Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 53-2 (2020) 15537–15543

Sampling-based Motion Planning with Temporal Logic Missions and Spatial Preferences [⋆]

Jesper Karlsson [*,1] Fernando S. Barbosa [*,1] Jana Tumova [*]

[*] *KTH Royal Institute of Technology, Stockholm, Sweden (e-mail: {jeskarl, fdsb, tumova}@kth.se)*

Abstract: While motion planning under temporal logic specifications has been addressed in several state-of-the-art works, spatial aspects have been so far largely neglected. In this work, we enrich the semantics of robot motion specifications by including preferences on spatial relations between its trajectory and various elements in its environment. The spatial preferences are given in a fragment of Signal Temporal Logic (STL) on top of complex missions in syntactically co-safe Linear Temporal Logic (scLTL). We propose a cost function with user-specified parameters, which determines the compromise between efficiency and spatial robustness of a trajectory. The proposed modification of the incremental sampling-based RRT* driven by this cost function guarantees that the motion plan (if found) simultaneously satisfies the mission and asymptotically minimize the cost. The paper includes several case studies showcasing the effects of the user-adjustable parameters on the resulting trajectories.

Keywords: Temporal Logic, Trajectory Planning, Path Planning, Formal Methods, Robotics

1. INTRODUCTION

Motion planning is at the heart of many autonomous systems, from drones and self-driving cars to automated vacuum cleaners. Traditionally, the motion planning problem is as follows: provided a dynamical system, an environment, an initial state and a goal state, find a sequence of control inputs so as to drive the system from the initial state to the goal state while fulfilling some constraint dictated by the environment, e.g., avoiding obstacles. As the robotic platforms became more and more advanced, so did the tasks provided to them, and motion planning problems have been associated with more complex goals. For instance, sampling-based algorithms have been enhanced to tackle deterministic mu-calculus specifications (Karaman and Frazzoli, 2009), or syntactically co-safe Linear Temporal Logic (scLTL) formulas (Bhatia et al., 2010).

In this work, we aim to enrich the class of tasks that can be handled by motion planning algorithms even further, moving from temporal logic specifications towards spatio-temporal ones. As a motivation scenario, consider a mobile robot that is requested to travel from A to B through an environment with WiFi routers. The robot should stay within a certain distance to the closest router at all times in order to guarantee connectivity to the WiFi network. In fact, it should stay as close to the routers as possible. At the same time, it should reach point B as soon as possible. As another example, consider a mobile robot tasked with navigating through a corridor with tight doorways. In order to stay as safe as possible, the robot should pass each door right though the middle and reach its final destination as soon as possible. Such types of tasks feature both *temporal* and *spatial* aspects. Our aim is to provide a means to express a variety of spatial preferences and a universal way to integrate them in a motion planning algorithm. However, temporal efficiency and obeying the spatial preferences do not necessarily go hand-in-hand; the fastest trajectory may be quite far from satisfying the spatial preferences and vice versa. Thus, we also allow a user to express the relative importance of the two so that a desired compromise is achieved.

When treating tasks including complex spatial preferences, approaches such as obstacle inflation do not work. For instance, when navigating through a corridor with doorways, obstacle inflation would either block the doors, keeping very narrow passages difficult to handle by motion planning algorithms, or keep too wide passage space, so that the robot would not be guaranteed to go in through the middle. In this particular case, the Vector Field Rapidly-exploring Random Trees (VFRRT) would provide desired trajectories (Ko et al., 2014). However, as opposed to our approach, VFRRT does not provide optimal paths with regard to different, potentially more complex, spatial and temporal specifications.

Linear Temporal Logic (LTL) and its syntactically co-safe fragment (scLTL) have been used as tools to specify time-sensitive missions to robots. In Ulusoy et al. (2013) a group of mobile robots is requested to satisfy a common LTL specification while optimizing the time interval between visits to certain locations. However, LTL and scLTL are traditionally interpreted over discrete sequences. Signal Temporal Logic (STL) was first proposed by Maler and

[⋆] This work was supported by the EU H2020 Research and Innovation Programme under GA No. 731869 (Co4Robots), and by the Swedish Research Council (VR).

[1] Both authors contributed equally to this work.

Nickovic (2004) as an extension to Metric Interval Temporal Logic (MITL) (Alur et al., 1996) to handle continuous signals. The concept of space and time robustness, alongside its combined version as well as quantitative semantics, is introduced in Donzé and Maler (2010). STL and its space robustness have been used with Model Predictive Controllers (MPC) for robust motion planning; Raman et al. (2014) encode STL specifications in Mixed Integer Linear Program (MILP) to formulate the MPC, while Lindemann and Dimarogonas (2017) treat a fragment of STL and propose the use of a Linear Program (LP) instead, creating a new robustness metric, called discrete average space robustness.

The asymptotically optimal counterpart of the RRT sampling-based algorithm, RRT* (Karaman and Frazzoli, 2011), has been used for trajectory generation going side-by-side with temporal logics task specifications and additional optimization criteria. Minimum-Violation RRT* has been introduced by Castro et al. (2013) and is capable of generating a control law for an autonomous vehicle that minimizes the level of unsafety with respect to a set of finite LTL formulas, while ensuring that a goal region is reached. Vasile et al. (2017b) propose a receding horizon solution for automated route generation for vehicles operating in road networks subject to road rules. Therein, ideas of least-violating planning in networks from Tumova et al. (2016) are employed alongside an RRT*-based algorithm, allowing a vehicle with limited sensing to achieve guaranteed performance.

The work by Vasile et al. (2017a) is, to our best knowledge, the one most closely related to ours. The authors present a sampling-based method for synthesizing control policies that maximize the spatial robustness of STL specifications. In contrast, our work aims to balance between spatial robustness and time robustness. Another related work, Haghighi et al. (2016) focus on control of robot swarms under spatio-temporal properties given in Spatio Temporal Logic (SpaTeL). SpaTeL formulas (Haghighi et al., 2015) are, however, interpreted over quad trees in contrast to trajectories, and are suitable for describing complex spatial patterns rather than spatial relations between a trajectory and elements in an environment. Lastly, Barbosa et al. (2019) presents a similar, but simpler, version of the approach introduced in this paper. Our work introduces a user-defined weight structure and demonstrates the integration with complex task assignments.

Our contribution can be summarized as follows: we design an extension to the RRT* that allows for specifying complex missions in scLTL and spatial preferences in a fragment of STL. Our approach measures the cost of a motion plan in terms of a newly developed combined space-time robustness that includes user-adjustable parameters. These can be used to tune the prioritization of temporal vs. spatial aspects of the specification. The motion planning algorithm preserves the important properties of RRT*, including asymptotic optimality; furthermore, it guarantees that, if a trajectory that satisfies the scLTL mission exists, it will be found in the asymptotic limit. The user can specify a spatial preference, such as "keep at least 0.5 m away from any wall", but when satisfying the preference is impossible, it is at least followed as closely as possible. Thanks to the user-adjustable parameters, the same framework can provide many different types of trajectories based on the user's preference. In one case, a detour could be preferable over a risky, but more efficient route (e.g., in an autonomous driving scenario). In another, a trajectory that violates the preference might be better, as long as the mission is completed quickly (e.g., in the case of a robotic vacuum cleaner). Our approach provides functionality similarly to what can be found using penalty methods in constrained optimization, without the risk of numerical ill-conditioning. We present examples that illustrate different compromises between temporal and spatial aspects based on different user-given priorities.

2. PRELIMINARIES

A dynamical system $\mathcal{S}$ is a tuple $\mathcal{S} = (f, X, U, W, x_{init})$, where $X \subset \mathbb{R}^m, U \subset \mathbb{R}^n$ are the state and control spaces of the system and $W \subset \mathbb{R}^2$ its bounded workspace. The dynamics are given by:

$$\dot{x} = f(x,u),\ x(0) = x_{init} \tag{1}$$

where x_{init} is the initial state at time $t = 0$, and $f : X \times U \to X$ is Lipschitz continuous. Let x_1 and x_2 refer to the coordinates of state x in the workspace W.

The workspace W is partitioned into k regions, denoted by $\Pi = \{\pi_1, \pi_2, \ldots, \pi_k\}$. Let $W_{obs} \subset W$ denote the set of obstacles. Without loss of generality, assume that $W_{obs} \subseteq \bigcup_{i\in I} \pi_i, i \in \{1, ..., k\}$. We consider a set of atomic propositions p, denoted by Σ. Each region is associated with a subset of Σ by the labeling function $L : \Pi \to 2^\Sigma$. $L(\pi_i)$ is the set of atomic propositions that are true in region π_i.

We denote by $\mathbf{x}$ the trajectory of the system. The trajectory is associated with a possibly infinite sequence, $\mathbf{p} = \pi_1\pi_2\ldots$ of regions in Π that are visited along the trajectory. We denote the *word* corresponding to a given sequence, $\mathbf{p}$, by $\omega(\mathbf{p}) = \omega_1, \omega_2, \ldots = L(\pi_1)L(\pi_2)\ldots$. For convenience we write, where the context is clear, the *word* for a path as ω.

A *syntactically co-safe Linear Temporal Logic formula* (scLTL) over a set of atomic propositions Σ is defined through the following grammar:

$$\varphi ::= p \mid \neg p \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \mathsf{F}\varphi \mid \varphi_1 \mathsf{U} \varphi_2 \tag{2}$$

where F (eventually) and U (until) are temporal operators. The formula φ is satisfied by a trajectory $\mathbf{x}$ of the dynamical system in (1) if the resulting word $\omega(\mathbf{p})$ satisfies φ, in other words, $\omega(\mathbf{p}) \models \varphi$.

Each trajectory that satisfies an scLTL formula φ contains a so-called satisfying prefix and any trajectory that begins with this satisfying prefix satisfies φ, too. The duration of the satisfying prefix is the *mission completion time*, which we denote T_φ. As a consequence, any scLTL formula can be translated into a language-equivalent deterministic finite automaton. A deterministic finite automaton is a tuple $\mathcal{A} = (Q, q_{init}, \Sigma, \delta, F)$ where Q is a set of states, q_{init} is the initial state, Σ is a finite alphabet, $\delta = Q \times \Sigma \to Q$ is a transition relation and $F \subseteq Q$ is a set of final states.

3. PROBLEM FORMULATION AND APPROACH

In this work, we aim to find a motion plan, i.e. a trajectory, for a robot that compromises between fulfilling a complex *mission* in scLTL as quickly as possible, while ensuring that given *spatial preferences* are violated minimally. Intuitively, a spatial preference gives the desired bounds on the (minimal or maximal) distance of the robot to other features in the workspace, in our case the regions. For instance, consider that a robot's mission is to follow an optimal trajectory through a set of regions in a particular order, and the spatial preference is to keep a certain minimal distance to all obstacles. While the optimal trajectory might not satisfy the spatial preference, the trajectories satisfying the spatial preference might be far from optimal. We aim to formalize and solve the problem of compromising between these two aspects based on user-defined priorities.

We consider a mobile robot modeled as a dynamical system in (1), whose mission is expressed in scLTL over the atomic propositions Σ that label the regions of the workspace as described in Sec. 2. Spatial preferences are expressed in a fragment of Signal Temporal Logic (STL) with alternative semantics: Let a predicate μ be defined as follows.

$$\mu = \begin{cases} \top \iff g(x(t)) \geq 0 \\ \bot \iff g(x(t)) < 0, \end{cases}$$

where $g(x(t))$ is a real-valued evaluation function, that provides an abstraction of the continuous signal $x(t)$, such that $g(x(t)) : \mathbb{R}^m \to \mathbb{R}$. A spatial preference over a set of predicates $\{\mu_1, \ldots, \mu_\ell\}$ is defined as a formula $\Psi \equiv \widehat{\mathsf{G}}_{[0,T_\varphi]}\psi$, where

$$\psi ::= \mu \mid \neg\mu \mid \psi_1 \wedge \psi_2 \mid \psi_1 \vee \psi_2 \tag{3}$$

The $\widehat{\mathsf{G}}$ operator expresses that ψ should hold at all times and in that sense it is similar to the G operator from STL. However, the quantitative semantics of $\widehat{\mathsf{G}}$ is in this work defined differently and tailored to the specific purpose of evaluating the degree of satisfaction of a spatial preference on a trajectory. Full details will be given in Sec. 4.

Example 1. Consider a mobile robot represented by a single integrator with a bounded input $||u(t)|| \leq u_{max}$ deployed in a 2D workspace $W = \{(x_1, x_2) \in \mathbb{R}^2 \mid 0 < x_1 < 6$ and $0 < x_2 < 6\}$ illustrated in Fig. 1. There are four regions $\Pi = \{\pi_1, \pi_2, \pi_3, \pi_4\}$ respectively labeled with atomic propositions $\Sigma = \{\text{EmptySpace}, \text{Obstacle1}, \text{Obstacle2}, \text{Goal}\}$. Let the function $dist(x)$ be the distance from the point (x_1, x_2) to the closest obstacle.

Let us consider a very simple mission in this case, namely to reach the goal region, $\varphi = \mathsf{F}\,\text{Goal}$. The spatial preference Ψ is to stay at least 0.5 units away from any obstacle during the mission,

$$\Psi = \widehat{\mathsf{G}}_{[0,T_\varphi]}\psi = \widehat{\mathsf{G}}_{[0,T_\varphi]}(dist(x(t)) - 0.5),$$

in which $g(x(t)) = dist(x(t)) - 0.5$ is the evaluation function.

From Fig. 1 we see that there are two possible classes of trajectories: i) going through the passage between Obstacle1 and Obstacle2, or ii) taking the "detour" above Obstacle2. If the spatial preference is of a very high importance, the robot should take the detour, where they can be fully met at the cost of the suboptimality of the mission completion time T_φ. On the other hand, if the spatial preference is only of mild importance in comparison to efficiency of achieving the mission, the robot should compromise by going through the narrow passage in a sufficient distance from the obstacles, slightly disobeying the spatial preference.

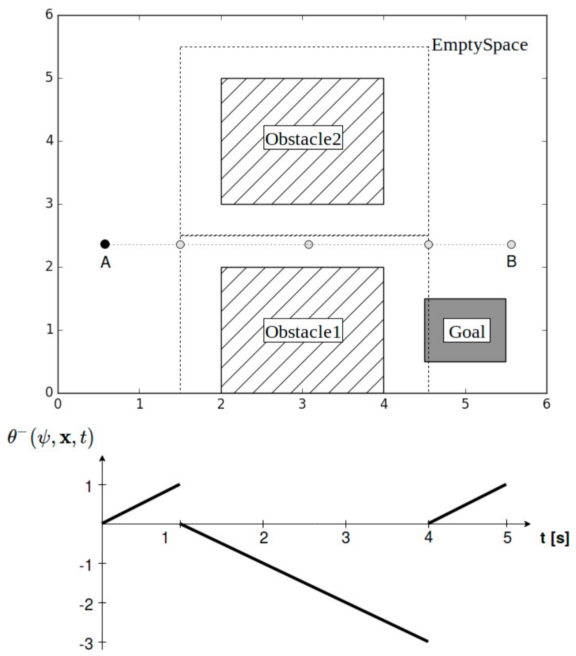


Fig. 1. a) Workspace defined in Ex. 1. It has two obstacles and one goal region; the rest is empty space, b) The 0.5 spatial preference is presented as a dashed line around the obstacles. A trivial trajectory from A to B is used to illustrate the left time robustness.

Given a dynamical system $\mathcal{S}$ operating in a labeled workspace W, a mission φ, a spatial preference Ψ, the problem we address is to:

(i) define a suitable cost function J that assigns a cost to each trajectory $\mathbf{x}$ and takes into consideration i) the mission completion time T_φ, ii) the degree of satisfaction of the spatial preference Ψ, and iii) the priority of satisfying the spatial preference Ψ; and
(ii) given the function J, find a state trajectory $\boldsymbol{x}$ of $\mathcal{S}$ that satisfies φ, and minimizes J.

4. SPATIO-TEMPORAL RRT*

In this section, we suggest a cost function J, incorporate this function into RRT*, and analyze the solution.

4.1 Cost function J

We propose to define the cost function J as follows:

$$J(\mathbf{x}) = J_\varphi(\mathbf{x}) + J_\Psi(\mathbf{x}), \tag{4}$$

where $J_\varphi(\mathbf{x})$ is the time duration of trajectory $\mathbf{x}$. Note that if $\mathbf{x}$ satisfies φ then $J_\varphi(\mathbf{x})$ is the mission completion time, $T_\varphi J_\Psi(\mathbf{x})$, which we define below, represents the (weighted) degree to which the spatial preference Ψ is satisfied.

The definition of $J_\Psi(\mathbf{x})$ is based on STL robustness (see, e.g., Donzé and Maler (2010) for full details). It is possible to measure how robust an STL formula is in two ways: in terms of spatial and time robustness. Loosely speaking, the former evaluates to which extent a trajectory (a signal in STL) deviates from the desired values, while the latter shows for how long this has been happening. Here, we present only the parts relevant for treating our STL fragment. The spatial robustness ρ of a preference ψ on a trajectory $\mathbf{x}$ at time t is defined as

$$\begin{aligned}\rho(\mu, \mathbf{x}, t) &= g(\mathbf{x}(t))\\ \rho(\neg\psi, \mathbf{x}, t) &= -\rho(\psi, \mathbf{x}, t)\\ \rho(\psi_1 \wedge \psi_2, \mathbf{x}, t) &= \min(\rho(\psi_1, \mathbf{x}, t), \rho(\psi_2, \mathbf{x}, t))\\ \rho(\psi_1 \vee \psi_2, \mathbf{x}, t) &= \max(\rho(\psi_1, \mathbf{x}, t), \rho(\psi_2, \mathbf{x}, t))\end{aligned}$$

and the left time robustness θ^- of a formula ψ on a trajectory $\mathbf{x}$ at time t as

$$\theta^-(\psi, \mathbf{x}, t) = \mathcal{X}(\psi, \mathbf{x}, t) \cdot max\{d \geq 0 \mid \forall t' \in [t-d, t]\\ \mathcal{X}(\psi, \mathbf{x}, t') = \mathcal{X}(\psi, \mathbf{x}, t)\},$$

where $\mathcal{X}(\psi, \mathbf{x}, t) = sign(\rho(\psi, \mathbf{x}, t))$. For a predicate μ, the spatial robustness is negative if the predicate is violated on trajectory $\mathbf{x}$ at time t. Time robustness at time t is then also negative and it indicates the time duration leading up to t for which μ has been violated. The lower the value of the spatial robustness, the more severe the violation of μ. The lower the value of the time robustness, the longer the violation of μ. Similarly, the spatial robustness is positive if the predicate is satisfied on trajectory $\mathbf{x}$ at time t. The time robustness is then positive and indicates the duration for which μ has been satisfied.

Example 2. Fig. 1 (top) gives an example of a trajectory $\mathbf{x}$ leading from A to B that is subject to a spatial preference $\psi = (dist(x(t)) - 0.5)$. Fig. 1 (bottom) shows the evolution of the left time robustness $\theta^-(\psi, \mathbf{x}, t)$ for the trajectory over time. The trajectory keeps at least 0.5 unit away from any obstacle for all $t \in [0, 1)$, as desired by the spatial preference, thus its left time robustness is growing positive. As soon as the trajectory gets closer to the obstacle than what is specified in the spatial preference, the left time robustness becomes zero and starts growing negative, as seen for $t \in [1, 4)$. The left time robustness value becomes zero and grows positively again as the trajectory gets farther than 0.5 from any obstacle, as for $t \in [4, 5]$.

We define the cost $J_\Psi(\mathbf{x})$, from (4), which can be also viewed as a quantitative semantics of the alternative always operator $\widehat{\mathsf{G}}$ as:

$$J_\Psi(\mathbf{x}) = -\int_0^{T_\varphi} \theta^\star(\psi, \mathbf{x}, t) \cdot w(\rho(\psi, \mathbf{x}, t))dt, \qquad (5)$$

where $\theta^\star$ is a modified version of left time robustness as defined below in (6) and w is a positive function defined by (7) that weighs the spatial robustness.

Modified left time robustness $\theta^\star(\psi, \mathbf{x}, t)$ is defined as

$$\theta^\star(\psi, \mathbf{x}, t) = \min(\theta^-(\psi, \mathbf{x}, t), 0) \qquad (6)$$

This definition guarantees that the cost function J_ψ is monotone and hence J is also monotone. This property will become important later on when we incorporate J into RRT* as we will discuss in Sec. 4.3.

Lemma 3. Given $\theta^\star(\psi, \mathbf{x}, t)$ in (6) and a positive weight function w, $J_\Psi(\mathbf{x})$ in (5) is monotonically increasing.

Proof. The proof follows directly from the fact that $\theta^\star(\psi, \mathbf{x}, t) \leq 0$ and $w(\rho(\psi, \mathbf{x}, t)) \geq 0$, and therefore their product is always upper-bounded by zero.

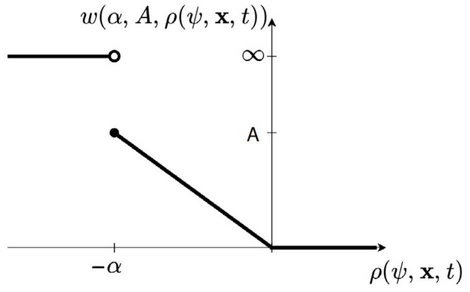


Fig. 2. The proposed weight function

Weight function $w(\rho(\psi, \mathbf{x}, t))$ is a positive function that can feature user-defined parameters. It allows the user to express the compromise between a time-efficient trajectory and a spatially-preferred one. We propose a function that takes into consideration two parameters, i) a lowest spatial robustness bound α, which signifies the lowest acceptable spatial robustness. For instance, in our running example, α could be set to d_r, the radius of the robot; ii) a parameter A indicating the pace at which the trajectory is penalized as it approaches the desired bounds. It can be viewed as the relative weighting between violating trajectories and in a way, it provides a measure of flexibility of the spatial preference. The proposed function $w(\alpha, A, \rho(\psi, \mathbf{x}, t))$ is depicted in Fig. 2 and formalized as follows:

$$w(\alpha, A, \rho(\psi, \mathbf{x}, t)) = \begin{cases}\infty, \iff \rho(\psi, \mathbf{x}, t) < -\alpha\\ 0, \iff \rho(\psi, \mathbf{x}, t) > 0\\ \dfrac{-A}{\alpha}\rho(\psi, \mathbf{x}, t) \iff otherwise.\end{cases} \qquad (7)$$

The proposed weight function splits points on a trajectory into three types: i) The points on the trajectory that violate the minimum requirements of the user (as specified by α), ii) the points that satisfy the spatial preference, and iii) the points that violate the spatial preference, but not so much as to violate the minimum threshold as defined by α. We put an infinite weight on the type i) to signify that the violation of the spatial preferences is unacceptable in those points; in fact, $\alpha = 0$ can be used to set hard constraints. We put zero weight on type ii) to signify that the spatial preferences are fully satisfied there, and therefore should not be penalized.

Altogether, the weight function w contributes to the cost function J_Ψ in a way that puts a very high weight on trajectories that violate the minimum requirements, differentiates trajectories that violate the specification, based on how much and how long they do so, and adds no additional weight on segments of a trajectory that do not violate the safety specifications.

Example 4. If a low α is chosen in the scenario from Ex. 1, there is a large region around the obstacles that the system should not enter, even if the mission will be delayed. α can be viewed as a minimum tolerance bound on the level of violation of the spatial aspect of the cost J. Parameter A specifies how quickly the cost increases as we move closer to obstacles. If A is high, the trajectory is likely to be longer.

4.2 Spatio-temporal RRT*

Given the definition of the cost function, we address the motion planning problem via an extension to *RRT** that was originally designed for asymptotically optimal A-to-B motion planning (Karaman and Frazzoli, 2011). Two modifications to *RRT** are needed: i) in order to find a trajectory satisfying φ, the vertices of the tree are elements of the space defined by the cartesian product between the system state space and the finite automaton capturing the mission φ, and ii) in order to return the trajectories that minimizes the violation of the spatial preference, the cost function triggering the updates to the edges of the tree is now J as defined in (4).

The main algorithm is outlined in Alg. 1. Each iteration starts by drawing a sample x_{rand} from X using the function $sample(X, q)$, where q is a state of the mission specification automaton. The algorithm continues by attempting to connect the drawn sample using the *near*, *steer*, and *update* procedures. The main idea of RRT* is to rewire the states of the grown graph based on a cost function.

The basic procedures used in Alg. 1-2 are as follows:

- *sample:* A sample (x_{rand}, q_{rand}) is drawn by the *sample* procedure from a uniform distribution over the state space X and the automaton states.
- *near:* The $near(x, q)$ returns the set of nodes within the ball $\mathcal{B}$ with radius $r = \gamma(\frac{\log n}{n})^{1/d}$ (γ a constant, $n = |V|$ and d is the dimension of the state-space) from x that are on the automaton state q.
- *steer:* The $steer(x', x)$ procedure computes the trajectory from x' to x and returns a state x_{new} that is the farthest state within radius r from x' in the direction of the trajectory from x' to x.
- *obstacle_free:* The *obstacle_free* procedure returns whether a candidate connection between two states in X enters W_{obs} or not.
- *update:* The *update* procedure is outlined in Alg. 2. First, the mission specification automaton state is set (line 2). Second, the cost of the new candidate connection is computed and compared with the best current cost. Here, $J(x', x_{new})$ denotes the cost of the trajectory from x' to x_{new} obtained by *steer*, which is computed according to (4). $cost(x')$ and $cost(x_{new})$ denote the best cost of a trajectory from x_{init} to x' and x_{new} computed so far. If a connection from the new candidate parent offers an improvement, reconnection is performed (lines 3-12).

Algorithm 1 Spatio-temporal *RRT**

Input: $\mathcal{S}$ - system model, W - workspace, φ - mission given in the form of an automaton $\mathcal{A} = (Q, q_{init}, \Sigma, \delta, F)$, ψ - spatial preference, N - number of iterations
Output: $\mathbf{x}$ - trajectory

```
 1: $V \leftarrow \{(x_{init}, q_{init})\}$; $E \leftarrow \emptyset$; $i \leftarrow 0$
 2: while $i < N$ do
 3:   $G \leftarrow (V, E)$;
 4:   $(x_{rand}, q_{rand}) \leftarrow sample(X, Q)$; $i \leftarrow i + 1$;
 5:   for $(x', q') \in near(x_{rand}, q_{rand})$ do
 6:     if $x_{new} \leftarrow steer(x', x_{rand})$ then
 7:       $update((x', q'), x_{new})$
 8:     end if
 9:   end for
10:   if $(x_{new}, q) \in V$ then
11:     for $(x', q') \in near(x_{new}, q)$ do
12:       if $x' = steer(x_{new}, x')$ then
13:         $update((x_{new}, q), x')$
14:       end if
15:     end for
16:   end if
17: end while
18: if $\mathbf{x} \leftarrow$ optimal trajectory corresponding to the path
    connecting $(x_{init}, q_{init})$ to a state $(x, q_f)$, where $x \in X$
    and $q_f \in F$ then
19:   return $x$
20: else
21:   return Fail
22: end if
```

Algorithm 2 $update((x', q'), x_{new})$

```
 1: if $obstacle\_free(x', x_{new})$ then
 2:   $q \leftarrow \delta(q', L(x_{new}))$
 3:   $C \leftarrow J(x', x_{new})$
 4:   if $C + cost(x') < cost(x_{new})$ then
 5:     if $(x_{new}, q) \notin V$ then
 6:       $V \leftarrow \{V \cup (x_{new}, q)\}$
 7:     else
 8:       $E \leftarrow E \setminus \{(parent(x_{new}, q), (x_{new}, q))\}$
 9:     end if
10:     $E \leftarrow E \cup \{((x', q'), (x_{new}, q))\}$
11:     $parent(x_{new}, q) \leftarrow (x', q')$
12:   end if
13: end if
```

4.3 Analysis

In this section, the optimality and complexity of the modified RRT* are considered and analyzed briefly. We discuss the mission satisfaction and comment on the asymptotic optimality of our algorithm.

Theorem 5. (Mission satisfaction): Any trajectory $\mathbf{x}$ found by the modified RRT* guarantees the satisfaction of the mission φ.

Proof. The vertices of the tree G are elements of the space defined by the cartesian product $(X \times Q)$. Suppose that the search for optimal path in G from (x_{init}, q_{init}) to (x, q_f), where $x \in X$ and $q_f \in F$ is successful. This path corresponds to a trajectory of the robot in the workspace, which is associated with a sequence of regions in Π, visited along this trajectory. This sequence of regions corresponds to a word, the sequence of sets of atomic propositions that hold true in the respective regions and this word is by construction accepted by $\mathcal{A}$. Hence, the trajectory satisfies the scLTL formula φ.

Given the fact that the solution is generated by RRT* with an alternative optimality criterion, the returned trajectory is indeed optimal with respect to this criterion.

The complexity of our algorithm depends heavily on the evaluation of the cost function (which can be arbitrarily expensive). Assuming the cost function proposed in this paper, the complexity is equal to that of the base RRT*.

Finally, let us remark that similarly as RRT*, our algorithm is an anytime incremental algorithm.

5. CASE STUDIES

This section demonstrates the viability of the proposed approach in three illustrative case studies, each of which was designed to show a particular aspect of the algorithm. We have implemented the solution in Python 2.7. All examples were run on an Intel Core i7 computer and 16GB RAM under Ubuntu 16.04.

5.1 Running Example

In the first case study we investigate the running example used in Sections 3 and 4. The objective here is to reach the goal region on the lower-right corner of the environment, starting from the lower-left corner, while maintaining 0.5 distance to the two obstacles present.

Fig. 3a shows that a larger value for A results in a safer path, while a zero-valued A results in a trajectory that tangents the safety limit set by α.

5.2 Office Environment

The second case study deals with a large and obstacle-filled office depicted in Fig. 3b. It has tables, walls, a corridor and four doors; three regular sized and one smaller, barely the size of the robot. The mission and spatial preference are:

$$\varphi = \mathsf{F}\, G_1 \wedge \mathsf{F}\, G_2$$
$$\Psi = \widehat{\mathsf{G}}_{[0,T_\varphi]}(dist(x(t)) - 0.5),$$

where G_1, G_2 corresponds to the first and second goal regions, respectively (dark gray regions in Fig. 3b).

The resulting trajectories can be seen for different values of A in Fig. 3b. Besides different parameter values, the case also provides trajectories using vanilla RRT* and RRT* with obstacles inflated by the robot size, 0.3.

5.3 Motion Planning with Wireless Network Constraints

The third case study considers wireless network constraints in an extension of the previous case, where the robot also has to remain within Wifi range throughout the mission.

The mission φ is the same as in the previous case and the spatial preference is:

$$\Psi = \widehat{\mathsf{G}}_{[0,T_\varphi]}\Big((dist(x(t)) - 0.5) \wedge \big((4 - dist(x(t), r_1)) \vee (3.5 - dist(x(t), r_2)) \vee (2 - dist(x(t), r_3))\big)\Big),$$

with $dist(x(t), r_i)$ the distance to the ith router at time t.

The resulting trajectories are in Fig. 3c. As can be seen, the longer path have been effectively filtered out from consideration due to the router range specification.

5.4 Discussion

We presented three examples of increasing mission complexity to demonstrate the effectiveness of our approach. We use this section to discuss the results depicted in Fig. 3. The running example, Fig. 3a, was designed to showcase the influence of the user-defined value of A to be used in the cost function. One can note that setting $A = 0$ means that the weight function (7) will set either an infinite or no weight to the trajectory, not differentiating the ones that satisfy the spatial preference from those that slightly violate it, i.e. $\rho(\psi, \mathbf{x}, t) > -\alpha$. Fig. 3a can be interpreted as follows: vanilla RRT* finds the shortest path that satisfies the mission; our approach with $A = 0$ finds the shortest path with $\alpha = 0.3$ clearance, while $A = 1$ finds a trajectory that goes right in the center of the corridor between the obstacles; lastly, $A = 10$ sets such a high weight for violating the spatial preference that the approach returns a path that takes a detour beyond the upper obstacle, never violating Ψ.

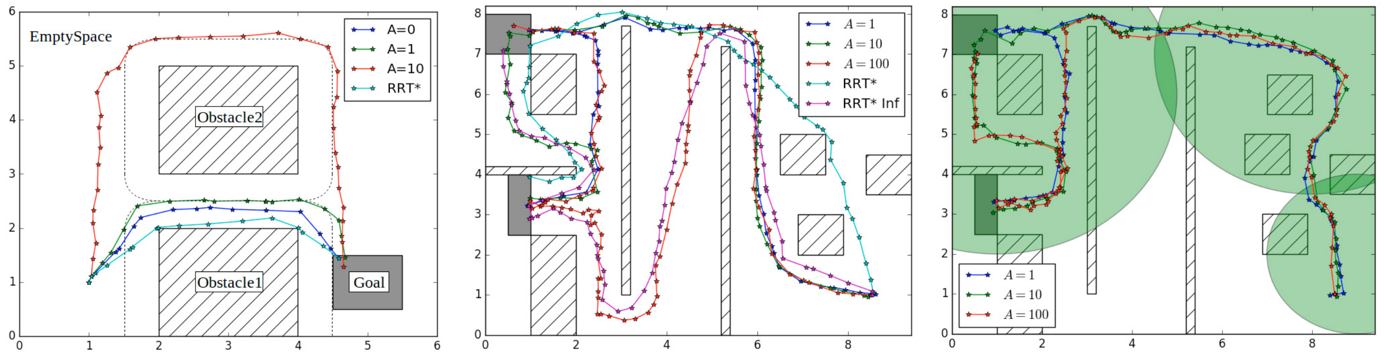
The second case study, Fig. 3b, shows how a higher value of A enforces a longer, but safer trajectory, when compared to lower A values and vanilla RRT*. Note that, in several parts of the trajectories, they pass right in the middle of the clearances in-between the obstacles and doors, such that the distance to the closest obstacle is maximized and, therefore, evaluating the cost function to a lower value. This allows our approach to find trajectories that would not be traversable if incorporation obstacle inflation to RRT*.

The third and last case, Fig. 3c, enriches the previous environment with connectivity limits, requiring not only that the mission is accomplished, but that the connectivity is not lost at any point. This case highlights the flexibility of our approach, and how the trajectories change according to the spatial preference. Note that, on the right-half part of the environment, in order to ensure connectivity, the trajectories moved from being on the safer part to being among several smaller obstacles. On the other half of the environment, the trajectories in respect to $A = 1$ and $A = 10$ are very similar to the ones presented in the previous case, since they were already inside the region to be covered by the WiFi; however, the one with $A = 100$ does not take the longest detour anymore, since that part is not covered by the WiFi, and goes right in the middle of the small passageway at the top. Constraining RRT* with inflated obstacles in a similar fashion makes it impossible to complete the task. Using our approach, the robot will be able to complete the task even though the path through the small door is suboptimal with regard to safety alone.

6. CONCLUSION

We have proposed a systematic way to enhance traditional motion planning with complex missions in scLTL and spatial preferences. A new cost reflecting spatio-temporal robustness quantification has been introduced. Unlike the standard STL semantics, it accounts for both spatial and temporal robustness at once. A modification of RRT* considering this cost maintains the important properties, including the asymptotic optimality (under certain assumptions), and anytime and incremental nature of the algorithm. We have provided an illustrative case study with scenarios of varying degree of complexity that illustrates the validity and flexibility of the proposed approach.

Future work includes expanding the procedure towards multi-agent systems,both for realistic traffic scenarios as



(a) Running example with four trajectories; one generated by the vanilla RRT⋆ and three with a choice of $\alpha = 0.3$ and varying choice of A.

(b) Office environment, with five trajectories; one generated by the vanilla RRT⋆, one by RRT⋆ with inflated obstacles and three with $\alpha = 0.3$ and varying choice of A.

(c) Scenario with wireless network constraints, with three trajectories with $\alpha = 0.3$ and varying choice of A.

Fig. 3. Case study results using different values of parameter A.

well as mobile robots. The main focus of this extension will be semantics that support measurement of robustness for time shifts in the future, which is a necessary component when considering multiple agents and/or stochastic behaviour in an environment. Future work will also focus on how to express the mutual priorities of the time and spatial aspects in one mission statement while maintaining the expressiveness of the current approach.

We also aim to expand the case studies in an experimental testbed, for both the single-agent case presented here and the multi-agent under development.

REFERENCES

Alur, R., Feder, T., and Henzinger, T.A. (1996). The benefits of relaxing punctuality. *Journal of the ACM (JACM)*, 43(1), 116–146.

Barbosa, F.S., Duberg, D., Jensfelt, P., and Tumova, J. (2019). Guiding autonomous exploration with signal temporal logic. *IEEE Robotics and Automation Letters*, 4(4), 3332–3339.

Bhatia, A., Kavraki, L.E., and Vardi, M.Y. (2010). Sampling-based motion planning with temporal goals. In *2010 IEEE International Conference on Robotics and Automation*, 2689–2696.

Castro, L.I.R., Chaudhari, P., Tumova, J., Karaman, S., Frazzoli, E., and Rus, D. (2013). Incremental sampling-based algorithm for minimum-violation motion planning. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, 3217–3224. IEEE.

Donzé, A. and Maler, O. (2010). *Robust Satisfaction of Temporal Logic over Real-Valued Signals*, 92–106. Springer Berlin Heidelberg, Berlin, Heidelberg.

Haghighi, I., Jones, A., Kong, Z., and Bartocci, E. (2015). SpaTeL: a novel spatial-temporal logic and its applications to networked systems. *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, 189–198.

Haghighi, I., Sadraddini, S., and Belta, C. (2016). Robotic swarm control from spatio-temporal specifications. *2016 IEEE 55th Conference on Decision and Control, CDC 2016*, 5708–5713.

Karaman, S. and Frazzoli, E. (2009). Sampling-based motion planning with deterministic μ-calculus specifications. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, 2222–2229.

Karaman, S. and Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7), 846–894.

Ko, I., Kim, B., and Park, F.C. (2014). Randomized path planning on vector fields. *The International Journal of Robotics Research*, 33(13), 1664–1682.

Lindemann, L. and Dimarogonas, D.V. (2017). Robust motion planning employing signal temporal logic. *2017 American Control Conference (ACC)*, (2), 2950–2955.

Maler, O. and Nickovic, D. (2004). Monitoring Temporal Properties of Continuous Signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, 152–166. Springer.

Raman, V., Donzé, A., Maasoumy, M., Murray, R.M., Sangiovanni-Vincentelli, A., and Seshia, S.A. (2014). Model predictive control with signal temporal logic specifications. In *53rd IEEE Conference on Decision and Control*, 81–87.

Tumova, J., Karaman, S., Belta, C., and Rus, D. (2016). Least-Violating Planning in Road Networks from Temporal Logic Specifications. In *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS)*, 1–9.

Ulusoy, A., Smith, S.L., Ding, X.C., Belta, C., and Rus, D. (2013). Optimality and Robustness in Multi-Robot Path Planning with Temporal Logic Constraints. *The International Journal of Robotics Research*, 32(8), 889–911. doi:10.1177/0278364913487931.

Vasile, C.I., Raman, V., and Karaman, S. (2017a). Sampling-based synthesis of maximally-satisfying controllers for temporal logic specifications. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3840–3847.

Vasile, C.I., Tumova, J., Karaman, S., Belta, C., and Rus, D. (2017b). Minimum-violation scLTL motion planning for mobility-on-demand. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 1481–1488.