Online Strategy Synthesis for Safe and Optimized Control of Steerable Needles

Sascha Lehmann[1] Antje Rogalla[1] Maximilian Neidhardt[2]
Alexander Schlaefer[2] Sibylle Schupp[1]
[1]Institute for Software Systems [2]Institute of Medical Technology
Hamburg University of Technology, Hamburg, Germany*
{s.lehmann, antje.rogalla, maximilian.neidhardt, schlaefer, schupp}@tuhh.de

Autonomous systems are often applied in uncertain environments, which require prospective action planning and retrospective data evaluation for future planning to ensure safe operation. Formal approaches may support these systems with safety guarantees, but are usually expensive and do not scale well with growing system complexity. In this paper, we introduce *online strategy synthesis* based on classical strategy synthesis to derive formal safety guarantees while reacting and adapting to environment changes. To guarantee safety online, we split the environment into region types which determine the acceptance of action plans and trigger local correcting actions. Using model checking on a frequently updated model, we can then derive locally safe action plans (prospectively), and match the current model against new observations via reachability checks (retrospectively). As use case, we successfully apply online strategy synthesis to medical needle steering, i.e., navigating a (flexible and beveled) needle through tissue towards a target without damaging its surroundings.

1 Introduction

In cyber-physical systems (CPS), we deal with the common task of directing a controllable entity through its environment towards a target state defined by local or global goals, without violating frame properties imposed by safety requirements. For applications of limited complexity with directly measurable environmental parameters (e.g., thermostats), practical control solutions already exist [21]. However, the environment of complex systems is often uncertain, changing, and not entirely measurable, rendering a comprehensive model of such environments for safety prediction infeasible. Deriving safety guarantees via formal methods imposes further limitations on feasible model complexities. Using a partial model instead, global safety guarantees might not hold anymore as soon as reality and model deviate.

In practice, safety is only required for the actual trace of the real system. An *online* approach based on model updates on the fly splits the task of deriving global guarantees into a series of safety verifications on locally valid models with limited scope. That way, we guarantee safety for all possible near futures of the current system state. However, three main problems persist: First, one still needs to guarantee that the system does not reach critical sections due to local deviations of the predicting model. Second, one needs to decide when to adapt the model or the real system. Third, among the locally safe solutions, one should choose those that keep the potential number of adaptations low. We approach these problems by splitting the environment into discrete regions, i.e., unknown, target, safe, critical, and detection regions, based on which we can derive safety-preserving entity actions and perform cost optimizations.

Overall, we approach the safety problem of dynamic CPS by the following four contributions:

*This study was partially funded by the TUHH i[3] lab initiative (T-LP-E01-WTM-1801-02), DFG SCHU 2479, and DFG SCHL 1844/6-1.

M. Farrell & M. Luckcuck (Eds.):
Third Workshop on Formal Methods
for Autonomous Systems (FMAS2021)
EPTCS 348, 2021, pp. 128–135, doi:10.4204/EPTCS.348.9

1. We introduce online strategy synthesis (OnSS) based on frequent model updates and local synthesis
2. We introduce a region interpretation of the environment (for the real system and model) to guarantee safety in the OnSS workflow
3. We investigate options of optimal action plan choice from synthesized strategies
4. We apply OnSS and the region interpretation to the application of autonomous medical needle steering

The remaining paper is structured as follows: We provide preliminary definitions and related work in Sec. 2. Then, in Sec. 3, we discuss the modeling and workflow, safety and optimization aspects of OnSS. Afterwards, we perform the needle steering experiments in Sec. 4, and conclude our work in Sec. 5.

2 Preliminaries and Related Work

A **game** is a mathematical model of interaction between different decision makers. A common setting is that of two (usually one *controllable* and one *uncontrollable*) adversarial players with alternating turns. One established use case is a game of a controllable user against an uncontrollable environment; in needle steering, these entities are the needle and the tissue, respectively. Evaluating a game allows determining solutions for winning or losing of a particular party involved.

A **timed automaton (TA)** is a finite-state machine extended by (real-valued) clocks [2]. It is a simple type of hybrid system, with only one differential equation, i.e., a constant change rate for all clocks. While clocks usually provide a notion of time (where clock invariants in locations and clock guards and reset on edges determine the timed transitions between states), they can be used to model continuous variables in general, including physical variables and accumulated costs.

A **timed game (TG)** extends the classical timed automaton with a notion of controllable and uncontrollable transitions. That way, one can model adversarial games between a controller and a (stochastic) environment in timed systems [6][10]. The combined formalism applies well to autonomous systems in particular, as they perform actions over time in a (partially) unknown and reactive environment.

Offline strategy synthesis (OffSS) is the automated approach of deriving a *winning strategy*, i.e., a strategy satisfying a given target property [3][11][22]. Strategy synthesis on timed games in particular is supported, e.g., by *Uppaal Stratego* [13]. There, a strategy consists of concrete decisions on actions in each state with controllable outgoing transitions. In needle steering, these decision points are motion actions, i.e., the push, rotate, and pull actions. OffSS provides the base for our online synthesis approach.

Our problem is an instance of controller synthesis and verification of timed games with partial information (about the environment). Environmental abstractions in static models inevitably lead to incomplete representations of real systems and possibly wrong system behavior at run time [16]. Several approaches therefore exist that consider environmental uncertainties already during the modeling phase. [12][14] present a transformation of a timed game with partial information into a timed game with complete information, [17] introduces a template-based controller synthesis approach based on automatic abstraction refinement, and [5] presents a new framework for synthesizing strategies for weighted timed games [9] with uncertainties restricted to weights. Other work introduces algebraic frameworks (e.g., based on risk factors [18]) for the design of correct safety controllers. Tools for static controller synthesis and verification include Kronos [15], FlySynth [1], Synthia [23], and the tool suite UPPAAL [19]. Timed controller synthesis and verification has been applied, among others, to online floor heating [20] and vehicle rerouting [8][7]. The latter involves uncertain traffic volume and periodically regenerates new strategies based on current traffic data. Our approach is similar but applies to cognitive or physiological uncertainties. Specifically, we transfer the static model of timed game needle steering [24] based on a nonholonomic model [25] into a model that is adapted online.

3 Online Strategy Synthesis

For online strategy synthesis, we use a single model combining the processes of synthesizing strategies and matching observation data. The general model consists of five components:

1. **Decision Maker** The decision maker who gives instructions to the controlled device
2. **Controlled Device** The entity controlled by the decision maker
3. **Environment** The uncontrollable environment
4. **State Checker** The acceptor model which checks properties on individual states
5. **Data Matcher** The acceptor model which matches observations against the action plan

The `Decision Maker`, `Controlled Device`, and `Environment` implement the concrete system entities, the `State Checker` accepts or rejects paths during strategy synthesis, and the `Data Matcher` validates the correctness of the current prediction model with observation data. Note that for autonomous devices, the `Decision Maker` and `Controlled Device` can be merged into a single `Actor`.

In needle steering, the `Decision Maker` is the surgeon (manual) or a stochastic action selector (autonomous), the `Controlled Device` is the needle, and the `Environment` is the tissue. The `State Checker` checks if critical states are entered, and whether the target is still reachable or already reached, and the `Data Matcher` matches the observed needle position data against the model-predicted motion.

In a game setting, the environment is especially important as it determines the safety and reachability of individual system states. It further determines when and to which extent particular characteristics can be measured. In an online setting, we need to know in which sections re-evaluation or adaptation of the model or real system is required to still ensure safe operation. Therefore, we split the complete state space into five regions based on the environment characteristics (see Fig. 1).

1. **Unknown regions (UR)** Regions not yet classified via a-priori knowledge or discovery
2. **Safe regions (SR)** Regions which do not violate safety
3. **Critical regions (CR)** Regions which violate safety
4. **Detection regions (DR)** Transitional regions between SRs and CRs where upcoming CRs (= safety violations) can be detected
5. **Target regions (TR)** Regions which we want to reach with the controllable entity

In needle steering, the regions map to spatial areas, i.e., the SRs are the uncritical tissue areas, the CRs are hardened tissue and organs, the DRs are pre-rupture deformation sections at which force increases steadily, and the TR is the targeted placement position.

Based on the system model and the regions, the workflow of online strategy synthesis is shown in Fig. 1. Generally speaking, the task is to reach a TR via SRs (or URs if knowledge on SRs is missing) without entering any CRs, which are early detected in surrounding DRs. The concrete workflow is as follows: First, the model is initialized (`Controlled Device`, `Environment`) with data of the real system's starting state. Then, an initial strategy is synthesized via reachability check on `State Checker`. Up to this point, the approach works offline. Afterwards, the system is executed and its tracked trace matched against the current motion plan model via `Data Matcher`. If an exceeding deviation is detected, the model is again updated, and a new strategy is synthesized. If no strategy can be found, or if a DR is reached, the system is *readjusted*, i.e., rolled back to a previously visited and known state. In needle steering, a local pullback of the needle is used for readjustment. Furthermore, the model is updated to the new system state after readjustment. If the initial state is re-reached via rollbacks, and again no strategy is found, the process is aborted (and may be started anew under different conditions). If the target region is reached, the process succeeded.

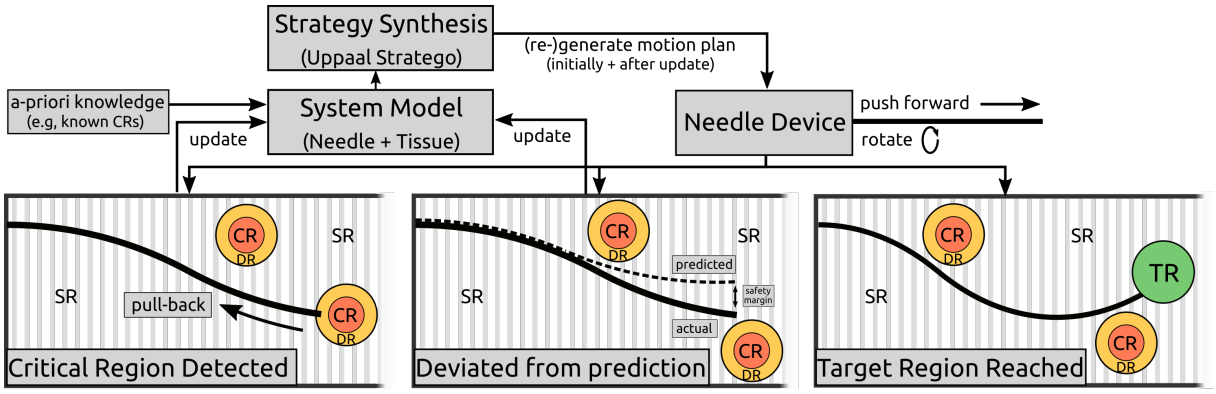


Figure 1: The workflow of the online needle steering application.

Online Safety Guarantees The online strategy synthesis combines safety and reachability requirements but prioritizes safety of the system over reaching a TR. We assume that every CR has been constructed so that it is surrounded by a DR. For each CR, we define a safe margin, e.g., a minimum required distance so that the CR can be detected early enough and that proper reaction in its DR is possible. The region sizes are computed based on the underlying system; for safety guarantees, it is necessary that the DRs are larger than the measurable step size under the given time resolution to ensure to halt the system in time when detection-related parameters change.

Theorem 1 (Safety) *A motion plan classified as safe via offline strategy synthesis is indeed safe under the momentary system assumptions ("local safety"). The system never reaches an unsafe state via online strategy synthesis ("global safety").*

Local safety follows directly from the soundness of the underlying strategy synthesizer. Global safety follows from the case distinction in Fig. 2 if one can show that the safety margins are determined so, that deviations from the model cannot lead the real system into known CRs and DRs.

The needle steering application indicates that the system assumptions are reasonable: We normally observe an increasing force when moving to another – possibly critical – tissue type due to deformations, which allows detection of upcoming CRs. Spanning around $5-6mm$, the deformation section is larger than the measurable step size of the used sensors, which provide new force and position data with a frequency of $150Hz$ (i.e., every $33.35\mu m$ of needle progress at a speed of $5mm/s$) with a conservative

		Real System			
		SR	CR	DR	TR
Model	UR	(3)	(1)	(3,*)	(4)
	SR	(3)	(1)	(3,*)	(4)
	CR	(2,3)	(1,2)	(2,3)	(2)
	DR	(2,3)	(1,2)	(2,3)	(2)
	TR	(3)	(1)	(3,*)	(4)

(1) Not possible as CRs not reachable in real system due to readjustment in surrounding DRs.
(2) Not possible as plans leading to CRs or DRs in the model are discarded during synthesis.
(3) Safe due to trivial safety of SRs and DRs.
(4) TR was safely reached and the process finishes successfully.
(*) Could affect termination (due to infinitely repeated readjustments), but is solved by safety margins.

Figure 2: Case distinction on regions for safety proof. (Note: Column for unknown regions (UR) in real system is left out, as each state of the real system is either known in advance or, when reached, directly classified as SR, CR, or DR based on measured data. Furthermore, URs in the model are treated as "safe" during strategy synthesis until they are further classified.)

maximum error of 3*mm*; the DRs are thus always measurable. Note that the error is mostly static and can thus be accounted for, unless optical tracking fails, e.g., due to air bubbles or surface reflections in gelatin, which may be prevented by ultrasound measurements in the future. Finally, rollbacks are possible as needle pullbacks will always follow the inversed insertion path, whose safety we discovered already.

Rollbacks increase the chance of finding safe plans but the OnSS algorithm does not guarantee that every safe plan is discovered. At the same time, termination of OnSS becomes non-trivial in the presence of rollbacks since one has to show that only a finite number of DRs is added. Due to space restrictions, we just state the following property without further proof.

Theorem 2 (Incompleteness and termination) *OnSS based on safety margins and on-the-fly discovered URs is incomplete. The OnSS instance for the needle-steering problem terminates.*

Action Plan Optimization In general, one can distinguish between *hard (H) and soft (S) requirements* for concrete action plans. In case of needle steering, the following requirements are given:

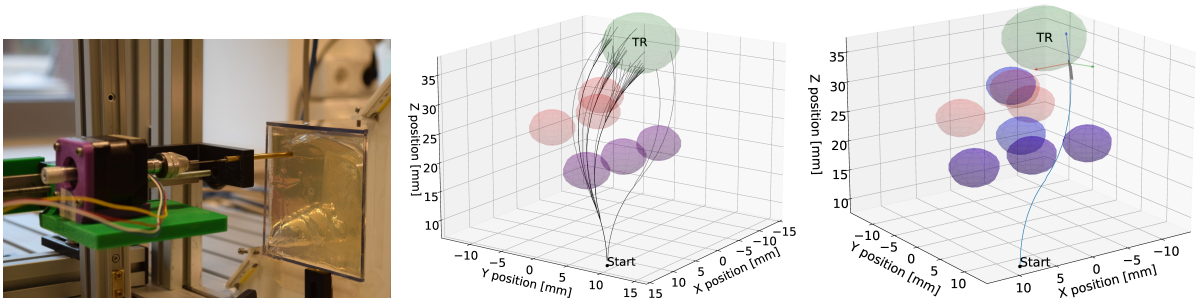
- **H1**: The target region is reached
- **S1**: As few rotations as possible are needed
- **S2**: The path is as short as possible
- **H2**: No critical region (e.g., critical tissue) is pierced
- **S3**: The critical regions are circumvented most spaciously
- **S4**: The path needs the fewest amount of readjustments
- **S5**: The target center is reached as close as possible

A synthesized strategy usually contains more than one possible action plan, which all automatically satisfy the binary hard requirements H1 and H2 regarding reachability and safety, respectively, based on the reachability query `EF StateChecker.Final_TR_Reached` and critical paths leading to deadlocks before. The soft requirements, in contrast, are continuous by nature, and their satisfaction differs for the accepted action plans. While some soft requirements are directly connected (e.g., S3 and S4), others usually contradict each other (e.g., S2 and S3), so that no universal optimum can be found. Therefore, a fixed weighting, i.e., cost assignment, of the soft requirements turns the action plan choice into an optimization problem. Depending on whether the system is discrete (using fixed time steps) or continuous (using ordinary differential equations), the costs assigned to the actions and distances in the model can be implemented either as integer variables incremented by cost deltas, or hybrid clocks, respectively.

The concrete cost values need to be provided a-priori. For needle steering, one source of knowledge for cost weighting are the underlying biological aspects. While the precise cost vector is still subject to ongoing research, plausible assumptions for cost assignments are that needle rotations inside the tissue impose more damage than a slightly longer path, and that – in the scope of "safe" regions – readjustments impose the highest damage.

4 Experiments

Following the model structure introduced in Sec. 3, we implemented a needle steering model in *Uppaal Stratego*. We developed an online strategy synthesis framework in *Python*, which then uses verification queries on the frequently updated model for the individual synthesis steps. We conducted two types of experiments based on randomly generated (**Experiment 1**) and real-measured reference needle paths (**Experiment 2**), where the setup shown in Fig. 3a is used for the latter. The reference paths are used to randomly place target and critical regions on and around that path, respectively, as base for each invidual experiment run; that way, we ensure that at least one safe motion plan towards the target region exists initially, so that the experiment is not aborted immediately. An experiment run *failed* if the starting state is re-reached via readjustments, or if the target region is not reached within 2 minutes.



(a) System setup (b) Initial strategy (c) Final needle trace

Figure 3: Experiment setup and needle motion plans with initially unknown (red), known, and discovered (both blue) CRs.

The experiment runs differ in the number ($\{0, 1, 2, \mathbf{5}, 10, 20\}$) and size ($\{1mm, 2mm, \mathbf{3mm}, 4mm, 5mm, 10mm\}$) of CRs, assumed size of identified CRs ($\{1mm, 2mm, \mathbf{3mm}, 4mm, 5mm, 10mm\}$), distance of the TR along the reference trace ($\{10mm, 20mm, \mathbf{30mm}, 40mm, 50mm\}$), and percentage of initially known CRs ($\{\mathbf{0}\%, 20\%, 40\%, 60\%, 80\%, 100\%\}$). One parameter is changed at a time, and the bold values are used for the other parameters that remain unchanged. For each of the 29 parametrizations derived hereby, we perform 20 experiment runs, i.e., 580 needle steering runs in total.

Fig. 3b and Fig. 3c exemplarily show the set of initially (offline) synthesized motion plans as well as the actually (online) traversed path for a single experiment run. Furthermore, the measurement results are shown in Tab. 1, and highlight four aspects: First, for experiments 1 and 2, the success rates of reaching a target are between $68 - 88\%$; the rates stay below 100% because of incompleteness (cf. Sec. 3) and imprecision of the underlying geometric model (only experiment 2). Second, the number of readjustments on average is comparatively low (0.57 and 2.94, respectively); some runs even need no readjustment at all. Third, the synthesis times lie between $0.04s$ and $6.99s$, which appear acceptable for an online application. Fourth, the successful runs took between $20.45s$ and $36.03s$ on average, so that the complete procedure (from insertion to target reaching) can be finished in reasonable time.

5 Conclusion and Future Work

We presented an online workflow to realize controller synthesis for real-timed games in (changing) environments of partial knowledge. We classify the environment into unknown, safe, critical, detection, and target regions and ensure safe action planning to reach a target state by periodically validating and updating our model. We applied our approach to medical needle steering.

In future work, we plan to replace the geometric approximation of needle motion by a physically accurate model and increase the "hostility" of the environment by permitting displacement of critical

	TR Reach	Readjustments	Synthesis Time	Overall Time
Experiment 1	88.28%	$(0.00, 0.57, 10.00)$	$(0.04s, 2.37s, 5.18s)$	$(7.42s, 20.45s, 85.82s)$
Experiment 2	68.39%	$(0.00, 2.94, 19.00)$	$(0.04s, 2.34s, 6.99s)$	$(7.61s, 36.03s, 119.50s)$

Table 1: Experiment results for each measure with (min,avg,max) data

regions due to respiration, deformed tissue layers (which affect the needle tip speed), and inhomogeneous properties of multi-layered tissue. Work on human-robot collaboration [4] may then provide a starting point for formal modeling and integration of more advanced human behavior.

References

[1] Karine Altisen & Stavros Tripakis (2002): *Tools for Controller Synthesis of Timed Systems*. In: *Proceedings 2nd Workshop on Real-Time Tools*.

[2] Rajeev Alur & David L. Dill (1994): *A Theory of Timed Automata*. *Theoretical Computer Science* 126, pp. 183–235, doi:10.1016/0304-3975(94)90010-8.

[3] Eugene Asarin, Oded Maler & Amir Pnueli (1995): *Symbolic controller synthesis for discrete and timed systems*. In: *Hybrid Systems II*, pp. 1–20, doi:10.1007/3-540-60472-3_1.

[4] Mehrnoosh Askarpour (2020): *How to Formally Model Human in Collaborative Robotics*. In: Proceedings Second Workshop on *Formal Methods for Autonomous Systems*, 329, pp. 1–14, doi:10.4204/EPTCS.329.1.

[5] Giovanni Bacci, Patricia Bouyer, Uli Fahrenberg, Kim G. Larsen, Nicolas Markey & Pierre-Alain Reynier (2021): *Optimal and robust controller synthesis using energy timed automata with uncertainty*. *Formal Aspects of Computing* 33(1), pp. 3–25, doi:10.1007/s00165-020-00521-4.

[6] Nathalie Bertrand & Sven Schewe (2012): *Playing Optimally on Timed Automata with Random Delays*. In: *FORMATS 2012 - Formal Modeling and Analysis of Timed Systems*, pp. 43–58, doi:10.1007/978-3-642-33365-1_5.

[7] Alexander Bilgram, Emil Ernstsen, Peter Greve, Harry Lahrmann, Kim G. Larsen, Marco Muñiz, Peter Taankvist & Thomas Pedersen (2021): *Online and Proactive Vehicle Rerouting with Uppaal Stratego*. *Transportation Research Record*, doi:10.1177/03611981211000348.

[8] Christopher Bischopink & Maike Schwammberger (2020): *Verification of Fair Controllers for Urban Traffic Manoeuvres at Intersections*. In: *FM 2019 - Formal Methods. FM 2019 International Workshops*, pp. 249–264, doi:10.1007/978-3-030-54994-7_18.

[9] Patricia Bouyer, Franck Cassez, Emmanuel Fleury & Kim Guldstrand Larsen (2004): *Optimal Strategies in Priced Timed Game Automata*. In: *FSTTCS 2004 - Foundations of Software Technology and Theoretical Computer Science*, pp. 148–160, doi:10.1007/978-3-540-30538-5_13.

[10] Patricia Bouyer & Vojtěch Forejt (2009): *Reachability in Stochastic Timed Games*. In: *ICALP 2009 - Automata, Languages, and Programming*, pp. 103–114, doi:10.1007/978-3-642-02930-1_9.

[11] Franck Cassez, Alexandre David, Emmanuel Fleury, Kim Guldstrand Larsen & Didier Lime (2005): *Efficient On-the-Fly Algorithms for the Analysis of Timed Games*. In: *CONCUR 2005 - Concurrency Theory*, pp. 66–80, doi:10.1007/11539452_9.

[12] Franck Cassez, Alexandre David, Kim G. Larsen, Didier Lime & Jean-François Raskin (2007): *Timed Control with Observation Based and Stuttering Invariant Strategies*. In: *ATVA 2007 - Automated Technology for Verification and Analysis*, pp. 192–206, doi:10.1007/978-3-540-75596-8_15.

[13] Alexandre David, Peter Gjøl Jensen, Kim Guldstrand Larsen, Marius Mikučionis & Jakob Haahr Taankvist (2015): *Uppaal Stratego*. In: *TACAS 2015 - Tools and Algorithms for the Construction and Analysis of Systems*, pp. 206–211, doi:10.1007/978-3-662-46681-0_16.

[14] Alexandre David, Kim Guldstrand Larsen & Thomas Chatain (2009): *Playing Games with Timed Games*. In: *ADHS 2003 - IFAC Proceedings Volumes*, pp. 238–243, doi:10.3182/20090916-3-ES-3003.00042.

[15] C. Daws, A. Olivero, S. Tripakis & S. Yovine (1996): *The tool Kronos*. In: *Hybrid Systems III*, pp. 208–219, doi:10.1007/BFb0020947.

[16] Angelo Ferrando, Louise A. Dennis, Rafael C. Cardoso, Michael Fisher, Davide Ancona & Viviana Mascardi (2021): *Toward a Holistic Approach to Verification and Validation of Autonomous Cognitive Systems*. *ACM Transactions on Software Engineering and Methodology* 30(4), doi:10.1145/3447246.

[17] Bernd Finkbeiner & Hans-Jörg Peter (2012): *Template-Based Controller Synthesis for Timed Systems*. In: *TACAS 2012 - Tools and Algorithms for the Construction and Analysis of Systems*, pp. 392–406, doi:10.1007/978-3-642-28756-5_27.

[18] Mario Gleirscher, Radu Calinescu & Jim Woodcock (2021): *RiskStructures: A Design Algebra for Risk-Aware Machines*. *Formal Aspects of Computing* 33, pp. 763–802, doi:10.1007/s00165-021-00545-4.

[19] Kim G. Larsen, Florian Lorber & Brian Nielsen (2018): *20 Years of UPPAAL Enabled Industrial Model-Based Validation and Beyond*. In: *Leveraging Applications of Formal Methods, Verification and Validation. Industrial Practice*, pp. 212–229, doi:10.1007/978-3-030-03427-6_18.

[20] Kim G. Larsen, Marius Mikučionis, Marco Muñiz, Jiří Srba & Jakob Haahr Taankvist (2016): *Online and Compositional Learning of Controllers with Application to Floor Heating*. In: *Tools and Algorithms for the Construction and Analysis of Systems*, pp. 244–259, doi:10.1007/978-3-662-49674-9_14.

[21] Yang Liu, Yu Peng, Bailing Wang, Sirui Yao & Zihe Liu (2017): *Review on cyber-physical systems*. *IEEE/CAA Journal of Automatica Sinica* 4(1), pp. 27–40, doi:10.1109/JAS.2017.7510349.

[22] Oded Maler, Amir Pnueli & Joseph Sifakis (1995): *On the synthesis of discrete controllers for timed systems*. In: *STACS 95 - Annual Symposium on Theoretical Aspects of Computer Science*, pp. 229–242, doi:10.1007/3-540-59042-0_76.

[23] Hans-Jörg Peter, Rüdiger Ehlers & Robert Mattmüller (2011): *Synthia: Verification and Synthesis for Timed Automata*. In: *CAV 2011 - Computer Aided Verification*, pp. 649–655, doi:10.1007/978-3-642-22110-1_52.

[24] Antje Rogalla, Sascha Lehmann, Maximilian Neidhardt, Johanna Sprenger, Marcel Bengs, Alexander Schlaefer & Sibylle Schupp (2020): *Synthesizing Strategies for Needle Steering in Gelatin Phantoms*. *Electronic Proceedings in Theoretical Computer Science* 316, pp. 261–274, doi:10.4204/eptcs.316.10.

[25] Robert J. Webster, Jin Seob Kim, Noah J. Cowan, Gregory S. Chirikjian & Allison M. Okamura (2006): *Nonholonomic Modeling of Needle Steering*. *International Journal of Robotics Research* 25(5-6), pp. 509–525, doi:10.1177/0278364906065388.