AFRL-RQ-WP-TR-2019-0114

FORMAL SPECIFICATION AND CORRECT-BY-CONSTRUCTION SYNTHESIS OF CONTROL PROTOCOLS FOR ADAPTABLE, HUMAN-EMBEDDED AUTONOMOUS SYSTEMS

Ufuk Topcu
Trustees of The University of Pennsylvania
The Clinical Practices of The University of Pennsylvania

MAY 2019
Final Report

AIR FORCE RESEARCH LABORATORY
AEROSPACE SYSTEMS DIRECTORATE
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7542
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the USAF 88th Air Base Wing (88 ABW) Public Affairs Office (PAO) and is available to the general public, including foreign nationals.

Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RQ-WP-TR-2019-0114 has been reviewed and is approved for publication in accordance with assigned distribution statement.

This report is published in the interest of scientific and technical information exchange and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
May 2019	Final	03 April 2015 – 31 May 2019

4. TITLE AND SUBTITLE
FORMAL SPECIFICATION AND CORRECT-BY-CONSTRUCTION SYNTHESIS OF CONTROL PROTOCOLS FOR ADAPTABLE, HUMAN-EMBEDDED AUTONOMOUS SYSTEMS

5a. CONTRACT NUMBER
FA8650-15-C-2546

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62201F

6. AUTHOR(S)
Ufuk Topcu

5d. PROJECT NUMBER
2403

5e. TASK NUMBER

5f. WORK UNIT NUMBER
Q18D

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Trustees of The University of Pennsylvania
The Clinical Practices of The University of Pennsylvania
3451 Walnut Street
Philadelphia, PA 19104-6205

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory
Aerospace Systems Directorate
Wright-Patterson Air Force Base, OH 45433-7542
Air Force Materiel Command
United States Air Force

10. SPONSORING/MONITORING AGENCY ACRONYM(S)
AFRL/RQQA

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-RQ-WP-TR-2019-0114

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited.

13. SUPPLEMENTARY NOTES
PA Clearance Number: 88ABW-2019-3628; Clearance Date: 26 July 2019

14. ABSTRACT
This project developed methods and computational tools for formal specification, analysis, and correct-by-construction synthesis of adaptable, hierarchical control protocols for human-embedded, multi-vehicle autonomous systems. The work made novel connections between controls, computer science formal methods, convex optimization, learning theory, and human factors. Its main research thrusts included 1) efficient and reliable computation engines for high-fidelity dynamic modeling, 2) formally embedding the operators into the execution of autonomy protocols, 3) automated feedback generation and inference of specifications, and 4) learning-based adaptation with provable correctness guarantees.

15. SUBJECT TERMS
formal methods, formal verification, synthesis, correct-by-construction, temporal logic, human-automation interaction, human-machine interaction, convex optimization, controls, learning theory, human factors, control protocols

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT:	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON (Monitor)
a. REPORT	b. ABSTRACT	c. THIS PAGE			Laura Humphrey
Unclassified	Unclassified	Unclassified	SAR	27	19b. TELEPHONE NUMBER (Include Area Code) N/A

TABLE OF CONTENTS

LIST OF FIGURES

1 SUMMARY

This project developed methods and computational tools for formal specification, analysis, and correct-by-construction (with respect to rich temporal logic specifications) synthesis of adaptable, hierarchical control protocols for human-embedded, multi-vehicle autonomous systems. The work has made novel connections between controls, formal methods from computer science, convex optimization, learning theory, and human factors. It was structured into four main research thrusts:

Thrust (1) Efficient and reliable computation engines for high-fidelity dynamic modeling: How can we incorporate high-fidelity dynamic models of the autonomous systems into autonomy protocol synthesis in a scalable way?

Thrust (2) Formally embedding the operators into the execution of autonomy protocols: How can we formally define human-embedded, flexibly adjustable autonomy and develop synthesis algorithms for the co-design of control protocols and the information content of operator interfaces?

Thrust (3) Automated feedback generation and inference of specifications: How can we render the automated synthesis procedure transparent and interpretable to human operators/designers by generating informative feedback and diverse design planning choices and inferring operator preferences and requirements as mathematically-based specifications?

Thrust (4) Learning-based adaptation with provable correctness guarantees: How can we develop protocols that both react to anticipated environmental changes and adapt to unforeseen contextual changes with provable guarantees of correctness with respect to temporal logic specifications?

Finally, a cross-cutting case study on multi-UAV operations with adversaries, cooperating assets, and human operators has served as a demonstration and assessment platform.

2 INTRODUCTION

The current generation of remotely-piloted vehicles has provided a proof-of-concept for unmanned systems. On the other hand, significant challenges remain to be addressed in order to realize the expected reductions in manpower requirements. Most unmanned vehicles feature low levels of autonomy. In current practice, human operators hand-code the waypoints the vehicle is to visit in order to complete the mission. Human encoding of detailed mission plans for multiple vehicles with rapidly changing user needs and requirements is not sustainable. The dynamic, possibly adversarial operating environments create a vast range of execution and contingency scenarios beyond human reasoning. Moreover, it is unrealistic to account for all run-time changes – often unforeseeable at design time – in the context in which the system operates.

The gap between the present capabilities and those needed for future mission scenarios calls for a new generation of control protocols that enable operator-autonomy interactions at higher levels of decision-making. Such control protocols shall be able to not only react to the changes – anticipated yet not necessarily known precisely at design time – but also adapt to unforeseen contextual changes. Furthermore, most of the systems are subject to strict assurance requirements. Despite all these complicating factors that unsustainably increase the development costs, the design and verification of autonomous systems and their interfaces to human operators are ad hoc. The process is rarely initiated with formal, mathematically-based specifications. A bottleneck leading to this primitive state is the lack of computational, automated tools.

In this project, we tackled these challenges by developing methods and tools for formal specification and correct-by-construction synthesis of control protocols that blend reactivity, adaptation, and human operators' inputs. In this report, we overview our results in six categories:

- Temporal-logic constrained planning and control
- Synthesis for human-autonomy interactions
- Synthesis under partial information
- Synthesis in multi-agent settings
- Constrained reinforcement learning
- Chance-constrained finite-time optimal control and set-to-set reachability computations

3 RESULTS

We now present the results in six categories: (i) temporal-logic-constrained planning and control; (ii) synthesis under partial information; (iii) synthesis in multi-agent settings; (iv) synthesis for human-autonomy interactions; (v) constrained reinforcement learning; (vi) chance-constrained finite-time optimal control and set-to-set reachability.

3.1 Temporal-Logic-Constrained Planning and Control

We present representative results that made novel connections between automata theory, approximate dynamic programming, satisfiability solving, automata learning and convex optimization.

3.1.1 Automata Theory Meets Approximate Dynamic Programming: Optimal Control with Temporal Logic Constraints

We re-visited, in (Papusha, et al., 2016), the problem of optimal control of dynamical systems under temporal logic specifications. Dynamical systems of interest to control are typically written as differential equations on a continuous state space, with inputs that can take on a continuum of values over a continuous time interval. However, temporal logic constraints that permit decidable synthesis must work with a finite or countable parameterization of time and space. As a result, a control designer must either forgo the continuous dynamics, create a discrete abstraction, or somehow re-express the temporal constraints within their optimal control framework.

Abstraction-based, hierarchical, and symbolic control methods have been proposed for continuous systems under temporal logic constraints. These classes of methods involve three general steps: 1) abstracting the dynamical system as a discrete finite-state system, 2) synthesizing a discrete control law using a product of the specification and the abstraction, and 3) compatibly implementing the discrete control law on the original continuous system. Approximate abstractions can be developed by reachability-based computational methods, counter-example guided abstraction refinement, and sampling-based methods. However, it is well-known that the abstraction process is computationally expensive. In addition, it is difficult to ensure the optimality of a control policy designed at the abstraction level with respect to a given continuous cost function.

The method we developed exploits the idea that continuous-time and continuous-state systems constrained by linear temporal logic (LTL) specifications can be formulated as a hybrid dynamical system through an augmentation of the continuous state space with the discrete states of the specification automaton. The resulting optimality conditions consist of mixed continuous-discrete Hamilton-Jacobi-Bellman (HJB) equations, which are difficult to solve in general. Therefore, approximate dynamic programming (ADP) can be used to approximate the value function, and to give an approximate policy. Our dynamic program makes use of a finite acceptance condition of the specification automaton by effectively treating controller synthesis as a kind of shortest path problem. As a result, our framework is also limited to a subset of LTL for its temporal specification language – in this case the fragment is called co-safe LTL. Importantly, co-safe LTL admits its own automaton construction method, which is more efficient than Buchi automata constructions for general LTL. The main parameters under the designer's

control are the value function bases used, rather than the fidelity of the discretization of time or space.

3.1.2 Maximum Realizability for Linear Temporal Logic Specifications

Automatic synthesis from temporal logic specifications is increasingly becoming a viable alternative for system design in a number of domains such as control and robotics. The main advantage of synthesis is that it allows the system designer to focus on what the system should do, rather than on how it should do it. Thus, the main challenge becomes providing the right specification of the system's required behavior. While significantly easier than developing a system at a lower level, specification design on its own is a difficult and error-prone task. For example, in the case of systems operating in a complex adversarial environment, such as robots, the specification might be over-constrained, and as a result unrealizable, due to failure to account for some of the possible behaviors of the environment. In other cases, the user might have several alternative specifications in mind, possibly with some preferences, and wants to know what the best realizable combination of requirements is. For instance, a temporary violation of a safety requirement might be acceptable, if it is necessary to achieve an important goal. In such cases it is desirable that, when the specification is determined to be unrealizable, the synthesis procedure provides a "best-effort" implementation either according to some user-given criteria, or according to the semantics of the specification language.

The challenges of specification design motivate the need to develop synthesis methods for the maximum realizability problem, where the input to the synthesis tool consists of a hard specification which must be satisfied by the system and soft specifications which describe other desired and possibly prioritized properties.

A key ingredient of the formulation of the maximum realizability problem is a quantitative semantics of the soft requirements. Broadly speaking, one can distinguish between two types of quantitative satisfaction: intrinsic, which is based on the semantics of the qualitative operators of the specification language, and extrinsic, which requires the user to provide certain quantitative information in terms of costs, weights, priority, or in terms of quantitative operators of the specification language. The approach to maximum realizability that we proposed in (Dimitrova, et al., 2018) is applicable to quantitative semantics from both classes.

In our recent studies, the main focus has been on soft specifications of the form $G\varphi_1 \wedge \ldots \wedge G\,\varphi_n$, where each $G\varphi_i$ is a syntactically safe LTL formula. For formulas of the form $G\varphi_i$, we considered a set of quantitative semantics that is typically used in the context of robustness. More precisely, we considered a set of intrinsic quantitative semantics which accounts for how often φ_i is satisfied. In particular, we considered truth values corresponding to φ_i being satisfied at every point of an execution, being violated only finitely many times, being both violated and satisfied infinitely often, or being continuously violated from some point on. We defined a function that determines the value in a given implementation of a conjunction $G\varphi_1 \wedge \ldots \wedge G\,\varphi_n$ of soft specifications based on this set of semantics. Our method then synthesizes an implementation that maximizes the value of the soft specifications. We further extended our proposed method to address quantitative semantics based on user-provided relaxations of the soft specification and weights capturing their priority.

The approach to maximum realizability that we developed is based on a bounded synthesis technique. Bounded synthesis is able to synthesize implementations of optimal size by leveraging the power of satisfiability (SAT), satisfiability modulo theory (SMT), or quantified Boolean formula (QBF) solvers. Since maximum realizability is an optimization problem, we reduced its bounded version to maximum satisfiability (MaxSAT). More precisely, we encoded the bounded maximum realizability problem with hard and soft specifications as a partial weighted MaxSAT problem, where hard specifications are captured by hard clauses in the MaxSAT formulation, and the weights of soft clauses encode the quantitative semantics of soft specifications. By adjusting these weights our approach can easily capture different variations of the quantitative semantics. Although the formulation encodes the bounded maximum realizability problem (where the maximum size of the implementation is fixed), by providing a bound on the size of the optimal implementation, we were able to establish the completeness of our synthesis method. The existence of such a completeness bound is guaranteed by considering quantitative semantics in which the values can be themselves encoded by LTL formulas.

3.1.3 An Automaton Learning Approach to Solving Safety Games over Infinite Graphs

We now overview our results in an automaton-learning-based approach for synthesis in two-player safety constrained games (Neider & Topcu, 2016). We proposed a method to construct finite-state reactive controllers for systems whose interactions with their adversarial environment are modeled by infinite-duration two-player games over (possibly) infinite graphs. The method targets safety games with infinitely many states or with such a large number of states that it would be impractical – if not impossible – for conventional synthesis techniques that work on the entire state space. We resort to constructing finite-state controllers for such systems through an automata learning approach, utilizing a symbolic representation of the underlying game that is based on finite automata. Throughout the learning process, the learner maintains an approximation of the winning region (represented as a finite automaton) and refines it using different types of counterexamples provided by the teacher until a satisfactory controller can be derived (if one exists). We presented a symbolic representation of safety games (inspired by regular model checking), proposed implementations of the learner and teacher, and evaluated their performance on examples motivated by robotic motion planning.

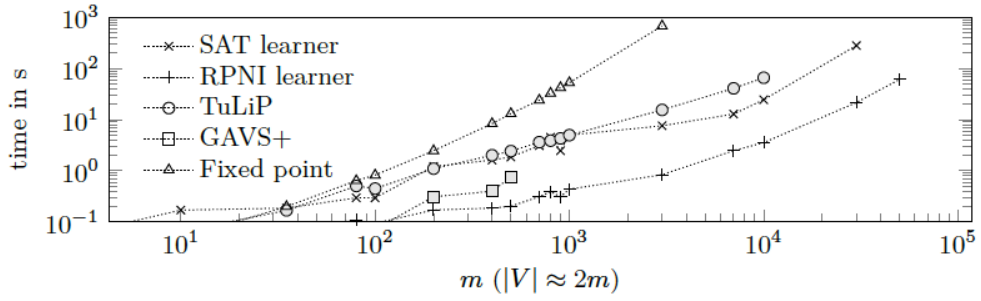

Figure 1: Sample results for automata learning for solving safety games.

Figure 2 shows the computation time for a series of finite-state motion planning examples over grid worlds of varying size. The figure compares three conventional synthesis algorithms and our learning-based algorithm with two different implementations of the teacher. The learning-based algorithms offer better scalability on these benchmark problems. Additionally, we applied the

technique on a series of examples from the literature with infinitely many states with finite input alphabet.

3.1.4 Sequential Convex Programming for Efficient Verification of Parametric Markov Decision Processes

We studied the applicability of convex optimization for the formal verification of systems that exhibit randomness and stochastic uncertainties (Cubuktepe, et al., 2017), (Cubuktepe, et al., 2018). Such systems are formally represented by so-called parametric Markov decision processes (MDPs).

Key requirements for applying model checking are a reliable system model and formal specifications of desired or undesired behaviors. As a result, most approaches to probabilistic model checking assume that models of the stochastic uncertainties are precisely given. For example, if a system description includes an environmental disturbance, the mean of that disturbance should be known before formal (probabilistic) statements are made about expected system behavior. However, the desire to treat many applications where uncertainty measures (e.g., faultiness, reliability, reaction rates, packet loss ratio) are not exactly known at design time gives rise to parametric probabilistic models. Here, transition probabilities are expressed as functions over system parameters, i.e., descriptions of uncertainties. In this setting, parameter synthesis addresses the problem of computing parameter instantiations leading to satisfaction of system specifications. More precisely, the functions are mapped to concrete probabilities that induce a resulting instantiated model that satisfies the specifications. A direct application is model repair, where a concrete model (without parameters) violates specifications. The model is changed, i.e., repaired, such that the specifications are satisfied. The repair is subject to a cost function, which penalizes deviations from some original instantiation; the underlying model is parametric.

Existing tools that address parameter synthesis, like PARAM, PRISM, or PROPhESY, compute rational functions over the parameters that express reachability probabilities or expected costs in a parametric Markov chain (MC). These optimized tools work on benchmarks having millions of states but are restricted to a few parameters as the computation of greatest common divisors does not scale well with the number of parameters. Moreover, the resulting functions are inherently nonlinear. In fact, they are often of high degree and very large. Evaluation by an SMT solver such as Z3 over nonlinear arithmetic suffers from the fact that the solving procedures are exponential in the degree of polynomials and the number of variables.

In this reporting period, we took another perspective. We presented a general nonlinear programming formulation for the verification of probabilistic MDPs (pMDPs). The powerful modeling capabilities of nonlinear programs (NLPs) enable incorporating multi-objective properties and penalties on the parameters of the pMDP. However, because of their generality, solving NLPs to find a global optimum is difficult. Even feasible solutions (satisfying the constraints) cannot always be computed efficiently. On the other hand, for the class of NLPs called convex optimization problems, efficient methods to compute feasible solutions and global optima even for large-scale problems are available.

We proposed an automated method of utilizing convex optimization for pMDPs with multiple specifications and an optimality objective. This method enables a direct and efficient synthesis method for similar problems that are formulated as NLPs. First, we restricted functions over parameters such that the problems we consider are formulated as signomial programs (SGPs), a certain class of nonconvex optimization problems. The restriction is mild and applies to a large class of widely studied benchmarks. The main steps of the method are as follows: (1) We relaxed nonconvex constraints in SGPs and applied a simple transformation to the parameter functions. The resulting programs are geometric programs (GPs), a class of convex programs. We showed that a solution to the relaxed GP induces feasibility (satisfaction of all specifications) in the original pMDP problem. Note that solving GPs is polynomial in the number of variables. (2) Given an initial feasible solution, we used a technique called sequential convex programming to improve a signomial objective. This local optimization method for nonconvex problems leverages convex optimization by solving a sequence of convex approximations of the original SGP.

3.2 Synthesis for Human-Autonomy Interactions

We present a range of results on synthesis of shared control protocols, run-time correction modules and explainable diagnostics for human-autonomy interactions.

3.2.1 Run-Time Shielding of Human-Embedded Autonomous Systems

Shields are generally used to correct the outputs of reactive systems, which comprise a broad range of systems whose exact runtime behaviors depend on external inputs. We recently showed how a shield could be used to enforce the desired properties in a human-embedded autonomous system interpreted as a reactive system (Humphrey, et al., 2016).

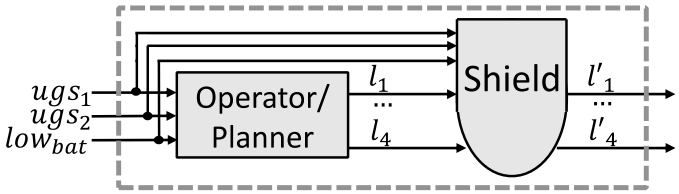

Figure 2: The interaction between the operator/planner and the shield.

Mission planning and execution can be very workload intensive, especially when operators are expected to control multiple unmanned systems simultaneously. Several errors can occur. For instance, in issuing high-level commands to the low-level planner, a human operator might neglect required safety properties due to high workload, fatigue, or an incomplete understanding of exactly how the autonomous planner might execute the command. The commands issued by the operator or planner could also be corrupted, e.g., by software that translates waypoint messages between ground station- and autopilot-specific formats or during transmission over the communication link. As the mission unfolds, waypoint commands may violate the specification, and this violation may go unnoticed by the operator due to high workload. In such cases, a shield that monitors system inputs may issue corrections (i.e., modifications to the waypoint commands) to ensure the desired properties.

We asked the question of how a shield should behave in human-embedded autonomous systems. We introduced the notions of best-effort strategies and admissible shields. If the human operator issues high-level commands to a low-level autonomous planner and wants to, for example, monitor a location in a restricted operating zone, she would like to simply command the unmanned vehicle to "monitor the location in the restricted zone and stay there"' with the planner handling the execution details. If the planner cannot do this while meeting all the safety properties, it is appropriate for another component such as the shield to revise its outputs. Yet, the operator would still expect his or her commands to be followed to the maximum extent possible; therefore, the shield should implement a best-effort strategy to minimize deviations from the operator's directives as executed by the planner. However, the shield will likely cause the unmanned vehicle to deviate from the route assumed by the planner, which may continue to send waypoint commands in a feed-forward manner, neglecting that the vehicle is taking a potentially longer route. This can cause the actual position of the vehicle to "fall behind" the position assumed by the planner, so that the next waypoint the planner issues is two or more steps away from the vehicle's current position. The shield should then implement a best-effort strategy to "synchronize"' the vehicle's actual position with that assumed by the planner. Though this cannot be guaranteed, the operator and planner are not adversarial towards the shield, so there will likely be opportunities to achieve this re-synchronization, e.g., when the vehicle goes back to a previous waypoint. We named these types of shields "admissible shields."

3.2.2 Synthesis of Shared Control Protocols with Provable Safety and Performance Guarantees

In shared control, a robot or an autonomous system executes a task to accomplish the goals of a human operator while adhering to additional safety and performance requirements. A human operator issues a command through an input interface, which maps the command directly to an action for the robot. The problem is that a sequence of such actions may fail to accomplish the task at hand e.g. due to limitations of the interface or failure of the human operator in comprehending the complexity of the problem. Therefore, a so-called *autonomy protocol* provides assistance to the human in order to complete the task according to the given requirements.

At the heart of the shared control problem is the design of the autonomy protocol. In the literature, there are two main directions. These are based on either *switching* the control authority between human and autonomy protocol or on *blending* their commands towards joined inputs for the robot.

One approach to switching the authority first determines the desired goal of the human operator with high confidence, then assists towards this goal. Switching the control authority between the human and autonomy protocol is done when necessary to ensure the satisfaction of specifications that are formally expressed in temporal logic. In general, switching of authority may be undesirable to the human, who usually prefers to retain as much control as possible.

The blending approach combines an automated command with the human operator's command. To introduce a more flexible trade-off between the human's control authority and the level of autonomous assistance, both commands are then blended to form a joined input for the robot. A

blending function determines the emphasis that is put on the autonomy protocol in the blending, that is, regulating the amount of assistance provided to the human. Switching of authority can be seen as a special case of blending, as the blending function may assign full control to the autonomy protocol or to the human. In general, putting more emphasis on the autonomy protocol in blending may lead to greater accuracy in accomplishing the task. However, humans prefer to retain control of the robot and may not approve if a robot issues a set of commands that is significantly different to the human's command. In any case, none of the existing blending approaches provide *formal correctness* guarantees that go beyond statistical confidence bounds. Correctness here refers to ensuring safety and optimizing performance according to the given requirements. Our goal in our recent work (Jansen, et al., 2017) was to design an autonomy protocol that admits formal correctness while rendering the robot behavior as close to the human's commands as possible, which is shown to enhance the human experience.

A human may be uncertain about which command to issue in order to accomplish a task. Moreover, a typical interface used to parse human commands, such as a brain-computer interface, is inherently imperfect. To capture such uncertainties and imperfections in the human's decisions, we introduce *randomness* to the commands issued by humans. It may not be possible to blend two different deterministic commands. If the human's command is "up" and the autonomy protocol's command is "right," we cannot blend these two commands to obtain another deterministic command. By introducing randomness to the commands of the human and the autonomy protocol, we ensure that the blending is always well-defined.

We modeled the behavior of the robot as an MDP, which captures the robot's actions inside a potentially stochastic environment. Problem formulations with MDPs typically focus on maximizing an expected reward (or, minimizing the expected cost). However, such formulations may not be sufficient to ensure safety or performance guarantees in a task that includes a human operator. Recently, it was shown that a reward structure is not sufficient to capture temporal logic constraints in general. We designed the autonomy protocol such that the resulting robot behavior satisfies probabilistic temporal logic specifications. Such verification problems have been extensively studied for MDPs and mature tools exist for efficient verification.

In what follows, we call a formal interpretation of a sequence of the human's commands the *human strategy*, and the sequence of commands issued by the autonomy protocol the *autonomy strategy*. In our previous work, we formulated the problem of designing the autonomy protocol as a *nonlinear programming problem*. However, solving nonlinear programs is generally intractable. Therefore, we later proposed a greedy algorithm that iteratively *repairs* the human strategy such that the specifications are satisfied without guaranteeing optimality. In (Jansen, et al., 2017), we proposed an alternative approach for the blending of the two strategies. We followed the approach of repairing the strategy of the human to compute an autonomy protocol. We ensured that the resulting robot behavior induced by the repaired strategy deviates minimally from the human strategy and satisfies safety and performance properties given in temporal logic specifications. We formally defined the problem as a *quasi-convex optimization problem*, which can be solved efficiently by checking feasibility of a number of convex optimization problems. The question remains how to obtain the human strategy in the first place. It may be unrealistic that a human can provide the strategy for an MDP that models a realistic scenario. To this end, we created a virtual simulation environment that captures the behavior of the MDP. We asked

humans to participate in two case studies to collect data about typical human behavior. We used *inverse reinforcement learning* to get a formal interpretation as a strategy based on human inputs. We modeled a typical shared control scenario based on autonomous wheelchair navigation in our first case study. In our second case study, we considered an unmanned aerial vehicle mission planning scenario, where the human operator is to patrol certain regions while keeping away from adversarial aerial vehicles.

3.2.3 Structured and Explainable Counter-Example Computation in UAV Mission Planning

In the formulation we developed in (Feng, et al., 2016), the execution of a UAV mission is composed (serially and/or concurrently) of a priori designed "plays." We considered each play as a finite-state MDP (or discrete-time MC) with particular entrance and exit conditions that characterize the types of compositions allowed. The composition of these plays, i.e., the execution, is then modeled as a large-scale MDP. While a human operator is considered to understand and be able to provide commands in terms of the plays, he or she is less likely to interpret all the details modeled in the composed MDP. Therefore, when there is a potential failure or an inconsistency between the high-level commands provided by the operator and the overall execution of the UAV mission (as captured in the composed MDP), an explanation of the causes in terms of the plays (rather than the individual, low-level states) is desirable.

In formal methods, counter-examples and counter-strategies are used as explanations of possible failures and inconsistencies. Motivated by the need of such explanations to be interpretable in UAV operations by human operators, we formulated a problem for the computation of structured counter-examples in MDPs and MCs. In this context, "structured" may refer to counter-examples that involve a small number of plays or those that primarily involve a subset of plays chosen by the operator. For discrete-time MCs, we formulated the problem as a mixed-integer linear programming problem and demonstrated its use on small running examples.

We later extended our method to produce counter-examples that accept "simple" structured natural-language-like explanations (Ghasemi, et al., 2018).

3.3 Synthesis of Strategies Subject to Information Limitations

We summarize our progress via three new methods for synthesis under information limitations.

3.3.1 Synthesis of Surveillance Strategies via Belief Abstraction

Performing surveillance, i.e. tracking the location of a target, has many applications. If the target is adversarial, these applications include patrolling and defense, especially in combination with other objectives, such as providing certain services or accomplishing a mission. Techniques for tracking non-adversarial but unpredictable targets have been proposed in settings like surgery to control cameras to keep a patient's organs under observation despite unpredictable motion of occluding obstacles. Mobile robots in airports have also been proposed to carry luggage for clients, requiring the robots to follow the human despite unpredictable motion and possibly sporadically losing sight of the target.

When dealing with a possibly adversarial target, a strategy for the surveilling agent can be seen as a strategy in a two-player game between the agent and the target. Since the agent may not always observe, or even know, the exact location of the target, surveillance is, by its very nature, a partial-information problem. It is thus natural to reduce surveillance strategy synthesis to computing a winning strategy for the agent in a two-player partial-information game. Game-based models for related problems have been extensively studied in the literature. Notable examples include pursuit-evasion games, patrolling games, and graph-searching games, where the problem is formulated as enforcing eventual detection, which is in its essence a search problem – once the target is detected, the game ends. For many applications, this formulation is too restrictive. Often, the goal is not to detect or capture the target, but to maintain certain level of information about its location over an unbounded (or infinite) time duration, or, alternatively, be able to obtain sufficiently precise information over and over again. In other cases, the agent has an additional objective, such as performing a certain task, which might prevent him from capturing the target but allow for satisfying a more relaxed surveillance objective.

In a recent paper (Bharadwaj, et al., 2018), we studied the problem of synthesizing strategies for enforcing *temporal surveillance objectives*, such as the requirement to never let the agent's uncertainty about the target's location exceed a given threshold or to recapturing the target every time it escapes. To this end, we considered surveillance objectives specified in linear temporal logic (LTL) with basic surveillance predicates. This formulation also allows for a seamless combination with other task specifications. Our computational model is that of a two-player game played on a finite graph, whose nodes represent the possible locations of the agent and the target and whose edges model the possible (deterministic) moves between locations. The agent plays the game with partial information, as it can only observe the target when it is in its field of view. The target, on the other hand, always has full information about the agent's location, even when the agent is not in view. In that way, we consider a model with one-sided partial information, making the computed strategy for the agent robust against a potentially more powerful adversary.

We formulated surveillance strategy synthesis as the problem of computing a winning strategy for the agent in a partial-information game with a surveillance objective. There is a rich theory on partial-information games with LTL objectives, and it is well known that even for very simple objectives the synthesis problem is EXPTIME-hard. Moreover, all the standard algorithmic solutions to the problem are based on some form of *belief set construction*, which transforms the imperfect-information game into a perfect-information game, and this may be exponentially larger since the new set of states is the powerset of the original one. Thus, such approaches scale poorly in general and are not applicable in most practical situations.

We addressed this problem by using *abstraction*. We introduced an *abstract belief set construction*, which underapproximates the information-tracking abilities of the agent (or alternatively overapproximates its belief, i.e., the set of positions it knows the target could be in). Using this construction we reduced surveillance synthesis to a two-player perfect-information game with an LTL objective, which we then solved using off-the shelf reactive synthesis tools. Our construction guarantees that the abstraction is sound, that is, if a surveillance strategy is found in the abstract game, it corresponds to a surveillance strategy for the original game. On the other hand, if such a strategy is not found, then the method automatically checks whether this is

due to the coarseness of the abstraction, in which case the abstract belief space is automatically refined. Thus, our method follows the general counterexample guided abstraction refinement scheme, which has successfully demonstrated its potential in formal verification and reactive synthesis.

3.3.2 Strategy Synthesis in Partially Observable Markov Decision Processes via Game-Based Abstractions

We studied in (Winterer, et al., 2019 (under review)) synthesis problems with constraints in partially observable Markov decision processes (POMDPs), where the objective is to compute a strategy for an agent that is guaranteed to satisfy certain safety and performance specifications. Verification and strategy synthesis for POMDPs are, however, computationally intractable in general. We alleviated this difficulty by focusing on planning applications and exploiting typical structural properties of such scenarios; for instance, we assumed that the agent has the ability to observe its own position inside an environment. We proposed an abstraction refinement framework which turns such a POMDP model into a (fully observable) probabilistic two-player game (PG). For the obtained PGs, efficient verification and synthesis tools allow for determining strategies with optimal safety and performance measures, which approximate optimal schedulers on the POMDP. If the approximation is too coarse to satisfy the given specifications, a refinement scheme improves the computed strategies. As a running example, we used planning problems where an agent moves inside an environment with randomly moving obstacles and restricted observability. We demonstrated that the proposed method advances the state of the art by solving problems several orders-of-magnitude larger than those that can be handled by existing POMDP solvers. Furthermore, this method gives guarantees on safety constraints, which is not supported by the majority of the existing solvers.

We considered in (Ahmadi, et al., 2018) a class of POMDPs with uncertain transition and/or observation probabilities. The uncertainty takes the form of probability intervals. Such uncertain POMDPs can be used, for example, to model autonomous agents with sensors with limited accuracy or agents undergoing a sudden component failure or structural damage. Given an uncertain POMDP representation of the autonomous agent, our goal was to propose a method for checking whether the system will satisfy an optimal performance objective while not violating a safety requirement (e.g., on fuel level or velocity). To this end, we cast the POMDP problem to a switched system scenario. We then exploited this switched system characterization and proposed a method based on barrier certificates for optimality and/or safety verification. We then showed that the verification task can be carried out computationally by sum-of-squares programming. We illustrated the efficacy of our method by applying it to a Mars rover exploration example.

3.4 Synthesis for Multi-Agent Systems

We overview several algorithms that we have developed over the course of the project for synthesizing control protocols for multi-agent systems.

3.4.1 Compositional Synthesis of Reactive Controllers for Multi-Agent Systems

We considered in (Alur, et al., 2016) a special class of multi-agent systems that are referred to as decoupled and are inspired by robot motion planning, decentralized control, and swarm robotics literature. Intuitively, in a decoupled multi-agent system the transition relations (or dynamics) of

the agents are decoupled, i.e., at any time-step, agents can make decisions on what action to take based on their own local state. For example, an autonomous vehicle can decide to slow down or speed up based on its position, velocity, etc. However, decoupled agents are coupled through objectives, i.e., an agent may need to cooperate with other agents or react to their actions to fulfill a given objective. For example, it would not be a wise decision for an autonomous vehicle to speed up when the front vehicle pushes the break if collision avoidance is an objective. In our framework, multi-agent systems consist of a set of controlled and uncontrolled agents. Controlled agents may need to cooperate with each other and react to the actions of uncontrolled agents in order to fulfill their objectives. Also, the controlled agents may be imperfect in the sense that they can only partially observe their environment, for example due to limitations in their sensors. The goal is to synthesize controllers for each controlled agent such that the objectives are enforced in the resulting system.

To solve the controller synthesis problem for multi-agent systems, one can directly construct a model of the system by composing those of the agents, then solve the problem centrally for the given objectives. However, the centralized method lacks flexibility, since any change in one of the components requires the repetition of the synthesis process for the whole system. Also, the resulting system might be exponentially larger than the individual parts, making this approach infeasible in practice. Compositional reactive synthesis aims to exploit the structure of the system by breaking the problem into smaller and more manageable pieces, then solving them separately. Then solutions to sub-problems are merged and analyzed to find a solution for the whole problem. The existing structure in multi-agent systems makes them a potential application area for compositional synthesis techniques.

To this end, we proposed a compositional framework for decoupled multi-agent systems based on automatic decomposition of objectives and compositional reactive synthesis using maximally permissive strategies. We assumed that the objective of the system was given in conjunctive form. We made an observation that in many cases, each conjunct of the global objective only referred to a small subset of agents in the system. We took advantage of this structure to decompose the synthesis problem: for each conjunct of the global objective, we only considered the agents that are involved, then computed the maximally permissive strategies for those agents with respect to the considered conjunct. We then intersected the strategies to remove potential conflicts between them and projected back the constraints to subproblems, solving them again with updated constraints, and repeating this process until the strategies become fixed.

We implemented the algorithms symbolically using binary decision diagrams (BDDs) and applied them to a robot motion planning case study where multiple robots were placed on a grid-world with static obstacles and other dynamic, uncontrolled, and potentially adversarial robots. We considered different objectives such as collision avoidance, keeping a formation, and bounded reachability. We showed that by taking advantage of the structure of the system, the proposed compositional synthesis algorithm could significantly outperform a centralized synthesis approach, both from time and memory perspective, and could solve problems where the centralized algorithm was infeasible. Furthermore, using compositional algorithms we managed to solve synthesis problems for systems with multiple agents, more complex objectives, and for grid worlds of sizes that were much larger than the cases considered in similar works. Our findings showed the potential of symbolic and compositional reactive synthesis methods as

planning algorithms in the context of a dynamically changing and possibly adversarial environment.

3.4.2 Shielding for Multi-Agent Systems

Since multi-agent systems exhibit complex interactions with their environment and between individual agents, they are often difficult to understand and are notoriously hard to design correctly. Individual agents have to not only fulfill their local objectives and meet their local requirements, but also abide by system-wide or global safety requirements such as avoiding collision with other agents. Distributed reactive synthesis is able to automatically transform a given correctness specification and a given architecture describing the individual agents' interaction into a correct-by-construction implementation. Unfortunately, except for a few restricted classes of architectures, the distributed synthesis problem is undecidable. Even the decidable versions of the problem lack practical solutions due to their non-elementary complexity.

To address this problem, there has been a large body of work in designing algorithms to perform agent coordination and task assignment for a wide array of applications. For example, software frameworks such as the Air Force Research Laboratory's Unmanned Systems Autonomy Services (UxAS) provide mission-level autonomy for multi-agent systems and include capabilities from high-level task assignment to path planning for unmanned systems. Such frameworks often allow for dynamic task reallocation as missions change, but in doing so, cannot necessarily account for potential violations of global safety specifications. This necessitates shielding the agents at runtime from a possible task assignment that can cause a violation of a global safety specification.

One approach in this direction is to perform runtime verification that allows checking whether a run of a system satisfies a given specification. An extension of this idea is to perform runtime enforcement of the specified property, by not only detecting property violations, but also altering the behavior of the system in a way that maintains the desired property.

Shield synthesis is a general method to automatically derive runtime enforcement implementations, called shields, from temporal logical specifications. A shield is attached to a reactive system, monitors the behavior of the system (i.e., its inputs and outputs), and corrects erroneous outputs instantaneously, but only if necessary and as infrequently as possible.

In a recent paper (Bharadwaj, et al., 2019), we introduced shield synthesis for multi-agent systems. A shield monitors and (if needed) corrects the output of one or more agents in the system, such that a given global safety specification is always satisfied. The distributed nature of the problem gives rise to a number of considerations to be made during the shield synthesis procedure. In order to explore the design space of possible shields for multi-agent systems, we categorize shields based on three criteria according to: (1) the interference of the shield processes with the individual agents, (2) the assumptions on the behavior of the agents the shield can rely on, and (3) the fairness of the shield with respect to the individual agents.

(1) Quantifying interference – By construction, a shield is guaranteed to enforce correct operation of the shielded system. However, we might prefer one shield over another, based on

how much the shield interferes with the system as a whole or how it interferes with the individual agents in case of an error. We developed a notion of interference cost in order to quantify the quality of a shield and synthesize cost-optimal shields that minimize the interference cost for the worst-case behavior of the multi-agent system. We have also developed algorithms to synthesize cost-optimal shields based on different cost functions.

(2) Assumptions on the multi-agent system – The shield synthesis procedure does not rely on the particular implementation of the system or specifications of each of the agents, which is the key to the practicability of the approach. Instead, a shield has to guarantee safety for any possible implementation. However, it is often realistic to make assumptions on the worst-case behavior of the system and synthesize optimal shields with respect to the chosen interference cost under those assumptions. A natural assumption is that wrong outputs occur rarely, i.e., the length of all sequences of wrong outputs is bounded. When such knowledge is available, we can compute a cost-optimal shield considering the worst-case behavior of any system satisfying the assumptions.

(3) Fair shielding – In the multi-agent setting, in which each individual agent might have to fulfill some individual goals, it is often important that a shield treats all agents fairly: in case of an error, a fair shield does not always interfere with the same agent repeatedly. We defined a fairness notion for shields and corresponding synthesis procedure.

3.4.3 Distributed Synthesis of Surveillance Strategies for Mobile Robots

We studied in (Bharadwaj, et al., 2018) the problem of synthesizing strategies for a mobile sensor network to conduct surveillance in partnership with static alarm triggers. We formulated the problem as a multi-agent reactive synthesis problem with surveillance objectives specified as temporal logic formulas. In order to avoid the state space blow-up arising from a centralized strategy computation, we proposed a method to *decentralize* the surveillance strategy synthesis by decomposing the multi-agent game into subgames that can be solved independently. We also decomposed the global surveillance specification into local specifications for each sensor, and showed that if the sensors satisfy their local surveillance specifications, then the sensor network as a whole will satisfy the global surveillance objective. Thus, our method guarantees global surveillance properties in a mobile sensor network while synthesizing completely decentralized strategies with no need for coordination between the sensors. We also presented a case study in which we demonstrate an application of decentralized surveillance strategy synthesis.

3.5 Constrained Reinforcement Learning

We now overview our results on constrained reinforcement learning.

3.5.1 Safety-Constrained Reinforcement Learning for Markov Decision Processes

Many formal system models are inherently stochastic. Consider for instance randomized distributed algorithms (where randomization breaks the symmetry between processes), security (e.g., key generation at encryption), systems biology (where species randomly react depending on their concentration), or embedded systems (interacting with unknown and varying environments). These various applications have made the verification of stochastic systems such

as discrete-time Markov chains (MCs) or MDPs an important research topic in the last decade, resulting in several tools like PRISM, LiQuoR, MRMC, and FMurph. The always growing set of case studies in the PRISM benchmark suite serve as witnesses to the applicability of MDP and MC model checking.

However, controller synthesis is a relatively new topic in this setting. Consider a controllable system such as a robot or some other machine which is embedded into an environment. Having a formal model of both the controllable entity and the environment, the goal is to synthesize a controller that satisfies certain requirements. Again, faithful models are often stochastic, e.g. due to sensor imprecisions of a robot, message loss, or unpredictable behavior of the environment. Moreover, it might be the case that certain information – such as cost caused by energy consumption – is not exactly known prior to exploration and observation.

We therefore studied the following problem (Junges, et al., 2016): Given an MDP with a cost structure, synthesize an optimal policy subject to safety constraints. This multi-objective model checking problem has been previously studied, but not when the cost function is unknown. Consider for instance a motion planning scenario in a grid-world where a robot wants to move to a certain position. Acting unpredictably, a janitor moves randomly through the grid. The robot reaches its goal safely if it moves according to a strategy that avoids the janitor. Moreover, each movement of the robot occasions cost depending on the surface. However, the robot only learns the actual costs by physically executing actions within the environment; this requires the exclusion of unsafe behavior prior to exploration. Consequently, we need to find a safe strategy for the robot which simultaneously induces minimal cost.

We model robot behavior as an MDP and the stochastic behavior of the environment as an MC. We assume a given a safety condition specified as a probabilistic reachability objective. Additionally, we assume a performance condition bounding the expected costs for reaching a certain goal. A significant problem is that the costs of certain actions are not known before they are executed. This calls for using reinforcement learning algorithms like Q-learning, where optimal strategies are obtained without prior knowledge about the system. While this is usually a suitable solution, in this case we have to ensure that no unsafe actions are taken during exploration to ensure an optimal and safe strategy.

The setting neither allows for using plain verification nor direct reinforcement learning. On one hand, verifying safety and performance properties – in the form of multi-objective model checking – is not possible because the costs of actions are not known. On the other hand, in practice learning means that the robot will explore parts of the system. In order to do that, we need to ensure that all unsafe behavior is avoided beforehand. Our solution to these problems is to use the notion of permissive schedulers. In contrast to standard schedulers, where for each system run the next action to take is fixed, more permissiveness is given in the sense that several actions are allowed. The first step is to compute a safe permissive scheduler which allows only safe behavior. The system is then restricted accordingly and therefore fit for safe exploration.

It would be desirable to compute a permissive scheduler which encompasses the set of all safe schedulers. Having this would ensure that via reinforcement learning a safe scheduler inducing optimal cost would be obtained. Unfortunately, there is no efficient representation of such a

maximal permissive scheduler. Therefore, we developed an iterative approach utilizing SMT-solving where a safe permissive scheduler is computed. Moreover, the computation can be done via mixed-integer linear programming (MILP). Out of this, reinforcement learning determines the locally optimal scheduler. In the next iteration, this scheduler is explicitly excluded and a new permissive scheduler is obtained. This is iterated until the performance criterion is satisfied or until the solution is determined to be globally optimal, which can be done using known lower bounds on the occurring costs.

An important goal of reinforcement learning (RL) is to make an agent learn to behave as desired through its experience. For problems modeled as stochastic games, the expectation of the discounted sum of rewards is commonly used as a performance criterion to encode the preference over strategies. It is well known that with discounted-sum objectives, pure memoryless strategies suffice for optimality, which significantly simplifies the learning algorithms. However, the discounted-sum objective suffers from several noticeable drawbacks in the task of describing the desired strategies. First, it is not applicable to strategies that require memory. As memoryless strategies are sufficient to achieve optimality, agents lack the incentive to learn the more complicated finite-memory strategies. Second, it cannot restrict behavior during the learning process. With rewards, an agent can only figure out preferable actions after it actually tries all of them, even fatal ones such as crashing into some obstacle, which is obviously unacceptable. Third, there is usually a lack of theoretical proof that any strategy solved with the given reward function is desirable, except in some simple scenarios. For example, multi-dimensional reward functions are generally necessary to represent the conjunction of several requirements, in which case a strategy usually cannot be simultaneously optimized with every single reward. It is hard to know intuitively from the reward function how different the learned strategy is from a desired one.

In order to compensate for these problems, in (Wen & Topcu, 2016) we used LTL specifications to complement the encoding of the desired strategies. In practice, it is relatively straightforward to extract LTL specifications from high-level task requirements in robot planning and control. Algorithmically, all LTL formulas can be transformed to deterministic Rabin or parity automata (DRA or DPA), which can be further used to construct product stochastic Rabin or parity games. Strategies synthesized for such product Rabin or parity games are guaranteed to satisfy the corresponding LTL specifications with probability one (i.e. almost surely), treating LTL specifications as “game rules” that should never be violated. Both the construction of DRA or DPA from LTL formulas and the synthesis can be performed using off-the-shelf tools. Although it has been shown that pure memoryless strategies suffice for almost-sure winning in the product stochastic Rabin or parity games, these strategies use memory in the original stochastic games. In this way, LTL specifications offer a systematic way of designing the memory for the desired strategies. We showed that with the pre-computation of almost-sure winning regions in the product games, we can keep the agent safe even during the learning procedure.

We used both discounted rewards and LTL specifications to encode task requirements. In particular, if an LTL specification is realizable and can be transformed into a deterministic Buchi automaton (DBA), we proved the existence of a memoryless strategy which is both almost-sure winning with respect to the Buchi objective and ε-optimal with respect to the discounted-sum objective. We also developed a probably approximately correct (PAC) algorithm to learn such a

strategy online when the reward function and the transition distributions are both unknown a priori.

3.5.2 Constrained Cross-Entropy Method for Safe Reinforcement Learning

In (Wen & Topcu, 2018) we studied a safe reinforcement learning problem in which the constraints are defined as the expected cost over finite-length trajectories. We proposed a constrained cross-entropy-based method to solve this problem. The method explicitly tracks its performance with respect to constraint satisfaction and thus is well-suited for safety-critical applications. We showed that the asymptotic behavior of the proposed algorithm could be almost-surely described by that of an ordinary differential equation. Then we gave sufficient conditions on the properties of this differential equation to guarantee the convergence of the proposed algorithm. At last, we showed with simulation experiments that the proposed algorithm could effectively learn feasible policies without assumptions on the feasibility of initial policies, even with non-Markovian objective functions and constraint functions.

3.6 Chance-constrained Finite-Time Optimal Control and Set-to-Set Reachability Computations

3.6.1 Voronoi-Partition-Based Scenario Reduction for Fast Sampling-Based Stochastic Reachability Computation

Reach-avoid analysis is an established verification tool for discrete-time stochastic dynamical systems, which provides probabilistic guarantees on safety and performance. In (Sartipizadeh & Acikmese, 2019), we focused on the finite-time horizon *terminal* hitting time stochastic reach-avoid problem (referred to here as the *terminal time problem*), that is, computation of the maximum probability of hitting a target set at the terminal time, while avoiding an unsafe set during all the preceding time steps using a controller that satisfies the specified control bounds.

The solution to the terminal time problem relies on grid-based dynamic programming and so lacks scalability. Researchers have proposed scalable approximations using approximate dynamic programming, Gaussian mixtures, particle filters, convex chance-constrained optimization, Fourier transform-based verification, Lagrangian approaches, and semi-definite programming. Currently, the largest system verified is a 40-dimensional chain of integrators using Fourier transform-based techniques. However, their high computational cost precludes their use in real-time applications.

In (Sartipizadeh & Acikmese, 2019), we reconsidered the sampling-based approach. Similar sampling-based approaches have been used successfully in robotics and in stochastic optimal control. In the sampling-based stochastic reach-avoid problem, we sample the stochastic disturbance to produce a finite set of *scenarios*, then formulate a mixed-integer linear program (MILP) to maximize the number of scenarios that satisfy the reach-avoid constraints. The approximated probability will converge to the true terminal time probability as the number of considered scenarios increases. However, the computational complexity of MILP increases exponentially with the number of scenarios, making the approach practically intractable when seeking high accuracy solutions.

The main contributions were two-fold. We first provided a lower bound on the number of scenarios needed to probabilistically guarantee a user-specified upper bound on the approximation error with a user-specified confidence level using concentration techniques. Using Hoeffding's inequality, we demonstrated that the number of scenarios that need to be considered was inversely proportional to the square of the desired upper bound on the estimate error. We proposed a Voronoi partition-based under-sampling technique that underapproximates the MILP-based solution in a computationally efficient manner. This approach allowed us to partially mitigate the exponential computational complexity and provided flexibility to the user to select the number of partitions based on the desired online computational complexity. We demonstrated the application of the proposed method in the problem of spacecraft rendezvous and docking.

3.6.2 Approximate Convex-Hull-Based Scenario Truncation for Chance-Constrained Trajectory Optimization

In (Sartipizadeh & Acikmese, 2019 (accepted)), we studied chance-constrained trajectory optimization of linear systems with general ellipsoidal and polytopic state-input constraints, where the constraints must be met with some prescribed confidence level. We used sampling techniques, specifically a scenario approach, due to their generality and tractability compared to analytical methods. To address the main drawback of the scenario approach, which may require a large number of samples, we introduced an approximate convex hull-based method to significantly reduce the number of samples. Based on the allowable computational complexity, the prominent samples were selected in a proper mapping and the rest were truncated. The truncation error was later compensated for by adjusting (buffering) the constraint set, so that the satisfaction of constraints with the desired confidence level could still be guaranteed. Simulation results confirmed the theoretical predictions with solid performance of the proposed method after discarding about 99% of samples from the scenario approach, which remarkably sped up the online computations.

3.6.3 Stochastic Motion Planning Using Successive Convexification and Probabilistic Occupancy Functions

In (Vinod, et al., 2018), we developed a method for real-time motion planning in stochastic, dynamic environments via a receding horizon framework that exploits computationally efficient algorithms for forward stochastic reachability analysis and non-convex optimization. Our method constructs a dynamically feasible trajectory for a robot, modeled as a linear time invariant (LTI) dynamical system, while ensuring 1) a desired probabilistic collision-avoidance guarantee is achieved, 2) state and control constraints are satisfied, and 3) a convex performance objective is minimized. We first compute "keep-out" regions at each time instant to ensure a probabilistic collision avoidance guarantee. These keep-out regions are convex and compact, and they can be tightly overapproximated by ellipsoids which may be computed in a grid-free, recursion-free, and sample-free manner. The regions are constraints in a non-convex optimization problem, solved via successive convexification. This algorithm uses interior point methods for real-time implementation.

4 RECOMMENDATIONS

Integration of autonomous systems at scale hinges on factors including (i) how capable they are in executing complicated missions in dynamic and a priori unknown environments and (ii) how we can establish trust that they will operate safely and correctly. These two factors are partly in conflict, since the former requires adapting to and learning new skills in unpredictable, dynamic environments. Even though the completed project – and other concurrent work – has taken initial steps, we still lack the means to formally characterize and systematically design for provably safe and correct behavior of such adapting and learning systems. The current practice based on testing is not sustainable due to the complexity of such systems, which stems in part from the excessive amount of operational and contingency scenarios they are expected to handle.

In a learning setting, the intent of the designer is typically encoded in a mathematical construct, called a reward function (a scalar-valued function over the relevant features, states, and inputs of the system). While it may be tempting to design the reward functions used in learning in order to "program" the system to operate in a desired manner, reward functions intrinsically lack the expressivity necessary to reason about complex requirements for safety-critical autonomous systems. Moreover, in their current form and given their current functionality, they are difficult for human designers and operators to work with as a way of reliably specifying tasks. Furthermore, reward values are often dependent on or specialized for particular environments. This dependence requires tedious adjustment for each new target environment, undermining the very purpose of learning. Finally, over-engineering the reward structures to encode complex requirements runs the risk of triggering negative side effects or hiding potential bugs that surface only when the learning agents are deployed into broader autonomous system operations.

While the conventional abstractions for learning have served well in static, isolated applications with significant domain understanding, they fall short in dynamical and adversarial settings. Additionally, the majority of DoD applications for autonomy are data-starved, which is in stark conflict with the prevailing desire of "delegating every need to data." Furthermore, existing learning abstractions and algorithms are often oblivious to the need for verifiability. We argue that an increasing success in learning without an explicit effort for verifiability will hinder its transition to real systems and limit its impact.

Consequently, we recommend support for an increasing amount of work at the interfaces of conventional disciplines and, in particular, in the intersection of control theory, formal methods, and learning theory.

5 REFERENCES

Ahmadi, M., Cubuktepe, M., Jansen, N. & Topcu, U., 2018. Verification of Uncertain POMDPs Using Barrier Certificates. *Proceedings of Allerton Conference on Communication, Control and Computing.*

Alur, R., Moarref, S. & Tocu, U., 2016. Compositional synthesis of reactive controllers for multi-agent system. *Proceedings of Computer-Aided Verification.*

Bharadwaj, S. et al., 2019. Synthesis of Minimum-Cost Shields for Multi-Agent Systems. *Proceedings of American Control Conference.*

Bharadwaj, S., Dimitrova, R. & Topcu, U., 2018. Distributed Synthesis of Surveillance Strategies for Mobile Sensors. *Proceedings of Conference on Decision and Control.*

Bharadwaj, S., Dimitrova, R. & Topcu, U., 2018. Synthesis of Surveillance Strategies via Belief Abstraction. *Proceedings of Conference on Decision and Control.*

Cubuktepe, M. et al., 2017. Sequential Convex Programming for the Efficient Verification of Parametric MDPs. *Proceedings of Tools and Algorithms for the Construction and Analysis of Systems.*

Cubuktepe, M. et al., 2018. Synthesis in pMDPs: A Tale of 1001 Parameters. *Proceedings of Automated Technology for Verification and Analysis.*

Dimitrova, R., Ghasemi, M. & Topcu, U., 2018. Maximum realizability for linear temporal logic specifications. *Proceedings of International Symposium on Automated Technology for Verification and Analysis.*

Feng, L., Humphrey, L., Lee, I. & Topcu, U., 2016. Human-interpretable diagnostic information for robotic planning systems. *Proceedings of International Conference on Intelligent Robots and Systems.*

Ghasemi, M., Feng, L., Chang, K.-W. & Topcu, U., 2018. Counterexamples for Robotic Planning Explained in Structured Language. *Proceedings of International Conference on Robotics and Automation.*

Humphrey, L., Konighofer, B. & Topcu, U., 2016. Synthesis of Admissible Shields. *Proceedings of Haifa Verification Conference.*

Jansen, N., Cubuktepe, M. & Topcu, U., 2017. Synthesis of shared control protocols with provable safety and performance guarantees. *Proceedings of American Control Conference.*

Junges, S. et al., 2016. Safety-Constrained Reinforcement Learning for MDPs. *Proceedings of Tools and Algorithms for the Construction and Analysis of Systems.*

Neider, D. & Topcu, U., 2016. An automaton learning approach to solving safety games over infinite graphs. *Proceedings of International Conference on Tools and Algorithms for the Construction and Analysis of Systems.*

Papusha, I., Fu, J., Topcu, U. & Murray, R., 2016. Automata theory meets approximate dynamic programming: Optimal control with temporal logic constraints. *Proceedings of Conference on Decision and Control.*

Sartipizadeh, H. & Acikmese, B., 2019 (accepted). Approximate Convex Hull Based Sscenario Truncation for Scenario Approach to Chance Constrained Trajectory Optimization. *Automatica.*

Sartipizadeh, H. & Acikmese, B., 2019. Voronoi Partition-based Scenario Reduction for Fast Samplig-based Stochastic Reachability COmputation of Linear Systems. *Proceedings of American Control Conference.*

Vinod, A. et al., 2018. Stochastic Motion Planning Using Successive Convexification and Probabilistic Occupancy Functions. *Proceedings of Conference on Decision and Control.*
Wen, M. & Topcu, U., 2016. Probably Approximately Correct Learning in Stochastic Games with Temporal Logic Specifications. *Proceedings of International Joint Conference on Artificial Intelligence.*
Wen, M. & Topcu, U., 2018. Constrained Cross-Entropy Method for Safe Reinforcement Learning. *Proceedings of Neural Information Processing Systems* .
Winterer, L. et al., 2019 (under review). Strategy Synthesis in POMDPs via Game-Based Abstractions. *Transactions on Automatic Control.*