

**Continuous Model**
- Accurate performance
- Computational efficiency
- Initial state must be known
- Inability to recover from catastrophic failures
- Inability to track Multiple Hypotheses

**Discrete Model**
- Perform even when initial pose is unknown
- Deal with noisy measurements
- Represent ambiguities
- Computational time scales heavily
- Localization accuracy limited

**Inherits strengths**
**Eliminates weaknesses**
Localization

Belief state is intractable
• Mixture of $M^T$ Gaussians
• Grows exponentially with time

Solution
• Selection (eg. Cox94, Jensfelt99, Roumeliotis00, Duckett01)
  *Only keep the most probable paths in model histories (Multiple Hypothesis Tracking)*
• Collapsing (eg. Murphy98)
  *Approximate the mixture of $M^T$ Gaussians with a mixture of $M^r$ Gaussians ($r$: small number, eg. 1,2,3)*
Localization – Discrete Model

Corner Point Visibility

- Dominant corner point
- Invisible corner point
- Visible but not dominant corner point
Localization – Discrete Model
Corner Point Visibility

\[ \theta = 180^\circ \]

\[ \alpha = -90^\circ \]
Localization - Discrete Model (Observation – Transition)

\[ P(s^k_t | y_1, y_2, ..., y_t) = \frac{1}{c_t} P(y_t | s^k_t) \prod_i P(s^k_t | s^i_{t-1}) P(s^i_{t-1} | y_1, y_2, ..., y_{t-1}) \]
Localization – Continuous Model (EKF)

Project State estimates forward (prediction step)

\[ \mu_{x_{t+1}} = \text{Exp}(F(\mu_x, \alpha_t)) \]
\[ \Sigma_{x_{t+1}} = \nabla F_x \Sigma_x \nabla F_x^T + \nabla F \alpha \Sigma \alpha \nabla F_{\alpha}^T \]

Predict already mapped features to the predicted state

\[ l_{t+1}^- = H(\mu_{x_{t+1}}) \]

Compute Kalman Innovation

\[ r_{t+1} = l_{t+1} - l_{t+1}^- \]
\[ \Sigma_{r_{t+1}} = \nabla F_{x_{t+1}}^T \Sigma_{x_{t+1}}^{-1} \nabla F_{x_{t+1}} + \Sigma_{l_{t+1}} \]

Compute Kalman Gain

\[ K_{t+1} = \Sigma_{x_{t+1}}^{-1} \nabla F_{x_{t+1}} \Sigma_{r_{t+1}}^{-1} \]
\[ \mu_{x_{t+1}} = \mu_{x_{t+1}} - K_{t+1} r_{t+1} \]
\[ \Sigma_{x_{t+1}} = \Sigma_{x_{t+1}} - K_{t+1} \Sigma_{r_{t+1}} K_{t+1}^T \]

Update Initial Prediction
Localization - Results (Simulated)
Localization - Results
(Real world – FORTH 1st floor)
Localization - Results
(Real world – Outside our lab)
Mapping
Problem Statement

Environment

Navigation

Robot localization

Environment Map

SLAM
Simultaneous Localization And Mapping
Simultaneously estimate the robot position as well as the positions of landmarks (stochastic mapping)

- Augment state vector to also include landmark positions

\[
x = \begin{bmatrix} x_r & y_r & x_{l1} & y_{l1} & x_{l2} & y_{l2} & \cdots & x_{ln} & y_{ln} \end{bmatrix}^T
\]
Mapping – Kalman Tracker

\[
x(k + 1) = x(k) + \begin{pmatrix}
1 & 0 \\
0 & 1 \\
0 & 0 \\
\vdots & \vdots \\
0 & 0 \\
\end{pmatrix}
\begin{pmatrix}
u_x(k) \\
u_y(k) \\
\end{pmatrix} + \begin{pmatrix}
v_{rx}(k) \\
v_{ry}(k) \\
0 \\
0 \\
\vdots \\
0 \\
\end{pmatrix}
\]

\[
x(k + 1) = Fx(k) + Gu(k) + v(k)
\]
Mapping – Discrete Bayesian Approach

- Recursive Bayesian filtering for estimating the robot positions along with a map of the environment

\[
P(x(1:k), m | u(0:k-1), y(1:k)) = aP(y(k) | x(k), m) \int A \cdot B \cdot dx(1:k-1)
\]

\[
A = (P(x(k) | u(k-1), x(k-1))
\]

\[
B = P(x(1:k-1), m | u(0:k-2), y(1:k-1)))
\]
Mapping – Discrete Bayesian Approach

- Estimating the full posterior is not tracktable
- Incremental scan matching

Let at time k-1 the localization and map estimates:

\[
\hat{x}(k - 1), \hat{m}(\hat{x}(1 : k - 1), y(1 : k - 1))
\]

At time k – after moving and getting a new measurement y(k)

\[
\hat{x}(k) = \arg \max_{x(k)} \left\{ P(y(k) \mid x(k), \hat{m}(\hat{x}(1 : k - 1), y(1 : k - 1)))P(x(k) \mid u(k - 1), \hat{x}(k - 1)) \right\}
\]
Mapping – Discrete Bayesian Approach

- Estimating the full posterior is not tractable
- FastSLAM

\[
P(x(1:k), m \mid u(0:k-1), y(1:k)) = P(m \mid u(0:k-1), y(1:k)) \cdot P(x(1:k) \mid u(0:k-1), y(1:k))
\]

- Usually implemented via particle filters
Mapping Challenge: Loops in the robot’s path

- As the robot moves and maps features, errors in both the state and the mapped features tend to increase with time.
- When already mapped areas are visited (a loop is detected), the mapping algorithm should be able to correct its state and eliminate the accumulated errors.
- Complicated robot paths, nested loops or loops that close simultaneously are difficult cases.
Our approach

Off-line Feature-mapping algorithm:

Loop detection is accomplished via a hybrid localizer with global localization capabilities (SSSM) that creates hypotheses whenever known areas (corner points) are visited

- All hypotheses created by the localizer, whenever loops are detected, are tracked individually within their own copy of the map.
- The best path through hypotheses histories is selected, a Kalman smoother redistributes errors and an iterative procedure corrects the map.

H. Baltzakis and P. Trahanias, ICRA 2006
Features

- **Line Segments**
  (used to localize the robot)
  \[
  l \approx N\left(l_f, l_d, \Sigma_f\right)
  \]

- **Corner Points**
  (used to create hypotheses)
  \[
  c \approx N\left(c_x, c_y, \Sigma_c\right)
  \]
Algorithm overview 1

- Mapping starts with one hypothesis (dominant)
- Existing line segments used for localization while the map is created.
- Non existing segments are inserted in the map
Algorithm overview 2

- Detected corner points result in creation of new hypotheses.
- Hypotheses eventually vanish if observation sequences do not confirm their validity.
Hypotheses maintain their own copy of the map (as many maps as hypotheses)
Algorithm overview 4

- Upon entering previously mapped areas (corners detected), new hypotheses are created at the correct robot poses.
- Correct hypotheses will eventually become more probable since observations confirm their validity.
Algorithm overview 5

- All hypotheses are tracked within their own copy of the map.

Multi-hypothesis Mapping

Haris Baltzakis & Panos Trahanias

Example - Artificial

PHASE A
Map Rectification - Iterative Algorithm

- **E-STEP**: Localize the robot using all available measurements. (obtain max a-posteriori estimates of robot states)
- **M-STEP**: Recalculate map features

Treat map features as parameters of the dynamical system according to which the robot’s state evolves.
EM Algorithm - Example

Operation of EM Algorithm (simulation)
Results (simulated running example)

Initial map

Multi-hypothesis Mapping
Haris Baltzakis & Panos Trahanias

Example - Artificial
PHASE B
Results (simulated running example)

Initial Map

Rectified Map

Ground Truth
Results (Di Castello Belgioioso)

Phase A

Multi-hypothesis Mapping
Haris Baltzakis & Panos Trahanias

Example - Castello di Belgioioso
PHASE A

Belgioioso dataset available from university of Freiburg
Results (Di Castello Belgioioso)

Phase B

Multi-hypothesis Mapping
Haris Baltzakis & Panos Trahanias

Example - Castello di Belgioioso
PHASE B

Belgioioso dataset available from university of Freiburg

July 2006 Panos Trahanias - Onassis Lecture Series 83/93
Results (Radish - cmu_nsh_level_a)

Multi-hypothesis Mapping
Haris Baltzakis & Panos Trahanias
Example - Radish - cmu_nsh_level_a
PHASE A

Multi-hypothesis Mapping
Haris Baltzakis & Panos Trahanias
Example - Radish - cmu_nsh_level_a
PHASE B

Radish cmu_nsh_level_a data set submitted by Nick Roy
Mapping - Results
(Real world – FORTH 1st floor)
Mapping - Results
(Real world – FORTH 1st FLOOR)

Initial Map

Rectified Map

Ground Truth
Visual Information Processing

- Laser range finders provide fast and accurate depth information for 2D slices of the environment.
- Various objects are invisible to the laser range finder.
- Vision can provide extra information for crucial tasks such as obstacle avoidance.
Step 1: From 2D laser measurements, form 3D model hypothesis

Step 2: From 1\textsuperscript{st} camera visual data and 3D model hypothesis predict visual data at the 2\textsuperscript{nd} camera. Then, compare actual and predicted visual data to evaluate the 3D model.

Step 3: Wherever 3D model hypothesis is invalid, extract metric structure information.
Visual Information Processing
(simulated example)
Depth computations take place only where inconsistencies are detected.
For collision avoidance depth computations can be further eliminated.

- Criterion 1. Visual range defers significantly to laser range data
- Criterion 2. Visual range is shorter than corresponding laser suggests
- Criterion 3. Visual range is neither too far nor too close to the robot.
Visual Information Processing
(Real world example – outside our lab)

H. Baltzakis, A. Argyros and P. Trahanias, MVA 2003
Real Application
(Robotic Tour-guide in exhibition site)

The TOURBOT & WebFAIR Projects:

- Autonomous mobile robots in populated environments (serving real-visitors)
- Also operating over the web (serving web-visitor)

P. Trahanias et al, IEEE RAM 2005
Thanks for your attention! Time for a break...