Five Challenge Problems in Cyber-Physical Systems

Rahul Mangharam, Houssam Abbas, Madhur Behl, Kuk Jang, Zhihao Jiang, Matthew O’Kelly and Yash Vardhan Pant
School of Engineering and Applied Sciences
University of Pennsylvania
Philadelphia, Pennsylvania. USA 19104
Email: {rahulm, habbas, mbehl, jangkj, zhihaoj, mokelly, yashpant}@seas.penn.edu

Abstract—The tight coupling of computation, communication and control with physical systems such as actuation of closed-loop medical devices within the human body, peak power minimization by coordination of controllers across large industrial plants, and fast life-critical decision making by autonomous vehicles, present a set of fundamental and unique challenges. Each of these require new approaches at the intersection of multiple scientific, human and systems disciplines. We discuss five such challenges which require creative insights and application of model-based design, control systems, scheduling theory, formal methods, statistical machine learning and domain-specific experimentation. We ask the following questions: (1) An autonomous medical device is implanted to control your heart over a period of 5-7 years. How do you guarantee the software in the device provides safe and effective treatment under all physiological conditions? (2) Electricity prices in the US have summer peaks that are over 32x their average prices and winter peaks that are 86x. How can buildings respond to massive swings in energy prices at fast time scales? (3) While wireless has been successfully used for open-loop monitoring and tracking, how can we operate closed-loop control systems over a network of wireless controllers. Furthermore, how can we ensure robust, optimal and secure control in the presence of node/link failures and topology changes? (4) While autonomous vehicles have driven over a million miles, the most basic maneuvers, such as a lane change, are yet to be certified to be safe. How do we verify the safety of the decision controller in an autonomous system under all driving scenarios - in other words how can we provide a driver’s license to a self-driving vehicle? (5) In autonomous vehicles, when the computational resources of the hardware platform become overloaded, the estimation delay can compromise control performance and even stability. How can we co-design adaptive perception algorithms to operate with a fraction of the computing resources while ensuring the control stability and performance?

I. INTRODUCTION

We describe five challenge problems at the foundations of Cyber-Physical Systems (CPS). CPS are the new generation of time-critical and safety-critical Real-Time Embedded Systems where computation and communication are tightly coupled to control large, complex and “messy” plants. Unlike classical Real-Time Systems where the system is designed in a constrained manner to limit the complexity for predictability, in current CPS, the plants are difficult to model precisely because they are non-deterministic, interactive and often scale to thousands of controllers. We outline research approaches to address these structural concerns in the design of future Cyber-Physical Systems through closed loop modeling, architectures, algorithms and platforms. The eventual goal of efforts such as this is to develop CPS to transform how we interact with and manipulate the physical world, just as the Internet transformed how we interact with information systems.

A key aspect of this cross-cutting work is that each challenge bridges two or more well-established scientific fields such as scheduling theory, control systems, formal methods, statistical machine learning, and experimentation. We illustrate this through five domain-specific problems spanning the safety of closed-loop medical devices, control over wireless, data-driven control of energy systems in volatile price markets, safe decision making in autonomous systems and co-design of computation and control for future automotive systems. We address the foundations of CPS across four themes: Modeling, Architectures, Algorithms and Platforms.

- **CPS Modeling**: From verified models into verified code for life-critical systems. We focus on the formal modeling, synthesis and certification of high-confidence medical device software and systems. We describe efforts on both implantable medical devices and physiological control systems to ensure that both the functional and formal aspects are verified, validated and tested within the closed-loop context of their physiological systems.

- **CPS Architectures**: Distributed Control over Wireless Networks. We describe radical architectural approaches for completely in-network computation for robust, optimal and secure control over unreliable infrastructure.

- **CPS Algorithms**: Data-driven Controller Synthesis. In highly volatile electricity pricing markets, we required buildings and built environments to rapidly respond to peak pricing spikes. We describe a new data-driven approach for synthesis of control strategies to achieve real-time demand response while ensuring custom climate conditions are maintained.

- **CPS Platforms and Test-beds**: To certify autonomous vehicles are safe we describe the experimental challenges with verifying decision controllers and executing them across a range of driving scenarios. As the perception and computation workload increase in such systems, we describe new approaches for the co-design of anytime computation with robust control. This allows the system to trade-off accuracy for faster responses and lower power requirements while ensuring the stability and tracking performance of the vehicle controls.
II. THE DESIGN OF SAFE AND EFFECTIVE CLOSED-LOOP MEDICAL DEVICES

The design of safe, bug-free, and efficient medical device software is challenging, especially in implantable devices that directly control and actuate partially understood organs. However, such design is also essential; safety recalls of pacemakers and implantable cardioverter defibrillators between 1990 and 2000 affected over 600,000 devices [1]. Importantly, 41% of these recalls were due to software issues [2]. According to the US Food and Drug Administration (FDA), in 1996, 10% of all medical device recalls were caused by software-related issues. This percentage rose to an average of 15% of recalls from 2008 to 2012. And, surprisingly given these numbers, there is currently no formal methodology or open experimental platform to test the correct operation of medical device software within the closed-loop context of the patient [3].

Unlike other industries such as aviation and automotive where the safety concern is focused on a well-defined physical plant [4], [5], in the medical device domain patient response is complex and nondeterministic. As a result there are no well-established standards for the development of medical device software that directly control and actuate the patient. For device manufacturers, this has prompted recent interest in applying formal modeling [6], [7], [8] and verification techniques in medical devices software development [9], [10].

How do you guarantee the device software will not adversely affect the patient under all physiological conditions?

An effective software verification methodology is therefore needed for the risk analysis and certification of medical device software during the FDA’s pre-market submission phase. Testing medical device software currently is ad hoc, open loop, and very expensive [11], [12]. Test generation must be interactive and adaptive and must consider the current state of the patient when generating the next input in a way that advances the purpose of the test. The problem, therefore, becomes one of controller synthesis and cannot be addressed by an off-the-shelf model checker [13]. The key challenge is in the generation of physiologically relevant software that does not provide inappropriate therapy or adversely affect the patient. This requires validated patient models of the appropriate abstraction levels and that testing is conducted within the closed-loop context of the patient model.

Consequently, there is a need for fundamentally novel approaches to the closed-loop modeling, analysis, and design of medical cyber-physical systems, as well as the development of holistic, heterogeneous physiological models, approaches, and tools that address the many different physical, functional and logical aspects of the device and patient interaction. The scientific agenda of this research on medical cyber-physical systems is to unify theories of formal methods, control of physiological systems, communication and computing systems. Our early efforts are conducted with implantable medical devices, such as cardiac pacemakers and defibrillators, and physiological control systems such as infusion pumps with networks of discrete sub-systems. Both feature tight coupling

Fig. 1. Pacemaker operating in a closed-loop with the heart. The leads sense cardiac electrophysiological activity from inside the heart tissue (AS/VS = Atrial/Ventricular Sense event) and actuate the heart (AP/VP = Atrial/Ventricular Pacing event to maintain the heart rate. With the patient-in-the-loop, exposing safety and efficacy risks of autonomous and automatic control of the body.

1. Implantable Medical Devices - Cardiac Pacemakers: The human heart is perhaps the most important realtime system, generating electrical impulses that determine the heart’s rhythm and proper function. Irregularities with timing, i.e. cardiac arrhythmias, cause inefficient and unsafe function of the blood-oxygen system, necessitating the maintenance of the heart rate artificially. The cardiac pacemaker is a rhythm management device that maintains the minimum heart rate and synchrony between its chambers, thereby improving the condition of patients with cardiac arrhythmias. Cardiac rhythm management devices have grown in complexity with over 80,000 to 100,000 lines of code [14]. The primary approach to system-level testing of medical devices is unit testing using a playback of pre-recorded electrogram and electrocardiogram signals. This tests if the input signal triggers a particular response by the pacemaker but cannot evaluate if the response was appropriate for the patient condition. Furthermore, this approach of open loop “tape testing” is unable to check for safety violations due to inappropriate stimulus by the pacemaker. Pacemaker Mediated Tachycardia (PMT), a condition where the pacemaker inappropriately drives the intrinsic heart-rate toward the maximum rate, is a strong example of why we need an interactive and adaptive closed-loop verification and testing of such systems. With a tape test, PMT would not be observed and the response of the pacemaker could be classified as appropriate therapy.

Our proposed model-based design (MBD) for Medical CPS begins with developing integrated functional and formal heart

| Model Checking | Non-deterministic | UPPAAL Model |
| Interpolation | Deterministic | Stateflow Chart |
| Simulation | HDL Coder | Simulink Coder |
| Implementation | FPGA Implementation | C Code Implementation |

Fig. 2. From closed-loop verification to simulation-based testing, code generation and platform evaluation.
models that interact with real and modeled pacemakers for closed-loop verification and testing [15]. As shown from the top of Fig. 2, the heart-pacemaker closed-loop systems is first modeled abstractly to facilitate verification of the basic pacemaker design with maximum coverage [16]. In our case, we use timed automata [17], [18] and the UPPAAL model checker [19], [20], [21] at this design stage. Next, the models are translated to more detailed models that take into account the complex dynamics of the heart and interaction with more detailed pacemaker model [22], [23], [24]. We use Stateflow and Simulink [25], [26] at this design stage. These models are validated by physicians for their clinical relevance. The automatic model translation procedure, from UPPAAL to Stateflow, ensures that abstract models used for verification over-approximate the more detailed models used downstream [27]. Once the detailed models pass simulation-based testing with closed-loop dynamics, they are automatically generated into code and are subject to platform-level integration testing [28] as shown in Fig. 3. This MBD approach ensures the closed-loop safety properties are retained through the design toolchain and facilitates the development of verified software from verified models.

2. Physiological Control Systems: The understanding of closed-loop safety analysis of single-system devices such as pacemakers has naturally broadened our work to physiological control systems featuring multiple networked devices with the patient-in-the-loop [29]. Such a clinical scenario is viewed as a control system, in which the patient is the plant, bedside monitors are sensors and drug infusion pumps are actuators [6]. In this setup, caregivers traditionally perform the role of the controller. In many cases automatic controllers for drug infusion can reduce the burden on the caregiver and avoid human errors. However, vendors of medical equipment continue to avoid closed-loop scenarios due to an insufficient understanding of the human body’s response to treatment. Furthermore, a particular challenge arises from the complex interplay between the continuous dynamics of the patient’s reaction to treatment, and discrete controller and network. Consequently, there is a need for model driven safety analysis of closed-loop medical systems within uncertain parameters. Both the abstract, formal model and the detailed, informal model are needed in the process of verification, validation, and regulatory approval of closed-loop medical device systems.

Formal models allow us to exhaustively explore the possible behaviors of the system and prove its safety, detailed models allow us to use high-fidelity simulation that take real system dynamics into account. Both kinds of results can be used to make the case for regulatory approval if the abstract model is guaranteed to over-approximate the patient’s dynamics with respect to the control algorithms used.

3. Model-based Clinical Trials: Regulatory authorities require that the safety and efficacy of a new high-risk medical device be proven in a Clinical Trial (CT), in which the effects of the device on a group of patients are compared to the effects of the current standard of care. Phase III trials can run for several years, cost millions of dollars, and pose an inherent risk to the patients by exposing them to an unproven device. With the use of computational modeling and simulation, we would like to investigate how to use a large model-based synthetic group of patients and device models to improve the planning and execution of a CT so as to increase the chances of a successful trial.

As an example, we apply our initial efforts in applying it to a real CT that compares two algorithms within implantable cardioverter defibrillators (ICDs) for the detection of potentially fatal cardiac arrhythmias [30]. In 2011, a CT posited that one algorithm (Boston Scientific’s) would be better than the other (Medtronic’s) but the results of the trial were opposite to this hypothesis [31]. We begin by modeling the heart and processing 100’s of real patients’ electrogram signals, mapping the timing and morphology components of the signals to the heart model. This is followed by generating a population of
10,000+ synthetic heart models, by perturbing the parameters of the initial heart models for different arrhythmias, and implementing diagnostic algorithms of two very commonly used ICD platforms in the USA, i.e. Boston Scientific and Medtronic. We conducted conformance testing to validate our device models against real ICDs. Now, using the closed-loop of the device models and synthetic patient population we conducted multiple trials to compare the performance of the two algorithms to appropriately discriminate between potentially fatal ventricular tachycardias (VT) and non-fatal SupraVentricular Tachycardias (SVTs).

The results of our model-based clinical trials (MBCT) indicate we could have accurately predicted this with our model, i.e. that Boston Scientific’s algorithm was less able to discriminate between SVT and VT and so may lead to inappropriate therapy. We further demonstrated that the result continues to hold if we vary the characteristics of the synthetic population and device parameters. While MBCTs do not seek to replace a CT, they may provide early insight into the factors which affect the outcome at a fraction of the cost to replace a CT, they may provide early insight into the factors which affect the outcome at a fraction of the cost and duration and without the ethical issues. This effort is an early step towards using computer modeling as regulatory-grade evidence for medical device certification.

III. DATA-DRIVEN MODELING, CONTROL AND TOOLS FOR CYBER-PHYSICAL ENERGY SYSTEMS

In 2013, a report by the U.S. National Climate Assessment provided evidence that the most recent decade was the nation’s warmest on record [32] and 2015 is very likely to become the hottest year on record since the beginning of weather recording in 1880 [33]. Heat waves in summer and polar vortexes in winter are growing longer and pose increasing challenges to an already over-stressed electric grid.

Furthermore, with the increasing penetration of renewable generation, the electricity grid is also experiencing a shift from predictable and dispatchable electricity generation to variable and non-dispatchable generation. This adds another level of uncertainty and volatility to the electricity grid as the relative proportion of variable generation vs. traditional dispatchable generation increases. The volatility due to the mismatch between electricity generation and supply further leads to volatility in the wholesale price of electricity. For e.g., the polar vortex triggered extreme weather events in the U.S. in January 2014, which caused many electricity customers to experience increased costs. Parts of the Pennsylvania-New Jersey-Maryland (PJM) electricity grid experienced a 86 fold increase in the price of electricity from $31/MWh to $2,680/MWh in a matter of a few minutes [34]. Similarly, the summer price spiked 32 fold from an average of $25/MWh to $800/MWh in July of 2015. Energy industry experts are now considering the concept that extreme weather, more renewables and resulting electricity price volatility, could become the new norm.

Across the United States, electric utilities and independent service operators (ISOs or grid coordinators) are devoting increasing attention and resources to demand response (DR) [35]. It is considered as a reliable means of mitigating the uncertainty and volatility of renewable generation and extreme weather conditions and improving the grid’s efficiency and reliability. The potential demand response resource contribution from all U.S. demand response programs is estimated to be nearly 72,000 megawatts (MW), or about 9.2 percent of U.S. peak demand [36] making DR the largest virtual generator in the U.S. national grid. The annual revenue to end-users from DR markets with PJM ISO alone is more than $700 million [37]. Global DR revenue is expected to reach nearly $40 billion from 2014 through 2023 [38].

In order to shield themselves from the volatility and risk of high prices, such consumers must be more flexible in their electricity demand. Consequently, large commercial, industrial and institutional customers are increasingly looking to demand response programs to help manage their electricity costs. As shown in Fig. 5, DR programs involve a voluntary response of a building to a price signal or a load curtailment request from the utility or the curtailment service provider. Upon successfully meeting the required curtailment level the end-users are financially rewarded, but may also incur penalties for under-performing and not meeting a required level of load curtailment.

A. Challenges

On the surface demand response may seem simple. Reduce your power when asked to and get paid. However, in practice, one of the biggest challenges with end-user demand response for large scale consumers of electricity is the following: Upon receiving the notification for a DR event, what actions must the end-user take in order to achieve an adequate and a sustained DR curtailment? This is a hard question to answer because of the following reasons:

1. **Modeling complexity and heterogeneity:** Unlike the automobile or the aircraft industry, each building is designed and used in a different way and therefore, it must be uniquely modeled. The user expertise, time, and associated sensor costs required to develop a model of a single building (e.g., with EnergyPlus [39]) is very high [40] - often taking 5-12 months to model and tune the model. This is because usually a building modeling domain expert requires the geometry of a building from the building design and equipment layout plans, detailed information about material properties, and of equipment and operational schedules.
2. Control complexity and scalability: Upon receiving a notification for a DR event, the building’s facilities manager must determine an appropriate DR strategy to achieve the required load curtailment. These control strategies can include adjusting zone temperature set-points, supply air temperature and chilled water temperature set-point, dimming or turning off lights, decreasing duct static pressure set-points and restricting the supply fan operation etc. For a large building, it is difficult to assess the effect of one control action on other sub-systems and on the building’s overall power consumption because the building sub-systems are tightly coupled. Therefore, it is extremely difficult for a human operator to accurately gauge the building’s or a campus’s response.

3. Interpretability of modeling and control: Predictive models for buildings, regardless how sophisticated, lose their effectiveness unless they can be interpreted by human experts and facilities managers in the field. For e.g., artificial neural networks (ANN) obscure physical control knobs and interactions and hence, are difficult to interpret by building facilities managers. Therefore, the required solution must be transparent, human centric and highly interpretable.

B. Real-Time Data-driven Demand Response

We have developed a method called DR-Advisor (Demand Response-Advisor) [41], which acts as a recommender system for the building’s facilities manager and provides the power consumption prediction and control actions for meeting the required load curtailment and maximizing the economic reward. Using historical meter and weather data along with set-point and schedule information, DR-Advisor builds a family of interpretable regression trees to learn non-parametric data-driven models for predicting the power consumption of the building (Figure 8). DR-Advisor can be used for real-time demand response baseline prediction, strategy evaluation and, most importantly, for control synthesis, without having to learn first principles based models of the building:

1. DR Baseline Prediction: A baseline is an estimate of the electricity that would have been consumed by a customer in the absence of a demand response event. Typical demand response programs rely upon financial incentive for customers based on the extent to which they reduce their energy consumption and therefore require a reliable system to measure the energy reduction. For this reason the measurement and verification of demand response is the most critical component of any DR program. Using regression trees based approaches, DR-Advisor achieves a prediction accuracy of 93% to 98.9% for baseline estimates of eight buildings on the Penn campus by just analyzing data as shown in Figure 8.

2. DR Strategy Evaluation: A DR strategy refers to what sequence of control actions, and at what times, a system (lighting, HVAC or plug loads) will actuate. Furthermore, there could be several of such fixed DR strategies, but only one specific strategy can be used at a time. This brings us to our question, how can we choose good DR strategies from a predetermined set of strategies? Instead of predicting the baseline power consumption $Y^\text{base}$, in this case we want the ability to predict the actual response of the building $Y^\text{b}W$ due to any given strategy. At the beginning of the DR event we use the auto-regressive tree for predicting the response of the building due to each rule-based strategy and choose the one which performs the best over the predicted horizon. The prediction and strategy evaluation is re-computed periodically throughout the event.

3. DR Control Synthesis: While black-box and data-driven machine learning approaches are suitable for prediction, our primary contribution is making them capable of controller synthesis. Unlike rule-base DR, which does not account for building state and external factors, in DR synthesis the optimal control actions are derived based on the current state of the building, forecast of outside weather and electricity prices. We introduce a novel model based control with regression trees (mbCRT) algorithm to enable control with regression trees for real-time DR synthesis. Using the mbCRT algorithm, we can optimally trade off thermal comfort inside the building against the amount of load curtailment. While regression trees are a popular choice for prediction based models, this is the first time regression tree based algorithms have been used for controller synthesis with applications in demand response. Our synthesis algorithm outperforms rule based DR strategy by 17% (based on guidelines from Siemens) while maintaining bounds on thermal comfort inside the building.

We have evaluated the performance of DR-Advisor using a mix of real data from 8 buildings (1.2 million square feet) on the campus of the University of Pennsylvania, a large office building in Philadelphia and data-sets from a virtual building test-bed for the Department of Energy’s (DoE) large commercial reference building. We also compared the performance of DR-Advisor against other data-driven methods using a bench-marking data-set from AHRAE’s great energy predictor shootout challenge [42] and rank second. The first place was achieved by the use of neural networks which are not interpretable, while DR-Advisor with regression trees is highly interpretable by facilities managers and provides them both guidance and provenance for decisions made to control their infrastructure.
IV. Closed-loop Control over Wireless Networks

The current generation of embedded wireless systems has largely focused on open-loop sensing and monitoring applications. To address actuation in closed-loop wireless control systems, there is a strong need to re-think the communication architectures and protocols for reliability, coordination and control [43]. Wireless networked control systems, or Networked Cyber-Physical Systems (Networked-CPS), fundamentally differ from standard distributed systems in that the dynamics of the network (variable channel capacity, probabilistic connectivity, topological changes, node and link failures) can change the operating points and physical dynamics of the closed-loop system [44], [45]. The most important objective of control in Networked-CPS is to provide stability of the closed-loop system. It is therefore necessary for the network (along with its interfaces to sensors and actuators) to be able to provide some form of guarantee of the control system's stability in the face of the non-idealities of the wireless links and the communication constraints of the wireless swarm network. A secondary goal in Networked-CPS is to allow for composition of additional controllers and plants within the same network without requiring reconfiguration of the entire network operation.

The most common approach to incorporating Networked-CPS into the feedback loop is to use it primarily as a communication medium: the nodes in the network simply route information to and from one or more dedicated controllers, which are usually specialized CPUs capable of performing computationally expensive procedures (see Fig. 7(b)). The use of dedicated controllers imposes a routing requirement along one or more fixed paths through the network, which must meet the stability constraints, encapsulated by end-to-end delay requirements [46], [47]. However, this assignment of routes is a static setup, which commonly requires global reorganization for changes in the underlying topology, node population and wireless link capacities.

Routing couples the communication, computation and control problems [48], [49], [50]. Therefore, when a new route is required due to topological changes, the computation and control configurations must also be recalculated. Merely inserting a WNCS into the standard network architecture “sensor → channel → controller/estimator → channel → actuator” requires the addition of significant software support [47], [51], as the overhead of completely recomputing the computation and control configurations, due to topological changes or packet drops, is too expensive and does not scale.

While providing a review of classical and recent approaches for control over wireless networks, we present two complementary approaches on maintaining stability in the presence of environment and network disturbances. The first approach adopts a “computer systems” perspective on the design of robust architectures for embedded wireless control and actuation. We call this scheme Embedded Virtual Machines (see Fig. 8) which provides software mechanisms to decouple controller functionality from the physical node – thus providing resilience to node, link and topology changes. The second approach adopts a “control theoretic” perspective on distributed control within the network (see Fig. 7(c)). This provides control mechanisms to remove controller functionality from a dedicated node to all nodes in the network - thus eliminating the need for routing and guaranteeing stability and optimal control in the presence of link, node and topology changes.

A. Embedded Virtual Machines

Current approaches for robust networked control [45] require the underlying network to satisfy a minimal set of requirements (e.g. guaranteed packet delivery rate, upper bound on network induced delay) and reduce the network model to that of a single channel with random delays. In addition, they do not address the spatial aspects of the network, i.e., how changes in the network topology affect the closed-loop system performance.

As the links, nodes and topology of wireless systems are inherently unreliable, such time-critical and safety-critical applications require programming abstractions where the tasks are assigned to the sensors, actuators and controllers as a single component, rather than statically mapping a set of tasks to a specific physical node at design time (as shown in Fig. 8). Such wireless controller grids are composed of many nodes that share a common sense of the control application but without regard to physical node boundaries. Our approach, is to decouple the functionality (i.e., tasks) from the inherently
unreliable physical substrate (i.e., nodes) and allows tasks to migrate/adapt (Fig. 8(d)) to changes in the topology.

To this end, we introduced the Embedded Virtual Machine (EVM), a powerful and flexible programming abstraction where a Virtual Component (VC) and its properties are maintained across node boundaries [47], [52], as shown in Fig. 2(c). EVMs differ from classical system virtual machines. In the enterprise or on PCs, one (powerful) physical machine may be partitioned to host multiple virtual machines for higher resource utilization. On the other hand, in the embedded domain, an EVM is composed across multiple physical nodes with the goal to maintain correct and high-fidelity operation even under changes in the physical composition of the network. The goal of the EVM is to maintain a set of functional invariants, such as a control law and para-functional invariants such as timeliness constraints, fault tolerance and safety standards across a set of controllers given the spatio-temporal changes in the physical network. Thus, the EVM introduces new degrees of freedom, task migration and routing which facilitates, at runtime, the network configuration (operating point, conditions) to meet the requirements of the networked control algorithms. However, the EVM does not provide explicit guarantees but only finds the optimal operation configuration in terms of routing and task assignment.

B. Distributed Control over Wireless Networks

Let us consider the problem of stabilizing a plant with a multi-hop network of resource constrained wireless nodes. In [53], we introduced the concept of a Wireless Control Network (WCN), which is a paradigm change for distributed control over a wireless network. In a WCN the entire network itself acts as a controller, as the computation is spread over the whole network, instead of assigning a particular node with the execution of the control procedure. We have devised a numerical design procedure that produces the coefficients of the linear combinations for each node and actuator to apply in order to stabilize the plant. The radio connectivity between nodes in the network induces topological constraints to the control algorithm, and this topology determines whether it is even possible to stabilize the system with the use of linear iterative strategies. In addition, a method to synthesize an optimal WCN, with respect to the standard cost functions, has been developed [54].

Given the fundamental unreliability of wireless communication, the WCN method handles topological constraints while maintaining mean square stability for packet drop rates up to 20% for a specific network topology and plant. This bridges the gap between the basic WCN and the theoretical upper bound of robustness to packet drops [55]. We also show a method to synthesize a WCN robust to a certain level of node failures, and then extend the synthesis procedures to allow for the use of the WCN for control of continuous-time plants.

While in the past efforts, we consider scenarios where the network topology is already set, in recent efforts [56], [57] we have investigated a dual problem, “how to synthesize the network so that a stable WCN configuration exists?” The topological conditions from [56], along with the results from [53] provide the essential building blocks for an integrated decentralized wireless control network design framework. Early experiments in an industrial process control case study of a distillation column in a process-in-the-loop testbed demonstrate optimal control of continuous-time physical processes which maintain system stability under the presence of node and link failures.

1) Advantages of the WCN: The WCN introduces very low communication and computation overhead. The linear iterative runtime procedure is computationally very inexpensive as each node only computes a linear combination of its value and values of its neighbors. This makes it suitable for resource constrained, low-power wireless nodes (e.g., Tmote). Furthermore, the communication overhead is also very small, as each node needs to transmit only its own state once per frame. In the case when a node maintains a scalar state it transmits only 2 bytes in each message, making it suitable to combine this scheme with periodic message transmissions in existing wireless systems.

The WCN utilizes a simple transmission schedule where each node is active only once during a TDMA cycle and the control-loop does not impose end-to-end delay requirements. This allows the network operator to decouple the computation schedule from the communication schedule, which significantly simplifies closed-loop system design and enables compositional design and analysis.
V. VERIFICATION OF AUTONOMOUS VEHICLE PLANNING

Each year there are an average of 1.24 million traffic fatalities around the world [58]. Estimates indicate that more than 90 percent of all accidents are due to driver error [58]. Competent autonomous vehicles (AVs) could drastically reduce the occurrence of such incidents, but a major question first needs be answered: how can we judge when an AV is ready to graduate from research laboratories to public roads? One thing is clear: the public will want significant evidence that AVs are indeed safe [59]. This raises significant ethical and legal questions about how the AV should behave, and technical questions about how to verify that it will always behave the way its designers intended.

How do you give a self-driving car a driver’s license?

Prototype AVs have driven millions of miles and are even being approved for testing on public roads in some states [60]; however, manufacturers cannot verify (i.e., guarantee) the safety of even the simplest of scenarios in the presence of other dynamic traffic participants. Compounding, the difficulty of vehicle certification, vehicle manufacturers such as Tesla are transitioning to frequent over-the-air software updates. Such practice eschews conventional vehicle development technique and greatly increases pressure on developers to deliver correct software at a rapid pace.

The net result of the growing pressure to release AVs is a wide gap between current regulations and our technological capabilities. Human drivers are not ‘certified’ to act safely in all situations, in fact, we know they don’t; however, they assume liability for their actions. Who is liable for the behavior of an AV? Currently, it appears that manufacturers will take one of two approaches: (1) assume liability for the actions of the vehicle and self-insure [61] or (2) force the human occupants of an AV to make all critical decisions and shift liability to the pilot [62]. In either case, even if AVs reduce accidents by 99 percent, it is likely that the 1 percent of remaining accidents will invariably spawn a myriad of legal actions against both manufacturers [63] and vehicle owners. If the legal question is not answered, these risks could stifle the development of the AV market.

Thus, regardless of where legal liability falls it is clear that we first need new, practical methods for verification and validation of the decision engines of each AV (as shown in Fig. 9). Furthermore, verification must be automatic, exhaustive, and expedient for clearly defined scenarios. Secondly, ethical considerations still underlie both the design and verification of AVs: how does the safety of the car’s passengers weigh against that of people in its environment? Whose morals are embedded in the decision engines of an AV?

A. New and Old Challenges

There are several fundamental safety challenges in verifying and implementing autonomous vehicles, including:

1) As shown in Fig. 10, in a given driving scenario, the vehicles planning controller must determine all possible pathways in the road and provide the supervisory controller with the safest trajectory while interacting with other agents (vehicles, pedestrians, bicyclists), obeying flow control infrastructure (stop signs, traffic lights, environment conditions) and adhering to traffic rules;

2) While the target vehicle is controllable, the state evolutions of other agents are uncertain and the target vehicle must continuously adapt its plan as the scenario evolves;

3) At any time, the driver may interject to take over control of the vehicle.

Some of the problems facing would-be AV manufacturers in 2016 are very different than those outlined by the teams in the 2007 DARPA Urban Challenge [64]. One problem highlighted by the first ever crash between AVs at the Urban Challenge was the difficulty of accurately predicting the behavior of other vehicles in a dynamic environment.

Thus, it is clear that new methods for verification and validation of autonomous vehicle decision engines are needed. These methods should be automatic, exhaustive, and expedient for clearly defined scenarios.
Challenge [65] is that there are no known “formal methods that would allow definitive statements about the completeness or correctness of a vehicle interacting with a static environment, much less a dynamic one” [66]. Today one must still consider the massive configuration space of each individual snapshot of the day-to-day life of an AV if verification is to be attempted. For example, in order to test the interaction between two 7 DOF AVs requires $10^{14}$ simulations for only 10 samples from each state. If each test takes 10 seconds, the resulting set of simulations will take 30 million years to complete. Furthermore, within a given scenario, errors in localization, sensing, and actuation imply we never know the state of the world exactly. Small state estimation errors can befuddle motion planning algorithms and mean a difference between collision and safety in tight spots. As many teams in the Urban Challenge noted, a means of verifying the safety of autonomous driving is paramount to putting self-driving cars in the hands of the public [66].

B. APEX: Autonomous vehicle Plan verification & EXecution

The APEX tool represents a new approach to solving both the problems of the Urban Challenge and investigating rare events encountered only through on road driving and testing. Unlike other tools capable of verifying controllers for hybrid systems we require almost no abstraction. Most current approaches look at only the behavioral layer and assume perfect implementation of plans at the motion planning layer. Instead, as shown in Fig. 11, we propose that the behavioral layer is used to generate sequences of problems to be investigated at the level of motion planning and trajectory tracking. Thus, APEX addresses the safety verification issue by leveraging new results in hybrid systems and reachability analysis [67] to convert a brute force search over real intervals (which is intractable) into a set of finite sequences of bounded reachability problems.

With APEX, we capture the output of trajectory generation and optimization based methods outside of the formal model of the system. Using such information, we generate a sequence of verification problems by running the trajectory through realistic vehicle dynamics and low level controls. The software which runs on the vehicle is used directly for verification with the Robot Operating System (ROS). If a rare event is encountered and recorded by a real vehicle and a new rule or controller is added to the AV it is imperative that the manufacturer have high confidence that the modification will not induce new errors which are not evident in a single trace. APEX can make the most of rare events and scenarios; given a template we could find every possible instantiation and prove it is impossible for solution to make a decision leading to crash.

The key features of our approach are: (1) Use of realistic planning software which runs on actual vehicles. (2) Modular vehicle model construction which can easily be replaced when new algorithms or alternate vehicle dynamics are necessary. (3) Non-conservative evaluation of safety and liveness at the level of the ego-vehicles actual spatial-temporal evolution.

Thus, verification of realistic scenarios under all possible configurations over a length of 30-60 seconds is entirely feasible. Using such an approach a manufacturer, regulator, or insurer can begin to build a library of scenarios on which to test new vehicle software or updates made to the behavioral layer made through a reinforcement learner.

C. Open Challenges for Autonomous Vehicle Safety

1. Scenario Composition for Safe Journeys

Safety certification will seek to verify that a complete mission, made up of several independently verified scenarios, can be composed to ensure the safety across a journey (i.e. sequence of scenarios). The greatest challenge will be ensuring that the transitions between scenarios will result in safe initial conditions. As not all interactions are observable or captured in formal models, work also needs to be performed on the integration of informal assurance case arguments in a structured manner within APEX. This will also assist in evaluating cooperative and collaborative interactions with the human-in-the-loop.

2. Conformance-based Controller testing within the scenario context

A framework for black-box conformance testing of autonomous vehicle controllers within the context of one or more scenarios is essential for the creation of high confidence code from verified systems and scenarios.

3. Property hierarchy for unrealizable specifications

When a human operator is faced with an unrealizable specification it is common for the driver to knowingly violate a lesser priority safety property i.e. the driver may swerve on to the shoulder lane to avoid a sudden braking vehicle in front. In the context of autonomous driving in general, one can consider that many unrealizable scenarios might arise due to the initialization of the environment. Thus, for each scenario we need to assign priorities to properties such that lower priority properties may be violated to ensure higher priority properties are maintained under extreme scenarios.

VI. Co-DESIGN OF COMPUTATION AND CONTROL FOR AUTONOMOUS SYSTEMS

A. Motivation

Real-time control of autonomous vehicles requires the processing of a large amount of sensor data from cameras, LI-
DAR, radars, encoders, inertial sensors and possibly information communicated by other vehicles or the road infrastructure. This data is used by the vehicle to determine its position in the world and that of neighboring obstacles, which we refer to as state estimation, and to calculate its next move. Google’s autonomous vehicles generate over 750MB/s of sensor data [68] which must be processed by the perception pipeline fast enough for timely control decisions. The perception pipeline uses run-to-completion algorithms: such algorithms must always meet a pre-defined termination criterion before returning a useful answer. If interrupted early, they don’t return anything useful to the controllers. Typical algorithms in a perception pipeline span tasks like edge detection, filtering, segmentation, object recognition and tracking, and sensor fusion.

Perception algorithms have variable execution times because their runtimes depend on the current context. E.g., a corner detector will take longer to detect all the corners in a rich scene than in a flat homogeneous scene (at a given processor frequency). Because of this variability, the hardware running these run-to-completion algorithms is over-engineered for the worst-case execution time. That is, the system is equipped with hardware that can always execute the software within a certain deadline. When the current context causes the perception pipeline to perform the most operations (e.g., the corner detector has to find many corners), this is appropriate. However, such worst-case situation rarely holds in practice. The rest of the time, the perception pipeline has fewer operations to perform within the deadline, so a slower (and less power-hungry) hardware execution would be appropriate. The downside with this over-engineering is unnecessarily large power consumption for the average case. For example, in many research autonomous vehicles, this leads to a requirement of up to 4KW in computational power [69]. This is a significant drain on the vehicle’s lithium-ion battery capacity, which is 24kWh in a Nissan Leaf, for example. It is also a significant compared to the average power drawn by the drive motors, which is around 3KW in a small electric vehicle modeled in ADVISOR [70] going through the Urban Dynamometer Drive Cycle.

In addition to the high power consumption, the rigid nature of the run-to-completion algorithms used for perception and state estimation restricts the range of maneuvers that autonomous systems can do, e.g. autonomous cars are not yet capable of fast or aggressive driving. Reacting faster to a dynamic scene requires a perception pipeline (and state estimation) that can deliver answers faster, even if the cost is slightly worsened quality of output.

B. Challenge

The challenge is to enable less power-hungry autonomous vehicles that still maintain the control performance of current systems. Note that the emphasis is on control performance, and not on the output quality of the perception algorithm. While in general, the two are positively correlated (better output quality means better control performance) there are situations where this is not the case. For example, if the lower quality output is produced faster, allowing a faster update of the control input.

From the above discussion it can be seen that we ought to design control algorithms that recognize how soon an update is needed from the perception and state estimation (P&SE) pipeline. Correspondingly, we should also design P&SE algorithms that can provide useful answers even when interrupted before their run-to-completion criterion has been met. The main challenges in such a system are:

- Enabling common perception algorithms to provide useful (if lower quality) answers early. This is a challenging task for any given algorithm as they are most often designed and implemented for maximum performance without consideration for the possibility of interruption.
- Development of control/scheduling algorithms that are aware of when to request faster updates for the optimal operation of the closed loop system. Existing work models the effect of delays on control and designs controllers that operate well despite the delay. Here we are suggesting that rather than view the delays as purely adversarial, the controller should be aware of the range of trade-offs afforded it by the P&SE algorithms and exploit these trade-offs for lower-power control.

C. Contract-based Co-design of Anytime Computation and Robust Control

Anytime algorithms [71] are a class of algorithms that can be interrupted at any point during their execution and still return a usable solution, usually one whose quality is monotonically improving with time. Contract algorithms [72] are a version of anytime algorithms where the interruption time is chosen from a set of pre-defined times. In our recent work [73], [74], we show how to convert off-the-shelf P&SE

---

**Fig. 12.** A perception algorithm for object detection is decomposed into three tasks, and various components used to compose the contract based perception algorithm and their representation as real-time tasks. For a given δ contract, knob settings are chosen at run-time resulting in a schedule to execute these sequential components, or tasks, to respect the contract.
contract-based perception-and-estimation algorithm, a robust control
can set contracts (deadlines) for the perception/estimation algorithm. Based on the contract
and the offline profiling, the contract based estimator decides at run-time what tasks to execute in order to best meet the contract.

algorithms into contract algorithms, and how to design a controller that takes advantage of this capability.

First, the PS&E is profiled off-line. Namely, the algorithm’s parameters are varied and for each setting of the parameters, it is run on a profiling data set. This yields a finite set of contracts: a contract is a pair (estimation delay, estimation error), and each setting of the parameters yields one contract when run on the profiling data set Fig. 12 shows this approach applied to an object detection algorithm. At run-time, when the estimator receives a (delay, error) contract request from the controller, it can adapt its execution paths to respect the contract, namely, to provide a state estimate within the requested error bound, and within the requested delay. Another dimension of decision making, explored in [74], is resource allocation for execution of tasks, e.g. whether the task gets executed on the CPU or the GPU, resulting in a power versus computation delay trade-off.

The controller is designed with the knowledge of the contracts that can be realized by the P&SE algorithm, and at each time step will request a contract. This gives the controller the ability to leverage the flexible nature of the estimation algorithm to maximize some measure of control performance. In [73] we develop a Robust Model Predictive Control algorithm that can leverage this trade-off offered by the contract based co-design (see Fig. 13 while optimizing for a joint cost of control performance and computation power consumption. We experimentally evaluated our approach on a hex-rotor with visual odometry and showed improved control performance and computation energy efficiency over a baseline MPC controller that does not leverage co-design.

Fig. 13 shows the closed loop architecture in a system with co-design of the anytime computation (estimator) and robust controller. In the co-designed system as presented in this paper, the controller can make the estimation algorithm switch to lower or higher time (and/or energy) consuming modes based on the control objective at the current time step. The main components of the co-design architecture are a contract based perception-and-estimation algorithm, a robust control algorithm that computes an input to be sent to the physical system being controller as well as the operating mode for the contract time estimator, and the interface between them. More details on these components are in the following sections.

REFERENCES
