Robot Motion Planning

K. Grover, F. Barbosa, J. Tůmová, J. Křetínský

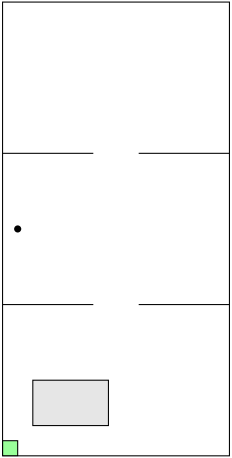
Technical University of Munich, KTH Royal Institute of Technology

June 3, 2021

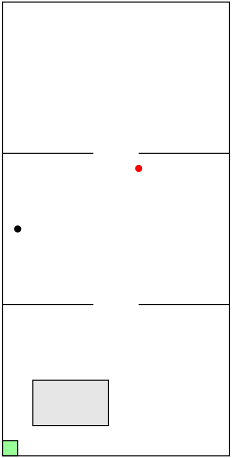
Outline

Motion Planning Problem

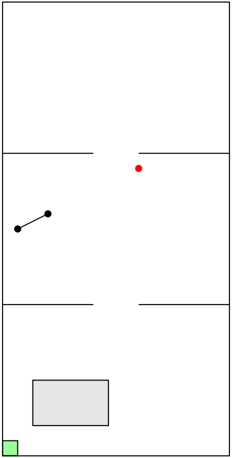
Motion Planning Problem



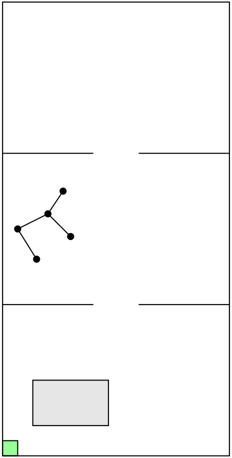
RRT/RRG



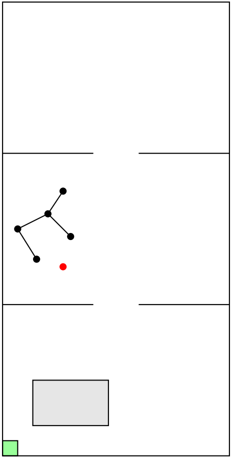
RRT/RRG



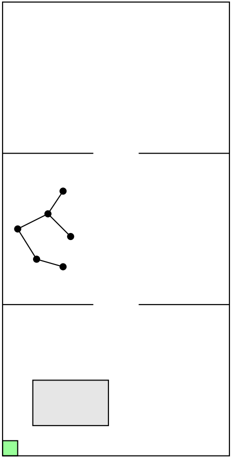
RRT/RRG



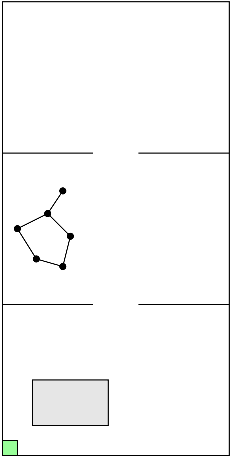
RRT/RRG



RRT



RRG



What about LTL?

- Algorithms already exists for LTL specifications.

What about LTL?

- Algorithms already exists for LTL specifications.
- But they only work if the robot knows the whole environment.

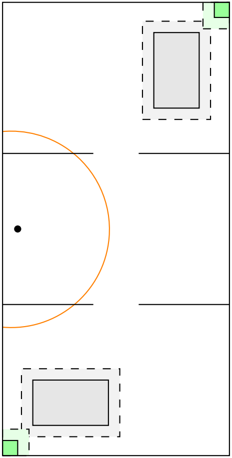
What about LTL?

- Algorithms already exists for LTL specifications.
- But they only work if the robot knows the whole environment.
- In reality, the robot know only about things inside a sensing radius and it cannot see beyond walls or obstacles.

What about LTL?

- Algorithms already exists for LTL specifications.
- But they only work if the robot knows the whole environment.
- In reality, the robot know only about things inside a sensing radius and it cannot see beyond walls or obstacles.
- To know the whole environment, the robot has to explore before start planning. This is usually done by a “frontier-based" exploration.

LTL Motion Planning



Specification: $F(r_1 \wedge b) \wedge F(r_2 \wedge b)$

Naive Algorithm

Naive Solution

- Explore the whole environment initially using frontier based approaches.

Naive Solution

- Explore the whole environment initially using frontier based approaches.
- Start building the RRG graph and construct an abstraction of the system on-the-fly.

Naive Solution

- Explore the whole environment initially using frontier based approaches.
- Start building the RRG graph and construct an abstraction of the system on-the-fly.
- Construct the product automaton with the property automaton.

Naive Solution

- Explore the whole environment initially using frontier based approaches.
- Start building the RRG graph and construct an abstraction of the system on-the-fly.
- Construct the product automaton with the property automaton.
- Find an accepting path in the product.

Naive Solution

- Explore the whole environment initially using frontier based approaches.
- Start building the RRG graph and construct an abstraction of the system on-the-fly.
- Construct the product automaton with the property automaton.
- Find an accepting path in the product.
- Lift this path to the original environment.

Can we do better?

- Too much effort is wasted in exploration.

Can we do better?

- Too much effort is wasted in exploration.
- Can we do exploration while trying to satisfy the path simultaneously?

Can we do better?

- Too much effort is wasted in exploration.
- Can we do exploration while trying to satisfy the path simultaneously?
- Identify places to go to, so that there is some progress in satisfying the property.

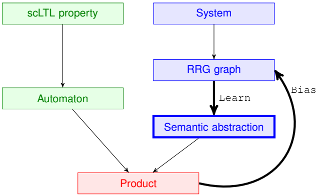
Better Solution

Our solution

- Sample a batch in the known area (initially, sensing radius) using biasing (initially, no biasing).

Our solution

- Sample a batch in the known area (initially, sensing radius) using biasing (initially, no biasing).
- Learn from the newly added edges and improve the bias.



Our solution

- Sample a batch in the known area (initially, sensing radius) using biasing (initially, no biasing).
- Learn from the newly added edges and improve the bias.
- Find the best move and go there.

Our solution

- Sample a batch in the known area (initially, sensing radius) using biasing (initially, no biasing).
- Learn from the newly added edges and improve the bias.
- Find the best move and go there.

Learn and Bias

- Learn from past experiences i.e. look for transitions that are similar to transitions that we have seen till now.

Learn and Bias

- Learn from past experiences i.e. look for transitions that are similar to transitions that we have seen till now.
- For each transition sampled in the current batch, add similar ‘maybe’ transitions in the abstraction as well.

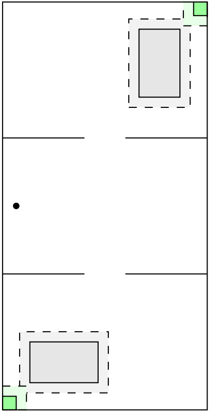
Learn and Bias

- Learn from past experiences i.e. look for transitions that are similar to transitions that we have seen till now.
- For each transition sampled in the current batch, add similar ‘maybe’ transitions in the abstraction as well.
- Use these ‘maybe’ transitions to bias the search.

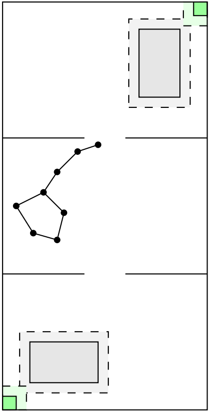
Learn and Bias

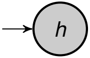
- What does similar mean here?
  $(r_1, t) \rightarrow (r_1, b) \implies (r_2, t) \rightarrow (r_2, b)$.
  - Compute domain of changes: Set of APs changing in a transition.
    $DOC = \{t, b\}$, $(s_1 \oplus s_2)$
  - Add transitions $s_1' \rightarrow s_2'$ where s_1' and s_2' are states which agree with s_1 and s_2 on DOC respectively.

Example

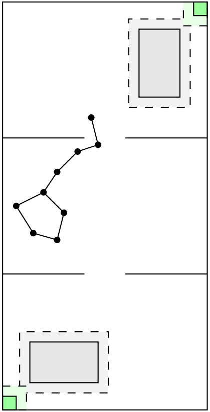


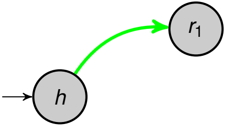
Example



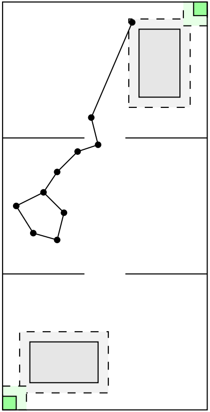


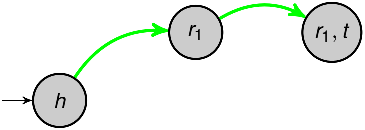
Example



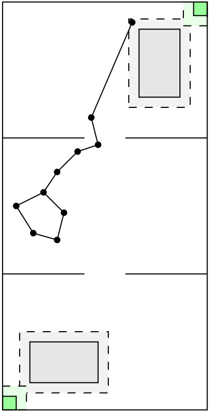


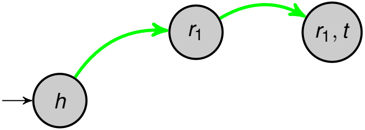
Example





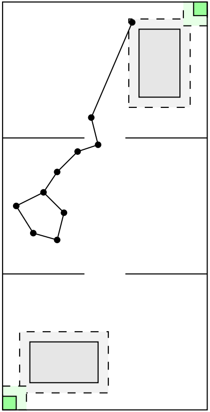
Example

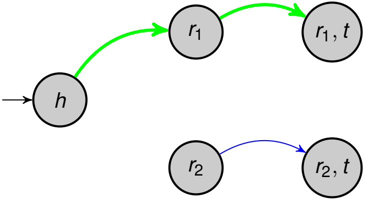




$$DOC = \{t\}$$

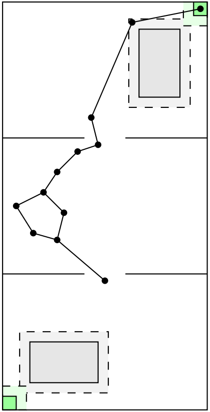
Example

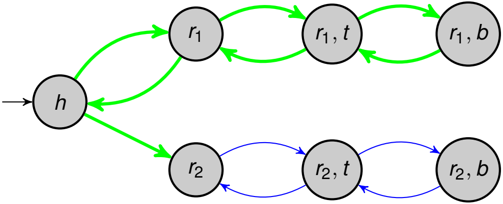




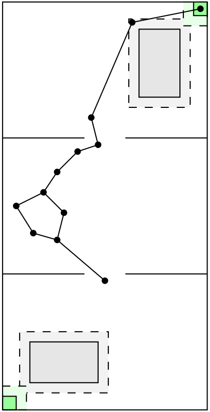
$$DOC = \{t\}$$

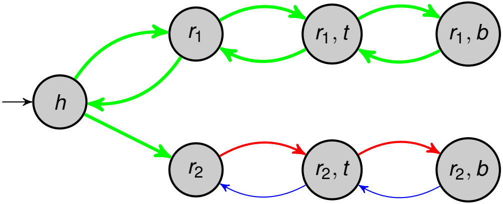
Example





Example





Our solution

- Sample a batch in the known area (initially, sensing radius) using biasing (initially, no biasing).
- Learn from the newly added edges and improve the bias. ✓
- Find the best move and go there.

Our solution

- Sample a batch in the known area (initially, sensing radius) using biasing (initially, no biasing).
- Learn from the newly added edges and improve the bias. ✓
- Find the best move and go there.

Where to move?

Possible options:

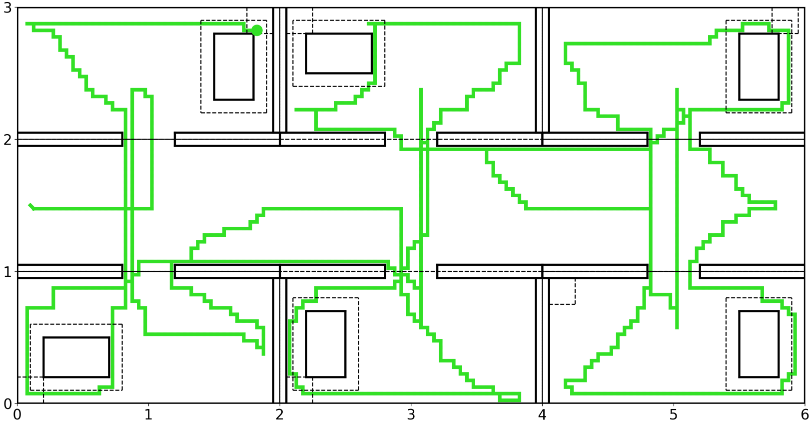
- One of the frontiers (as in frontier exploration).
- Points sampled in the current batch which were according to the bias. Define IG for these based on how far they are from the accepting state in the product automaton.

Our solution

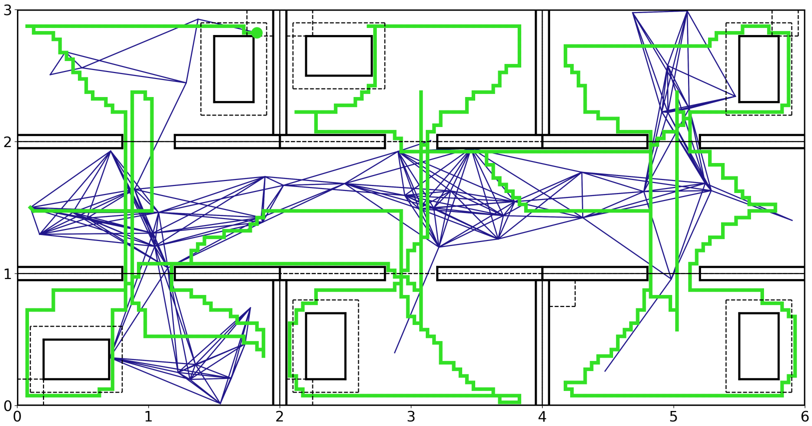
- Sample a batch in the known area (initially, sensing radius) using biasing (initially, no biasing).
- Learn from the newly added edges and improve the bias. ✓
- Find the best move and go there. ✓

Experiments

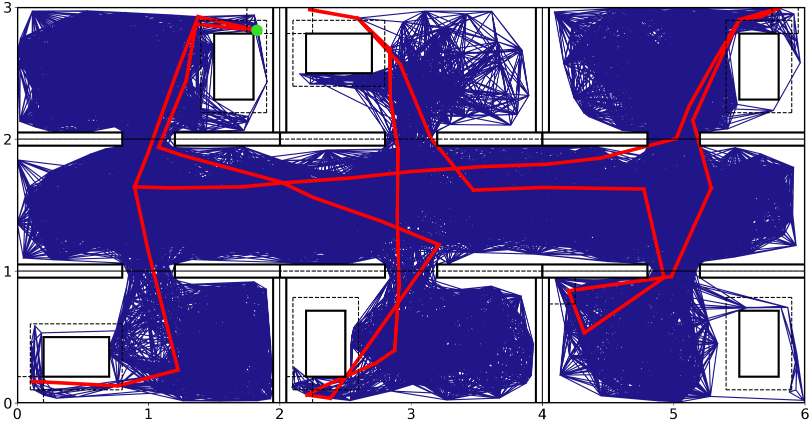
A run of “First Explore Then Plan"



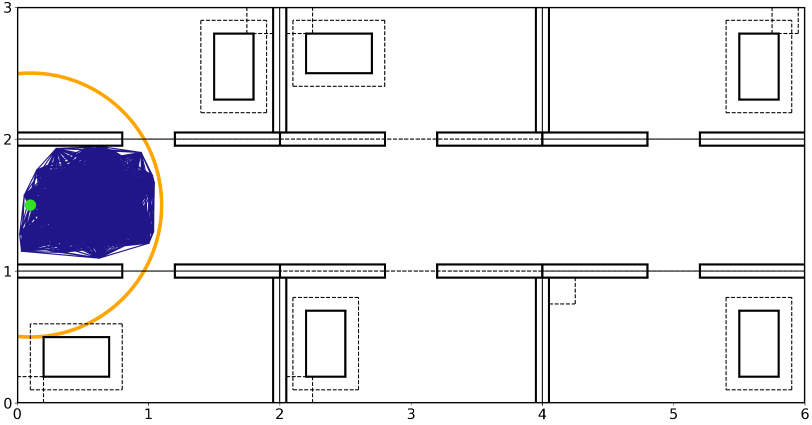
A run of “First Explore Then Plan"



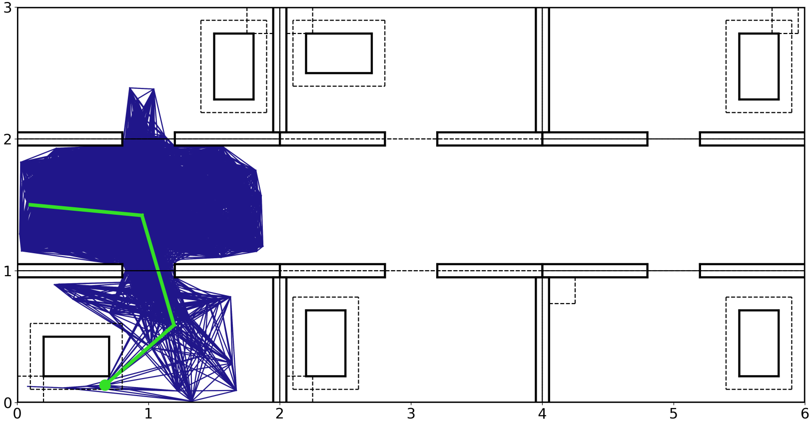
A run of “First Explore Then Plan"



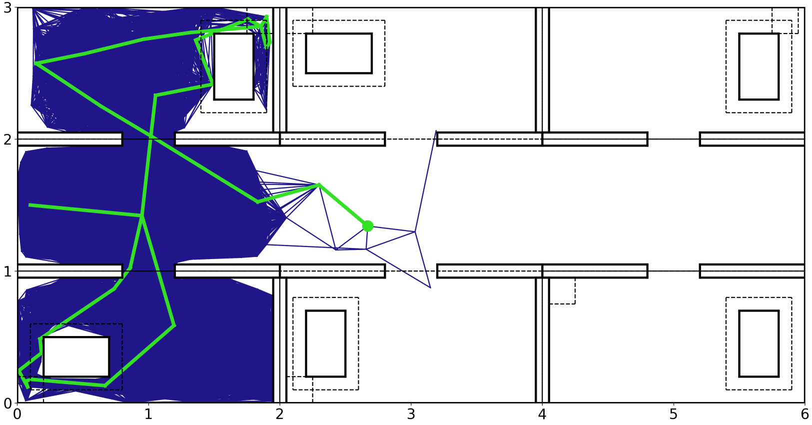
A run of “Simultaneously with biasing"



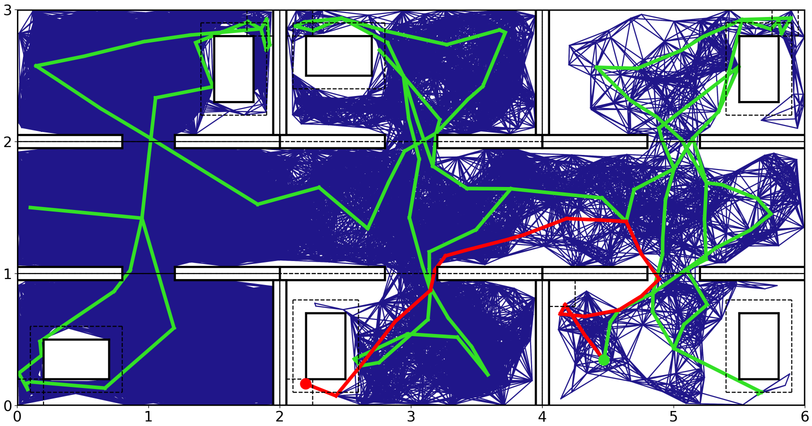
A run of “Simultaneously with biasing"



A run of “Simultaneously with biasing"



A run of “Simultaneously with biasing"



Experiments: See Through Desks

Compared different approaches on 100 randomly generated office like environments.

	See-through Desks		
	Explore, then plan	**Simultaneous**	**Simult. biased**
Total length	77.3 (7.5)	56.6 (8.0)	29.4 (5.0)
Exploration length	57.1 (3.2)	37.5 (7.1)	28.0 (4.9)
Remaining plan l.	20.2 (7.0)	19.1 (3.6)	1.3 (1.8)
Total Time	7.8 (2.0)	6.4 (2.3)	7.3 (1.9)
RRG size	1931.2 (460.9)	1938.6 (559.5)	1793.6 (312.1)

Experiments: Opaque Desks

Compared different approaches on 100 randomly generated office like environments.

	Opaque Desks		
	Explore, then plan	**Simultaneous**	**Simult. biased**
Total length	79.1 (7.1)	62.9 (16.5)	32.3 (11.8)
Exploration length	57.8 (4.9)	44.4 (16.6)	31.3 (12.1)
Remaining plan l.	21.3 (5.1)	18.5 (3.4)	1.1 (1.8)
Total Time	9.6 (2.5)	8.3 (3.2)	9.1 (2.4)
RRG size	2313.8 (550.9)	1868.7 (498.2)	1901.4 (301.2)

Conclusion and Future Work

Conclusion and Future Work

Conclusion

- Gave an algorithm for Motion Planning of scLTL missions in an unknown environment.

Conclusion and Future Work

Conclusion

- Gave an algorithm for Motion Planning of scLTL missions in an unknown environment.
- Use the previous experiences to help in the future.

Conclusion and Future Work

Conclusion

- Gave an algorithm for Motion Planning of scLTL missions in an unknown environment.
- Use the previous experiences to help in the future.
- Reduce the path traversed by more than 50% compared to the naive approach.

What's next?

- Can we do full LTL?.

Conclusion and Future Work

Conclusion

- Gave an algorithm for Motion Planning of scLTL missions in an unknown environment.
- Use the previous experiences to help in the future.
- Reduce the path traversed by more than 50% compared to the naive approach.

What's next?

- Can we do full LTL?.
- Experiments on actual robots.

Conclusion and Future Work

Conclusion

- Gave an algorithm for Motion Planning of scLTL missions in an unknown environment.
- Use the previous experiences to help in the future.
- Reduce the path traversed by more than 50% compared to the naive approach.

What's next?

- Can we do full LTL?.
- Experiments on actual robots.
- Figure out atomic propositions on-the-fly to make it work in a completely unknown environment.

Thank You!