Safe and Efficient Navigation in Dynamic Environments

Anirudh Vemula

Robotics Institute
Carnegie Mellon University

July 5th, 2017

Research Focus: Navigation in Dynamic Environments

Dynamic Environments: Presence of other dynamic agents

Scripted dynamic agents [1]

Unscripted dynamic agents [2]

Objective

Given a dynamic environment, a start location and a goal, find a

- safe,
- dynamically feasible,
- bounded cost suboptimal path for the robot

Challenges

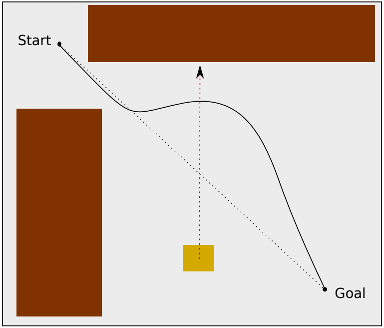
To navigate in dynamic environments, the robot needs to

1. quickly plan a path accounting for dynamic agents
2. have an accurate model of the dynamics of environment

In this thesis, we will tackle both challenges

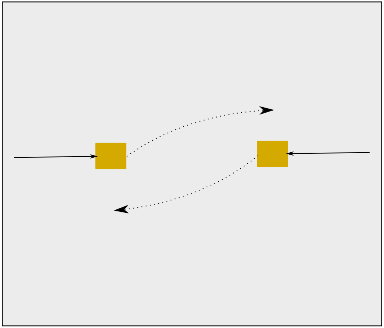
Outline

Path Planning



Find a path for the robot that avoids collisions with static and dynamic obstacles, and is dynamically feasible to execute

Trajectory Prediction



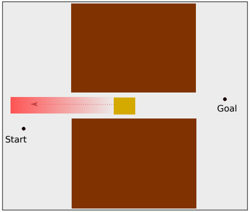
Given the past trajectories of dynamic agents in the environment, predict their future trajectories until a fixed time horizon

Outline

Problem Statement

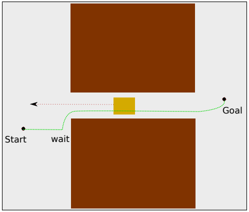
Assuming a perfect model of world dynamics **exists**, given a start and goal configuration, find a path that is safe and feasible to execute, with bounded cost suboptimality

Past Work : Planning in Dynamic Environments



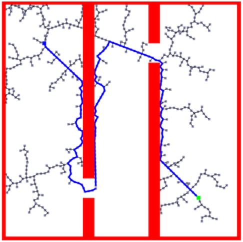
Planning without time dimension[3]

Incomplete



Safe Interval Path Planning[4]

Wait in-place

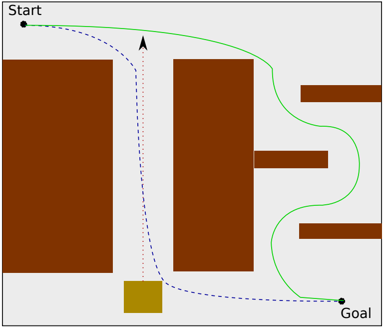


RRT[5]

No optimality guarantees

Motivation

Low-D heuristics tend to be wrong in dynamic environments



The solution path (green) is against heuristic

Hence, heuristic-based planners are extremely slow

Adaptive Dimensionality Graph

Core Idea : Consider time dimension only in regions where there is a potential dynamic obstacle collision. Plan in low-dimension elsewhere.

Example AD graph

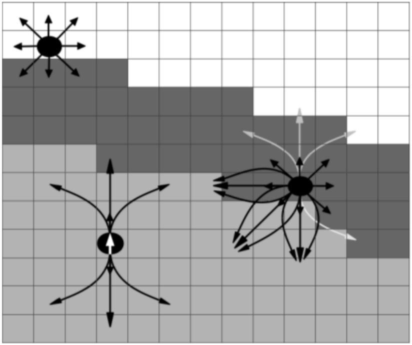


Figure : Low-D: (x, y), High-D: $(x, y, \theta)^6$

Extending to Planning in Dynamic Environments

State-spaces

- S^{ld} : Only spatial variables e.g. (x, y)
- S^{hd} : Spatio-temporal variables e.g. (x, y, t)
- Bound the time dimension in S^{hd} by a upper bound T

Transition sets

- T^{ld} : Transitions between states in S^{ld}
- T^{hd} : Transitions between states in S^{hd}
- Collision with dynamic obstacle only checked in T^{hd}

Extending to Planning in Dynamic Environments

Projections

- Projection function $\lambda : S^{hd} \rightarrow S^{ld}$
- Inverse projection function λ^{-1} is given by:

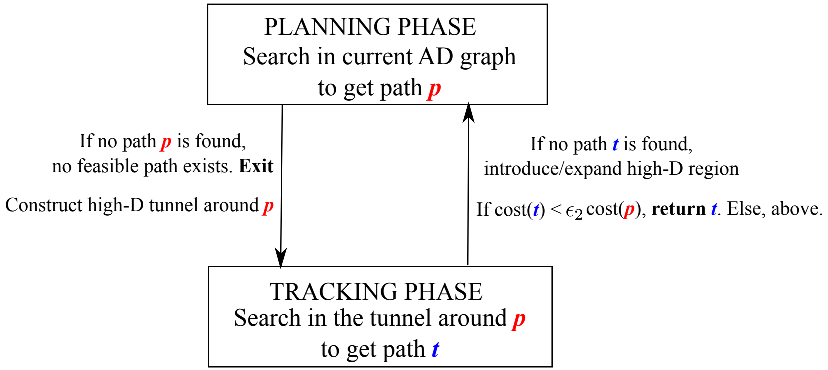
$$\lambda^{-1}(X^{ld}) = \{X^{hd} | \lambda(X^{hd}) = X^{ld}, t_d(X^{ld}) \leq t(X^{hd}) \leq T\}$$

- We obtain t_d using a time-optimal Dijkstra search in G^{ld} to find the least time taken to reach any state from start

Adaptive transitions

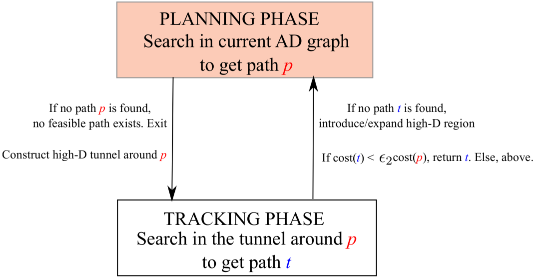
- $(X^{hd}, Y^{ld}) \in T^{ad}$, if $Y^{ld} = \lambda(X^{hd})$
- $(X^{ld}, Y^{hd}) \in T^{ad}$, if $(X^{hd}, Y^{hd}) \in T^{hd}$ and $X^{hd} \in \lambda^{-1}(X^{ld})$

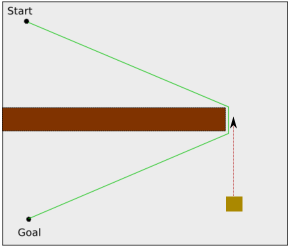
Main Loop



PLANNING PHASE
Search in current AD graph
to get path p
If no path p is found,
no feasible path exists. **Exit**
Construct high-D tunnel around p
If no path t is found,
introduce/expand high-D region
If $\mathrm{cost}(t) < \epsilon_2\,\mathrm{cost}(p)$, **return** t. Else, above.
TRACKING PHASE
Search in the tunnel around p
to get path t

Main Loop : Start with $G^{ad} = G^{ld}$





Main Loop : Construct High-D tunnel around path

PLANNING PHASE
Search in current AD graph
to get path p

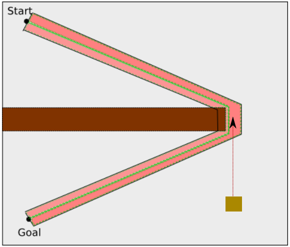
If no path p is found,
no feasible path exists. Exit

Construct high-D tunnel around p

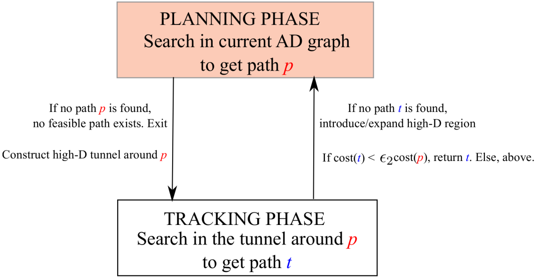
If no path t is found,
introduce/expand high-D region

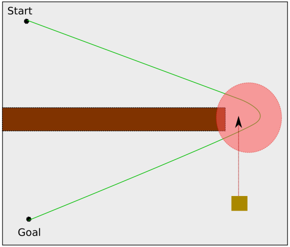
If $\text{cost}(t) < \epsilon_2 \text{cost}(p)$, return t. Else, above.

TRACKING PHASE
Search in the tunnel around p
to get path t

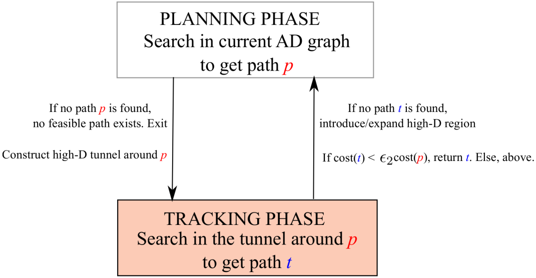


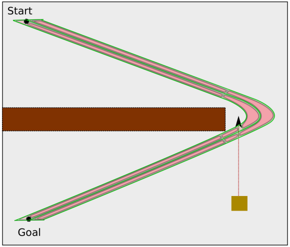
Main Loop : Add High-D region and search G^{ad}



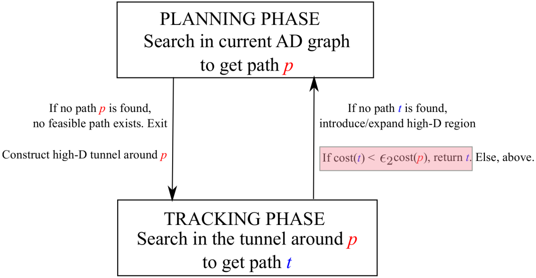


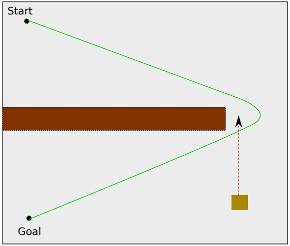
Main Loop : No collisions in High-D tunnel





Main Loop : Success





Theoretical Properties

Theorem 1 : Completeness

If no path $\pi_{G^{ad}}$ is found in planning phase, no collision-free, feasible path exists that can reach goal from start within time T in G^{hd}

Theorem 2 : Bounded cost suboptimality

If a weighted A* is used for both planning and tracking phase, then the cost of resulting path (if found) has cost no more than $\epsilon_{plan} \cdot \epsilon_{track} \cdot c(\pi^*_{G^{hd}}(X_S, X_G))$

3D Nonholonomic Robot

- Simulated 3D (x, y, θ) robot in randomly generated dynamic environments
- Hence, S^{hd} is a 4D-(x, y, θ, t) space. We chose S^{ld} to be the 2D-(x, y) space
- Cost of a path is the time taken to execute the path
- We compare our performance against 4D-(x, y, θ, t) HCA* with 2D-(x, y) Dijkstra heuristic

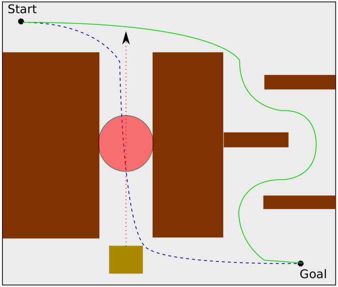
Quantitative Results

Algorithm	Number of Success (Out of 50)	Epsilon ($\epsilon_{plan} \cdot \epsilon_{track}$)	time (secs) mean	time (secs) std dev
Adaptive	**41**	1.1	**6.7**	0.8
HCA*	5	1.1	91.0	71.2
Adaptive	**43**	1.5	**11.7**	14.0
HCA*	21	1.5	70.3	86.7
Adaptive	**46**	2.0	**18.5**	26.6
HCA*	23	2.0	35.8	69.8

Table : Results on 50 environments with 30 dynamic obstacles.

Discussion

- Path suggested by Low-D heuristic blocked by dynamic obstacles, our approach outperforms baseline significantly
- Otherwise, we have comparable performance with baseline



Summary

PLANNING PHASE
Search in current AD graph
to get path p

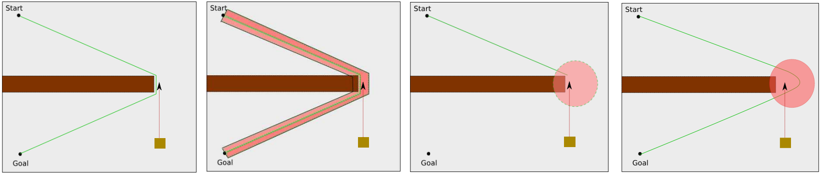
If no path p is found,
no feasible path exists. **Exit**

Construct high-D tunnel around p

TRACKING PHASE
Search in the tunnel around p
to get path t

If no path t is found,
introduce/expand high-D region

If $\text{cost}(t) < \epsilon_2 \, \text{cost}(p)$, **return** t. Else, above.



Outline

Problem Statement

Given sensory information regarding the past trajectories of agents in a dynamic environment, predict their future trajectories

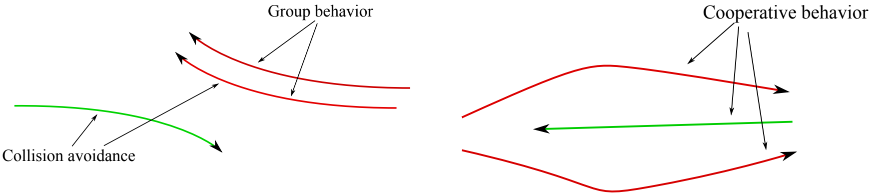
In this work, we will focus on human crowds

Cooperation in Crowds

Sparse[7] Moderate[8] Dense[8]

Humans navigate through crowds by adapting their trajectories to those of other people in the vicinity

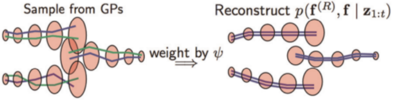
Crowd interactions



To capture such interactions, we need to model the joint distribution of trajectories

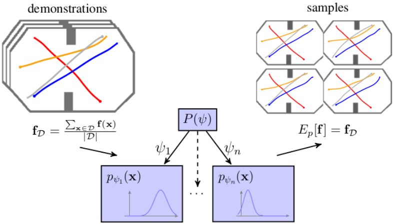
But, a joint distribution model scales with the number of agents in the crowd

Past Work : Navigation in Human Crowds



Interacting Gaussian Processes[3]

Handcrafted potential term



Inverse Reinforcement Learning[8]

Scales poorly

Notation

- $\mathbf{f}^{(i)}$: Trajectory of agent i, a sequence of (x, y) locations for discrete time-steps 1 to T
- $\mathbf{z}_{1:t}^{(i)}$: Observed locations of agent i at time-steps 1 to t
- $\mathbf{z}_{1:t}$: Observed locations of all agents
- $\mathbf{f}$: Trajectories of all agents
- $(\mathbf{v_x})_t^{(i)}, (\mathbf{v_y})_t^{(i)}$: x- and y-velocity of agent i at time t

Note: We assume there are a fixed set of goals G in the environment

Joint Prediction Model

Independent model :

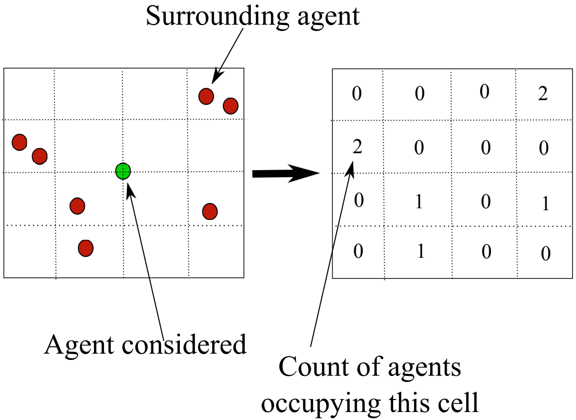
$$P(\mathbf{f}|\mathbf{z}_{1:t}) = \prod_{i=1}^{N} P(\mathbf{f}^{(i)}|\mathbf{z}_{1:t}^{(i)})$$

Interacting Gaussian processes (Trautman and Krause 2015) :

$$P(\mathbf{f}|\mathbf{z}_{1:t}) = \frac{1}{Z} \underbrace{\psi(\mathbf{f}^{(1)}, \cdots, \mathbf{f}^{(N)})}_{\text{Handcrafted potential}} \prod_{i=1}^{N} P(\mathbf{f}^{(i)}|\mathbf{z}_{1:t}^{(i)})$$

Instead of handcrafted potential, we couple predictions **f** using Occupancy Grids

Occupancy Grids



Scaling up using occupancy grids

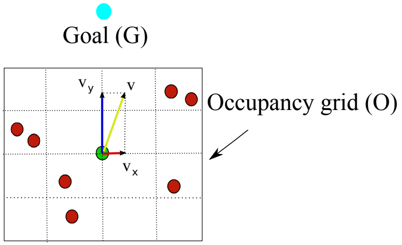
Given observed part of trajectories $\mathbf{z}_{1:t}$,

$$\begin{aligned} P(\mathbf{f}|\mathbf{z}_{1:t}) &= \sum_{\mathbf{g}} P(\mathbf{f}|\mathbf{g}, \mathbf{z}_{1:t}) P(\mathbf{g}|\mathbf{z}_{1:t}) \\ &= \sum_{\mathbf{g}} \prod_{i=1}^{N} \underbrace{P(\mathbf{f}^{(i)}|\mathbf{O}_{1:t}^{(i)}, \mathbf{g}^{(i)}, \mathbf{z}_{1:t}^{(i)})}_{\text{prediction}} \underbrace{P(\mathbf{g}^{(i)}|\mathbf{z}_{1:t}^{(i)})}_{\text{inferring goal}} \end{aligned}$$

where $\mathbf{O}_t^{(i)}$ is the occupancy grid of agent i at time t and $\mathbf{g}^{(i)}$ is the goal location of agent i.

Training Local Interaction Model

For each goal $\mathbf{g} \in G$, where G is the set of goals in the environment we estimate the distribution $P(\mathbf{v_x}|\mathbf{O}, \mathbf{g})$ and $P(\mathbf{v_y}|\mathbf{O}, \mathbf{g})$ using Gaussian Process regression



Learn $P(v|O, G)$ from training data

Goal Inference

$$P(\mathbf{f}|\mathbf{z}_{1:t}) = \sum_{\mathbf{g}} \prod_{i=1}^{N} \underbrace{P(\mathbf{f}^{(i)}|\mathbf{O}_{1:t}^{(i)}, \mathbf{g}^{(i)}, \mathbf{z}_{1:t}^{(i)})}_{\text{prediction}} \underbrace{P(\mathbf{g}^{(i)}|\mathbf{z}_{1:t}^{(i)})}_{\text{inferring goal}}$$

At inference time,

$$\begin{aligned} P(\mathbf{g}^{(i)}|\mathbf{z}_{1:t}^{(i)}) &\propto P(\mathbf{z}_{1:t}^{(i)}|\mathbf{g}^{(i)})P(\mathbf{g}^{(i)}) \quad \text{Bayes rule} \\ &\propto \underbrace{P((\mathbf{v_x})_{1:t}^{(i)}|\mathbf{O}_{1:t}^{(i)}, \mathbf{g}^{(i)})P((\mathbf{v_y})_{1:t}^{(i)}|\mathbf{O}_{1:t}^{(i)}, \mathbf{g}^{(i)})}_{\text{From trained model}} \end{aligned}$$

Trajectory Prediction

$$P(\mathbf{f}|\mathbf{z}_{1:t}) = \sum_{\mathbf{g}} \prod_{i=1}^{N} \underbrace{P(\mathbf{f}^{(i)}|\mathbf{O}_{1:t}^{(i)}, \mathbf{g}^{(i)}, \mathbf{z}_{1:t}^{(i)})}_{\text{prediction}} \underbrace{P(\mathbf{g}^{(i)}|\mathbf{z}_{1:t}^{(i)})}_{\text{inferring goal}}$$

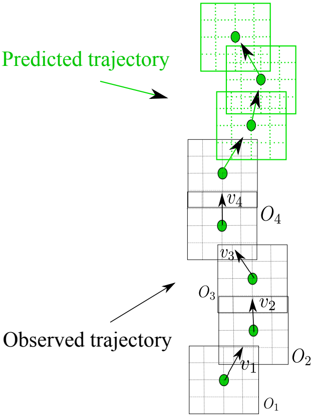
For each time-step $t' > t$, we sample from the predictive GP distribution

$$P((\mathbf{v_x})_{t'}^{(i)}|\mathbf{O}_{t'}^{(i)}, \mathbf{g}^{(i)}, (\mathbf{v_x})_{1:t}^{(i)}, \mathbf{O}_{1:t}^{(i)})$$

$$P((\mathbf{v_y})_{t'}^{(i)}|\mathbf{O}_{t'}^{(i)}, \mathbf{g}^{(i)}, (\mathbf{v_y})_{1:t}^{(i)}, \mathbf{O}_{1:t}^{(i)})$$

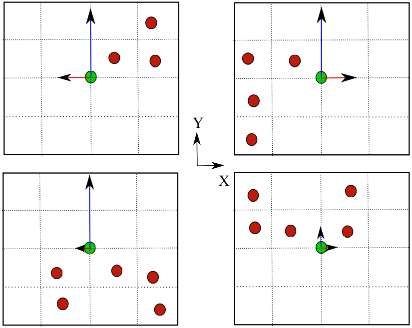
to compute future locations $\mathbf{f}_{t'}$ and occupancy grids $\mathbf{O}_{t'}$ for all agents

Trajectory Prediction



Predict velocities and compute
occupancy grids at each time-step

Results: Learned Behaviors



Collision avoidance

Cooperative behavior

Qualitative Results

IGP

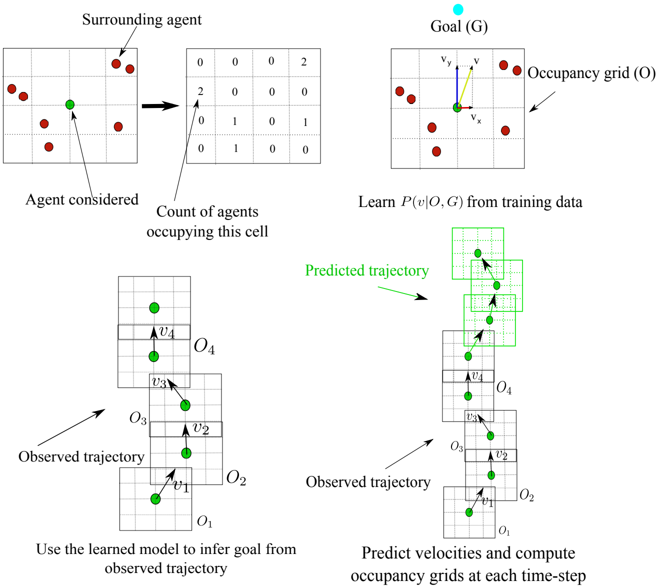
Our approach

Quantitative Results

Table : Prediction errors (in pixels) for IGP and our approach

Metric	Prediction horizon (H)	IGP	Our Approach
Avg. Disp. Error	1	**3.42**	4.42
	2	**5.66**	6.14
	5	15.75	**12.09**
	10	21.59	**18.23**
	20	41.51	**34.63**
Final Disp. Error	1	**3.42**	4.42
	2	**7.12**	7.78
	5	23.18	**19.77**
	10	38.75	**28.31**
	20	67.41	**54.2**

Summary



Surrounding agent
Agent considered
0 0 0 2
2 0 0 0
0 1 0 1
0 1 0 0
Count of agents occupying this cell
Goal (G)
v_y
v
v_x
Occupancy grid (O)
Learn $P(v|O,G)$ from training data
v_4
O_4
v_3
O_3
v_2
O_2
v_1
O_1
Observed trajectory
Use the learned model to infer goal from observed trajectory
Predicted trajectory
v_4
O_4
v_3
O_3
v_2
O_2
v_1
O_1
Observed trajectory
Predict velocities and compute occupancy grids at each time-step

Outline

Conclusion

In this thesis, we have successfully explored and answered the following two questions:

1. Given a model of world dynamics, how to obtain safe, feasible and bounded sub-optimal paths for robots in dynamic environments? Path Planning with Adaptive Dimensionality
2. How do you model complex world dynamics such as cooperative behavior in dynamic environments, specifically human crowds? Occupancy grid-based joint modeling

Future Work

Path Planning

- Verify the performance on a mobile robot in a scripted dynamic environment
- Make planner incremental - reuse search tree from previous iterations
- Account for uncertainty in the world dynamics model predictions to obtain frequency of replanning

Future Work

Modeling World Dynamics

- Account for static obstacles within the model
- Relax the assumption of fixed goals in the environment
- Local interaction assumption is not necessarily true. Attend to important surrounding agents (we have recently submitted a work in this direction)

Thank you. Questions?

Publications:

1. Anirudh Vemula, Katharina Muelling, Jean Oh. **Path Planning in Dynamic Environments with Adaptive Dimensionality**. SoCS 2016
2. Anirudh Vemula, Katharina Muelling, Jean Oh. **Modeling Cooperative Navigation in Dense Human Crowds**. ICRA 2017
3. Anirudh Vemula, Katharina Muelling, Jean Oh. **Social Attention: Modeling Attention in Human Crowds**. Submitted to CoRL 2017

[1] https://www.youtube.com/watch?v=z_R8feyCu-M

[2] https://www.youtube.com/watch?v=dPSxgQgPv28

[3] Trautman, Peter, and Andreas Krause. "Unfreezing the robot: Navigation in dense, interacting crowds." Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on. IEEE, 2010.

[4] Likhachev, Maxim, and Dave Ferguson. "Planning long dynamically feasible maneuvers for autonomous vehicles." The International Journal of Robotics Research 28, no. 8 (2009): 933-945.

[5] Phillips, Mike, and Maxim Likhachev. "Sipp: Safe interval path planning for dynamic environments." In Robotics and Automation (ICRA), 2011 IEEE International Conference on, pp. 5628-5635. IEEE, 2011.

[6] Bekris, Kostas E., and Lydia E. Kavraki. "Greedy but safe replanning under kinodynamic constraints." In Robotics and Automation, 2007 IEEE International Conference on, pp. 704-710. IEEE, 2007.

[7] Gochev, Kalin, Benjamin Cohen, Jonathan Butzke, Alla Safonova, and Maxim Likhachev. "Path planning with adaptive dimensionality." In Fourth annual symposium on combinatorial search. 2011.

[8] Lerner, Alon, Yiorgos Chrysanthou, and Dani Lischinski. "Crowds by example." In Computer Graphics Forum, vol. 26, no. 3, pp. 655-664. Blackwell Publishing Ltd, 2007.

[9] Kretzschmar, Henrik, Markus Spies, Christoph Sprunk, and Wolfram Burgard. "Socially compliant mobile robot navigation via inverse reinforcement learning." The International Journal of Robotics Research 35, no. 11 (2016): 1289-1307.