Certifiable and Efficient Autonomous Cyber-Physical Systems Design

Shengjie Xu[1], Clara Hobbs[1], Bineet Ghosh[2], Parasara Sridhar Duggirala[1], and Samarjit Chakraborty[1]
[1]The University of North Carolina at Chapel Hill, USA
[2]The University of Alabama, Tuscaloosa, USA

***Abstract*—The ability to certify the correctness of cyber-physical systems often rely on provisioning resources to account for worst-case behaviors. For example, timing certification necessitates that all software tasks are scheduled to be able to meet their deadlines. However, since the execution times of tasks have wide variances—and they continue to increase with growing software and processor complexity—provisioning resources for the worst case can be very pessimistic and also expensive. In this paper we outline some of our recent efforts to address this problem, and emerging techniques to ensure certification of autonomous cyber-physical systems, while ensuring their efficient implementation.**

***Index Terms*—Cyber-Physical Systems, Real-Time Systems, Safety, Controller Synthesis**

I. Introduction

Certifying the correctness of autonomous cyber-physical systems (CPS) [1], [2] is posing a major challenge in the widespread adoption of technologies like autonomous vehicles and robots. Here, most of the research literature has focused on the formal verification of control strategies [3]–[7] that implement the algorithmic core of many autonomous systems. More recently, the verification and testing of machine learning (ML) algorithms used for sensor and perception processing is also being actively pursued in the context of CPS design [8], [9].

However, a different form of challenge arises in closing the "*model-implementation gap*" [10]–[13] between the assumptions made when designing control strategies, versus the realities of modern distributed implementation platforms [14]. For example, as implementation platforms become more complex and distributed, the timing assumptions made during controller design [15], [16] are increasingly difficult to ensure, or even verify [17]. Even basic timing analysis tasks such as estimating safe *and* tight *worst case execution time* (WCET) estimates of software tasks is becoming a losing proposition [18], [19]. For WCET estimates to be safe, they are increasingly overestimated. Meeting all task deadlines with such overestimated WCET values leads to pessimistic or infeasible implementations. Further, in domains like automotive, in-vehicle architectures are rapidly moving away from "one function per ECU" or federated, to multiple functions sharing resources, *viz.*, "integrated" architectures [20]. The clear trend is that future architectures will be less "static" than before, as indicated by developments like AUTOSAR Adaptive [21] and service-oriented paradigms [22], [23]. As a result, the timing non-determinism experienced by control software running on such implementation platforms will only continue to increase. But in traditional design processes control engineers assume certain timing properties or deadlines that the control tasks need to satisfy for them to behave as desired, and the embedded systems engineers schedule them to meet those deadlines [24]. This ensures a clean separation of concerns, allowing the two groups of engineers to work independently.

System certification with unreliable components

As outlined above, such a design flow where system components are designed assuming certain deterministic reliability guarantees from the other components is increasingly breaking down in autonomous CPS design. This is not only the case with timing uncertainties, but is a broader design challenge. For example, when using ML components for state estimation — as it occurs when processing camera, radar or lidar data in autonomous cars — the results cannot be assumed to be perfectly correct. Similarly, 100% security [25] cannot be assumed at moderate cost during data transmission from the plant to the controller or the controller to the actuator. In all of these cases, ensuring 100% reliability comes at the expense of prohibitive cost or excessive design pessimism. Hence, an important question is: *Is it possible to certify system safety of autonomous CPS, while ensuring efficiency?*

In this paper we claim that it is possible to answer this question in the affirmative. Towards this we outline some of our recent work [26]–[28] on certifying system safety under implementation platform timing uncertainties. We discuss some related work in the area, and argue how such certification may be extended to cases beyond timing uncertainty.

Paper outline

In the next section we show how feedback control systems can be certified for system safety even when they experience uncertain timing behavior on an implementation platform. After this, we outline how to *synthesize* "imperfect" schedules that lead to better resource utilization but can nevertheless guarantee system-level safety. Finally, we list a broad class of work on safety and certifiability in CPS, before concluding the paper with an outline of our broad research vision. While we outline this vision in the special case of timing uncertainties, we argue that it extends to many other aspects of system design that are relevant for safety certification and efficiency.

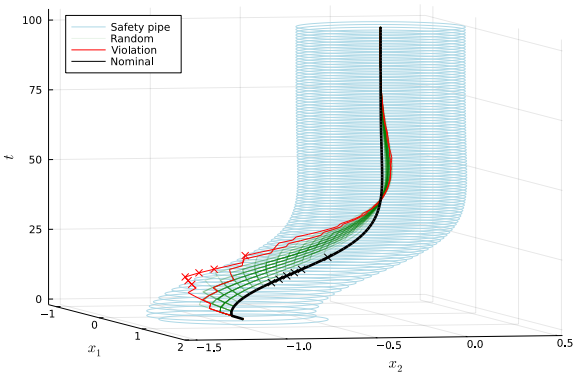

Figure 1: Safe & unsafe behaviors due to deadline misses [26].

II. Safety Certification with Timing Uncertainties

Instead of assuming that the platform implementing control software has an ideal timing behavior, *viz.,* no deadline misses, can we certify system safety when the control task misses deadlines [29]? The underlying idea is to first define a notion of *safety*. Towards this, we construct a *safety pipe* around the system trajectory in the state space that captures the *ideal* dynamics of the closed-loop system. Such an ideal dynamics or trajectory is when the system experiences an ideal timing behavior, *e.g.*, no deadline misses. In the presence of deadline misses, the system's dynamics will deviate from the ideal trajectory, and the extent of the deviation will generally correspond to the number of deadline misses the control task experiences. But as long as the deviation is not significant, and in particular is contained within the safety pipe, the system can be certified to be safe. The diameter or shape of the safety pipe can be specified based on the desired notion of safety.

This notion of system safety is illustrated in Figure 1 using the model of an automotive electric steering system from [26]. It shows the evolution of the closed-loop feedback control system in the (x_1, x_2) state space. Here, the solid black line shows the ideal system trajectory, where the control task always meets its deadline. The light blue envelope/spiral around it is the safety margin, or the set of potential system behaviors that can be certified to be safe. Finally, the trajectories in red and green are those arising when the control task experiences deadline misses, and because of which its dynamics deviates from the ideal behavior (the black line). The red trajectories violate the specified safety property since they go out of the safety envelope, and where the violations occur have been marked with "$\times$". The corresponding time instants have been marked with a black $\times$ on the nominal trajectory, with the distance between a black $\times$ and its corresponding red $\times$ exceeding the maximum allowed deviation.

This figure shows that in contrast to the prevalent practice of associating safety with *meeting all deadlines*, we (i) lift the notion of safety to system-level behaviors, and (ii) slightly relax the notion of safety to include multiple system trajectories rather than allowing a single behavior. With this revised notion of safety, we see that safety certification is possible even without a perfect timing behavior of the implementation platform, thereby mitigating some of the obstacles outlined in Section I. But the main open question is: *Which platform timing behaviors or deadline hit/miss patterns satisfy a given safety specification?*

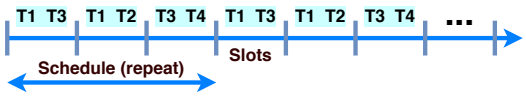

Figure 2: Scheduling control tasks with deadline misses.

Towards answering this, note that the dynamics of the system depend on both — the deadline hit/miss pattern [30], and how deadline misses are handled. The latter has two aspects: (i) in the event of the control task missing its deadline, which control input is applied, and (ii) how is the incomplete control task handled, *e.g.*, does it continue to execute or is it killed? If it is continued to execute, then subsequent instances of jobs of this control task needs to be skipped. The work in [26] studies different possibilities that were proposed in [31]—holding the previous control input versus applying a zero control input, along with killing the incomplete task versus letting it continue till completion. For any of these choices, the dynamics of the closed-loop system might evolve differently. For a given deadline hit/miss pattern and a strategy to handle misses, it is easy to check whether the safety property is satisfied over any finite time horizon. For stable systems, it is possible to identify such finite time horizons, verification over which is sufficient. If the system is unstable then there are standard techniques for checking so [32], which can be adapted to also check whether the notion of safety outlined above is satisfied.

But since the number of timing behaviors exhibited by an implementation platform can be exponentially large, it is not feasible to check the safety of each one of them individually. To address this, the work in [26] has resorted to safe but approximate reachability analysis, where *safe* implies all the timing behaviors exhibited by the platform lead to trajectories that are within the specified safety envelope. But *unsafe* means that the exhibited timing behaviors might or might not be safe. Using such approximate reachability analysis, it is possible to certify system-level safety for realistic systems.

III. Synthesis of Certifiable Components

The procedure for checking system-level safety in the presence of timing uncertainties may also be extended to *synthesize* schedules for multiple control tasks. Here, given safety specifications for each control task, the question is: *Can these tasks be scheduled, while allowing deadline misses, such that the safety specification for each controller is satisfied?*

To illustrate this question, consider a set of four control tasks $T_1, \ldots, T_4$ that need to be scheduled on a processor. Here, time is partitioned into equal-sized slots (see Figure 2), and for simplicity of exposition, let us assume that this slot size is also equal to the sampling period of each of the control tasks. The WCET of each of $T_1, \ldots, T_4$ is such that at most two of them may be scheduled in each time slot, and if a task T_i is not scheduled in a particular slot, it misses its deadline. This means that either a zero control input, or the previous control input (or some other scheme to compute a missing

control input) needs to be used to actuate the plant, and its dynamics will deviate from the ideal case.

Figure 2 shows a schedule for the four tasks that satisfy the scheduling constraint that no more than two tasks can be scheduled in one time slot. Note from this figure that the schedule for the task T_1 is $110110\ldots$, that of T_2 is $010010\ldots$, T_3 is $101101\ldots$, and finally, that of T_4 is $001001\ldots$, where a 1 denotes the deadline being met and a 0 a deadline miss. The next question is, do these deadline hit/miss patterns satisfy the specified safety requirements of the four tasks? This can be answered using the techniques discussed in Section II.

Here, given a schedule, we can check whether it satisfies the scheduling/resource constraint and the safety constraint. But since the number of potential schedules can be exponentially large, how can we *efficiently synthesize* valid schedules that satisfy both classes of constraints? We have shown [28] that this can be done by (i) characterizing deadline hit/miss patterns as regular languages, (ii) using approximate reachability analysis techniques to check whether all hit/miss patterns admitted by a given regular language meet specified safety properties using techniques from Section II, and finally (iii) using automata theoretic techniques check whether these schedules satisfy the scheduling/resource constraint, *e.g.*, that no more than two tasks can be scheduled in each time slot, as shown in Figure 2.

This illustrates the possibility of synthesizing certifiable CPS components, while not requiring perfect behavior (all deadlines are met) from other components, such as the scheduler in this case. Such flexibility allows more efficient system design, while still ensuring certification. While we have discussed how this works in the case of timing uncertainty, such principles may be extended to other situations too, *e.g.*, where all control messages need not be authenticated, thereby saving computation resources, or where a ML component is allowed to make inaccurate estimations without compromising system safety.

IV. Related Work on Safety & Certification

Safety and certification is a major design concern in most autonomous CPS, and especially those in the safety-critical domain, such as autonomous vehicles. In this section we broadly discuss some of the existing literature on these topics, with some emphasis on prior work done by us.

Since the complexity of CPS hardware architectures has been rapidly growing, are they susceptible to a variety of manufacturing variabilities, transient faults and aging issues, that might impact the software execution and the results they produce [33]–[36]. The increasing complexity of software in autonomous CPS and the associated scheduling [37]–[39] and management [40]–[42] of such software also contributes to growing safety concerns. This has led to a variety of work on testing [43]–[45] and compositional formal verification of real-time control software and CPS architectures [6], [46]–[49].

As outlined earlier in this paper, feedback control loops form the core of autonomous functionality. Hence, there are close ties between safety analysis and control theory in autonomous systems design. Towards this, control-theoretic methods, reachability analysis, and cross-layer design techniques have been proposed for safety verification [26], [50]–[52]. We earlier discussed the problem of ensuring controller safety in the presence of timing uncertainties. These uncertainties increase as CPS architectures continue to become more distributed and use heterogeneous architectures and communication protocols. As a result, control signals are subject to varying delays that can compromise the safety of the closed-loop system, even if the control strategy is functionally correct. Several papers have addressed this problem [31], [53]–[55]. In addition to the schedule synthesis technique discussed in Section III, the orthogonal problem of synthesizing delay-tolerant controllers has been studied in [56]–[58], including the synthesis of safe controllers that exploit features of the implementation platform and its underlying operating system [59]–[61]. Further, the problem of *co-synthesizing* controllers and their underlying task schedules have been explored in [62]–[65]. All of these problems also have connections to providing timing isolation to critical control software [24], [66], scheduling to meet timing constraints [67]–[72] and the scheduling of mixed-criticality tasks [73], [74]. In the case of vision-based control systems, the accuracy of the (often ML-based) vision processing system [75]–[77] plays an important role in the safety certification of the closed-loop system. Recent work has focused on emerging CPS topics like electric vehicles [78]–[81], autonomous vehicles [82], [83], CPS security [25], [84] and safety and certification issues arising in them. With the growing adoption of electric vehicles and drones, there has been increasing focus on the safety and reliability of battery systems [34], [85], [86]. Work on this topic also relies on battery ageing models as studied in the context of mobile devices [87], but is substantially different from prior work on energy management for such devices [88]–[90], and is currently much less developed.

V. Concluding Remarks

We have outlined a selection of recent work on safety and certification of CPS as they occur in the autonomous and semi-autonomous systems like cars and drones. For any CPS with multiple components — such as one or more controllers, schedulers determining the timing behaviors experienced by the controllers, ML components determining the accuracy of the state estimations seen by the controllers, and security components determining the integrity of the estimated system state and the control signals — current workflow attempts to design each of these components to perfection. For example, the scheduler attempts to meet *all* deadlines in the system. While this enables separation of concerns and design modularity, in the process of ensuring system safety and certification, it also results in undue pessimism. We have argued that by relaxing the notion of safety and allowing each of the system components to admit some error or failure, it would be possible to ensure *both* — safety and efficiency. We have shown this in the case of timing and its impact on system-level safety. Our future research agenda is to determine how this strategy

would also extend to other system components, such as those responsible for security and ML.

Acknowledgments

This work is partially funded by the US NSF grant 2038960, entitled "Design Automation for Automotive Cyber-Physical Systems."

REFERENCES

[1] U. D. Bordoloi *et al.*, "Autonomy-driven emerging directions in software-defined vehicles," in *Design, Automation & Test in Europe Conference (DATE)*, 2023.

[2] S. Chakraborty, M. A. A. Faruque, W. Chang, D. Goswami, M. Wolf, and Q. Zhu, "Automotive cyber-physical systems: A tutorial introduction," *IEEE Design & Test*, vol. 33, no. 4, pp. 92–108, 2016.

[3] F. Alegre, E. Feron, and S. Pande, "Using ellipsoidal domains to analyze control systems software," *CoRR*, vol. abs/0909.1977, 2009. [Online]. Available: http://arxiv.org/abs/0909.1977

[4] P. Tabuada, *Verification and Control of Hybrid Systems - A Symbolic Approach*. Springer, 2009.

[5] D. Goswami *et al.*, "Model-based development and verification of control software for electric vehicles," in *The 50th Annual Design Automation Conference (DAC)*, 2013.

[6] M. Broy *et al.*, "Cross-layer analysis, testing and verification of automotive control software," in *11th International Conference on Embedded Software (EMSOFT)*, 2011.

[7] M. Kauer *et al.*, "Fault-tolerant control synthesis and verification of distributed embedded systems," in *Design, Automation & Test in Europe Conference (DATE)*, 2014.

[8] C. Hsieh, Y. Li, D. Sun, K. Joshi, S. Misailovic, and S. Mitra, "Verifying controllers with vision-based perception using safe approximate abstractions," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 41, no. 11, pp. 4205–4216, 2022.

[9] Z. Wang *et al.*, "Bounding perception neural network uncertainty for safe control of autonomous systems," in *Design, Automation & Test in Europe Conference (DATE)*, 2021.

[10] S. Tripakis, R. Limaye, K. Ravindran, and G. Wang, "On tokens and signals: Bridging the semantic gap between dataflow models and hardware implementations," in *Intl. Conf. on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS)*, 2014.

[11] S. Tripakis, "Bridging the semantic gap between heterogeneous modeling formalisms and FMI," in *International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS)*, 2015.

[12] P. Caspi, N. Scaife, C. Sofronis, and S. Tripakis, "Semantics-preserving multitask implementation of synchronous programs," *ACM Trans. Embedded Comput. Syst.*, vol. 7, no. 2, pp. 15:1–15:40, 2008.

[13] D. Roy *et al.*, "Semantics-preserving cosynthesis of cyber-physical systems," *Proceedings of the IEEE*, vol. 106, no. 1, pp. 171–200, 2018.

[14] S. Chakraborty *et al.*, "Cross-layer interactions in CPS for performance and certification," in *DATE*, 2019.

[15] L. Ju *et al.*, "Context-sensitive timing analysis of Esterel programs," in *46th Design Automation Conference (DAC)*, 2009.

[16] L. Ju, B. K. Huynh, A. Roychoudhury, and S. Chakraborty, "Timing analysis of Esterel programs on general-purpose multiprocessors," in *47th Design Automation Conference (DAC)*, 2010.

[17] M. Geier, T. Burghart, M. Hackl, and S. Chakraborty, "In situ latency monitoring for heterogeneous real-time systems," in *32nd International Conference on VLSI Design (VLSID)*, 2019.

[18] R. Wilhelm, "Determining reliable and precise execution time bounds of real-time software," *IT Professional*, vol. 22, no. 3, pp. 64–69, 2020.

[19] L. Thiele and R. Wilhelm, "Design for timing predictability," *Real-Time Systems*, vol. 28, no. 2-3, pp. 157–177, 2004.

[20] R. Obermaisser, C. E. Salloum, B. Huber, and H. Kopetz, "From a federated to an integrated automotive architecture," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 28, no. 7, pp. 956–965, 2009.

[21] C. Menard, A. Goens, M. Lohstroh, and J. Castrillón, "Achieving determinism in adaptive AUTOSAR," in *Design, Automation and Test in Europe Conference (DATE)*, 2020.

[22] E. Fraccaroli, P. Joshi, S. Xu, K. Shazzad, M. Jochim, and S. Chakraborty, "Timing predictability for SOME/IP-based service-oriented automotive in-vehicle networks," in *Design, Automation & Test in Europe Conference (DATE)*, 2023.

[23] A. Arestova, M. Martin, K.-S. J. Hielscher, and R. German, "A service-oriented real-time communication scheme for AUTOSAR adaptive using OPC UA and time-sensitive networking," *Sensors*, vol. 21, no. 7, 2021.

[24] A. Masrur *et al.*, "VM-based real-time services for automotive control applications," in *16th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2010.

[25] P. Waszecki *et al.*, "Automotive electrical and electronic architecture security via distributed in-vehicle traffic monitoring," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 36, no. 11, pp. 1790–1803, 2017.

[26] C. Hobbs, B. Ghosh, S. Xu, P. S. Duggirala, and S. Chakraborty, "Safety analysis of embedded controllers under implementation platform timing uncertainties," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 41, no. 11, pp. 4016–4027, 2022.

[27] B. Ghosh *et al.*, "Statistical hypothesis testing of controller implementations under timing uncertainties," in *28th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2022.

[28] S. Xu *et al.*, "Safety-aware flexible schedule synthesis for cyber-physical systems using weakly-hard constraints," in *28th Asia and South Pacific Design Automation Conference (ASPDAC)*, 2023.

[29] P. Pazzaglia *et al.*, "DMAC: deadline-miss-aware control," in *31st Euromicro Conference on Real-Time Systems (ECRTS)*, 2019.

[30] ——, "Beyond the weakly hard model: Measuring the performance cost of deadline misses," in *30th Euromicro Conference on Real-Time Systems (ECRTS)*, 2018.

[31] M. Maggio, A. Hamann, E. Mayer-John, and D. Ziegenbein, "Control-system stability under consecutive deadline misses constraints," in *32nd Euromicro Conference on Real-Time Systems (ECRTS)*, 2020.

[32] D. Liberzon, *Switching in Systems and Control*, ser. Systems & Control: Foundations & Applications. Birkhäuser, 2003.

[33] G. Georgakos *et al.*, "Reliability challenges for electric vehicles: from devices to architecture and systems software," in *50th Annual Design Automation Conference (DAC)*, 2013.

[34] W. Chang *et al.*, "Reliable CPS design for mitigating semiconductor and battery aging in electric vehicles," in *3rd IEEE International Conference on Cyber-Physical Systems, Networks, and Applications (CPSNA)*, 2015.

[35] ——, "Battery- and aging-aware embedded control systems for electric vehicles," in *35th IEEE Real-Time Systems Symposium (RTSS)*, 2014.

[36] F. H. Gandoman *et al.*, "Status and future perspectives of reliability assessment for electric vehicles," *Reliab. Eng. Syst. Saf.*, vol. 183, pp. 1–16, 2019.

[37] S. Chakraborty, T. Erlebach, and L. Thiele, "On the complexity of scheduling conditional real-time code," in *7th International Workshop on Algorithms and Data Structures (WADS)*, 2001.

[38] L. Zhang *et al.*, "Schedule management framework for cloud-based future automotive software systems," in *22nd IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2016.

[39] R. Schneider *et al.*, "Multi-layered scheduling of mixed-criticality cyber-physical systems," *Journal of Systems Architecture - Embedded Systems Design*, vol. 59, no. 10-D, pp. 1215–1230, 2013.

[40] S. Ramesh *et al.*, "Specification, verification and design of evolving automotive software," in *54th Annual Design Automation Conference (DAC)*, 2017.

[41] F. Sagstetter *et al.*, "Multischedule synthesis for variant management in automotive time-triggered systems," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 35, no. 4, pp. 637–650, 2016.

[42] S. Samii *et al.*, "Dynamic scheduling and control-quality optimization of self-triggered control applications," in *31st IEEE Real-Time Systems Symposium (RTSS)*, 2010.

[43] G. Tibba *et al.*, "Testing automotive embedded systems under X-in-the-loop setups," in *35th International Conference on Computer-Aided Design (ICCAD)*, 2016.

[44] S. Kramer, D. Ziegenbein, and A. Hamann, "Real world automotive benchmark for free," in *International Workshop on Analysis Tools and Methodologies for Embedded and Real-Time Systems*, 2015.

[45] J. Oetjens et al., "Safety evaluation of automotive electronics using virtual prototypes: State of the art and research challenges," in *51st Design Automation Conference (DAC)*, 2014.

[46] Z. Wang, H. Liang, C. Huang, and Q. Zhu, "Cross-layer design of automotive systems," *IEEE Des. Test*, vol. 38, no. 5, pp. 8–16, 2021.

[47] L. Guo, Q. Zhu, P. Nuzzo, R. Passerone, A. L. Sangiovanni-Vincentelli, and E. A. Lee, "Metronomy: A function-architecture co-simulation framework for timing verification of cyber-physical systems," in *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, 2014.

[48] L. T. X. Phan *et al.*, "Composing functional and state-based performance models for analyzing heterogeneous real-time systems," in *28th IEEE Real-Time Systems Symposium ((RTSS)*, 2007.

[49] R. Schneider *et al.*, "Compositional analysis of switched Ethernet topologies," in *Design, Automation and Test in Europe (DATE)*, 2013.

[50] D. Roy *et al.*, "Automated synthesis of cyber-physical systems from joint controller/architecture specifications," in *Forum on Specification and Design Languages (FDL)*, 2016.

[51] W. Chang *et al.*, "Model-based design of resource-efficient automotive control software," in *35th International Conference on Computer-Aided Design (ICCAD)*, 2016.

[52] P. Kumar *et al.*, "A hybrid approach to cyber-physical systems verification," in *Design Automation Conference (DAC)*, 2012.

[53] D. Goswami, R. Schneider, and S. Chakraborty, "Relaxing signal delay constraints in distributed embedded controllers," *IEEE Trans. Contr. Sys. Techn.*, vol. 22, no. 6, pp. 2337–2345, 2014.

[54] M. Balszun *et al.*, "Effectively utilizing elastic resources in networked control systems," in *23rd IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2017.

[55] H. Voit *et al.*, "Optimizing hierarchical schedules for improved control performance," in *5th IEEE International Symposium on Industrial Embedded Systems (SIES)*, 2010.

[56] D. Goswami, R. Schneider, and S. Chakraborty, "Co-design of cyber-physical systems via controllers with flexible delay constraints," in *16th Asia South Pacific Design Automation Conference (ASP-DAC)*, 2011.

[57] S. Mohamed, D. Goswami, V. Nathan, R. Rajappa, and T. Basten, "A scenario- and platform-aware design flow for image-based control systems," *Microprocess. Microsystems*, vol. 75, p. 103037, 2020.

[58] W. Chang and S. Chakraborty, "Resource-aware automotive control systems design: A cyber-physical systems approach," *Foundations and Trends in Electronic Design Automation*, vol. 10, no. 4, pp. 249–369, 2016.

[59] W. Chang *et al.*, "OS-aware automotive controller design using non-uniform sampling," *ACM Transactions on Cyber-Physical Systems TCPS*, vol. 2, no. 4, pp. 26:1–26:22, 2018.

[60] ——, "Memory-aware embedded control systems design," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 36, no. 4, pp. 586–599, 2017.

[61] ——, "Cache-aware task scheduling for maximizing control performance," in *Design, Automation & Test in Europe (DATE)*, 2018.

[62] R. Schneider *et al.*, "Constraint-driven synthesis and tool-support for flexray-based automotive control systems," in *9th International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, 2011.

[63] D. Roy *et al.*, "Multi-objective co-optimization of FlexRay-based distributed control systems," in *22nd IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2016.

[64] L. Bhatia, I. Tomic, A. Fu, M. Breza, and J. A. McCann, "Control communication co-design for wide area cyber-physical systems," *ACM Trans. Cyber Phys. Syst.*, vol. 5, no. 2, pp. 18:1–18:27, 2021.

[65] R. Mahfouzi, A. Aminifar, S. Samii, A. Rezine, P. Eles, and Z. Peng, "Breaking silos to guarantee control stability with communication over ethernet TSN," *IEEE Des. Test*, vol. 38, no. 5, pp. 48–56, 2021.

[66] F. Sagstetter, M. Lukasiewycz, and S. Chakraborty, "Generalized asynchronous time-triggered scheduling for FlexRay," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 36, no. 2, pp. 214–226, 2017.

[67] L. T. X. Phan, S. Chakraborty, and P. S. Thiagarajan, "A multi-mode real-time calculus," in *29th IEEE Real-Time Systems Symposium (RTSS)*, 2008.

[68] L. Ju, S. Chakraborty, and A. Roychoudhury, "Accounting for cache-related preemption delay in dynamic priority schedulability analysis," in *Design, Automation and Test in Europe (DATE)*, 2007.

[69] L. Zhang *et al.*, "Task- and network-level schedule co-synthesis of ethernet-based time-triggered systems," in *19th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2014.

[70] F. Sagstetter *et al.*, "Schedule integration framework for time-triggered automotive architectures," in *51st Annual Design Automation Conference (DAC)*, 2014.

[71] M. Lukasiewycz *et al.*, "Modular scheduling of distributed heterogeneous time-triggered automotive systems," in *17th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2012.

[72] R. Jacob *et al.*, "TTW: A time-triggered wireless design for CPS," in *Design, Automation & Test in Europe Conference (DATE)*, 2018.

[73] S. Chakraborty *et al.*, "Embedded systems and software challenges in electric vehicles," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2012.

[74] D. Goswami *et al.*, "Time-triggered implementations of mixed-criticality automotive software," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2012.

[75] M. Balszun, M. Geier, and S. Chakraborty, "Predictable vision for autonomous systems," in *IEEE 23rd International Symposium on Real-Time Distributed Computing (ISORC)*, 2020.

[76] T. Amert, M. Balszun, M. Geier, F. D. Smith, J. H. Anderson, and S. Chakraborty, "Timing-predictable vision processing for autonomous systems," in *Design, Automation & Test in Europe Conference (DATE)*, 2021.

[77] N. Otterness *et al.*, "An evaluation of the NVIDIA TX1 for supporting real-time computer-vision workloads," in *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, G. Parmer, Ed., 2017.

[78] M. Lukasiewycz *et al.*, "System architecture and software design for electric vehicles," in *50th Annual Design Automation Conference (DAC)*, 2013.

[79] ——, "Cyber-physical systems design for electric vehicles," in *15th Euromicro Conference on Digital System Design (DSD)*, 2012.

[80] G. Xu, K. Xu, C. Zheng, X. Zhang, and T. Zahid, "Fully electrified regenerative braking control for deep energy recovery and maintaining safety of electric vehicles," *IEEE Trans. Veh. Technol.*, vol. 65, no. 3, pp. 1186–1198, 2016.

[81] R. Aalund, W. Diao, L. Kong, and M. G. Pecht, "Understanding the non-collision related battery safety risks in electric vehicles a case study in electric vehicle recalls and the LG chem battery," *IEEE Access*, vol. 9, pp. 89 527–89 532, 2021.

[82] K. Samal, M. Wolf, and S. Mukhopadhyay, "Closed-loop approach to perception in autonomous system," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2021.

[83] C. Hobbs *et al.*, "Perception computing-aware controller synthesis for autonomous systems," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2021.

[84] P. Mundhenk *et al.*, "Security analysis of automotive architectures using probabilistic model checking," in *52nd Annual Design Automation Conference (DAC)*, 2015.

[85] S. Park, L. Zhang, and S. Chakraborty, "Battery assignment and scheduling for drone delivery businesses," in *International Symposium on Low Power Electronics and Design (ISLPED)*, 2017.

[86] S. Narayanaswamy, M. Kauer, S. Steinhorst, M. Lukasiewycz, and S. Chakraborty, "Modular active charge balancing for scalable battery packs," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 25, no. 3, pp. 974–987, 2017.

[87] A. Pröbstl, B. Islam, S. Nirjon, N. Chang, and S. Chakraborty, "Intelligent chargers will make mobile devices live longer," *IEEE Des. Test*, vol. 37, no. 5, pp. 42–49, 2020.

[88] Y. Gu and S. Chakraborty, "A hybrid DVS scheme for interactive 3d games," in *14th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2008.

[89] ——, "Power management of interactive 3d games using frame structures," in *21st International Conference on VLSI Design*, 2008.

[90] N. Peters *et al.*, "Web browser workload characterization for power management on HMP platforms," in *11th International Conference on Hardware/Software Codesign and System Synthesis (CODES)*, 2016.