Autonomous navigation in dynamic uncertain environment using probabilistic models of perception and collision risk prediction

Chiara Fulgenzi

INRIA Rhône-Alpes, Grenoble France

PhD thesis presentation
June 8, 2009

Thesis Advisor
Christian Laugier

Co-advisor
Anne Spalanzani
Problem Definition

Autonomous navigation in unknown dynamic environment

Move autonomously in an unknown environment among moving vehicles or people
Autonomous navigation in unknown dynamic environment

Move autonomously in an unknown environment among moving vehicles or people
Problem Definition

Autonomous navigation in unknown dynamic environment

Static environment is explored:

- finite range
- sensor errors and accuracy
- hidden zones

Moving obstacles are detected and tracked:

- model uncertainty and errors
- model validity in time
- new obstacles entering the scene
Autonomous navigation in unknown dynamic environment

Static environment is explored:
- finite range
- sensor errors and accuracy
- hidden zones

Moving obstacles are detected and tracked:
- model uncertainty and errors
- model validity in time
- new obstacles entering the scene
Problem Definition

**Autonomous navigation in unknown dynamic environment**

Information about the environment is incomplete and uncertain in both time and space:

- Configuration Sensing
- Configuration Prediction
- Environment Sensing
- Environment Prediction
Problem Definition

**Autonomous navigation in unknown dynamic environment**

Information about the environment is incomplete and uncertain in both time and space:

- Configuration Sensing
- Configuration Prediction
- Environment Sensing
- Environment Prediction
Constraints

The navigation algorithm must take into account that:

- The environment changes dynamically
  - limited time to take decisions
  - models and decisions must be continuously updated

- The current state of the environment is uncertain
  - represent the limits of information
  - represent the quality of information

- The future state is of the environment is uncertain
  - predict
  - represent the quality of information
Constraints

The navigation algorithm must take into account that:

- **The environment changes dynamically**
  - limited time to take decisions
  - models and decisions must be continuously updated
- **The current state of the environment is uncertain**
  - represent the limits of information
  - represent the quality of information
- **The future state of the environment is uncertain**
  - predict
  - represent the quality of information
Navigation algorithms based on the risk of collision

- **Part I: Reactive method**
  - dynamic occupancy grid (BOF, [Coué, 03])
  - probabilistic velocity obstacles (VO, [Shiller, 98])

- **Part II: Motion Planning based on target-tracking**
  - mapping and target tracking ([Vu, 06])
  - RRTs ([LaValle, 99])
  - Partial Motion Planning ([Fraichard, 05])

- **Part III: Motion Planning based on typical patterns**
  - Hidden Markov Models ([Vasquez, 07])
  - Gaussian Processes ([Tay, 07])
Reactive Navigation

Reactive Navigation Methods: only the next control is computed at each step.

Potential Fields [Khatib, 85]
Vector Field Histogram [Borenstein, 91]
Curvature Velocity [Simmons, 96]
Lane Curvature Velocity [Simmons, 98]
Dynamic Window [Fox, 97]
Nearness Diagram [Montano, 00]
Obstacle Restriction [Minguez, 05]
Inevitable Collision States [Fraichard, 03]
Velocity Obstacles [Shiller, 98]
Dynamic Object Velocity [Montano, 05]

DYNAMIC ENVIRONMENT
PERCEPTION UNCERTAINTY
### Reactive Navigation

Reactive Navigation Methods: only the next control is computed at each step.

<table>
<thead>
<tr>
<th>Method</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Potential Fields</td>
<td>Khatib, 85</td>
</tr>
<tr>
<td>Vector Field Histogram</td>
<td>Borenstein, 91</td>
</tr>
<tr>
<td>Curvature Velocity</td>
<td>Simmons, 96</td>
</tr>
<tr>
<td>Lane Curvature Velocity</td>
<td>Simmons, 98</td>
</tr>
<tr>
<td>Dynamic Window</td>
<td>Fox, 97</td>
</tr>
<tr>
<td>Nearness Diagram</td>
<td>Montano, 00</td>
</tr>
<tr>
<td>Obstacle Restriction</td>
<td>Minguez, 05</td>
</tr>
<tr>
<td>Inevitable Collision States</td>
<td>Fraichard, 03</td>
</tr>
<tr>
<td>Velocity Obstacles</td>
<td>Shiller, 98</td>
</tr>
<tr>
<td>Dynamic Object Velocity</td>
<td>Montano, 05</td>
</tr>
</tbody>
</table>

**DYNAMIC ENVIRONMENT**

**PERCEPTION UNCERTAINTY**

Chiara Fulgenzi
Autonomous navigation in dynamic uncertain environment
### Reactive Navigation

Reactive Navigation Methods: only the next control is computed at each step.

<table>
<thead>
<tr>
<th>Method</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Potential Fields</td>
<td>Khatib, 85</td>
</tr>
<tr>
<td>Vector Field Histogram</td>
<td>Borenstein, 91</td>
</tr>
<tr>
<td>Curvature Velocity</td>
<td>Simmons, 96</td>
</tr>
<tr>
<td>Lane Curvature Velocity</td>
<td>Simmons, 98</td>
</tr>
<tr>
<td>Dynamic Window</td>
<td>Fox, 97</td>
</tr>
<tr>
<td>Nearness Diagram</td>
<td>Montano, 00</td>
</tr>
<tr>
<td>Obstacle Restriction</td>
<td>Minguez, 05</td>
</tr>
<tr>
<td>Inevitable Collision States</td>
<td>Fraichard, 03</td>
</tr>
<tr>
<td>Velocity Obstacles</td>
<td>Shiller, 98</td>
</tr>
<tr>
<td>Dynamic Object Velocity</td>
<td>Montano, 05</td>
</tr>
</tbody>
</table>

**DYNAMIC ENVIRONMENT**

**PERCEPTION UNCERTAINTY**
The Bayesian Occupancy Filter [Coué, 03]

- The probability of occupancy in the space
- A distribution function over a discrete set of velocities

Velocity Obstacles [Shiller, 98]

- Geometric method
- Tells if a linear velocity of the robot is in collision with moving obstacles
Related Work

The Bayesian Occupancy Filter [Coué, 03]
- The probability of occupancy in the space
- A distribution function over a discrete set of velocities

Velocity Obstacles [Shiller, 98]
- Geometric method
- Tells if a linear velocity of the robot is in collision with moving obstacles
Probabilistic Velocity Obstacles

Compute the risk of collision for linear velocities of the robot

- with a cell-to-cell approach
- using a clustered grid

\[
P(t_{coll} \in (t_0, t]|v_r, v_n) = \max_{o \in \text{SO}_t} P_o(\text{Occ}) \cdot P_o(v_n)
\]

\[
P(t_{coll} \in (t_0, t]|v_r) = 1 - \prod_{n=1}^{N}(1 - P_{\text{coll}}(v_r, v_n))
\]

Navigation function dependent on the risk of collision

\[
T_{\text{safe}}(v) = \epsilon + T_{brake}(v)
\]

\[
K^*(v) = K(v, \text{goal}) \cdot \left((1.0 - P(t_{coll}(v) \in (t_0, t_0 + k\tau])) \cdot \frac{k\tau}{\max_v(T_{\text{safe}}(v))}\right)
\]
Results: simulation setup

- Holonome Robot
- Distance sensor with limited range
Results: cell-to-cell VS clustering

BOF

\[ P(t_{coll} \in (0, 3]) \]

\[ P(t_{coll} \in (0, t]) \]
Results

scenario

perfect

vel uncertainty

+ limited range
Conclusions

Complexity
- cell-to-cell: depends on the size of the grid, not on the number of obstacles
- computation is parallelizable for each \((v_r, v_n)\)

Contributions
- computation of the risk of collision for linear velocities from a dynamic occupancy grid
- uncertainty rising from occlusion, limited range, velocity estimation uncertainty directly influences the choice of the next control
Conclusions

Limitations

One-step ahead reasoning

- efficiency issues
- safety issues
Outline

1 Introduction
   - Problem Definition
   - Contribution

2 Part I: Reactive Navigation
   - State of The Art
   - Contribution
   - Results

3 Part II: Motion Planning based on target-tracking
   - State of the Art
   - Contribution
   - Results

4 Part III: MP with Typical Patterns
   - Gaussian Processes representation
   - Hidden Markov Models representation
   - Results

5 Conclusions
Motion planning in unknown dynamic environment

Path planning approaches: compute a complete path

<table>
<thead>
<tr>
<th>Static or Dynamic, but deterministic environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combinatorial</td>
</tr>
<tr>
<td>RoadMaps [Canny,88]</td>
</tr>
<tr>
<td>Cell decomposition [Schwartz,83]</td>
</tr>
<tr>
<td>Potential Fields [Khatib,80]</td>
</tr>
<tr>
<td>Sampling Based</td>
</tr>
<tr>
<td>Probabilistic RM [Kavraki,95]</td>
</tr>
<tr>
<td>Discretized $C_{free}$ [Brooks,85]</td>
</tr>
<tr>
<td>Randomized PF [Barraquand,90]</td>
</tr>
<tr>
<td>Ariadne’s clew [Bessière,93]</td>
</tr>
<tr>
<td>RRTs [LaValle,99]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Unknown but Static environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDP</td>
</tr>
<tr>
<td>POMDP [Foka, 07]</td>
</tr>
<tr>
<td>Anytime-RRTs [Ferguson,06]</td>
</tr>
<tr>
<td>Particle-RRTs [Melchior,07]</td>
</tr>
</tbody>
</table>
State of the Art

Motion planning in unknown dynamic environment

Path planning approaches: compute a complete path

<table>
<thead>
<tr>
<th>Static or Dynamic, but deterministic environment</th>
<th>Sampling Based</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combinatorial</td>
<td></td>
</tr>
<tr>
<td>RoadMaps [Canny, 88]</td>
<td>Probabilistic RM [Kavraki, 95]</td>
</tr>
<tr>
<td>Cell decomposition [Schwartz, 83]</td>
<td>Discretized $C_{free}$ [Brooks, 85]</td>
</tr>
<tr>
<td>Potential Fields [Khatib, 80]</td>
<td>Randomized PF [Barraquand, 90]</td>
</tr>
<tr>
<td></td>
<td>Ariadne’s clew [Bessière, 93]</td>
</tr>
<tr>
<td></td>
<td>RRTs [LaValle, 99]</td>
</tr>
</tbody>
</table>

Unknown but Static environment

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MDP</td>
<td>Anytime-RRTs [Ferguson, 06]</td>
</tr>
<tr>
<td>POMDP [Foka, 07]</td>
<td>Particle-RRTs [Melchior, 07]</td>
</tr>
</tbody>
</table>
State of the Art

Motion planning in unknown dynamic environment

**D* [Stentz, 95]**
- computes the best path with the current knowledge
- execution time depends on the dimensions of the space

**Hybrid methods: plan-react-replan**
- static environment is known
- time for planning and replanning is not limited

**Partial Motion Planning [Fraichard, 03]**
- gives a partial safe path at anytime
- satisfies real-time constraints
- deterministic environment
## State of the Art

### Comparison

<table>
<thead>
<tr>
<th></th>
<th>Environment</th>
<th></th>
<th>Configuration</th>
<th></th>
<th>Time Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sensing</td>
<td>Prediction</td>
<td>Sensing</td>
<td>Prediction</td>
<td></td>
</tr>
<tr>
<td>Complete</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D*</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MDP</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>POMDP</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Sampling Based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Anytime-RRTs</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Particle-RRTs</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>PMP</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
</tbody>
</table>
## Motivation

### State of the Art

<table>
<thead>
<tr>
<th></th>
<th>Environment Sensing</th>
<th>Environment Prediction</th>
<th>Configuration Sensing</th>
<th>Configuration Prediction</th>
<th>Time Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complete</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D*</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MDP</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>POMDP</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Sampling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Anytime-RRTs</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Particle-RRTs</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>PMP</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Needed Properties</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
</tbody>
</table>

PMP → real-time constraints

Complete methods are not suitable in dynamic environment → Sampling Based Method

RRT → incremental

→ non-holonomic constraints

Integrate and update uncertain information in the decision process
1. Each configuration $q$ is deterministically known: $q \in C_{\text{free}}$ or $q \in C_{\text{obs}}$

2. A random point $p \in C_{\text{free}}$ is chosen

3. The nearest node of the current tree is expanded toward $p$

4. The search ends when the goal configuration is in the tree or it continues till some other condition is satisfied

5. The path is retrieved from the goal to the root
**Rapidly-exploring Random Trees**

1. Each configuration $q$ is deterministically known: $q \in C_{\text{free}}$ or $q \in C_{\text{obs}}$
2. A *random* point $p \in C_{\text{free}}$ is chosen
3. The nearest node of the current tree is expanded toward $p$
4. The search ends when the goal configuration is in the tree or it continues till some other condition is satisfied
5. The path is retrieved from the goal to the root
Rapidly-exploring Random Trees

1. Each configuration $q$ is deterministically known: $q \in C_{\text{free}}$ or $q \in C_{\text{obs}}$

2. A random point $p \in C_{\text{free}}$ is chosen

3. The nearest node of the current tree is expanded toward $p$

4. The search ends when the goal configuration is in the tree or it continues till some other condition is satisfied

5. The path is retrieved from the goal to the root
### Rapidly-exploring Random Trees

1. Each configuration \( q \) is deterministically known: \( q \in C_{\text{free}} \) or \( q \in C_{\text{obs}} \)
2. A *random* point \( p \in C_{\text{free}} \) is chosen
3. The nearest node of the current tree is expanded toward \( p \)
4. The search ends when the goal configuration is in the tree or it continues till some other condition is satisfied
5. The path is retrieved from the goal to the root
Rapidly-exploring Random Trees

1. Each configuration \( q \) is deterministically known: \( q \in C_{\text{free}} \) or \( q \in C_{\text{obs}} \)
2. A *random* point \( p \in C_{\text{free}} \) is chosen
3. The nearest node of the current tree is expanded toward \( p \)
4. The search ends when the goal configuration is in the tree or it continues till some other condition is satisfied
5. The path is retrieved from the goal to the root
Probabilistic RRTs

1. Each configuration $q$ has a probability of collision $P_{coll}(q)$
2. A random point $p$ is chosen
3. The node with the *most likely* path is expanded toward $p$
4. The search ends when the available time is out
5. The path is retrieved from the *most likely* node to the root
Probabilistic RRTs

1. Each configuration $q$ has a probability of collision $P_{\text{coll}}(q)$
2. A *random* point $p$ is chosen
3. The node with the *most likely* path is expanded toward $p$
4. The search ends when the available time is out
5. The path is retrieved from the *most likely* node to the root
Probabilistic RRTs

1. Each configuration \( q \) has a probability of collision \( P_{\text{coll}}(q) \)
2. A \textit{random} point \( p \) is chosen
3. The node with the \textit{most likely} path is expanded toward \( p \)

4. The search ends when the available time is out
5. The path is retrieved from the \textit{most likely} node to the root
**Probability of collision**

\[ P_{coll}(q) \]: risk of collision of \( q = (s, t) \), state \( s \) at time \( t \)

\[
P_{coll}(q) = P_{cs}(s) + (1 - P_{cs}(s)) \cdot P_{cd}(s, t) = P_{occ}(s) + (1 - P_{cs}(s)) \cdot \left( 1 - \prod_{o=1}^{0} (1 - P_{cd}(s, t, o)) \right)
\]

\[ P_{coll}(\pi) \]: risk of collision of path \( \pi \) from root \( q_0 \) to node \( q_N \)

\[
P_{coll}(\pi) = 1 - \prod_{i=0}^{N} (1 - P_{coll}(q_i))
\]

\[ L_{\pi}(q_N) \]: probability of success of path \( \pi \)

\[
L_{\pi}(q_N) = 1 - P_{coll}(\pi)
\]
**Probability of collision**

\[ P_{\text{coll}}(q) : \text{risk of collision of } q = (s, t), \text{ state } s \text{ at time } t \]

\[
P_{\text{coll}}(q) = P_{cs}(s) + (1 - P_{cs}(s)) \cdot P_{cd}(s, t) = \\
P_{\text{occ}}(s) + (1 - P_{cs}(s)) \cdot \left( 1 - \prod_{o=1}^{0} (1 - P_{cd}(s, t, o)) \right)
\]

\[ P_{\text{coll}}(\pi) : \text{risk of collision of path } \pi \text{ from root } q_0 \text{ to node } q_N \]

\[
P_{\text{coll}}(\pi) = 1 - \prod_{i=0}^{N} (1 - P_{\text{coll}}(q_i))
\]

\[ L_{\pi}(q_N) : \text{probability of success of path } \pi \]

\[
L_{\pi}(q_N) = 1 - P_{\text{coll}}(\pi)
\]
Probability of collision

\[ P_{\text{coll}}(q): \text{risk of collision of } q = (s, t), \text{ state } s \text{ at time } t \]

\[ P_{\text{coll}}(q) = P_{\text{cs}}(s) + (1 - P_{\text{cs}}(s)) \cdot P_{\text{cd}}(s, t) = \]

\[ P_{\text{occ}}(s) + (1 - P_{\text{cs}}(s)) \cdot \left(1 - \prod_{o=1}^{0} (1 - P_{\text{cd}}(s, t, o))\right) \]

\[ P_{\text{coll}}(\pi): \text{risk of collision of path } \pi \text{ from root } q_0 \text{ to node } q_N \]

\[ P_{\text{coll}}(\pi) = 1 - \prod_{i=0}^{N} (1 - P_{\text{coll}}(q_i)) \]

\[ L_{\pi}(q_N): \text{probability of success of path } \pi \]

\[ L_{\pi}(q_N) = 1 - P_{\text{coll}}(\pi) \]
A weight is computed for each partial path:

1. The likelihood is normalized
2. The estimated length of the path to Goal is considered
3. The path with the highest weight is chosen

\[ L_\pi(q_N) = 1 - P_{coll}(\pi) \]
\[ \tilde{L}_\pi(q_N) = \frac{N}{\sqrt{L_\pi(q_N)}} \]
\[ w_{q_N} = \frac{\tilde{L}_\pi(q_N)}{\text{dist}(q_0, q_N, \text{Goal})} \]
Most likely node, most likely path

\[ L_\pi(q_N) = 1 - P_{\text{coll}}(\pi) \]

\[ \tilde{L}_\pi(q_N) = \frac{N}{\sqrt{L_\pi(q_N)}} \]

\[ w_{q_N} = \frac{\tilde{L}_\pi(q_N)}{\text{dist}(q_0, q_N, \text{Goal})} \]

A weight is computed for each partial path:

1. The likelihood is \textit{normalized}.
2. The estimated length of the path to \textit{Goal} is considered.
3. The path with the highest weight is chosen.
Updating the tree

- Environment is explored
- Environment is dynamic

→ Update the tree and the path with new information

Partial Motion Planning [Fraichard, 05]

- Real-time constraints
- Safety issues (no ICS)

→ Deterministic prediction up to infinite time
Given the path chosen at the previous step, $\pi(q_N) = \{q_0...q_N\}$:

1. The robot moves to $q_1$
2. The search tree is pruned: $q_1$ is the new root
3. The tree is updated according to the new information
4. The tree is grown in the remaining time
5. A new path is chosen
Given the path chosen at the previous step, $\pi(q_N) = \{q_0...q_N\}$:

1. The robot moves to $q_1$
2. The search tree is pruned: $q_1$ is the new root
3. The tree is updated according to the new information
4. The tree is grown in the remaining time
5. A new path is chosen
Updating the tree

Given the path chosen at the previous step, $\pi(q_N) = \{q_0...q_N\}$:

1. The robot moves to $q_1$
2. The search tree is pruned: $q_1$ is the new root
3. The tree is updated according to the new information
4. The tree is grown in the remaining time
5. A new path is chosen
Updating the tree

Given the path chosen at the previous step, $\pi(q_N) = \{q_0 \ldots q_N\}$:

1. The robot moves to $q_1$
2. The search tree is pruned: $q_1$ is the new root
3. The tree is updated according to the new information
4. The tree is grown in the remaining time
5. A new path is chosen
Probabilistic RRTs: example
PRRTs with short term prediction

\[ \text{[Vu,07]} \]

Chiara Fulgenzi  Autonomous navigation in dynamic uncertain environment  28/48
Results

PP-RRTs with target tracking

[click me]
Probabilistic RRTs with target tracking

Introduction

Reactive Navigation

Motion Planning based on target-tracking

MP with Typical Patterns

Conclusions

Results

Chiara Fulgenzi

Autonomous navigation in dynamic uncertain environment
Probabilistic RRTs with target tracking: results

Target Tracking: $\sim 10Hz$  
PRRT: $2Hz$
**Probabilistic RRTs with target tracking**
Probabilistic RRTs with target tracking
Conclusions

Contributions

- Probabilistic RRTs:
  - Static environment uncertainty → occupancy grid
  - Velocity estimation uncertainty → target-tracking based

- On-line information and decision updating

Target-tracking based prediction

<table>
<thead>
<tr>
<th>PROs</th>
<th>CONs</th>
</tr>
</thead>
<tbody>
<tr>
<td>low a priori information</td>
<td>short-term</td>
</tr>
<tr>
<td>reactive to behavior changes</td>
<td>linear</td>
</tr>
</tbody>
</table>
Conclusions

Contributions

- Probabilistic RRTs:
  - Static environment uncertainty $\rightarrow$ occupancy grid
  - Velocity estimation uncertainty $\rightarrow$ target-tracking based
- On-line information and decision updating

Target-tracking based prediction

<table>
<thead>
<tr>
<th>PROs</th>
<th>CONs</th>
</tr>
</thead>
<tbody>
<tr>
<td>low \textit{a priori} information</td>
<td>short-term</td>
</tr>
<tr>
<td>reactive to behavior changes</td>
<td>linear</td>
</tr>
</tbody>
</table>
Conclusions
Outline

1. Introduction
   - Problem Definition
   - Contribution

2. Part I: Reactive Navigation
   - State of The Art
   - Contribution
   - Results

3. Part II: Motion Planning based on target-tracking
   - State of the Art
   - Contribution
   - Results

4. Part III: MP with Typical Patterns
   - Gaussian Processes representation
   - Hidden Markov Models representation
   - Results

5. Conclusions
Typical Patterns

Given an observed environment
- Moving objects follow typical patterns
- Patterns are learned off-line and modeled with HMMs or GPs
- Based on observations, probabilistic prediction is performed

Prediction
- Takes into account the structure of the environment
- Uncertainty is limited around typical patterns
### Gaussian Processes for Pattern Modelling [Tay, 2007]

A Typical Path $\rightarrow$ D dimensional Gaussian

$$D = \# \text{ of points observed along a path}$$

$p(O_m) = \sum_{k=1}^{K} I_{k,O_m} P(O_m|k)$

- Dataset
- Learned GP means
- Gaussian mixture

---

Chiara Fulgenzi

Autonomous navigation in dynamic uncertain environment
Gaussian Processes representation

**PPRRT with Gaussian Processes**

\[
P_{cd}(q, O_m, k) \quad \text{collision with obstacle } O_m \text{ in pattern } k
\]

\[
P_{cd}(q, O_m) = \sum_{k=1}^{K} l_{k, O_m} \cdot P_\pi(q, O_m, k)
\]

**Tree update**

The weight of each Gaussian component \(l_{k, O_m}\) is updated.
Hidden Markov Models for Pattern Modelling [Vasquez, 07]

HMM: Bayesian filter for discrete state-space approach

A Typical Path $\rightarrow$ a directed graph

nodes $\rightarrow$ discrete states  edges $\rightarrow$ transition probabilities

A HMM-graph represents all typical motions toward 1 goal

- States are given by position, velocity and intended goal
- Discretization is uniform or learned on the dataset
- Transition probabilities are learned
Hidden Markov Models based prediction

Discrete prediction obtained letting the graph evolve under the transition probabilities

\[
P(S_{t+k} | O_t) = \sum_{S_{t+k-1}} P(S_{t+k} | S_{t+k-1})P(S_{t+k-1} | O_t)
\]
Results for 100 goals reached; average obtained on 10 iterations.

<table>
<thead>
<tr>
<th># obsts</th>
<th># colls</th>
<th>with $v_r \neq 0$</th>
<th>% time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1.5</td>
<td>0</td>
<td>1.14</td>
</tr>
<tr>
<td>6</td>
<td>2.3</td>
<td>0</td>
<td>1.43</td>
</tr>
<tr>
<td>8</td>
<td>3.8</td>
<td>0</td>
<td>1.65</td>
</tr>
<tr>
<td>10</td>
<td>6.3</td>
<td>0</td>
<td>1.79</td>
</tr>
<tr>
<td>12</td>
<td>7.4</td>
<td>0</td>
<td>1.87</td>
</tr>
</tbody>
</table>
Simulation Results based on GPs

Results for 100 goals reached; average obtained on 10 iterations.

<table>
<thead>
<tr>
<th># obs</th>
<th># colls</th>
<th>with νr ≠ 0</th>
<th>% time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1.5</td>
<td>0</td>
<td>1.14</td>
</tr>
<tr>
<td>6</td>
<td>2.3</td>
<td>0</td>
<td>1.43</td>
</tr>
<tr>
<td>8</td>
<td>3.8</td>
<td>0</td>
<td>1.65</td>
</tr>
<tr>
<td>10</td>
<td>6.3</td>
<td>0</td>
<td>1.79</td>
</tr>
<tr>
<td>12</td>
<td>7.4</td>
<td>0</td>
<td>1.87</td>
</tr>
</tbody>
</table>
Simulation results based on HMMs

Results for 100 goals reached; average obtained on 10 iterations.

<table>
<thead>
<tr>
<th># obst</th>
<th># colls</th>
<th>with $v_r \neq 0$</th>
<th>% time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0.2</td>
<td>0</td>
<td>1.33</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>0</td>
<td>1.37</td>
</tr>
<tr>
<td>8</td>
<td>3.2</td>
<td>0</td>
<td>1.60</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
<td>0</td>
<td>1.75</td>
</tr>
<tr>
<td>12</td>
<td>7</td>
<td>0</td>
<td>1.84</td>
</tr>
</tbody>
</table>
Comparison

<table>
<thead>
<tr>
<th></th>
<th>HMM</th>
<th>GP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Velocity representation</td>
<td>discrete</td>
<td>continuous</td>
</tr>
<tr>
<td>Prediction</td>
<td>discrete</td>
<td>continuous</td>
</tr>
<tr>
<td>Prediction Update</td>
<td>complex</td>
<td>simple</td>
</tr>
</tbody>
</table>
Conclusions

Contributions:
- Reactive Method
- Partial Probabilistic RRTs:
  - Target tracking based algorithm
  - Typical patterns based prediction

Properties:
- Probabilistic uncertainty of environment perception and prediction is meaningfully integrated into the navigation strategy
- Risk of collision is updated on-line with incoming estimation
- Known typical patterns
  - allow more reliable and non-linear predictions
  - more complex robot behaviors
Perspectives

**Future work:**

- From simulator to tests on the real robot
- Short and medium-term prediction used together in one framework (off-board platform)
Perspectives

Future work:

- From simulator to tests on the real robot
- Short and medium-term prediction used together in one framework (off-board platform)
Perspectives

Future work:

- From simulator to tests on the real robot
- Short and medium-term prediction used together in one framework (off-board platform)
Perspectives

Future work:

- From simulator to tests on the real robot
- Short and medium-term prediction used together in one framework (off-board platform)
Perspectives:

- Use the probabilistic framework to perform reflexive prediction
- Multiple robot coordination
Publications

- Fulgenzi, C., Spalanzani, A., Laugier, C. "Dynamic Obstacle Avoidance in uncertain environment combining PVOs and Occupancy Grid.", IEEE ICRA 2007

