Where's Waldo?
Sensor-Based Temporal Logic Motion Planning

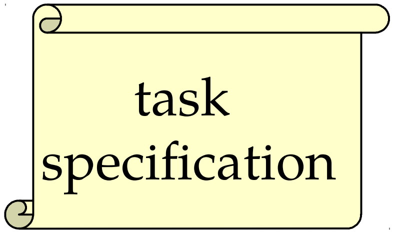
Hadas Kress-Gazit, Georgios E. Fainekos and George J. Pappas

Presented by Marios Xanthidis

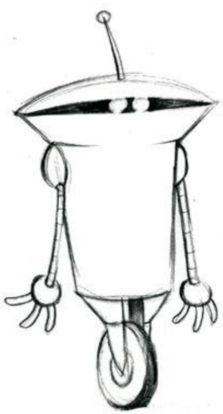
The problem

- How can we describe symbolic high level tasks and automatically formulate them into sensing and control?
- The paper:
  - presents a framework to develop hybrid controllers that satisfy high level specifications.
  - solves both the motion planning and the task planning problems.

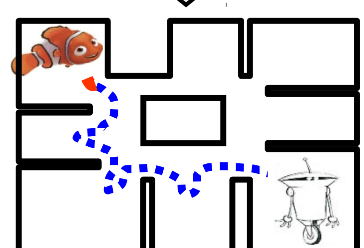
The problem



dynamic environment



robot model



correct robot motion and action

slide by Hadas Kress-Gazit

Related work

Bottom-up motion planning:

- "**Principles of Robot Motion: Theory, Algorithms, and Implementations**"

  H. Choset, K. M. Lynch, L. Kavraki, W. Burgard, S. A. Hutchinson, G. Kantor, and S. Thrun

- "**Planning Algorithms**"

  S. M. LaValle

- "**Exact robot navigation using artificial potential functions**"

  Elon Rimon and Daniel E. Kodischek

Related work

Top-down task planning:

- “**Artificial Intelligence, A Modern Approach**”

  S. Russell and P. Norvig

- “**OBDD-based universal planning for synchronized agents in non-deterministic domains**”

  R.M. Jensen and M. M. Veloso

- “**MBP: A model based planner**”

  P. Bertoli, A. Cimatti, M. Pistore, M. Roveri, , and P. Traverso

Related work

Integrated systems:

- “**Temporal logic motion planning for mobile robots**”

  Georgios E. Fainekos, Hadas Kress-Gazit, and George J. Pappas
- “**Hybrid controllers for path planning: A temporal logic approach**”

  Georgios E. Fainekos, Hadas Kress-Gazit, and George J. Pappas
- “**A fully automated framework for control of linear systems from LTL specifications**”

  M. Kloetzer and C. Belta
- “**Composition of Local Potential Functions for Global Robot Control and Navigation**”

  David C. Conner, Alfred A. Rizzi, and Howie Choset
- “**Integrated planning and control for convex-bodied nonholonomic systems using local feedback control policies**”

  D. Conner, H. Choset, and A. Rizzi
- “**Computing smooth feedback plans over cylindrical algebraic decompositions**”

  S. Lindemann and S. LaValle

Related work

Integrated systems:

- “**Temporal logic motion planning for mobile robots**”

  Georgios E. Fainekos, Hadas Kress-Gazit, and George J. Pappas
- “**Hybrid controllers for path planning: A temporal logic approach**”

  Georgios E. Fainekos, Hadas Kress-Gazit, and George J. Pappas
- “**A fully automated framework for control of linear systems from LTL specifications**”

  M. Kloetzer and C. Belta
- “**Composition of Local Potential Functions for Global Robot Control and Navigation**”

  David C. Conner, Alfred A. Rizzi, and Howie Choset
- “**Integrated planning and control for convex-bodied nonholonomic systems using local feedback control policies**”

  D. Conner, H. Choset, and A. Rizzi
- “**Computing smooth feedback plans over cylindrical algebraic decompositions**”

  S. Lindemann and S. LaValle

The Novelties

- The linear temporal logic formulation considers sensor inputs.
- The use of General Reactivity reduces the complexity to polynomial from double exponential.
- But only if the environment is modeled properly and satisfies the assumptions.

Problem Formulation

- Robot model: $\dot{p}(t) = u(t) \quad p(t) \in P \subseteq \mathbb{R}^2 \quad u(t) \in U \subseteq \mathbb{R}^2$
  $$\mathcal{Y} = \{r_1, r_2 \ldots, r_n\}$$
- Admissible environments (ϕ_e): $\mathcal{X} = \{x_1, x_2, \ldots, x_m\}$
- System Specification (ϕ_s):
  - Coverage
  - Sequencing
  - Conditions
  - Avoidance
- Given the robot model, $p(0)$, and ϕ_e construct a controller that satisfies ϕ_s.

Linear Temporal Logic (LTL)

$$\phi ::= \pi | \neg\phi | \phi_1 \vee \phi_2 | \phi_1 \wedge \phi_2 | \bigcirc \phi | \Box\phi | \Diamond\phi | \phi_1 U \phi_2$$

π : Atomic proposition.

$\neg\phi$: Negation.

$\phi_1 \vee \phi_2$: Disjunction.

$\phi_1 \wedge \phi_2$: Conjunction.

$\bigcirc\phi$: ϕ is true in the next step.

$\Box\phi$: ϕ is true always.

$\Diamond\phi$: ϕ is true at least one time (eventually).

$\phi_1 U \phi_2$: ϕ_1 is true until ϕ_2 becomes true.

Special class of LTL formulas

$$\varphi = (\varphi_e \Rightarrow \varphi_s)$$

$$\varphi_e = \varphi_i^e \wedge \varphi_t^e \wedge \varphi_g^e$$

$$\varphi_s = \varphi_i^s \wedge \varphi_t^s \wedge \varphi_g^s$$

φ_i^e, φ_i^s : Formulas constraining the initial values for sensor and system proposition.

φ_t^e : The possible evolution of state of the environment.

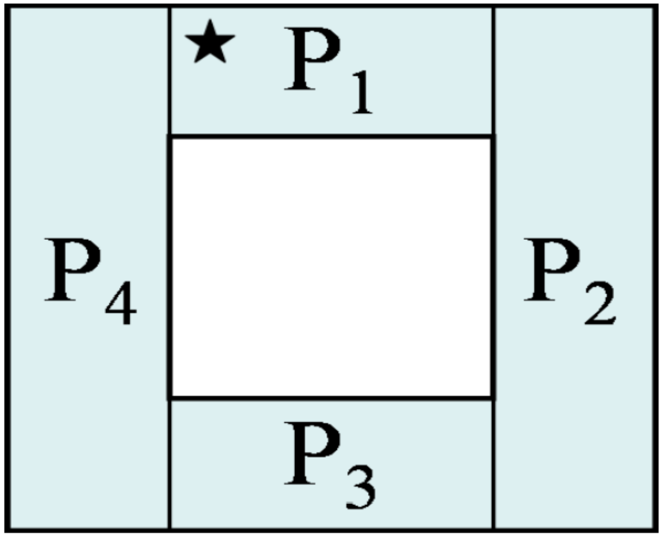
$$\Box B_i, \quad B_i \text{ constructing from subformulas in } X \cup Y \cup \bigcirc X$$

φ_t^s : The possible evolution of the state of the system.

$$\Box B_i, \quad B_i \text{ constructing from subformulas in } X \cup Y \cup \bigcirc X \cup \bigcirc Y$$

φ_g^e, φ_g^s : Goal assumptions for the environment and the system.

Example



$$Y = \{r_1, r_2, r_3, r_4\}$$

$$X = \{s^{Waldo}\}$$

$$\varphi_i^e = (\neg s^{Waldo})$$

$$\varphi_t^e = \begin{cases} \Box((\neg r_2 \wedge \neg r_4) \;\Rightarrow\; (\bigcirc s^{\mathrm{Waldo}} \Leftrightarrow s^{\mathrm{Waldo}})) \\ \bigwedge \Box(s^{\mathrm{Waldo}} \;\Rightarrow\; \bigcirc s^{\mathrm{Waldo}}) \end{cases}$$

$$\varphi_g^e = \Box\Diamond True$$

$$\varphi_i^s = (r_1 \wedge \neg r_2 \wedge \neg r_3 \wedge \neg r_4)$$

$$\varphi_t^s = \begin{cases} \bigwedge \Box(r_1 \;\Rightarrow\; (\bigcirc r_1 \vee \bigcirc r_2 \vee \bigcirc r_4)) \\ \bigwedge \Box(r_2 \;\Rightarrow\; (\bigcirc r_1 \vee \bigcirc r_2 \vee \bigcirc r_3)) \\ \bigwedge \Box(r_3 \;\Rightarrow\; (\bigcirc r_2 \vee \bigcirc r_3 \vee \bigcirc r_4)) \\ \bigwedge \Box(r_4 \;\Rightarrow\; (\bigcirc r_1 \vee \bigcirc r_3 \vee \bigcirc r_4)) \\ \bigwedge \Box(\; (\bigcirc r_1 \wedge \neg \bigcirc r_2 \wedge \neg \bigcirc r_3 \wedge \neg \bigcirc r_4) \\ \qquad \vee(\neg \bigcirc r_1 \wedge \bigcirc r_2 \wedge \neg \bigcirc r_3 \wedge \neg \bigcirc r_4) \\ \qquad \vee(\neg \bigcirc r_1 \wedge \neg \bigcirc r_2 \wedge \bigcirc r_3 \wedge \neg \bigcirc r_4) \\ \qquad \vee(\neg \bigcirc r_1 \wedge \neg \bigcirc r_2 \wedge \neg \bigcirc r_3 \wedge \bigcirc r_4) \;) \\ \bigwedge_{i \in \{2,4\}} \Box(\; (r_i \wedge \bigcirc s^{\mathrm{Waldo}}) \;\Rightarrow\; \bigcirc r_i \;) \end{cases}$$

$$\varphi_g^s = \Box\Diamond(r_2 \vee s^{\mathrm{Waldo}}) \bigwedge \Box\Diamond(r_4 \vee s^{\mathrm{Waldo}})$$

Discrete Synthesis

- The synthesis can be seen as a game between the system and the environment.
- Each step the environment makes a transition and then the system makes its own transition.
- When the robot wins, no matter the transition of the environment, we extract an automaton for the system, else the desired behavior is unrealizable.
- The winning condition is given as a General Reactivity formula

Discrete Synthesis

- Initial states of the players: $\varphi_i^e, \quad \varphi_i^s$
- Transitions: $\varphi_t^e, \quad \varphi_t^s$
- Winning condition: $\varphi = \varphi_g^e \Rightarrow \varphi_g^s$
- Output automaton is modeled as a tuple: $A = (X, Y, Q, q_0, \delta, \gamma)$

X : Set of input propositions.

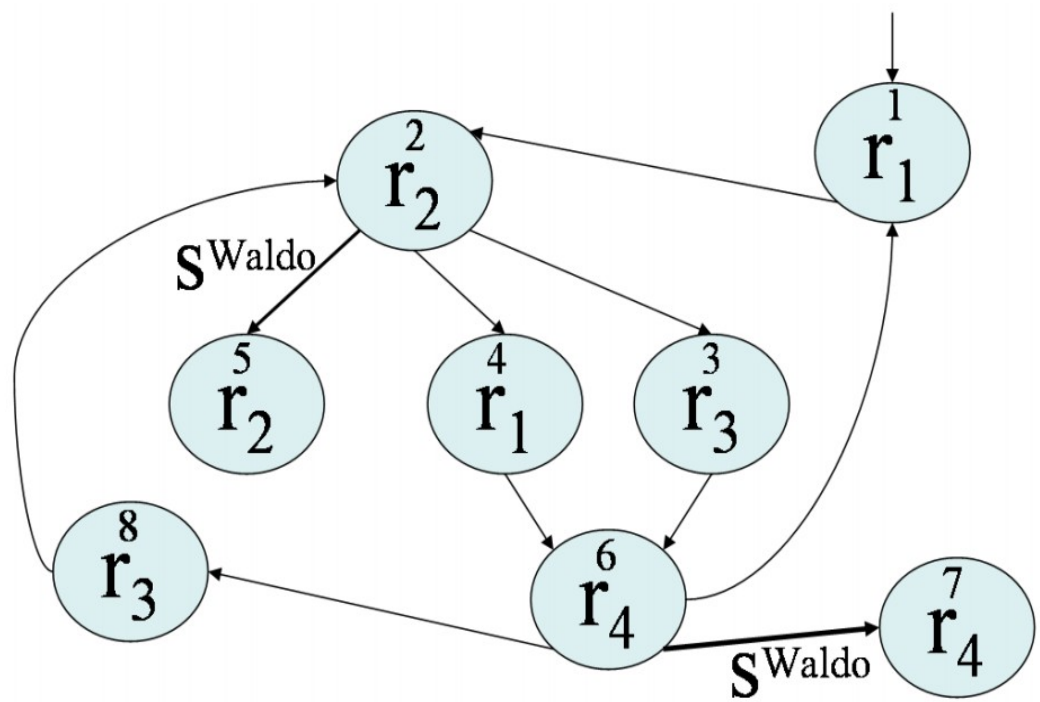
Y : Set of output propositions.

Q : Set of states.

q_0 : Initial state.

δ : Transition relation.

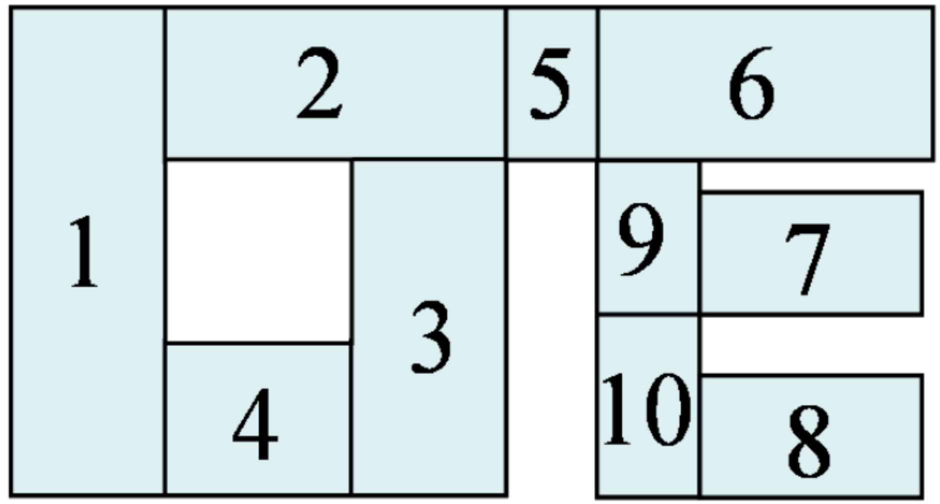
γ : State labeling function.



Controller Composition

- According to the execution of the automaton simple controllers are used for every transition to construct a hybrid controller.
- For each step:
  - The robot determines the sensors' readings.
  - The next automaton state is selected according to δ.
  - The next region the robot should go is extracted from γ.
  - When the robot reaches the new region proceed to the next step.

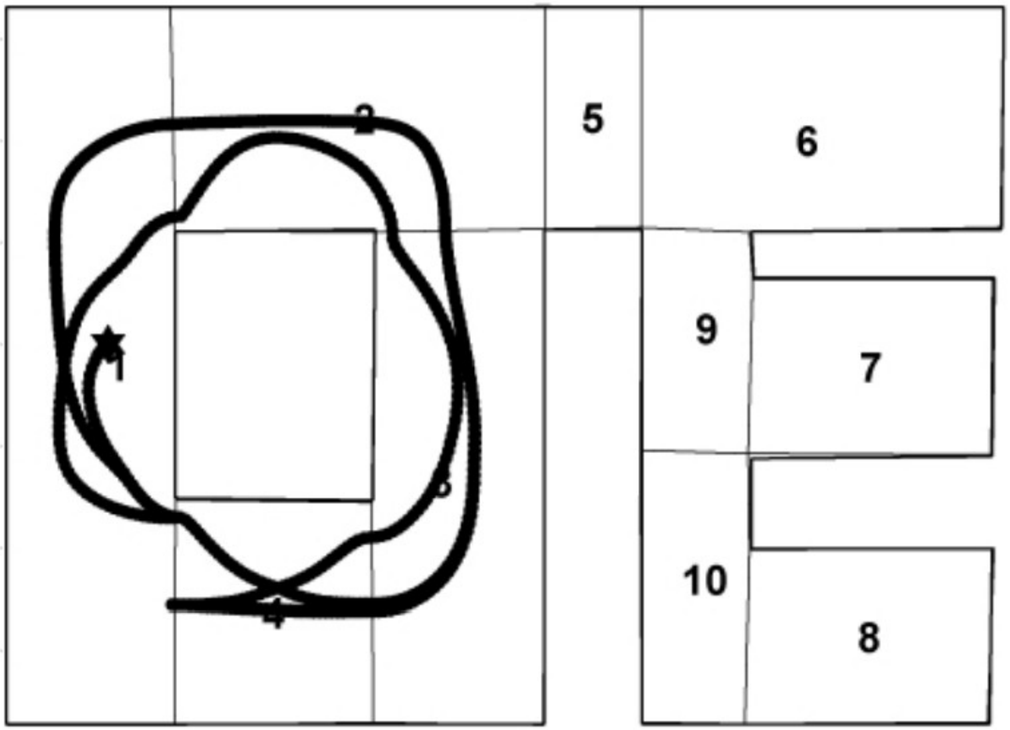
Single Robot - Nursery Scenario



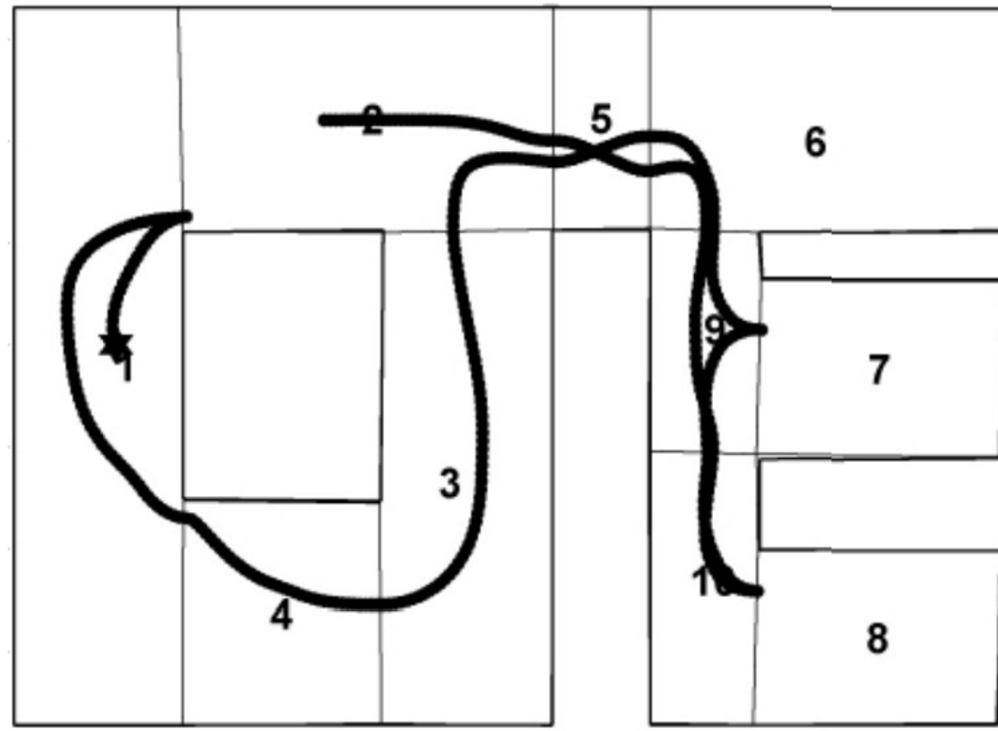
$$\varphi_e = \begin{cases} CkBby \\ \bigwedge \Box(((r_2 \vee r_4) \wedge \neg CkBby) \rightarrow (\neg \bigcirc CkBby))) \\ \bigwedge \Box(((r_6 \vee r_7 \vee r_8) \wedge CkBby) \rightarrow (\bigcirc CkBby))) \\ \bigwedge \Box(\neg(r_2 \vee r_4 \vee r_6 \vee r_7 \vee r_8) \\ \qquad \rightarrow (\bigcirc CkBby \leftrightarrow CkBby)) \\ \bigwedge \Box\Diamond True \end{cases}$$

$$\varphi_s = \begin{cases} r_1 \wedge_{i=2,\ldots,10} \neg r_i \\ \bigwedge Transitions \bigwedge Mutual\ Exclusion \\ \bigwedge_{i\in\{2,4\}} \Box\Diamond(r_i \vee \neg CkBby) \\ \bigwedge_{i\in\{6,7,8\}} \Box\Diamond(r_i \vee CkBby) \end{cases}$$

Single Robot - Nursery Scenario



(a) Babies are not crying



(b) A baby in region 4 cries and the adult is in region 8

Multi Robot - Search and Rescue

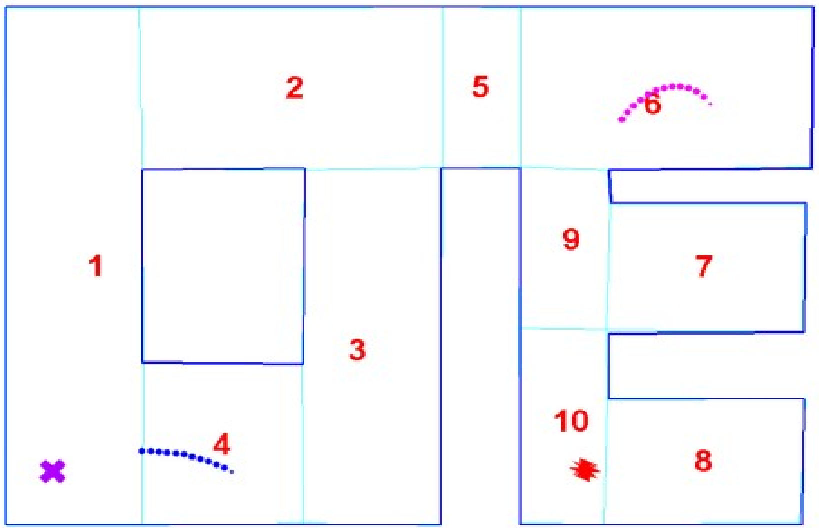
$$\varphi_1^e = \begin{cases} \bigwedge \neg r_1^3 \bigwedge \neg r_3^3 \bigwedge \neg r_7^3 \bigwedge \neg r_8^3 \\ \bigwedge \mathit{Mutual\ Exclusion\ between}\ r_i^3,\ i \in \{1,3,7,8\} \\ \bigwedge \Box\Diamond \mathit{True} \end{cases}$$

$$\varphi_1^s = \begin{cases} r_4^1 \wedge_{i=1,2,3,5,\ldots,10} \neg r_i^1 \\ \bigwedge\ \mathit{Transitions}\ \bigwedge\ \mathit{Mutual\ Exclusion} \\ \bigwedge_{i\in\{1,3,7,8\}} \Box\Diamond(r_i^1 \vee r_i^3) \end{cases}$$

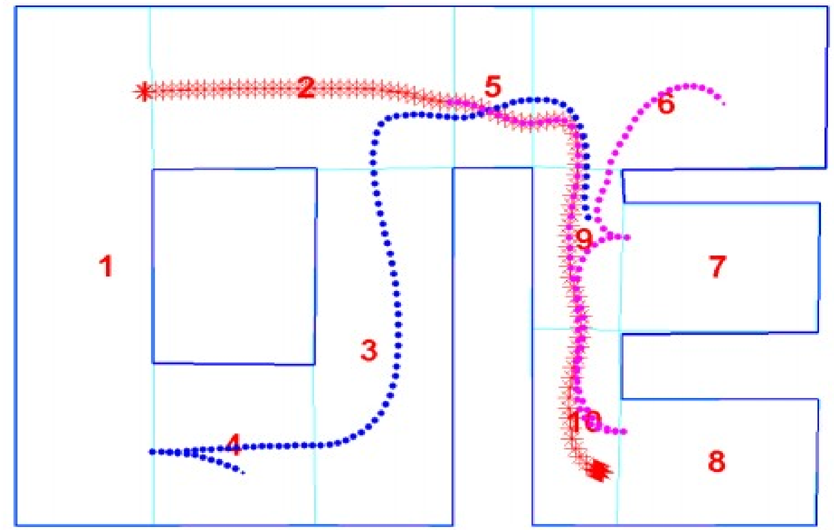
$$\varphi_3^e = \begin{cases} \wedge_{i\in\{1,3,7,8\}} \neg help_i \\ \bigwedge_{i\in\{1,3,7,8\}} \Box(r_i^3 \longrightarrow \neg \bigcirc help_i) \\ \bigwedge_{i\in\{1,3,7,8\}} \Box((\neg r_i^3 \wedge help_i) \longrightarrow \bigcirc help_i) \\ \bigwedge \Box\Diamond \mathit{True} \end{cases}$$

$$\varphi_3^s = \begin{cases} r_{10}^1\ \wedge_{i=1,\ldots,9}\ \neg r_i^1 \\ \bigwedge\ \mathit{Transitions}\ \bigwedge\ \mathit{Mutual\ Exclusion} \\ \bigwedge \Box((\wedge_{i\in\{1,3,7,8\}} \neg \bigcirc help_i) \\ \qquad \Rightarrow (\wedge_{j\in\{1,\ldots,10\}} \bigcirc r_j^3 \leftrightarrow r_j^3)) \\ \bigwedge_{i\in\{1,3,7,8\}} \Box\Diamond(r_i^3 \vee \neg help_i) \end{cases}$$

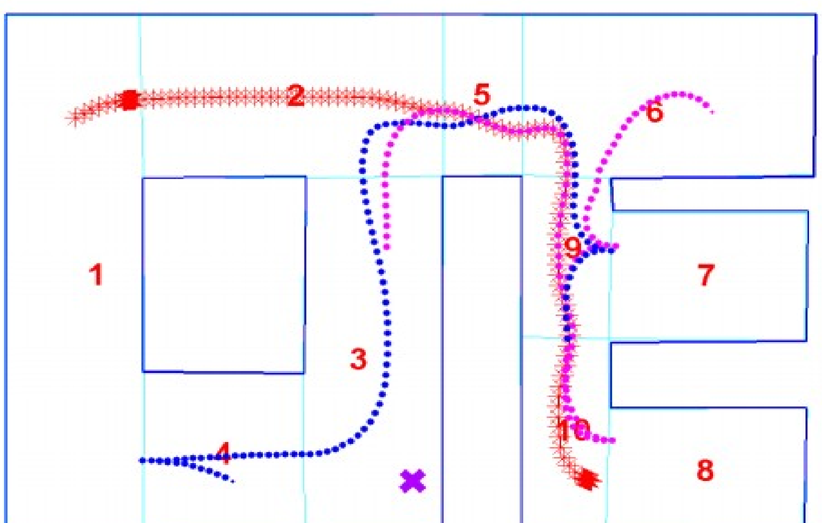
Multi Robot - Search and Rescue



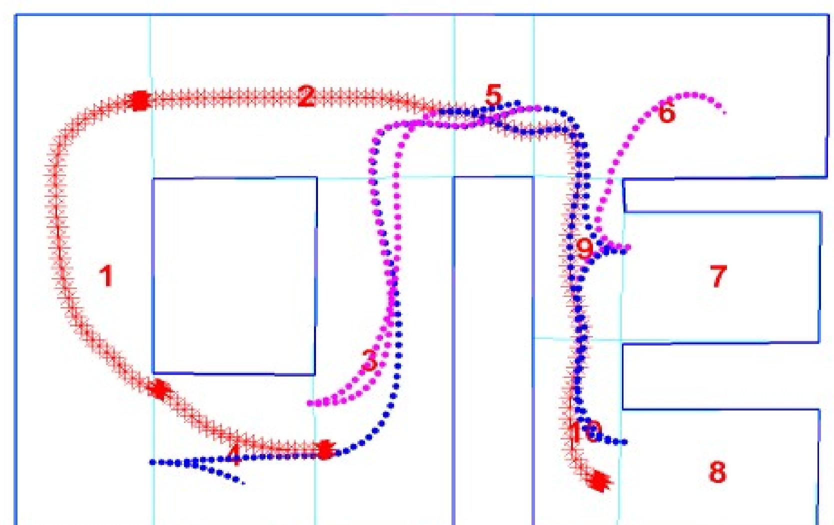
(a) Robot 1 found someone in 1



(b) Robot 3 arrived at 1



(c) Robot 2 found someone in 3



(d) Robot 3 arrived at 3

Discussion

- The method requires a fair amount of experience with temporal logic.
- The method is very sensitive to wrong formulation of the environment and there is no feedback or straight forward method to find the mistakes.
- The paper does not compare the algorithm with another method such as [8] [9] [10].
- The method may have some serious problems with real multi-robot systems (the 2 UAVs should fly in different altitudes) computationally and while developing.
- It can be improved computationally by finding some redundancy in the system and simplify the propositions.