# Monitoring Spatio-Temporal Properties (Invited Tutorial)

Laura Nenzi[1,2(✉)], Ezio Bartocci[1], Luca Bortolussi[2], Michele Loreti[3], and Ennio Visconti[1]

[1] TU Wien, Vienna, Austria
laura.nenzi@tuwien.ac.at
[2] DMG, University of Trieste, Trieste, Italy
[3] University of Camerino, Camerino, Italy

**Abstract.** From the formation of traffic jams to the development of troublesome, whirlpool-like spirals in the heart's electrical activity, spatio-temporal patterns are key in understanding how complex behaviors can emerge in a network of locally interacting dynamical systems. One of the most important and intriguing questions is how to specify spatio-temporal behaviors in a formal and human-understandable specification language and how to monitor their onset efficiently. In this tutorial, we present the spatio-temporal logic STREL and its expressivity to specify and monitor spatio-temporal behaviors over complex dynamical and spatially distributed systems. We demonstrate our formalism's applicability to different scenarios considering static or dynamic spatial configurations and systems with deterministic or stochastic dynamics.

## 1 Introduction

Spatio-temporal patterns are central to the understanding of how complex behaviors can emerge in a network of locally interacting dynamical systems.

A prominent example is the electrical currents that regularly traverse the cardiac tissue and are responsible for the heart's muscle fibers to contract. These electrical impulses travel as a planar wave smoothly and unobstructed in a healthy heart. In certain circumstances, myocytes can partially or entirely lose excitability [46,47,60], that is their ability to propagate and reinforce an electric stimulus. Lack of excitability can cause the formation of whirlpool-like spirals (see Fig. 1) of electrical activity that are a precursor to a variety of cardiac arrhythmias, including atrial fibrillation (AF), an abnormal rhythm originating in the upper chambers of the heart [11]. This type of behavior is called *emergent* because it emerges as the result of the local interactions of several (potentially heterogeneous) entities. Thus, these behaviors cannot be studied analyzing the individual entities, but they can be reproduced only by simulating/observing

This research has been partially supported by the Austrian FWF projects ZK-35 and W1255-N23, by the Italian PRIN project "SEDUCE" n. 2017TWRCNB and by the Italian PRIN project "IT-MaTTerS" n, 2017FTXR7S.

© Springer Nature Switzerland AG 2020

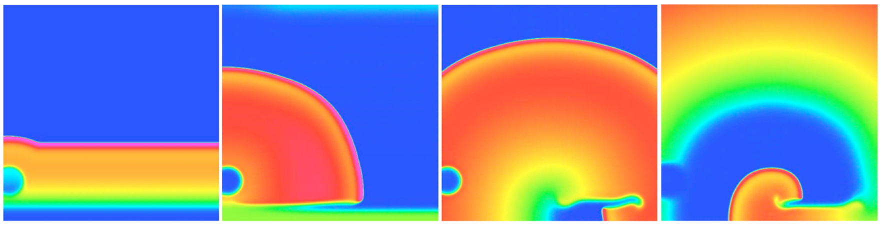


**Fig. 1.** Tachycardic spiral wave induced by the left hand side disc of unexcitable myocytes.

their collective behavior. A typical mechanism responsible for the onset of emergent behaviors is the change in space and time of the concentration of one or more chemical substances. In the cardiac tissue example, the electrical polarization results from a complex interplay between ion pumps and ionic channels embedded in the cell membrane that are responsible for the inward/outward flows of different ionic species (e.g., sodium, potassium, calcium).

Turing's reaction–diffusion systems [71] are suitable mathematical tools commonly used to model and simulate such physical phenomena. Mathematically, they consist of semi-linear parabolic partial differential equations (PDEs) with two terms that sum to each other: (a) a *reaction term* that describes, via nonlinear ordinary differential equations (ODEs), how the chemical species are transformed into each other in time; (b) a *diffusion term* modeling the flow/the spread of species in space.

Turing's reaction–diffusion systems can be employed to model a wide range of dynamical processes (not necessary chemical) ranging from wave-like phenomena in excitable media to the formation of other spatio-temporal self-organized patterns that are at the very origin of morphogenesis (e.g., the stripes of a zebra, the spots on a leopard and the filaments in Anabaena [45]) and developmental biology [14]. They are also at the core of self-assembly technologies, tissue engineering [72], and amorphous computing [2,21]. Such patterns are also known as "Turing's patterns".

Spatio-temporal patterns are not only pervasively present in nature, but they can arise also in human engineered artefacts such as Collective Adaptive Systems [54] (CAS) and Cyber-Physical Systems [64] (CPS).

CAS consist of a large number of heterogeneous components featuring complex interactions among themselves, with humans and other systems. Each component in the system may exhibit autonomic behavior and operates in open and non-deterministic environments where entities may enter or leave the CAS at any time. Decision-making in CAS is very challenging because the local interactions among components may introduce novel and sometimes undesired emergent behaviors. A typical example of CAS is a bike sharing system (BSS) [42], a service that allows people to borrow bicycles on a short term basis either for a price or for free. Users can pick-up and return bikes in special stations (spatially distributed in a city) equipped with bike racks. In such case the undesired behavior is to have the pick-up stations completely full or empty.

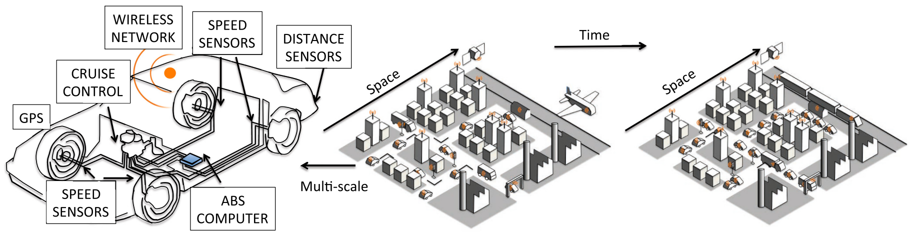


**Fig. 2.** An example of Cyber-Physical Systems in the automotive scenario.

Cyber Physical Systems (CPS) share similar characteristics with CAS. The term CPS was first coined in 2006 by Helen Gill [53], the Program Director for the Embedded & Hybrid Systems at US National Science Foundation. She defined CPS as "engineering, physical and biological systems whose operations are integrated, monitored, and/or controlled by a computational core. Components are networked at every scale. Computing is deeply embedded into every physical component, possibly even into materials. The computational core is an embedded system, usually demands real-time response, and is most often distributed. The behavior of a cyber-physical system is a fully-integrated hybridisation of computational (logical) and physical action."

Examples of CPS include contact tracing devices, self-driving cars, robotics teams, mobile ad-hoc sensor networks and smart cities. CPS behavior is characterised by the evolution in time of physical quantities measured by sensors and discrete states of computational, connected and spatially arranged entities.

Figure 2 shows an instance of CPS in the automotive scenario where the extensive integration of sensor networks and computational cores into automotive systems has led to the development of various driving assistance features that facilitate the driver during monotonous maneuvers and protect the passengers from hazardous situations. Furthermore, the advent of 5G mobile-network technology will support Vehicle To Vehicle (V2V) and Vehicle To Infrastructure (V2I) communication technologies very soon. These technologies will enable the exchange of information between vehicles and roadside units about position and speed of vehicles, driving conditions on a particular road, accidents, or traffic jams. This will allow to distribute the traffic load among several roads during rush hour and to prevent accidents.

The safety-critical nature of these systems [64] requires the engineers to check their correct execution with respect to rigorously defined spatial and temporal requirements. However, the complexity of these large scale systems often limits the possibility to analyze them using exhaustive verification techniques. A common approach consists instead in simulating their design with different initial conditions, parameters and inputs. The generated traces (e.g., mixed-analog signals) are then monitored [10,15] with respect to a formal specification of the behavioral property to satisfy. The verdict of such specification-based monitoring approach can return either a Boolean value specifying whether the traces satisfy or not the specification or a real value indicating how much the specification is satisfied or violated according to a chosen notion of distance [12,39,49,50,65].

Specification-based monitoring is nowadays the basic functionality upon which are built several other computer-aided verification and synthesis techniques such as statistical model checking [24,33,75], falsification analysis [1,67, 68,73], failure explanation/debugging [16–18], parameter synthesis for system identification or design [9,23,35,37]. The majority of specification languages and tools available for CPS supports only the monitoring of temporal properties. Examples are Metric Temporal Logic (MTL) [52], Signal Temporal Logic (STL) [57,58] and Timed Regular Expressions (TRE) [6].

However, one of the most important and intriguing questions in monitoring these systems is how to formally specify in a human-understandable language also spatio-temporal emergent behaviour, and how to efficiently monitor its onset.

In this tutorial, we present the Spatio-Temporal Reach and Escape Logic (STREL), a spatio-temporal specification language originally introduced in [13]. STREL enables the specification of spatio-temporal requirements and their monitoring over the execution of mobile and spatially distributed components. In this framework, space is represented as a weighted graph, describing the topological configurations in which the single components (nodes of the graph) are arranged. Both nodes and edges have attributes modelling physical and logical quantities that can change in time. STREL extends the Signal Temporal Logic [57] with two spatial operators *reach* and *escape* from which is possible to derive other spatial modalities such as *everywhere*, *somewhere* and *surround*. These operators enable a monitoring procedure where the satisfaction of the property at each location depends only on the satisfaction of its neighbours. Furthermore, we show how STREL can be interpreted according different semantics (Boolean, real-valued) semantics based on constraint semirings, an algebraic structure suitable for constraint satisfaction and optimisation.

The rest of this paper is organized as follows. Section 2 introduces the model we consider to represent the spatio-temporal signals, while Sect. 3 provides the syntax and the semantics of STREL. In Sects. 4, 5 and 6, we demonstrate the applicability of our formalism to different scenarios considering static (Sect. 4 and Sect. 5) or dynamic (Sect. 6) spatial configurations and systems with deterministic or stochastic dynamics. Section 7 discusses the related work, while Sect. 8 draws our conclusions and discusses future works.

## 2 Space Model, Signals and Traces

In this section we introduce some preliminary notions, including the model of space we consider and the related concept of spatio-temporal signal, illustrating them through a working example.

**Sensor Network.** As a running example, let us consider a network [4] of $n$ devices, equipped with a sensor to monitor for example the temperature. Two nodes can communicate with each other if their Euclidean distance is within their communication range.

## 2.1 Discrete Space as a Graph

The design of a spatial logic is strictly related to the description of *space* in which the dynamics takes place. In this tutorial, we consider a discrete space, described as a weighted direct graph (undirect graphs can be treated as direct graph with a symmetric relation). The reason why we focus our attention on discrete space is that many applications, like bike sharing systems, smart grid and sensor networks are naturally framed in a discrete spatial structure. Moreover, in many circumstances continuous space is abstracted as a grid or as a mesh. This is the case, for instance, of many numerical methods that simulate the spatio-temporal dynamics using partial differential equations (PDEs). Hence, this class of models can be dealt with by checking properties on such a discretization.

**Definition 1 (Space Model).** *We define the spatial model $\mathcal{S}$ as a pair $\langle L, \mathbf{W}\rangle$ where:*

- *$L$ is a set of nodes that we call* locations*;*
- *$\mathbf{W} \subseteq L \times \mathbb{R} \times L$ is a* proximity function *associating a label $w \in \mathbb{R}$ to distinct pair $\ell_1, \ell_2 \in L$. If $(\ell_1, w, \ell_2) \in \mathbf{W}$ it means that there is an edge from $\ell_1$ to $\ell_2$ with weight $w \in \mathbb{R}$, i.e. $\ell_1 \overset{w}{\mapsto} \ell_2$.*

Considering our running example, let us define a sensor space model. $L$ is given by the set of devices, i.e. each device represents a location. As proximity function $\mathbf{W}_C$, we can consider the connectivity graph, i.e. a location $\ell_i$ is next to a location $\ell_j$ if and only if they are within their communication range. Another possibility as proximity function is the dual graph of the Voronoi diagram [7] which partitions the plane into set of $n$ regions, one per location, assigning each point of the plane to the region corresponding to the closest location. These two examples of graphs can be seen in Fig. 3.

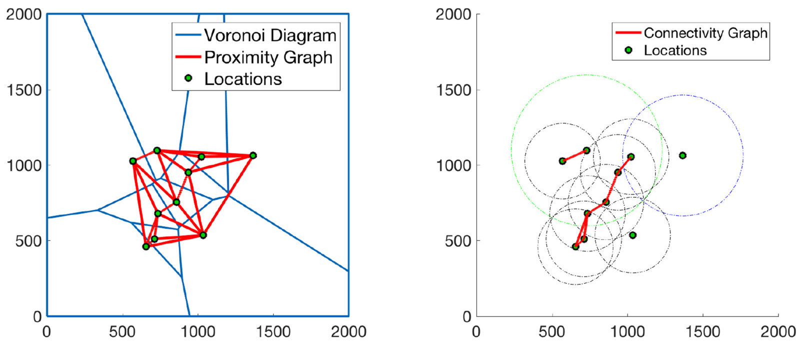


**Fig. 3.** Proximity graph (left) and connectivity graph (right)

Given a spatial model we can define *routes*.

**Definition 2 (Route).** *Let* $\mathcal{S} = \langle L, \mathbf{W} \rangle$, *a* route $\tau$ *is an infinite sequence* $\ell_0 \ell_1 \cdots \ell_k \cdots$ *in* $\mathcal{S}$ *such that for any* $i \geq 0$, $\ell_i \overset{w}{\mapsto} \ell_{i+1}$.

Let $\tau = \ell_0 \ell_1 \cdots \ell_k \cdots$ be a route, $i \in \mathbb{N}$ and $\ell \in L$, we use: $\tau[i]$ to denote the $i$-th node $\ell_i$ in $\tau$, $\tau[i..]$ to indicate the suffix route $\ell_i \ell_{i+1} \cdots$, and $\tau(\ell)$ to denote the first occurrence of $\ell$ in $\tau$:

$$\tau(\ell) = \begin{cases} \min\{i | \tau[i] = \ell\} & \text{if} \ell \in \tau \\ \infty & \text{otherwise} \end{cases}$$

We also use $Routes(\mathcal{S})$ to denote the set of routes in $\mathcal{S}$, while $Routes(\mathcal{S}, \ell)$ denotes the set of routes starting from $\ell \in L$.

We can use routes to define the *distance* among two locations in a *spatial model*. This distance is computed via an appropriate function $f$ that combines all the weights in a route into a value.

**Definition 3 (Route Distance).** *Let* $\mathcal{S} = \langle L, \mathbf{W} \rangle$, $\tau$ *a route in* $\mathcal{S}$, *the distance* $d_\tau^f[i]$ *up-to index* $i$ *is:*

$$d_\tau^f[i] = \begin{cases} 0 & i = 0 \\ f(d_{\tau[1..]}^f[i-1], w) & (i > 0) \textit{ and } \tau[0] \overset{w}{\mapsto} \tau[1] \end{cases}$$

*Given a location* $\ell \in L$, *the distance over* $\tau$ *up-to* $\ell$ *is then* $d_\tau[\ell] = d_\tau^f[\tau(\ell)]$ *if* $\ell \in \tau$, *or* $\infty$ *otherwise*[1].

Considering again the sensor example, we can be interested in different types of distance. For example we can count the number of *hops*, simply using the function *hops* defined as $hops(v, w) := v + 1$ and in this case $d_\tau^{hops}[i] = i$. We can also consider the distances with respect the weighted label $w$ in the edges, in that case we have $weight(v, w) = v + w$ and $d_\tau^{weight}[i]$ is the sum the weights in the edges of the route until the $i$-th node $\ell_i$.

**Definition 4 (Location Distance)** *The distance between two locations* $\ell_1$ *and* $\ell_2$ *is obtained by choosing the distance values along all possible routes starting from* $\ell_1$ *and ending in* $\ell_2$*:*

$$d_{\mathcal{S}}[\ell_1, \ell_2] = \min\{d_\tau[\ell_2] | \tau \in Routes(\mathcal{S}, \ell_1)\}.$$

In the sensor network example, the distance between two locations $\ell_1$ and $\ell_2$, will be the minimum hop-length or weight-length over all paths connecting $\ell_1$ and $\ell_2$ for the *hops* or *weight* distance function respectively.

[1] We restrict here only to the tropical semiring, a more general definition can be found in [13].

### 2.2 Signal, Trace and Dynamic Models

We assume to have *piecewise constant temporal signal* $\nu = [(t_0, d_0), \ldots, (t_n, d_n)]$ with $t_i \in \mathbb{T} = [0, T] \subseteq \mathbb{R}_{\geq 0}$ a time domain and $d_i \in D$. Different kinds of signals can be considered: signals with $D = \{true, false\}$ are called *Boolean signals*; with $D = \mathbb{R}^{\infty}$ are called real-valued or *quantitative signals*, signal with $D = \mathbb{Z}$ are integer signals. We use $\mathcal{T}(\nu)$ to denote the sequence of $(t_0, \ldots, t_n)$ of *time steps* in $\nu$.

**Definition 5 (Spatio-temporal signal)** *Let $\mathcal{S} = \langle L, \mathbf{W} \rangle$ be a space model and $\mathbb{T} = [0, T]$ a time domain, a spatio-temporal signal is a function*

$$\sigma : L \to \mathbb{T} \to D$$

*that associates a temporal signal $\sigma(\ell) = \nu$ at each location. We use $\sigma@t$ to denote the* spatial signal *at time $t$, i.e. the signal $\mathbf{s}$ such that $\mathbf{s}(\ell) = \sigma(\ell)(t)$, for any $\ell \in L$.*

**Definition 6 (Spatio-Temporal Trace)** *Let $\mathcal{S} = \langle L, \mathbf{W} \rangle$ be a space model, a spatio-temporal trace is a function*

$$\boldsymbol{x} : L \to \mathbb{T} \to D^n$$

*such that for any $\ell \in L$ yields a vector of temporal signals $\boldsymbol{x}(\ell) = (\nu_1, \ldots, \nu_n)$. Note that this means that a spatio-temporal trace is composed by a set of spatio-temporal signals. In the rest of the paper we will use $\boldsymbol{x}(\ell, t)$ to denote $\boldsymbol{x}(\ell)(t)$.*

We can consider a spatio-temporal trace of our sensor network as $\boldsymbol{x} : L \to \mathbb{T} \to \mathbb{R} \times \mathbb{R}$ that associates a set of temporal signals $\boldsymbol{x}(\ell) = (\nu_B, \nu_T)$ at each location, where $\nu_B$ and $\nu_T$ correspond for example to the temporal signals of the battery and the temperature respectively in location $\ell$.

We also consider spatial models that can dynamically change their configurations. For example, the devices can move in space and change their position and connectivity pattern. For this reason, we need to define a structure that returns the spatial configuration at each time.

**Definition 7 (Dynamic Spatial Model)** *Let $L$ be a set of locations and $(t_0, \ldots, t_n)$ a set of time step with $t_i \in \mathbb{T} = [0, T] \subseteq \mathbb{R}_{\geq 0}$, a* Dynamic Spatial Model *is a function associating each element $t_i$ with a space model $\mathcal{S}_i$ that describes the spatial configuration at that time step, i.e. $(t_i, \mathcal{S}_i)$ for $i + 1, ..., n$ and $\mathcal{S}(t) = \mathcal{S}_i$ for all $t \in [t_i, t_{i+1})$.*

In case of a static spatial model we assume that $\mathcal{S}(t) = \mathcal{S}$ for all $t$.

## 3 Logic and Monitoring Procedures

Here we consider the specification of spatio-temporal properties by the *Spatio-Temporal Reach and Escape Logic* (STREL). STREL is an extension of the *Signal Temporal Logic* (STL) [34,36,57], with a number of spatial modal operators. Signal Temporal Logic is a linear-time temporal logic, it integrates the dense-semantics of the *Metric Interval Temporal Logic* (MITL) [5] with a set of parametrised numerical predicates playing the role of atomic proposition $\mu$, these are inequality of the form $(g(\nu_1, \ldots, \nu_n) \geq 0)$, for $g : \mathbb{R}^n \to \mathbb{R}$. Considering our wireless sensor network, example of atomic propositions are: $v_B > 0.5$, i.e. the level of the battery should be greater than 0.5, or $v_T < 30$, i.e. the value of temperature should be less than 30°.

The syntax of STREL is given by

**Definition 8 (STREL Syntax)**

$$\varphi := true \mid \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \,\mathrm{U}_I\, \varphi_2 \mid \varphi_1 \,\mathrm{S}_I\, \varphi_2 \mid \varphi_1 \,\mathcal{R}^f_d\, \varphi_2 \mid \mathcal{E}^f_d\, \varphi$$

where *true* is the Boolean *true* constant, $\mu$ is an *atomic predicate* ($AP$), *negation* $\neg$ and *conjunction* $\wedge$ are the standard Boolean connectives, the temporal modalities are the *until* ($\mathrm{U}_I$) and the *since* ($\mathrm{S}_I$), where $I$ is a non singular positive real interval, while *reachability* ($\mathcal{R}^f_d$) and the *escape* ($\mathcal{E}^f_d$) are the spatial operators, with $f$ a *Distance Function* described in the previous section (e.g. the *hops* function) and $d$ a non singular positive real interval. Both $I$ and $d$ can be infinite intervals, in case of using all $\mathbb{R}^{\infty}_{\geq 0}$ the interval can be omitted. In addition, we can derive the *disjunction* operator ($\vee$), the future *eventually* ($\mathrm{F}_I$) and *always* ($\mathrm{G}_I$) operators and the past *once* ($\mathrm{O}_I$) and *historically* ($\mathrm{H}_I$). We can derive also three other spatial operators: the *somewhere*, the *everywhere* and the *surround*. Below, we describe in detail the semantics of the spatial operators, we will see the temporal operators directly in the next Sections within the case studies, for more detail about temporal operators of STL we refer the reader to [34,36,57].

## 3.1 Boolean and Quantitative Semantics

The logic presents two semantics: a Boolean semantics, $(\mathcal{S}, \boldsymbol{x}, \ell, t) \models \varphi$, with the meaning that the spatio-temporal trace $\boldsymbol{x}$ in location $\ell$ at time $t$ with spatial model $\mathcal{S}$, satisfies the formula $\varphi$ and a quantitative semantics, $\rho(\varphi, \mathcal{S}, \boldsymbol{x}, \ell, t)$, that can be used to measure the quantitative level of satisfaction of a formula for a given trajectory. The function $\rho$ is also called the *robustness* function. The robustness is compatible with the Boolean semantics since it satisfies the soundness property: if $\rho(\varphi, \mathcal{S}, \boldsymbol{x}, \ell, t) > 0$ then $(\mathcal{S}, \boldsymbol{x}, \ell, t) \models \varphi$; if $\rho(\varphi, \mathcal{S}, \boldsymbol{x}, \ell, t) < 0$ then $(\mathcal{S}, \boldsymbol{x}, \ell, t) \not\models \varphi$. Furthermore it satisfies also the correctness property, which shows that $\boldsymbol{x}$ measures how robust is the satisfaction of a trajectory with respect to perturbations. We refer the reader to [36] for more details.

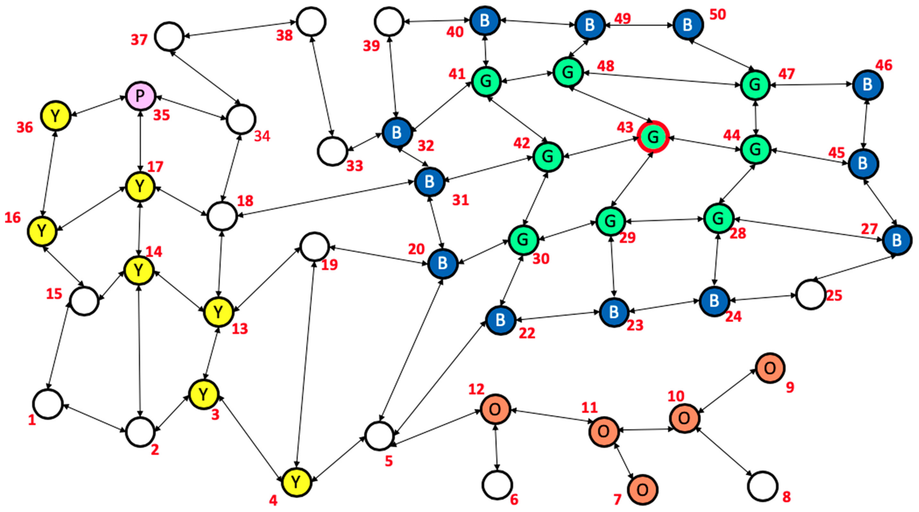


**Fig. 4.** Example of spatial properties. $\ell_3$ satisfies $yellow \mathcal{R}^{hops}_{[1,4]} pink$ while $\ell_4$ does not. $\ell_9$ satisfies $\mathcal{E}^{hops}_{[3,\infty]} orange$ while $\ell_{10}$ does not. $\ell_1$ satisfies $\diamonddiamond^{hops}_{[3,5]} pink$ and $\boxbox^{hops}_{[2,3]} yellow$. All green points satisfy $green \circledcirc^{hops}_{[0,100]} blue$. $\ell_{43}$ (the green point in the middle with a boild red circle) is the only location that satisfies $green \circledcirc^{hops}_{[2,3]} blue$. The letters inside the nodes indicate the color and the numbers indicate the enumeration of the locations. (Color figure online)

**Reach.** The quantitative semantics of the reach operator is:
$\rho(\varphi_1 \, \mathcal{R}^{f}_{[d_1,d_2]} \, \varphi_2, \mathcal{S}, \boldsymbol{x}, \ell, t) =$

$$= \max_{\tau \in Routes(\mathcal{S}(t),\ell)} \; \max_{\ell' \in \tau : \left(d_\tau^f[\ell'] \in [d_1,d_2]\right)} (\min(\rho(\varphi_2, \mathcal{S}, \boldsymbol{x}, \ell', t), \min_{j < \tau(\ell')} \rho(\varphi_1, \mathcal{S}, \boldsymbol{x}, \tau[j], t)$$

The Boolean semantics can be derived substituting $\min, \max$ with $\vee, \wedge$ and considering the Boolean satisfaction instead or $\rho$. $(\mathcal{S}, \boldsymbol{x}, \ell, t)$, a spatio-temporal trace $\boldsymbol{x}$, in location $\ell$, at time $t$, with a (dynamic) spatial model $\mathcal{S}$, satisfies $\varphi_1 \, \mathcal{R}^{f}_{[d_1,d_2]} \, \varphi_2$ iff it satisfies $\varphi_2$ in a location $\ell'$ reachable from $\ell$ through a route $\tau$, with a length $d^f_\tau[\ell'] \in [d_1, d_2]$, and such that $\tau[0] = \ell$ and all its elements with index less than $\tau(\ell')$ satisfy $\varphi_1$. Practically, the reach operator $\phi_1 \mathcal{R}^{f}_{[d_1,d_2]} \phi_2$ describes the behaviour of reaching a location satisfying property $\phi_2$ passing only through locations that satisfy $\phi_1$, and such that the distance from the initial location and the final one is greater than $d_1$ and less than $d_2$. In Fig. 4, we report an example of reachability property, considering f as the *hops* function described in the previous section. In the graph, the location $\ell_3$ (meaning the trajectory $\boldsymbol{x}$ at time t in position $\ell_3$ with spatial model $\mathcal{S}(t)$ as in the figure) satisfies $yellow \mathcal{R}^{hops}_{[1,4]} pink$. Indeed, there exists a route $\tau = \ell_3 \ell_{13} \ell_{14} \ell_{17} \ell_{35}$ such that $d_\tau^{hops}[\ell_{35}] = 4$, where $\tau[0] = \ell_3$, $\ell_{35}$ satisfies the pink property (i.e. it is pink) and all the other elements of the route satisfy the yellow property. Instead, for example, the location $\ell_4$ does not satisfy the property because it does not satisfies the distance constraint.

**Escape.** The quantitative semantics of the escape operator is:

$$\rho(\mathcal{E}^f_{[d_1,d_2]}\,\varphi, \mathcal{S}, \boldsymbol{x}, \ell, t) = \max_{\tau \in Routes(\mathcal{S}(t),\ell)} \max_{\ell' \in \tau : \left(d^f_{\mathcal{S}(t)}[\ell,\ell'] \in [d_1,d_2]\right)} \min_{i \leq \tau(\ell')} \rho(\varphi, \mathcal{S}, \boldsymbol{x}, \tau[i], t).$$

The Boolean semantics can be derived substituting $\min, \max$ with $\vee, \wedge$, and considering the Boolean satisfaction instead of $\rho$. $(\mathcal{S}, \boldsymbol{x}, \ell, t)$, a spatio-temporal trace $\boldsymbol{x}$, in location $\ell$, at time $t$, with a (dynamic) spatial model $\mathcal{S}$, satisfies $\mathcal{E}^f_{[d_1,d_2]}\,\varphi$ if and only if there exists a route $\tau$ and a location $\ell' \in \tau$ such that $\tau[0] = \ell$, $d_{\mathcal{S}}[\tau[0], \ell'] \in [d_1, d_2]$ and all elements $\tau[0], ...\tau[k]$ (with $\tau(\ell') = k$) satisfy $\varphi$. Practically, the escape operator $\mathcal{E}^f_{[d_1,d_2]}\phi$ describes the possibility of escaping from a certain region passing only through locations that satisfy $\phi$, via a route with a distance that belongs to the interval $d$.

In Fig. 4, we report an example of escape property. In the graph, the location $\ell_9$ satisfies $\mathcal{E}^{hops}_{[3,\infty]}$*orange*. Indeed, there exists a route $\tau = \ell_9\ell_{10}\ell_{11}\ell_{12}$ such that $\tau[0] = \ell_9$, $\tau[3] = \ell_{12}$, $d^{hops}_{S}[\ell_9, \ell_{12}] = 3$ and all elements $\tau[0], \tau[1], \tau[2], \tau[3]$ satisfy the orange property. Note that the route $\ell_{10}\ell_{11}\ell_{12}$ is not a good route to satisfy the property because the distance $d^{hops}_{S}[\ell_{10}, \ell_{12}] = 2$.

Now we describe the other three derived operators.

**Somewhere.** $\diamondsuit^f_{[d_1,d_2]}\varphi := true\mathcal{R}^f_{[d_1,d_2]}\varphi$ holds for $(\mathcal{S}, \boldsymbol{x}, \ell, t)$ iff there exists a location $\ell'$ in $\mathcal{S}(t)$ such that $(\mathcal{S}, \boldsymbol{x}, \ell', t)$ satisfies $\varphi$ and $\ell'$ is reachable from $\ell$ via a route $\tau$ with length $d^f_{\tau}[\ell'] \in [d_1, d_2]$. In Fig. 4, $\ell_1$ satisfies the property $\diamondsuit^{hops}_{[3,5]}$*pink* because there is a path $\tau = \ell_1 \ldots \ell_{35}$ with a length $d^{hops}_{\tau}(k) \in [3, 5]$, where $\tau[0] = \ell_1$, $\tau[k] = \ell_{35}$, and $\ell_{35}$ satisfies the pink property.

**Everywhere.** $\boxdot^f_{[d_1,d_2]}\varphi := \neg \diamondsuit^f_{[d_1,d_2]} \neg\varphi$ holds for $(\mathcal{S}, \boldsymbol{x}, \ell, t)$ iff all the locations $\ell'$ reachable from $\ell$ via a path,with length $d^f_{\tau}[\ell'] \in [d_1, d_2]$, satisfy $\varphi$. In Fig. 4, $\ell_1$ satisfies $\boxdot^{hops}_{[2,3]}$*yellow* because all the locations at a distance between 2 and 3 from $\ell_1$ satisfy the yellow property, while $\ell_2$ does not satisfies because $\ell_{18}$ is at a distance less than 3 but does not satisfy the yellow property.

**Surround.** $\varphi_1 \circledcirc^f_{[d_1,d_2]} \varphi_2 := \varphi_1 \wedge \neg(\varphi_1 \mathcal{R}^f_{[d_1,d_2]} \neg(\varphi_1 \vee \varphi_2) \wedge \neg(\mathcal{E}^f_{[d_2,\infty]}(\varphi_1))$ holds for $(\mathcal{S}, \boldsymbol{x}, \ell, t)$ iff there exists a $\varphi_1$-region that contains $\ell$, all locations in that region satisfies $\varphi_1$ and are reachable from $\ell$ via a path with length less than $d_2$. Furthermore, all the locations that do not belong to the $\varphi_1$-region but are directly connected to a location in $\varphi_1$-region must satisfy $\varphi_2$ and be reached from $\ell$ via a path with length in the interval $[d_1, d_2]$. Practically, the surround operator expresses the topological notion of being surrounded by a $\varphi_2$-region, while being in a $\varphi_1$-region, with additional metric constraints. The idea is that one cannot escape from a $\varphi_1$-region without passing from a node that satisfies $\varphi_2$ and, in any case, one has to reach a $\varphi_2$-node at a distance between $d_1$ and $d_2$. In Fig. 4, the green points satisfy *green* $\circledcirc^{hops}_{[0,100]}$ *blue*. Indeed, for each green point we can find a region that contains the point, such that all its points are green and all the points connected with an element that belongs to the region are blue and satisfy the metric constraint. Instead, the property $\mathit{green} \circledcirc^{hops}_{[2,3]} \mathit{blue}$ is satisfied only by location $\ell_{43}$ (the location with a bold red circle), indeed $\ell_{43}$ is the only location for which there exists a region (the green region) such that all its elements are at a distance less than 3 from $\ell_{43}$ and satisfy the green property; and all the locations directly connected with the green region are at a distance between 2 and 3 from $\ell_{43}$ and satisfy the blue property.

## 3.2 Offline Monitoring Algorithm

At the moment the logic supports only offline monitoring. The monitor takes as inputs a static or dynamic spatial model $\mathcal{S}$, a trace $\boldsymbol{x}$ and a formula $\phi$ and returns the spatio-temporal signal $\sigma$ representing the monitoring of $\phi$, a Boolean spatio-temporal signal for the Boolean Semantics and a real-value spatio-temporal signal for the quantitative one. The monitor of the whole trace corresponds to $\sigma@0$, i.e. the spatial Boolean or real-value signal at time zero. This means that the monitor of the whole trace corresponds to the evaluation at time $t = 0$ in each point in space: $(\mathcal{S}, \boldsymbol{x}, \ell) \models \varphi$ iff $(\mathcal{S}, \boldsymbol{x}, \ell, 0) \models \varphi$ and $\rho(\varphi, \mathcal{S}, \boldsymbol{x}, \ell) := \rho(\varphi, \mathcal{S}, \boldsymbol{x}, \ell, 0)$. We made this choice because we assume no privilege direction or location so we cannot consider a zero location as for the time.

Like in STL, monitoring of temporal operators is linear in the length of the signal times the number of locations in the spatial model. This because the monitoring procedure is performed at each location by using the same (linear) algorithm proposed in [36]. Monitoring of spatial properties is more expensive. These algorithms, formally described in [13], are based on a variations of the classical Floyd-Warshall algorithm. The number of operations to perform is polynomial on the size of the model times the length of the signal.

## 3.3 Application to Stochastic Systems

The analysis of spatio-temporal properties can be applied also on stochastic systems considering methodologies as Statistical Model Checking [74] (SMC). SMC combines simulation of the stochastic model (i.e. an algorithm that samples traces according to the probability distribution of the model in the Skorokhod space) with a monitoring routine for the property $\phi$. Stochastic systems induce a probability measure on the space of all possible traces (i.e. on the so-called Skorokhod space, the space of càdlàg functions, which are piecewise continuous functions of time, taking real values). If we define a *stochastic process* $\mathcal{M} = (\mathcal{T}, \mathcal{A}, \mu)$, where $\mathcal{T}$ is a trajectory space and $\mu$ is a probability measure on a $\sigma$-algebra $\mathcal{A}$ of $\mathcal{T}$, a quantity for measuring how a certain STREL formula $\varphi$ is satisfied by $\mathcal{M}$ is the *satisfaction probability* $S(\varphi, t)$, i.e. the probability that a trajectory generated by the stochastic process $\mathcal{M}$ satisfies the formula $\varphi$ at the time $t$: $\mathbb{E}[s(\varphi, \xi, t)] = \int_{\xi \in \mathcal{T}} s(\varphi, \xi, t) d\mu(\xi)$ where $s(\varphi, \xi, t) = 1$ if $(\xi, t) \models \varphi$ and 0 otherwise. The quantitative counterpart of the satisfaction probability is the *expected robustness*, defined as $\langle \rho(\varphi, t) \rangle := \mathbb{E}[\rho(\varphi, \xi, t)] = \int_{\xi \in \mathcal{T}} \rho(\varphi, \xi, t) d\mu(\xi)$ that is the expectation of the robustness computed over the trajectories of $\mathcal{M}$.

More specifically, SMC for satisfaction probability works by pipelining the generation of traces and their monitoring: every time a trace is generated by the simulator, it is passed to the Boolean monitor, which returns either 0 (false) or 1 (true). Probabilistically, this can be seen as a sample of a Bernoulli random variable, having probability $p(\phi)$ of observing 1. From a finite sample of such values, we can rely on standard statistical tools to estimate $p(\phi)$ and to compute the confidence level of such an estimate. Estimation of average robustness works in a similar way. Examples of spatio-temporal model checking to compute the approximated probabilistic satisfaction can be found in [14]. Analyzing these systems through the computation of satisfaction probability and/or average robustness, can therefore bring key insights in assessing and evaluating the design choices being made. The combination of Statistical Model Checking with quantitative semantics has been explored earlier for STL in [9] and applied to tasks like system design and parameter synthesis [9,25].

## 4 Static Space and Regular Grid: The Formation of Patterns

We consider here the simplest scenario, a regular grid, with only hop distance function and a deterministic model. In particular, in this example, we show how to exploit the surround operator to specify the formation of patterns and some other spatio-temporal related properties.

**Model and Trace.** The space model is a $K \times K$ grid treated as a weighted undirected graph, where each cell $(i, j) \in \{1, \ldots, K\} \times \{1, \ldots, K\}$ is a location, edges connect each pairs of neighbouring nodes along four directions and they have only one label which corresponds to the hop distance function, i.e. if two cells are neighbors the distance is equal to one.

The spatio-temporal trace describes the concentration of two proteins $A$ and $B$ in each cell of the grid at each time step. It is generated by a reaction-diffusion system, discretised according to a Finite Difference scheme [63], as a system of ODEs whose variables are organised in the $K \times K$ rectangular grid. Figure 5 reports the concentration of A for a number of time steps. It can be seen that from time $t = 20$ the shape of the pattern is apparent and remains stable; the pattern consists in a almost equidistant distribution of (blue) spots which have a low concentration of A surrounded by regions with a high concentration of A. For protein B (not shown) happens the opposite (high density regions surrounded by low density regions). More details about the model can be found here [62].

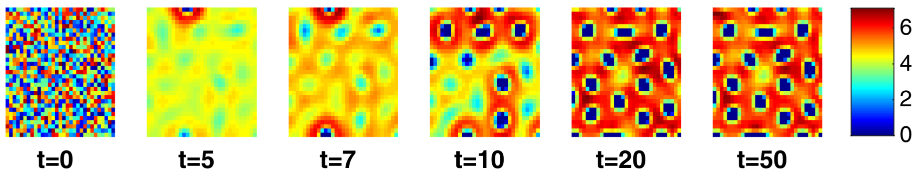


**Fig. 5.** Concentration of protein $A$ for the reaction-diffusion system for the frames with $t = 0, 5, 7, 12, 20, 50$ time units. The initial conditions (i.e. the initial concentration of A and B) are set randomly. The colour map for the concentration is specified in the legend on the right. (Color figure online)

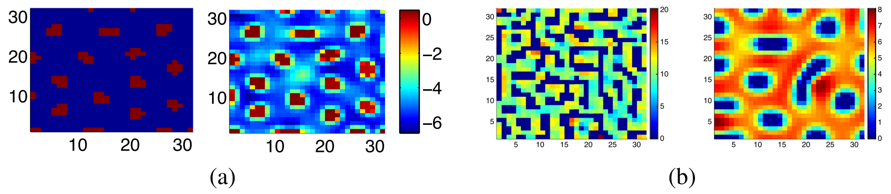


**Fig. 6.** (a) Boolean (left) and quantitative (right) satisfaction of the (spot formation) property; in the Boolean case the cells that satisfy the formula are in red; (b) Snapshots at time $t = 50$ of protein $A$ for the reaction-diffusion model with different diffusion rates for which we have the formation of different patterns. (Color figure online)

## 4.1 Properties

**Spot.** ($\phi_{\text{spot}} := (A <= h) \circledcirc^{hops}_{[d_1,d_2]} (A > h)$) holds in sub-regions that have low concentration of A, surrounded by a high concentrations of A. In detail, this property holds in the location $\ell$ that belongs to a region $L'$ of the grid where all elements satisfy the atomic proposition $A <= h$ and their distance from $\ell$ belong to the interval is less than $d_2$. Furthermore, each element directly connected with $L'$ satisfy $A > 0$, and its distance from $\ell$ belongs to $[d_1 d_2]$. The elements in the boundary correspond to all elements directly connected to a location of $L'$. Note that the use of distance bounds in the surround operator allows one to constrain the size/ diameter of the spot to $[d_1 d_2]$. If we have only one type of distance function, the name in the formula can be even omitted.

**Spot Formation.** ($\mathrm{F}_{[T,T+\delta]}\mathrm{G}(\mathit{spot})$) means that from a point in the future between $T$ and $T + \delta$ the *spot* property should always hold. In Fig. 6(a) we can see the Boolean and quantitative satisfaction of the Spot Formation formula with $h = 0.5, T = 19, \delta = 1, d_1 = 1, d_2 = 6$ for the trajectory reported in Fig. 5.

**Pattern.** ($\boxdot \diamondsuit_{[0,d_{spot}]} spotformation$) means that each node in the grid should be connected to a node at a distance less than $d_{spot}$ where the *spot* property holds, where $d_{spot}$ represent the maximum distance between spots. This property permits to describe a global behaviour. As we pointed in the description of the logic the monitor is done in each location, differently from the temporal part where we define the satisfaction of the whole trajectory as the satisfaction at time zero. Using the everywhere operator ($\boxdot$) without distance constraints means to cover all the locations of the grid, i.e. all the locations will have the same evaluation and this means that checking this formula in a random location of our space is enough to verify the presence of the pattern. In Fig. 6(b), we can see a snapshot of two different patterns generated by the same reaction-diffusion system, changing the diffusion rate. Changing the diffusion rate can affects the shape and size of the spots or even disrupts them. Trajectories generating patterns as in Fig. 6(b) do not satisfy the pattern formula (and the quantitative semantics returns a value around 0.5) while trajectory generating pattern as in Fig. 5 satisfy it.

## 5 Static Space and Stochastic Systems: Availability of Bikes

In this example we show how to combine temporal operators with the somewhere operator, for specifying some key properties of stochastic bike sharing systems. In particular, we observe how spatio-temporal properties can be used to analyse the availability of bikes or free slots in proximity of users.

**Model and Trace.** We consider the London Santander Cycles Hire network, modelled as a Population Continuous Time Markov Chain (PCTMC) with time-dependent rates [40]. We set the model parameters using historic journey and bike availability data from January 2015 to March 2015.

The Bike-Sharing System (BSS) is composed of a number of bike stations, distributed over a geographic area. Each station has a fixed number of bike slots. The users can pick up a bike, use it for a while, and then return it to another station in the area. The model, given the number of bikes/free slots in a station at time t, computes the probability distribution of the number of bikes/free slots in that station at time $t+h$ with $h \in [0,40]$ min. We can describe the model as a transition system where $B_i$ (respectively $S_i$) represents the bike agent (respectively the slot agent) in the $i^{th}$ station, $T^i_j$ is the bike agent travelling from pick-up station $i$ to return station $j$, while $N$ is the total number of stations. Each transition describes a possible event changing the state of the system. A transition rule like $B_i \rightarrow S_i + T^i_j$ models that an agent $B_i$ is removed from the system (a bike leaves station $i$), while new agents $S_i$ and $T^i_j$ are added to the system (a free slot is added to station $i$, while the bike is set to travel towards station $j$). The rate of each transition encodes the mean frequency with which it happens, considering historic journey data. For a detailed understanding of the model, the interested reader can refer to [62].

We consider a model with 733 bike stations (each with 20–40 slots) and a total population of 57,713 agents (users) picking up and returning bikes. We simulate the model using Simhya [22], a Java tool for the simulation of stochastic and hybrid systems, using the Gibson-Bruck (GB) algorithm. We are considering in particular the trajectories only of the bike ($B$) and slot ($S$) agents, in each station. Our spatio-temporal trace is then $\boldsymbol{x}(i,t) = (B_i(t), S_i(t))$, associating at each station $i$ the number of bikes and free slots at each time. The space is represented by a weighted graph, where the nodes are the bike stations and the edges describe the connection between each station. Two nodes are connected if they are at a distance less or equal than 1 km.

## 5.1 Properties

We use STREL to study spatio-temporal properties of the system and to explore their robustness considering a set of parameter values for the formulas. In the following, we will consider the distance induced by the function $\Delta(v,(x,y)) = v + \|(x,y)\|_2$, where $(x,y)$ are the coordinates of the distance vector between two adjacent nodes, while $v$ is the distance incrementally computed by $\Delta$.

**Local Availability.** One of the main problems of these systems consists in the availability of bikes or free slots in each station. The most interesting question related to this issue from a user's point of view is "If I don't find a bike (free slot, resp.) here, can I find another station close enough where there is an available bike (resp. free slot)?" This concern can be expressed by the STREL property described below:

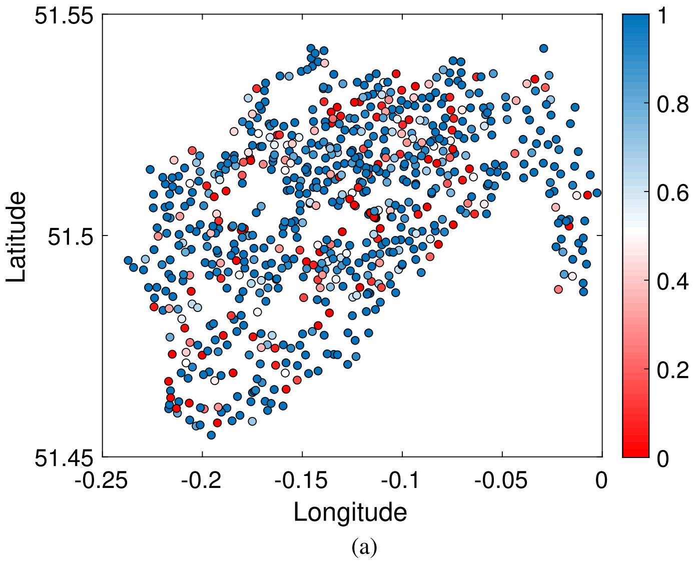
$$\phi_1 = \mathrm{G}_{[0,T_{end}]}\{\diamondsuit^{\Delta}_{[0,d]}(B > 0) \wedge \diamondsuit^{\Delta}_{[0,d]}(S > 0)\} \tag{1}$$

A station $\ell$ satisfies $\phi_1$ if and only if it is always true that, between 0 and $T_{end}$ minutes, there exists a station at a distance less than or equal to $d$, that has at least one bike and a station at a distance less or equal to $d$ that has at least one free slot.

In the analysis, we investigate the value of parameter $d \in [0,1]$ kilometres to see how the satisfaction of the property changes in each location. Figure 7 shows the approximate satisfaction probability $p_{\phi_1}$ for 1000 runs for all the stations, for (a) $d = 0$, and (b) $d = 0.3$ For $d = 0$, we can see that many stations have a high probability to be full or empty (indicated by red points), i.e. low values of satisfaction probability, with standard deviation of all the locations in the range [0, 0.0158] and mean standard deviation 0.0053. However, increasing $d$ to $d = 0.3$ km, i.e. allowing a search area of up to 300 metres from the station that currently has no bikes, or no slots respectively, we greatly increase the satisfaction probability of $\phi_1$, with a standard deviation that remains in the same range and mean standard deviation of 0.0039. For $d = 0.5$, the probability of $p_{\phi_1}$ is greater than 0.5 for all the stations; standard deviation is in the range [0, 0.0142] and mean stdv is 0.0002. Figure 8 (a) shows the satisfaction probability of some BBS stations vs distance d=[0,1.0].

**Timed Availability.** The property we analyzed previously did not consider that a user will need some time to reach a nearby station. Property $\varphi_1$ can be refined to take this aspect into consideration by considering a nested spatio-temporal property:

$$\psi_1 = \mathrm{G}_{[0,T_{end}]}\{\diamondsuit^{\Delta}_{[0,d]}(\mathrm{F}_{[t_w,t_w]}B > 0) \wedge \diamondsuit^{\Delta}_{[0,d]}(\mathrm{F}_{[t_w,t_w]}S > 0)\} \tag{2}$$



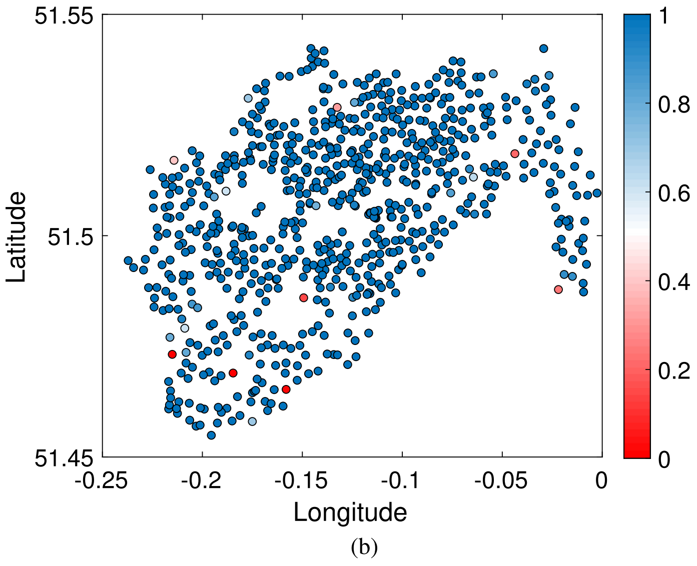


**Fig. 7.** Approximate satisfaction probability of formula $\phi_1$ for 1000 runs for each BSS station for (a) $d = 0$, and (c) $d = 0.3$. The value of the probability is given by the color legend. (Color figure online)

A station $\ell$ satisfies $\psi_1$ if and only if it is always true between 0 and $T_{end}$ minutes that there exists a station at a distance less than or equal to $d$, that, eventually in a time equal to $t_w$ (the walking time), has at least one bike and a station at

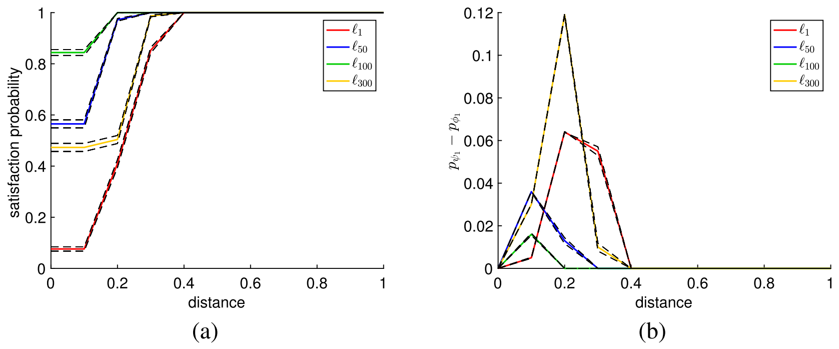


**Fig. 8.** Approximate satisfaction probability for property $\phi_1$ (a). (b) $p_{\psi_1} - p_{\phi_1}$ vs the distance d = [0,1.0]. Both evaluated over 1000 runs of BSS stations $\ell_1$,$\ell_{50}$, $\ell_{100}$, and $\ell_{300}$.

a distance less than or equal to $d$, that, eventually in a time equal to $t_w$ has at least one free slot.

We consider an average walking speed of 6.0 km/h, this means for example that if we evaluate a distance $d = 0.5$ km, we consider a walking time $t_w = 6$ min. The results of $\psi_1$ are very similar to the results of $\phi_1$. This means that there is not much difference between looking at $t = 0$ or after the walking time. Figure 8(b) shows the difference between the satisfaction probability of properties $\psi_1$, $\phi_1$ for the same locations.

## 6 Dynamic Space: Connectivity and Reliability in a MANET

In the previous sections we have considered two scenarios where the structure of the space does not change in time. Differently, in this section, we consider a scenario where the structure of the space is dynamic.

**Model and Trace.** We consider a mobile ad-hoc sensor network (MANET). This kind of systems can consist of up ten thousands of mobile devices connected wirelessly, usually deployed to monitor environmental changes such as pollution, humidity, light and temperature.

Each sensor node is equipped with a sensing transducer, data processor, a radio transceiver and an embedded battery. A node can move independently in any direction and indeed can change its links to other devices frequently. Two nodes can communicate each other if their Euclidean distance is at most their communication range as depicted in Fig. 3. Moreover, the nodes can be of different type and their behaviour and communication can depend on their types.

In this scenario, the spatial model of time $t$ is obtained by considering the graph where each device represents a node/location of the network. Edges are labelled with both their Euclidean distance and with the integer value 1. This last value is used to compute the hop (shortest path) count between two nodes, that is the number of intermediate network nodes through which data must pass between source node and target one. The signal associated with each location contains three values: the type of node (coordinator, router, end-device), the level of battery, and the temperature.

## 6.1 Properties

We can use STREL to specify, and monitor, properties related to connectivity of the network.

**Connectivity.** The first property one is interested to monitor is the *connectivity*. That is, each node that is an *end device* is directly connected either to a *router* or to a *coordinator*:

$$\psi_1 = end\_dev\, \mathcal{R}^{hops}_{[0,1]}\, (router \vee coord)$$

The formula above holds if from a node satisfying the atomic proposition *end_dev* (indicating an *end device*), we can reach a node satisfying either *router* or *coord* (that are the atomic proposition satisfied by *coordinators* or a *routers*), following a path in the spatial graph such that the *hops* distance along this path (i.e. its number of edges) is not bigger than 1.

More sophisticated properties can be specified with STREL. For instance, the following property can be used to specify that an *end device* is either connected to the *coordinator* or can reach it via a chain of at most of 5 routers:

$$\psi_2 = end\_dev\, \mathcal{R}^{hops}_{[0,1]}\, (router \mathcal{R}^{hops}_{[0,5]} coord)$$

**Delivery.** Another property that one could be interested in monitoring is the ability of the system to forward a message at a given distance. The ability of a component to forward a message is related to its battery level. To express this property, we can use the escape operator:

$$\psi_3 = \mathcal{E}^{hops}_{[5,\infty]}(battery > 0.5)$$

This property states that from a given location, we can find a path of (hops) length at least 5 such that all nodes along the path have a battery level greater than 0.5, i.e. that a message will be forwarded along a connection with no risk of power failure.

**Reliability.** Spatial and temporal operators can be mixed to specify properties regarding the evolution of the space in time. For instance, the following property is satisfied by the nodes with a battery level less than 0.5 that can reach in less than 10 hops another component that will eventually have a the battery level greater than 0.5:

$$\psi_4 = (battery < 0.5)\, \mathcal{R}^{hops}_{[0,10]}\, \mathrm{F}\,(battery > 0.5)$$

Moreover, the following property can be used to state that the correct spatial configuration is preserved in each time step:

$$\psi_5 = \mathrm{G}\psi_2$$

where $\psi_2$ is the formula defined above.

## 7 Related Work

**Machine Learning vs Specification Languages.** Pattern recognition is a well-established research area in machine learning. A typical approach consists in using a classifier trained with a labeled data set that assigns each data item to one class. Once the classifier is built from data using one of the several machine learning (ML) techniques [66] available, it can be used to detect/monitor whether a new data "most likely" matching to the classes. An important step is the choice and the extraction of the distinctive features [19,32,51,55] with relevant information from a set of input data representing the pattern of interest. Although extremely powerful, ML techniques lack generally of interpretability: they typically provide black-box data descriptors (e.g., deep neural networks) that are generally far from the human comprehension and ability to reason. Our approach to specify spatio-temporal patterns is instead based on a logic-based specification language. The declarative nature of the language offers an high-level and abstract framework enabling generalisation. STREL specifications are re-usable, compositional and generally closer to the human understanding.

**Spatio-Temporal Models.** In [69], Talcott introduced the notion of spatial-temporal event-based model to monitor spatial-temporal properties over CPS executions. In this model, actions (e.g. the exchange of messages, or a physical changes) are labeled with time and space stamps and they trigger events that are further processed by a monitor. In [70] the model was further extended to enable different space representations. Although the approaches in [69,70] provide a valuable algorithmic framework, they lack a specification language and the monitors cannot be automatically generated.

Other mathematical structures such as *topological spaces*, *closure spaces*, *quasi-discrete closure spaces* and *finite graphs* [61] have been investigated to reason about spatial relations (e.g. *closeness* and *neighborhood*) in the context of *collective adaptive systems* [31]. *Quad-trees* spatial structures [41] have been proposed in [44,47,48] to reason about fractal-like spatial patterns or spatial superposition properties in a grid, such as electrical spiral formation in cardiac tissues [47] or power management requirements in a smart grid [48]. Despite these models are suitable for offline and centralised monitoring of model-based simulations, they do not scale well for the runtime monitoring of spatially distributed CPS.

**Spatial and Spatio-Temporal Logics.** Spatial logics have been the subject of theoretically investigation since at least almost a couple of decades [3]. The work in [3] focuses on theoretically investigation, expressivity and decidability, often in continuous space. Less attention has been placed on more practical aspects, especially in the verification procedures. In particular, model checking techniques for spatial models have been introduced more recently. In [29], the authors introduce a *Spatial Logic for Closure Spaces* (SLCS) that leverages a discrete and topological notion of space, based on closure spaces [43]. An extension of the SLCS with temporal aspects, as "snapshot" models, was proposed later in [30]. This extends SLCS with the temporal modality of the branching logic *Computation Tree Logic* [38]. However, the algorithms to check snapshot models are very computational expensive and are susceptible to state-space explosion problems because the spatial formulae need to be recomputed at every state.

Relevant works are also those on spatial logics for process algebra with locations such as in [27,28], or spatial logic for rewrite theories [8]. Other logic-based formalisms have been introduced to reason about the topological [20] or directional [26] aspects of locally interacting components. In the topological approach [20], the entities are sets of points in the space and the relation between them is preserved under translation, scaling and rotation. If the relation between objects depends on their relative position then the spatial model supports the directional reasoning. These logics are highly computationally complex [26] or even undecidable [59] and indeed impractical to use.

**Table 1.** Table comparing the main features of different spatio-temporal logics.

| Specification language | Temporal logic | Spatial model | Static/dynamic space |
|---|---|---|---|
| STREL [13] | STL [57] | Weighted graph | Static/dynamic |
| SpaTeL [48] | STL [57] | Quad-trees | Static |
| STLCS [30] | CTL [38] | Closure spaces | Static |
| SaSTL [56] | STL [57] | Weighted graph | Static |
| SSTL [61] | STL [57] | Weighted graph | Static |

Monitoring spatial-temporal behaviors has recently started to receive more attention with *Spatial-Temporal Logic* (SpaTeL) [48], *Signal Spatio-Temporal Logic* SSTL [61], *Spatial Aggregation Signal Temporal Logic* (SaSTL) [56] and *Spatio-Temporal Reach and Escape Logic* STREL [13]. SpaTeL is the unification of *Signal Temporal Logic* [57] (STL) and *Tree Spatial Superposition Logic* (TSSL) introduced in [44] to classify and detect spatial patterns. Spatial properties are expressed using ideas from image processing, namely *quad trees* [41]. This allows one to capture very complex spatial structures, but at the price of a complex formulation of spatial properties, which are in practice only learned from some template images. SSTL combines STL with two spatial modalities, one expressing that something is true *somewhere* nearby and the other capturing the notion of being *surrounded* by a region that satisfies a given spatio-temporal property. SSTL has two possible semantics a Boolean and a real-valued one. SSTL [62] operates over a static topological space while STREL on the contrary can monitor entities over a dynamic topological space. Furthermore, STREL generalizes

SSTL spatial modalities with the *reach* and *escape* operators, simplifying the monitoring that can be computed locally with respect to each node. Finally, SaSTL [56] is recently proposed specification language that augment STL with two new logical operators for expressing spatial aggregation and spatial counting characteristics that are typical in monitoring spatio-temporal requirements in a smart city. Similarly to SSTL, also SaSTL operates on a static topological space. We summarize the main characteristics of the different spatio-temporal logics discussed in section in Table 1.

## 8 Conclusion

Spatio-temporal logics are high level languages that permit us to specify complex behaviours of dynamical systems distributed in space. In this tutorial paper we presented STREL, a modal logic which permits to specify spatio-temporal requirements and to monitor them automatically over a spatio-temporal trace. We discuss how STREL is suitable to capture behaviours of different scenarios: emergent Turing patterns, bike sharing systems, mobile sensor networks. These scenarios have both static and dynamic spatial structure and deterministic or stochastic dynamics. This flexibility of the logic, however, does not result in a high computational complexity of monitoring algorithms. This is a consequence of endowing the logic with spatial operators which are existential modal operators. Currently, STREL supports only offline monitoring. In order to properly apply STREL to real-time scenarios, we are designing dedicated online and distributed monitoring algorithms. Future work also includes the use of STREL within design and control loops of cyber-physical systems, leveraging the work done with STL in this respect [68].

## References

1. Abbas, H., Fainekos, G.E., Sankaranarayanan, S., Ivancic, F., Gupta, A.: Probabilistic temporal logic falsification of cyber-physical systems. ACM Trans. Embed. Comput. Syst. **12**(s2), 95:1–95:30 (2013). https://doi.org/10.1145/2465787.2465797
2. Abelson, H., et al.: Amorphous computing. Commun. ACM **43**(5), 74–82 (2000). https://doi.org/10.1145/332833.332842
3. Aiello, M., Pratt-Hartmann, I., van Benthem, J. (eds.): Handbook of Spatial Logics. Springer, Heidelberg (2007). https://doi.org/10.1007/978-1-4020-5587-4
4. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A survey on sensor networks. IEEE Commun. Mag. **40**(8), 102–114 (2002). https://doi.org/10.1109/MCOM.2002.1024422
5. Alur, R., Feder, T., Henzinger, T.: The benefits of relaxing punctuality. J. ACM **43**, 116–146 (1996)
6. Asarin, E., Caspi, P., Maler, O.: Timed regular expressions. J. ACM **49**(2), 172–206 (2002). https://doi.org/10.1145/506147.506151
7. Aurenhammer, F.: Voronoi diagrams—a survey of a fundamental geometric data structure. ACM Comput. Surv. **23**(3), 345–405 (1991). https://doi.org/10.1145/116873.116880

8. Bae, K., Meseguer, J.: A rewriting-based model checker for the linear temporal logic of rewriting. Electron. Notes Theoret. Comput. Sci. **290**, 19–36 (2012). https://doi.org/10.1016/j.entcs.2012.11.009
9. Bartocci, E., Bortolussi, L., Nenzi, L., Sanguinetti, G.: System design of stochastic models using robustness of temporal properties. Theoret. Comp. Sci. **587**, 3–25 (2015). https://doi.org/10.1016/j.tcs.2015.02.046
10. Bartocci, E., et al.: Specification-based monitoring of cyber-physical systems: a survey on theory, tools and applications. In: Bartocci, E., Falcone, Y. (eds.) Lectures on Runtime Verification. LNCS, vol. 10457, pp. 135–175. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75632-5_5
11. Bartocci, E., Fenton, F.H.: Teaching cardiac electrophysiology modeling to undergraduate students: laboratory exercises and GPU programming for the study of arrhythmias and spiral wave dynamics. Adv. Physiol. Educ. **35**(4), 427–437 (2011). https://doi.org/10.1152/advan.00034.2011
12. Bartocci, E., Bloem, R., Nickovic, D., Roeck, F.: A counting semantics for monitoring LTL specifications over finite traces. In: Chockler, H., Weissenbacher, G. (eds.) CAV 2018. LNCS, vol. 10981, pp. 547–564. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96145-3_29
13. Bartocci, E., Bortolussi, L., Loreti, M., Nenzi, L.: Monitoring mobile and spatially distributed cyber-physical systems. In: Proceedings of MEMOCODE 2017: The 15th ACM-IEEE International Conference on Formal Methods and Models for System Design, pp. 146–155. ACM (2017). https://doi.org/10.1145/3127041.3127050
14. Bartocci, E., Bortolussi, L., Milios, D., Nenzi, L., Sanguinetti, G.: Studying emergent behaviours in morphogenesis using signal spatio-temporal logic. In: Abate, A., Šafránek, D. (eds.) HSB 2015. LNCS, vol. 9271, pp. 156–172. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26916-0_9
15. Bartocci, E., Falcone, Y., Francalanza, A., Reger, G.: Introduction to runtime verification. In: Bartocci, E., Falcone, Y. (eds.) Lectures on Runtime Verification. LNCS, vol. 10457, pp. 1–33. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75632-5_1
16. Bartocci, E., Ferrère, T., Manjunath, N., Nickovic, D.: Localizing faults in simulink/stateflow models with STL. In: Proceedings of HSCC 2018: The 21st International Conference on Hybrid Systems: Computation and Control (part of CPS Week), pp. 197–206. ACM (2018). https://doi.org/10.1145/3178126
17. Bartocci, E., Manjunath, N., Mariani, L., Mateis, C., Ničković, D.: Automatic failure explanation in CPS models. In: Ölveczky, P.C., Salaün, G. (eds.) SEFM 2019. LNCS, vol. 11724, pp. 69–86. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30446-1_4
18. Bartocci, E., Manjunath, N., Mariani, L., Mateis, C., Nickovic, D., Pastore, F.: CPSDebug: a tool for explanation of failures in cyber-physical systems. In: Proceedings of ISSTA 2020: 29th ACM SIGSOFT International Symposium on Software Testing and Analysis, Virtual Event, USA, 18–22 July 2020, pp. 569–572. ACM (2020). https://doi.org/10.1145/3395363
19. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. IEEE Trans. Pattern Anal. Mach. Intell. **24**(4), 509–522 (2002). https://doi.org/10.1109/34.993558
20. Bennett, B., Cohn, A.G., Wolter, F., Zakharyaschev, M.: Multi-dimensional modal logic as a framework for spatio-temporal reasoning. Appl. Intell. **17**(3), 239–251 (2002). https://doi.org/10.1023/A:1020083231504

21. Bhattacharyya, A.: Morphogenesis as an amorphous computation. In: Proceedings of the Third Conference on Computing Frontiers, 2006, Ischia, Italy, 3–5 May 2006, pp. 53–64. ACM (2006). https://doi.org/10.1145/1128022.1128032
22. Bortolussi, L., Galpin, V., Hillston, J.: Hybrid performance modelling of opportunistic networks. In: Proceedings of QAPL 2012: The 10th Workshop on Quantitative Aspects of Programming Languages and Systems, pp. 106–121 (2012). https://doi.org/10.4204/EPTCS.85.8
23. Bortolussi, L., Milios, D., Sanguinetti, G.: U-check: model checking and parameter synthesis under uncertainty. In: Campos, J., Haverkort, B.R. (eds.) QEST 2015. LNCS, vol. 9259, pp. 89–104. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-22264-6_6
24. Bortolussi, L., Milios, D., Sanguinetti, G.: Smoothed model checking for uncertain continuous-time markov chains. Inf. Comput. **247**, 235–253 (2016)
25. Bortolussi, L., Silvetti, S.: Bayesian statistical parameter synthesis for linear temporal properties of stochastic models. In: Beyer, D., Huisman, M. (eds.) TACAS 2018. LNCS, vol. 10806, pp. 396–413. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-89963-3_23
26. Bresolin, D., Sala, P., Monica, D.D., Montanari, A., Sciavicco, G.: A decidable spatial generalization of metric interval temporal logic. In: 2010 17th International Symposium on Temporal Representation and Reasoning, pp. 95–102 (2010). https://doi.org/10.1109/TIME.2010.22
27. Caires, L., Cardelli, L.: A spatial logic for concurrency (part I). Inf. Comput. **186**(2), 194–235 (2003). https://doi.org/10.1016/S0890-5401(03)00137-8
28. Cardelli, L., Gordon, A.D.: Anytime, anywhere: modal logics for mobile ambients. In: Proceedings of POPL 2000: the 27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, pp. 365–377. ACM, New York (2000). https://doi.org/10.1145/325694.325742
29. Ciancia, V., Latella, D., Loreti, M., Massink, M.: Specifying and verifying properties of space. In: Diaz, J., Lanese, I., Sangiorgi, D. (eds.) TCS 2014. LNCS, vol. 8705, pp. 222–235. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44602-7_18
30. Ciancia, V., Gilmore, S., Grilletti, G., Latella, D., Loreti, M., Massink, M.: Spatio-temporal model-checking of vehicular movement in public transport systems (2015, Submitted)
31. Ciancia, V., Latella, D., Loreti, M., Massink, M.: Spatial logic and spatial model checking for closure spaces. In: Bernardo, M., De Nicola, R., Hillston, J. (eds.) SFM 2016. LNCS, vol. 9700, pp. 156–201. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-34096-8_6
32. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proceedings of CVPR 2005: The IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 886–893, June 2005
33. David, A., Larsen, K.G., Legay, A., Mikučionis, M., Wang, Z.: Time for statistical model checking of real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, pp. 349–355. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22110-1_27
34. Donzé, A., Maler, O.: Robust satisfaction of temporal logic over real-valued signals. In: Chatterjee, K., Henzinger, T.A. (eds.) FORMATS 2010. LNCS, vol. 6246, pp. 92–106. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15297-9_9

35. Donzé, A., Clermont, G., Legay, A., Langmead, C.J.: Parameter synthesis in nonlinear dynamical systems: application to systems biology. In: Batzoglou, S. (ed.) RECOMB 2009. LNCS, vol. 5541, pp. 155–169. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02008-7_11
36. Donzé, A., Ferrère, T., Maler, O.: Efficient robust monitoring for STL. In: Sharygina, N., Veith, H. (eds.) CAV 2013. LNCS, vol. 8044, pp. 264–279. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39799-8_19
37. Donzé, A., Krogh, B., Rajhans, A.: Parameter synthesis for hybrid systems with an application to simulink models. In: Majumdar, R., Tabuada, P. (eds.) HSCC 2009. LNCS, vol. 5469, pp. 165–179. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00602-9_12
38. Emerson, E.A., Halpern, J.Y.: "Sometimes" and "not never" revisited: on branching versus linear time. In: Wright, J.R., Landweber, L., Demers, A.J., Teitelbaum, T. (eds.) Proceedings of POPL 83: The Tenth Annual ACM Symposium on Principles of Programming Languages, pp. 127–140. ACM Press (1983). https://doi.org/10.1145/567067.567081
39. Fainekos, G.E., Pappas, G.J.: Robustness of temporal logic specifications for continuous-time signals. Theoret. Comput. Sci. **410**(42), 4262–4291 (2009). https://doi.org/10.1016/j.tcs.2009.06.021
40. Feng, C., Hillston, J., Reijsbergen, D.: Moment-based probabilistic prediction of bike availability for bike-sharing systems. In: Agha, G., Van Houdt, B. (eds.) QEST 2016. LNCS, vol. 9826, pp. 139–155. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-43425-4_9
41. Finkel, R.A., Bentley, J.L.: Quad trees: a data structure for retrieval on composite keys. Acta Informatica **4**, 1–9 (1974). https://doi.org/10.1007/BF00288933
42. Fishman, E.: Bikeshare: a review of recent literature. Transp. Rev. **36**(1), 92–113 (2016). https://doi.org/10.1080/01441647.2015.1033036
43. Galton, A.: The mereotopology of discrete space. In: Freksa, C., Mark, D.M. (eds.) COSIT 1999. LNCS, vol. 1661, pp. 251–266. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48384-5_17
44. Gol, E.A., Bartocci, E., Belta, C.: A formal methods approach to pattern synthesis in reaction diffusion systems. In: Proceedings of CDC 2014: The 53rd IEEE Conference on Decision and Control, pp. 108–113. IEEE (2014). https://doi.org/10.1109/CDC.2014.7039367
45. Golden, J.W., Yoon, H.S.: Heterocyst formation in anabaena. Curr. Opin. Microbiol. **1**(6), 623–629 (1998). https://doi.org/10.1016/S1369-5274(98)80106-9
46. Grosu, R., et al.: From cardiac cells to genetic regulatory networks. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, pp. 396–411. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22110-1_31
47. Grosu, R., Smolka, S.A., Corradini, F., Wasilewska, A., Entcheva, E., Bartocci, E.: Learning and detecting emergent behavior in networks of cardiac myocytes. Commun. ACM **52**(3), 97–105 (2009). https://doi.org/10.1145/1467247.1467271
48. Haghighi, I., Jones, A., Kong, Z., Bartocci, E., Grosu, R., Belta, C.: SpaTeL: a novel spatial-temporal logic and its applications to networked systems. In: Proceedings of HSCC 2015: The 18th International Conference on Hybrid Systems: Computation and Control, pp. 189–198. IEEE (2015). https://doi.org/10.1145/2728606.2728633
49. Jaksic, S., Bartocci, E., Grosu, R., Nguyen, T., Nickovic, D.: Quantitative monitoring of STL with edit distance. Formal Methods Syst. Des. **53**(1), 83–112 (2018). https://doi.org/10.1007/s10703-018-0319-x

50. Jaksic, S., Bartocci, E., Grosu, R., Nickovic, D.: An algebraic framework for runtime verification. IEEE Trans. CAD Integr. Circuits Syst. **37**(11), 2233–2243 (2018). https://doi.org/10.1109/TCAD.2018.2858460
51. Julesz, B.: Textons, the elements of texture perception, and their interactions. Nature **290**, 91–97 (1981)
52. Koymans, R.: Specifying real-time properties with metric temporal logic. Real-Time Syst. **2**(4), 255–299 (1990). https://doi.org/10.1007/BF01995674
53. Lee, E.A., Seshia, S.A.: Introduction to Embedded Systems, A Cyber-Physical Systems Approach, 2nd edn. MIT Press, Berlin (2017)
54. Loreti, M., Hillston, J.: Modelling and analysis of collective adaptive systems with CARMA and its tools. In: Bernardo, M., De Nicola, R., Hillston, J. (eds.) SFM 2016. LNCS, vol. 9700, pp. 83–119. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-34096-8_4
55. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the International Conference on Computer Vision, vol. 2, pp. 1150–1157 (1999)
56. Ma, M., Bartocci, E., Lifland, E., Stankovic, J.A., Feng, L.: SaSTL: spatial aggregation signal temporal logic for runtime monitoring in smart cities. In: Proceedings of ICCPS 2020: The 11th ACM/IEEE International Conference on Cyber-Physical Systems, pp. 51–62. IEEE (2020). https://doi.org/10.1109/ICCPS48487.2020.00013
57. Maler, O., Nickovic, D.: Monitoring temporal properties of continuous signals. In: Lakhnech, Y., Yovine, S. (eds.) FORMATS/FTRTFT -2004. LNCS, vol. 3253, pp. 152–166. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30206-3_12
58. Maler, O., Ničković, D.: Monitoring properties of analog and mixed-signal circuits. STTT **15**(3), 247–268 (2013)
59. Marx, M., Reynolds, M.: Undecidability of compass logic. J. Logic Comput. **9**(6), 897–914 (1999). https://doi.org/10.1093/logcom/9.6.897
60. Murthy, A., et al.: Curvature analysis of cardiac excitation wavefronts. IEEE/ACM Trans. Comput. Biol. Bioinform. **10**(2), 323–336 (2013). https://doi.org/10.1109/TCBB.2012.125
61. Nenzi, L., Bortolussi, L., Ciancia, V., Loreti, M., Massink, M.: Qualitative and quantitative monitoring of spatio-temporal properties. In: Bartocci, E., Majumdar, R. (eds.) RV 2015. LNCS, vol. 9333, pp. 21–37. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23820-3_2
62. Nenzi, L., Bortolussi, L., Ciancia, V., Loreti, M., Massink, M.: Qualitative and quantitative monitoring of spatio-temporal properties with SSTL. Log. Methods Comput. Sci. **14**(4) (2018). https://doi.org/10.23638/LMCS-14(4:2)2018
63. Olver, P.J.: Finite differences. Introduction to Partial Differential Equations. UTM, pp. 181–214. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-02099-0_5
64. Ratasich, D., Khalid, F., Geissler, F., Grosu, R., Shafique, M., Bartocci, E.: A roadmap toward the resilient Internet of Things for cyber-physical systems. IEEE Access **7**, 13260–13283 (2019). https://doi.org/10.1109/ACCESS.2019.2891969
65. Rodionova, A., Bartocci, E., Ničković, D., Grosu, R.: Temporal logic as filtering. In: Proceedings of HSCC 2016, pp. 11–20. ACM (2016). https://doi.org/10.1145/2883817.2883839
66. Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice Hall, Upper Saddle River (2002)
67. Sankaranarayanan, S., Kumar, S.A., Cameron, F., Bequette, B.W., Fainekos, G., Maahs, D.M.: Model-based falsification of an artificial pancreas control system. SIGBED Rev. **14**(2), 24–33 (2017). https://doi.org/10.1145/3076125.3076128

68. Silvetti, S., Policriti, A., Bortolussi, L.: An active learning approach to the falsification of black box cyber-physical systems. In: Polikarpova, N., Schneider, S. (eds.) IFM 2017. LNCS, vol. 10510, pp. 3–17. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66845-1_1
69. Talcott, C.: Cyber-physical systems and events. In: Wirsing, M., Banâtre, J.-P., Hölzl, M., Rauschmayer, A. (eds.) Software-Intensive Systems and New Computing Paradigms. LNCS, vol. 5380, pp. 101–115. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89437-7_6
70. Tan, Y., Vuran, M.C., Goddard, S.: Spatio-temporal event model for cyber-physical systems. In: 2009 29th IEEE International Conference on Distributed Computing Systems Workshops, pp. 44–50. IEEE (2009). https://doi.org/10.1109/ICDCSW.2009.82
71. Turing, A.M.: The chemical basis of morphogenesis. Philos. Trans. R. So. London B: Biol. Sci. **237**, 37–72 (1952)
72. Weiss, R.: Synthetic biology: from modules to systems. In: Proceedings of the 20th ACM Great Lakes Symposium on VLSI 2009, Providence, Rhode Island, USA, 16–18 May 2010, pp. 171–172. ACM (2010). https://doi.org/10.1145/1785481
73. Yaghoubi, S., Fainekos, G.: Hybrid approximate gradient and stochastic descent for falsification of nonlinear systems. In: Proceedings of ACC 2017: The 2017 American Control Conference, pp. 529–534. IEEE (2017). https://doi.org/10.23919/ACC.2017.7963007
74. Younes, H.L.S., Kwiatkowska, M., Norman, G., Parker, D.: Numerical vs. statistical probabilistic model checking: an empirical study. In: Jensen, K., Podelski, A. (eds.) TACAS 2004. LNCS, vol. 2988, pp. 46–60. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24730-2_4
75. Younes, H.L.S., Simmons, R.G.: Statistical probabilistic model checking with a focus on time-bounded properties. Inf. Comput. **204**(9), 1368–1409 (2006). https://doi.org/10.1016/j.ic.2006.05.002