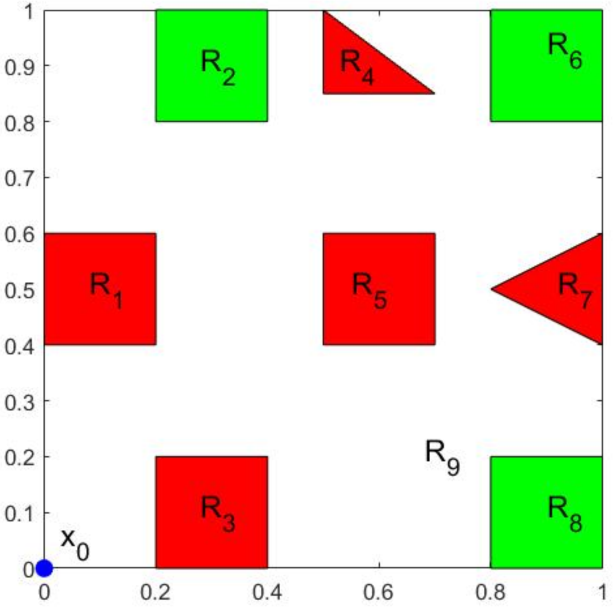
Path Planning Using Formal Methods

TEXAS
The University of Texas at Austin

Problem Formulation

- Configuration space: $\mathcal{D} \subseteq R^n, n \in N, n \geq 2$
- $\mathcal{R}$: a set of disjoint regions in $\mathcal{D}$
- $AP = \{\text{obstacle}, \text{goal}_i, \text{free}\}$
- $L: \mathcal{R} \rightarrow 2^{AP}$: properties associated to the region in $\mathcal{R}$.
- x_0: initial configuration of the robot



$\mathcal{D} \subseteq R^2$

Problem Formulation continued…

- Goal: find a control policy $\mathcal{C}$ that generates a satisfying trajectory π (in the absence of uncertainties) or that maximizes the probability of satisfying an LTL formula ϕ (in the presence of uncertainties)
- Planning in the absence of uncertainties: finite transition system, $\mathcal{T} = (X, x_0, \Delta, AP, J)$
- Planning in the presence of uncertainties: MDP, $\mathcal{M} = (Q, q_0, \mathcal{A}, P, AP, K)$

Planning in the Absence of Uncertainties

- Mechanism:
  1. Construct a graph $G = (V, E)$ using a PRM (Probabilistic RoadMap) based algorithm
  2. Design $\mathcal{T}$ using G
  3. Generate a satisfying trajectory using closed system synthesis

Construct a Graph G

$$G = (V, E), V \subset \mathcal{D}, E \subseteq V \times V$$

Algorithm 1: Modified-PRM

1: $\mathrm{V} \leftarrow \{x_0\} \cup \{Sample_i\}_{i=1,\ldots,m}$; $E \leftarrow \emptyset$

2: **foreach** $v \in V$ **do**

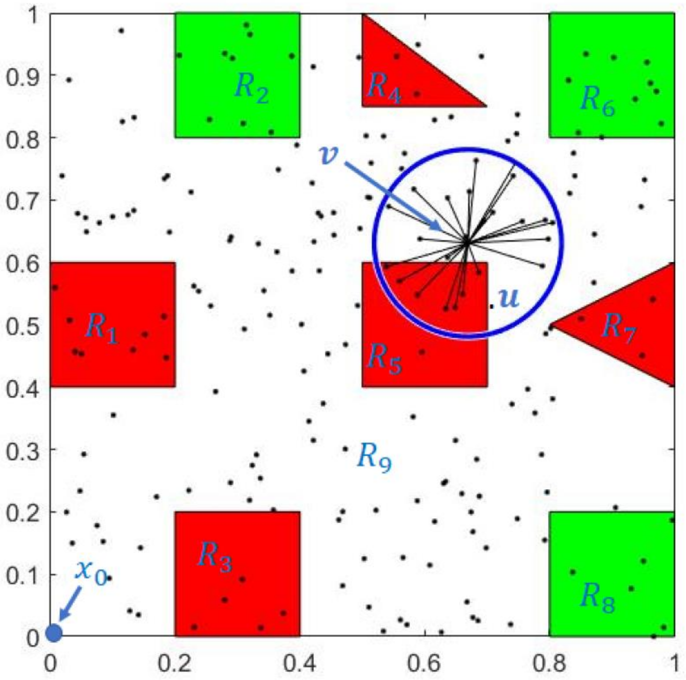
  $U \leftarrow Near(G = (V, E), v, r) \setminus \{v\}$;

  foreach $u \in U$ **do**

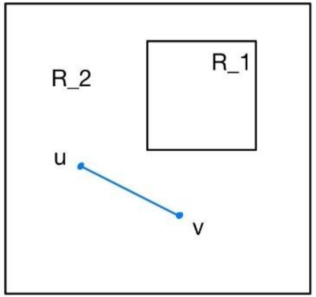
    if $isSimpleEdge(v, u)$ **then**

      $E \leftarrow E \cup \{(v, u), (u, v)\}$

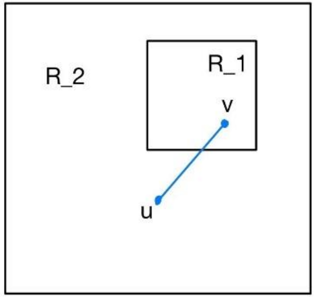
3: **return** $G = (V, E)$



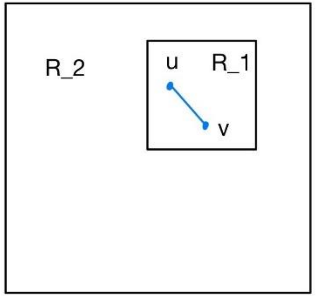
Construct a Graph G: $isSimpleEdge(v,u)$



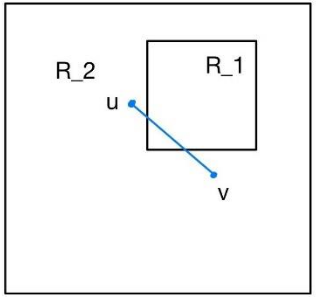
isSimpleEdge(v, u)=1



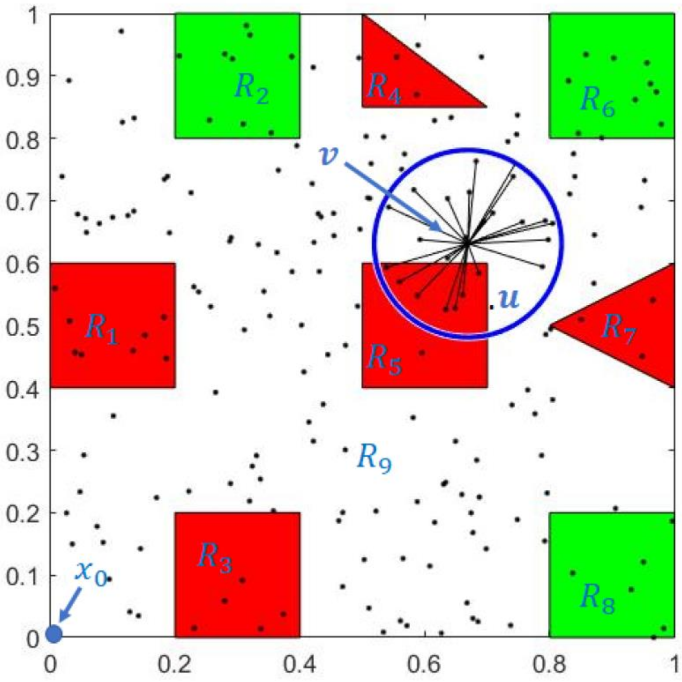
isSimpleEdge(v, u)=1



isSimpleEdge(v, u)=1

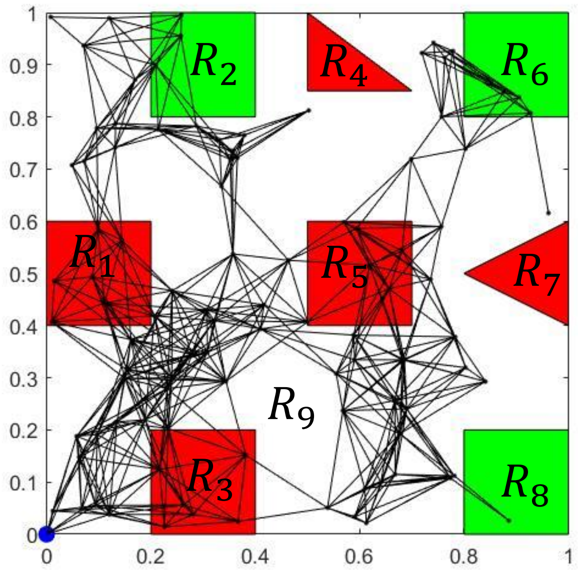


isSimpleEdge(v, u)=0



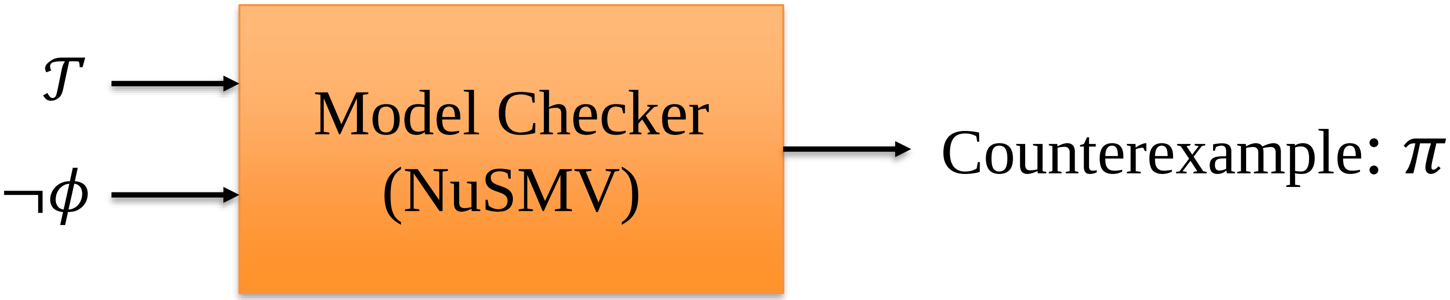
Construct a Transition System $\mathcal{T}$

- Environment $\mathcal{E} = (\mathcal{D}, x_0, \mathcal{R}, AP, L)$
- Graph $G = (V, E)$
- $\mathcal{T} = (X, x_0, \Delta, AP, J)$
  - $X = V$
  - $\Delta = E$
  - $\forall x \in X, J(x) = L(R_k)$, where $R_k \in \mathcal{R}$ is a region in $\mathcal{D}$ such that $x \in R_k$



Closed System Synthesis

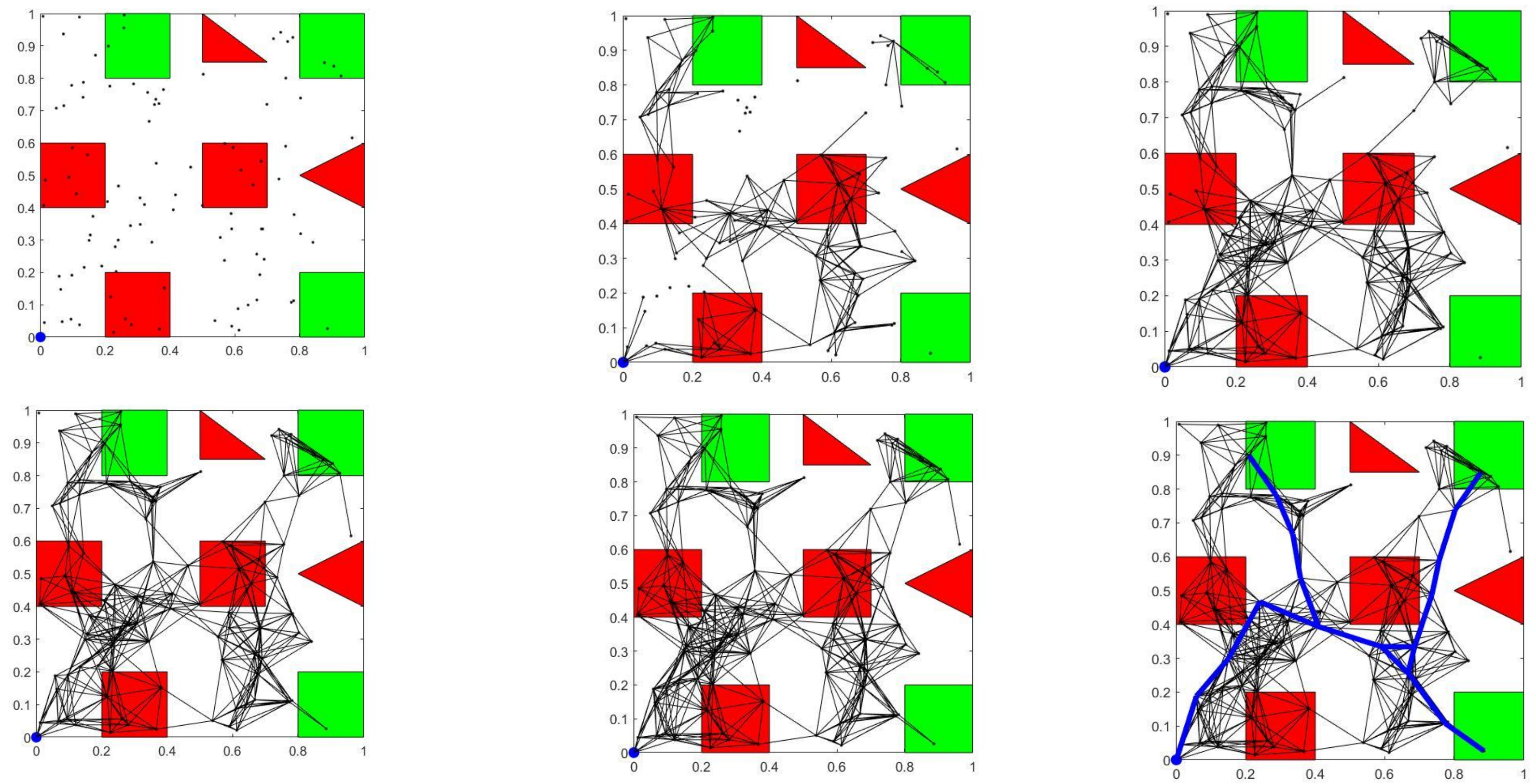
- $\phi = G\big((\neg obstacle) \wedge (F goal_1) \wedge (F goal_2) \wedge (F goal_3)\big)$
- Find a trajectory, π: $\pi \vDash \phi$.



Things to Note…

- The transition system $\mathcal{T}$ is an under-approximation of the Environment, $\mathcal{E} = (\mathcal{D}, x_0, \mathcal{R}, AP, L)$: π can be implemented in $\mathcal{E}$
- modified-PRM is probabilistically complete: as the number of samples in the construction of the graph, $m \to \infty$, the proposed mechanism finds a satisfying trajectory with probability 1 if such a trajectory exists

Results



Planning in the Presence of Uncertainties

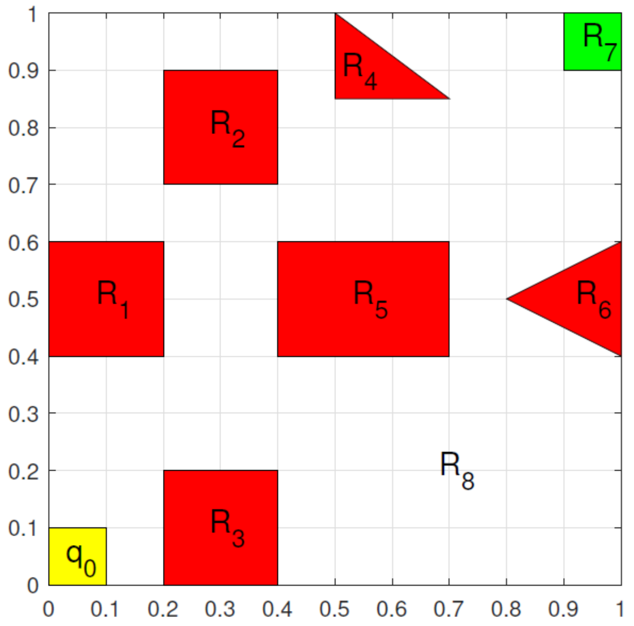
- Mechanism:

  1. Abstraction of the $\mathcal{E}$ into an MDP $\mathcal{M}$

  2. Synthesis of a control policy $\mathcal{C}$ for $\mathcal{M}$ that maximizes the probability of satisfying ϕ using probabilistic synthesis tools

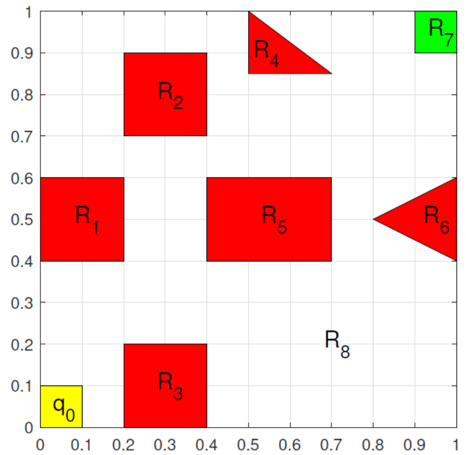
Construction of $\mathcal{M} = (Q, q_0, \mathcal{A}, P, AP, K)$

- Q = cells of the grid
- Initial state q_0: $x_0 \in q_0$
- $\forall q \in Q, \mathcal{A}(q) = \{right, up, left, down\}$
- Transition probabilities:
  - 0.8: successful transition
  - 0.1: move ±45 deg to the intended direction
- The robot bounces back to its original state when it hits the boundary
- Labeling function K:
  - $\forall\ q \in Q$, if $q \cap R_k \neq \emptyset$ and $L(R_k) = \{obstacle\}$ then $K(q) = L(R_k)$
  - $\forall\ q \in Q$, if $q \subseteq R_k$ and $L(R_k) = \{free\}$ or $\{goal_i\}$ then $K(q) = L(R_k)$



Synthesis of the Optimal Control Policy $\mathcal{C}^*$

- $\phi = \neg obstacle \, \mathcal{U} \, goal_1$
- $\mathcal{P}_{max}[\phi]$=?, $\mathcal{C}^*$=?



Synthesis of the Optimal Control Policy $\mathcal{C}^*$

- $Q = Q^{yes} \cup \mathrm{Q}^{no} \cup Q^{?}$
  - Q^{yes}: states that satisfy ϕ with probability 1 for some $\mathcal{C}$
  - Q^{no}: states that don't satisfy ϕ for all $\mathcal{C}$
  - $Q^{?}$: remaining states
- y_q: probability of satisfying ϕ from state q

$$min_{y_q} \sum_{q \in Q} y_q$$

subject to:

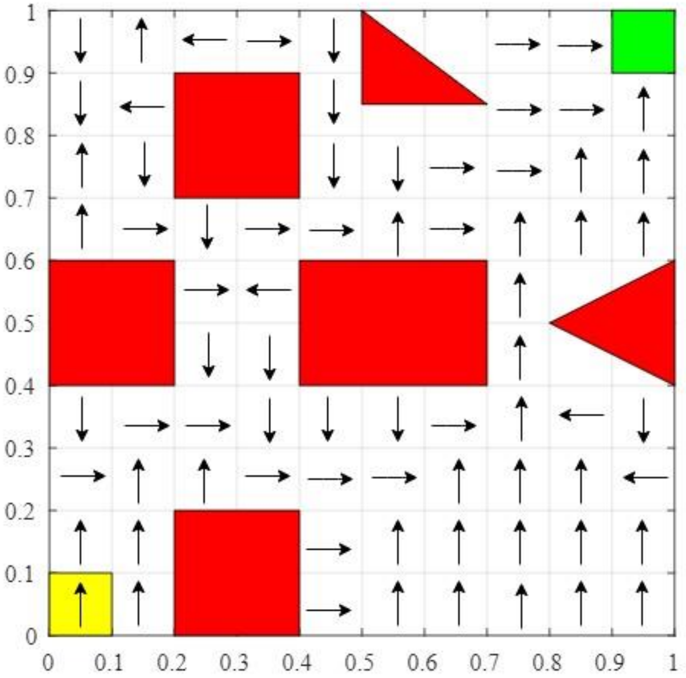
$$y_q = 1 \quad \forall q \in Q^{yes}$$

$$y_q = 0 \quad \forall q \in Q^{no}$$

$$y_q \geq \sum_{q' \in Q} P(q, \alpha, q') \cdot y_{q'} \quad \forall \alpha \in \mathcal{A}(q)$$

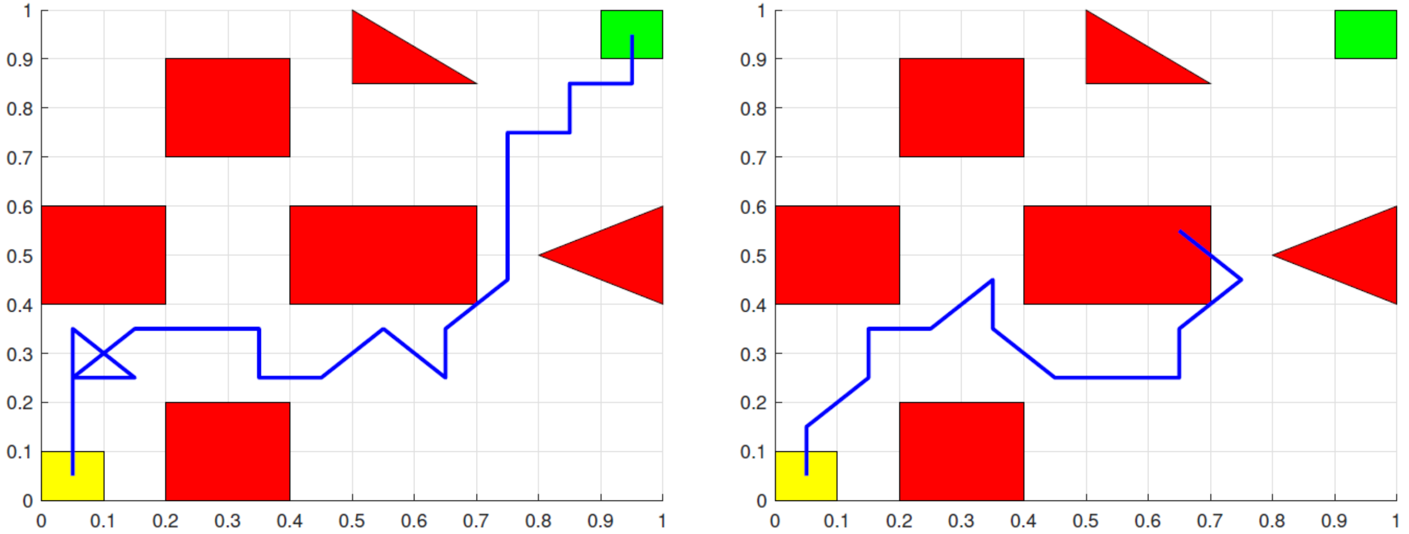
$$\forall q \in Q \backslash (Q^{yes} \cup Q^{no})$$

Optimal Control Policy $\mathcal{C}^*$



Optimal Control Policy $\mathcal{C}^*$

$$\mathcal{P}_{max}[\phi] = 0.79$$



(a) Trajectory satisfies ϕ (b) Trajectory doesn't satisfy ϕ

References

- S. Karaman, and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” International Journal of Robotics Research, vol. 30, pp. 846-894, 2011.
- C. Vasile and C. Belta, “Sampling-based temporal logic path planning,” in Int. Conf. on Intelligent Robots and Systems. IEEE, Nov 2013, pp. 4817–4822.
- Yoo C, Fitch R and Sukkarieh S (2013) Provably-correct stochastic motion planning with safety constraints. In: Proceedings of IEEE ICRA, pp. 981–986.
- M. Lahijanian, J. Wasniewski, S. B. Andersson, and C. Belta, “Motion planning and control from temporal logic specifications with probabilistic satisfaction guarantees,” in Proc. ICRA, 2010, pp. 3227–3232.