QRONOS: Towards Quality-Aware Responsive Real-Time Control Systems

Peter Ulbrich*, Maximilian Gaukler†
Department of Computer Science, *Distributed Systems and Operating Systems
Department of Electrical Engineering, †Automatic Control
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)

***Abstract*—A key design goal of safety-critical control systems is the verifiable compliance with a specific quality objective in the sense of the quality of control. Corresponding to these requirements, the underlying real-time operating system has to provide resources and a certain quality of service, mainly in the form of timing guarantees.**

For the design of efficient real-time control systems, considering only the quality of service is insufficient as it is firmly intertwined with the quality of control: First of all, the actual timing has a significant and nontrivial influence on the quality of control. Vice versa, the temporal precision required to provide a certain quality of control may vary considerably with environmental situation and disturbance. Consequently, quality-of-service requirements are not fixed but may vary depending on the execution context.

We present our ongoing work on quality-aware adaptive real-time control systems, addressing three challenges: evaluating quality of control under consideration of varying timing, static worst-case verification, and quality-aware scheduling at runtime.

I. Introduction

Compliance with an application-specific physical specification is a primary design objective of real-time control systems: in a vehicle, this is, for example, to keep lane in a centimeter tolerance range. Further improvement (i.e., millimeter accuracy) does not lead to further increase in specification compliance or general benefit. Accordingly, from a control-theoretical point of view, the system must be designed and assessed to provide a sufficient Quality of Control (QoC) under all possible environmental conditions (e.g., wind). Typically, the QoC is quantified using a quadratic cost function

$$J = x^T Q x + u^T R u$$

based on the state error x and the control-signal u: small deviation from the desired state $x = 0$ and small actuation correspond to minimum cost J and therefore maximum QoC.

Control systems periodically sample the state of the physical system via sensors, compute the required control signal, and send it to the actuators. Due to this close connection to the outside physical world, real-time control is particularly sensitive to timing variations: In the example of a moving car, sampling the position a later time results in a different value because the car has continued moving. This measurement deviation may reduce the precision of a lane keeping system.

In general, any deviation from the assumed temporal properties may negatively impact the QoC [1]–[3]. Thus, the real-time operating system is tuned for accurate timing of computation and input/output to provide an appropriate Quality of Service (QoS) to the control application running on top. Here, accurate refers to common assessment criteria such as deadline adherence or periodicity (i.e., absence of jitter). In practice, the prevailing point of view is that overall QoC should be optimized by maximizing the QoS, which boils down to tightening temporal bounds [2], [4].

In contrast, current trends in real-time systems foster a well-directed renouncement from this rigid interpretation by moving away from achieving the best possible QoS towards one that is good enough: approaches such as dynamically reconfigurable systems or mixed-criticality scheduling trade accuracy to boost average performance while easing system design as well as worst-case handling. For example, mixed-criticality scheduling [5], [6] provides multiple criticality levels, each with the expectation of a certain quality. Such approaches are, however, typically limited to QoS-guarantees for each criticality level (e.g., control tasks may change timing or even be omitted) . Consequently, it is assumed that there is a static mapping between the QoS and the actually relevant QoC. This static assumption is, for example, also shared by feedback scheduling techniques [1], [7], [8]. Ultimately, deadlines may even be intentionally violated for runtime adaptivity. A vivid example is weakly-hard scheduling of control tasks [9] such that in any window of m execution periods deadlines only have to be met for at least $n < m$ times. In summary, control applications will be faced with more dynamic real-time computing systems, whose timing behavior will be less predictable than it used to be.

Although environmental conditions and QoS (i.e., deviations from the assumed input/output timing) are both determining factors for the QoC, the latter are neither typically considered in the traditional design process nor is the relationship between QoC and QoS trivial.

In previous work [10], [11], we showcased that the dynamic behavior caused by varying timing (QoS) can be counterintuitive. Figure 1 illustrates an example of this effect on the controller of an inverted pendulum. The theoretical framework will be presented later, whereas here we will focus on the results from a real-time systems perspective. The system is subject to varying input and output timing ($\Delta t/T$), that is reading the sensor and writing the actuator values is not performed periodically as assumed during controller design, but with a certain jitter. At first ($t < 10$), the system is operated without delays. Increasing the actuation delay at $t = 10$ has a limited immediate effect but instead causes a gradual increase

in cost J (i.e., decrease in QoC). Upon switching back to better timing at $t = 20$, a short-time adverse effect occurs before the costs have returned to acceptable levels at $t = 21$. A static approximation of QoC as a function of the current timing, an assumption often underpinning embedded control systems design [12]–[14], fails to describe these memory-like behaviors, in which the QoC also depends on the history: At $t = 10$ and $t = 19$, the timing is the same, however the QoC is completely different.

Additionally, the influence of QoC varies between different various sensors and actuators, which is important for the design: In this example the sensor delay ($t = 30 \dots 40$) has less impact than the actuator delay, which suggest that the actuator should be given a higher priority than the sensor.

Existing approaches to evaluate the QoC under consideration of the actual runtime behavior, such as JITTERBUG [15], typically operate on stationary scenarios. In the example of a car, this translates to constant driving conditions and a fixed level of criticality for the control tasks. This means that dedicated QoS levels (timing conditions) are considered individually and the effects are only evaluated time-averaged. Therefore, the aforementioned behavior at transitions between different conditions cannot be analyzed.

II. Problem Statement

In summary, control applications will be faced with more dynamic real-time computing systems, whose timing behavior will be less predictable than it used to be. Runtime adaptivity and scheduling typically focus on QoS bounds, disregarding the susceptibility of control applications to timing variations. At the same time, verifiable compliance with a specific QoC is a key design goal in many settings. Consequently, any stability verification has to factor in non-perfect timing, which is, as illustrated by our example, a non-trivial task.

We identified the primary problem to be the nontrivial mapping of QoS and QoC as well as the fundamentally different approaches to the development and verification for control and real-time systems.

In this paper, we therefore address three challenges to ease the design of adaptive yet verifiable real-time control systems: (1) Evaluation of a time-dependent QoC for varying sensor/actuator timing in adaptive real-time systems. (2) Static analysis[1] and verification of feedback control systems under consideration of timing variations. (3) A real-time executive that saves resources by adapting QoS, but still respects application-specific QoC goals.

III. The QRONOS Approach

We present our vision on the design and verification of adaptive real-time control systems with non-deterministic input and output timing. These complex real-time systems with multiple applications and controllers can significantly profit from dynamic reconfigurability and mixed-criticality scheduling to boost average performance. Our goal is to achieve the benefits of such approaches without loosing the indispensable feature of traditional static scheduling: guaranteed QoC.

[1]Note to readers with a control systems background: The term *static analysis* from software engineering refers to analysis which happens "offline" before a program or system is run, in contrast to dynamic analysis. Despite its name, static analysis does indeed consider the system dynamics (transient behavior); it should not be confused with "analysis of the stationary case".

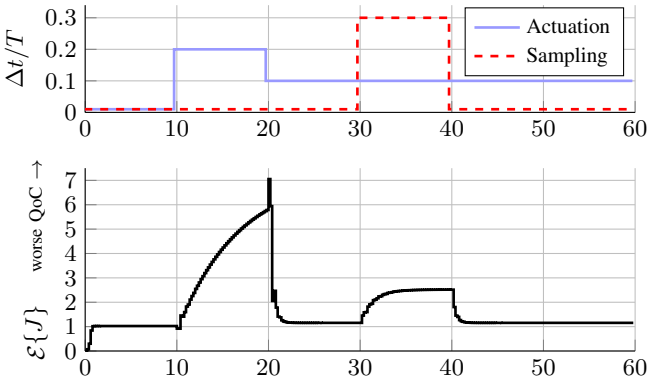

Figure 1. Quality-of-control evaluation for a controlled inverse pendulum for time-varying actuation and sampling delays [11]. The top figure shows the delay normalized to the control period, where 0 is the perfect timing for which the controller was designed. The bottom figure shows the cost over time, where 1 is the performance for perfect timing and larger values correspond to worse Quality of Control, i.e., larger amplitude of error and control signal.

We, therefore, present our ongoing work on Quality-Aware Responsive Real-Time Control Systems (QRONOS), an approach to (a) model and quantify average-case QoC in a time-dependent manner, (b) incorporate non-deterministic input and output timing in the design of controllers and ease verification of the resulting worst-case QoC, and (c) leverage that knowledge by a quality-aware design of the real-time operating system executing the controllers.

In the following sections, we go through these aspects and detail our previous and ongoing work as well as provide an outlook on our future challenges and steps.

A. Average-Case Analysis of Quality of Control

As a first step, we focused on average-case QoC evaluation, i.e., how well the system performs typically. Besides QoS in the form of non-perfect input/output timing, we consider the influence of stochastic physical disturbance (e.g., side wind), measurement noise and control situation (e.g., fast curve vs. straight road). For this complex system model, we developed a QoC evaluation scheme in [11] that can quantify the combined negative impacts of said effects. To gain insight into the dynamic behavior at changing timing, e.g., due to a changing criticality level in mixed-criticality-scheduling, we introduce a noise-averaged, but time-dependent QoC, roughly equivalent to the performance over time for typical disturbance.

Formally, this is modeled as the time-dependent expectation value $\mathcal{E}_{\mathcal{N}}\{J(t)\}$ about the noise $\mathcal{N}$, where $J(t)$ is the cost function from Section I, which weights physical state and control signal amplitude. To compute this averaged QoC without averaging over a multitude of simulations with different random sequences for disturbance and measurement noise, a scheme to directly evaluate this expectation was developed. It is based on first reformulating the problem as a linear impulsive system, which combines continuous and

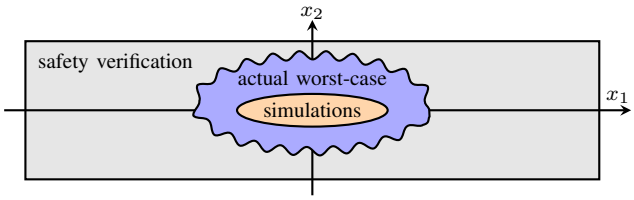

Figure 2. The bounds for the physical state x obtained by a finite number of simulations are too optimistic, whereas safety verification is too pessimistic.

discrete dynamics to model the plant as well as sampling and actuation of the discrete-time controller respectively. As a second step, a stochastic discretization is applied, from which the QoC is computed. The algorithms are currently available for deterministic or discrete stochastic timing [11].

Figure 1 gives the results of our QoC model for the inverted pendulum example discussed in Section I. Here, a deterministic (non-random) timing sequence was used for simplicity. This example is particularly suited to our approach as it demonstrates the possible efficiency gains: The model returns the exact result in a fraction of a second, whereas averaging over a multitude of simulations requires over five hours for an approximation with about 3 percent of remaining error [11].

With this, we offer a systematic approach for evaluating the temporal development of the QoC. We consider this a vital step towards an accurate usage of QoC as an evaluation metric in dynamic and adaptive real-time settings, such as mixed-criticality scheduling, and as a basis for further research on co-design of real-time control systems. Since our approach takes traditionally-designed control systems as input, it can be applied to evaluate the impact of timing on existing systems.

Outlook: We are currently working on further reduction of the computational effort and extending the efficiency gains to a wider problem class. The aim is to use the model (a) also for complex control systems with multiple inputs and outputs and (b) at runtime for QoC-aware timing adaptation.

B. Worst-Case Analysis of Quality of Control

The average-case QoC discussed in the previous section is important to quantify how the system will behave typically. On the other hand, it is equally important to show that the physical system always stays within safety bounds, even in the rare but possible worst case.

While randomized simulation is a pragmatic approach to assess the average performance, it is generally incapable of proving worst-case properties, due to the infeasible number of possible execution flows and timings. As visualized in Fig. 2, simulations can only determine an optimistic lower bound of the worst case. Instead, we opt for a sound overapproximation of worst-case behavior by verification methods.

Currently, we are working on worst-case verification of real-time control systems with uncertain input and output timing by modeling them as hybrid automata [16]. As with the linear impulsive systems used in Section III-A, hybrid automata allow combining the discrete-time and continuous-time aspects of a real-time control system. Unlike linear impulsive systems, which typically require additional informal explanation of the timing model, hybrid automata are a machine-readable precise formal description directly suitable for automatic verification.

As with any form of static analysis, the fundamental challenges are soundness, feasibility, and tight bounds: For the example of a car, we strive to prove that the worst-case track deviation is less than a few centimeters, not meters. Figure 2 illustrates that the bounds shown by verification can significantly exceed the actual worst case, requiring unnecessary safety margins in the design. Our preliminary experiments with existing tools indicate that verification is feasible with useful bounds in some cases yet challenging in general [16].

Therefore, we pursue a parallel approach to solve this problem: instead of proving stability in the presence of jitter, we eliminate the jitter for input and output operations altogether. This obligation requires the real-time system to increase its QoS to the maximum. For its implementation, a well-established method is to resort to a completely static schedule and sound WCET analysis of all control activities. Additionally, sensors and actuators must admit deterministic response times, which typically excludes smart sensors with internal signal processing. In turn, we can resort to traditional stability verification of feedback control loops. We show how to nonetheless benefit from adaptive real-time system techniques and our QoC model in the next section.

Outlook: While numerous techniques for the efficient verification of discrete-time controllers exist, to the best of our knowledge, none of them supports timing uncertainties as presented in our timing model [16], which addresses real-time systems with multiple sensors and actuators by introducing periodic timing windows. Therefore, future work will entail an extension of existing techniques such as [17], [18].

C. Quality-Aware Real-Time Executive

To tackle the challenge of saving resources without violating the application-specific QoC requirements, we propose a quality-aware real-time executive. That is, operating system support and scheduling instrumentation to make the QoC a first-class citizen equal to QoS, i.e., temporal parameters.

Therefore, we are working on an additional scheduler module that applies to jobs with control activities. Based on a simplified variant of the QoC-model from Section III-A, the module adapts release times and deadlines such that it leverages the situation-dependent reserves (i.e., margin between current and specified QoC) to boost average performance and overall runtime flexibility. At the same time, it ensures that adverse effects of varying timing (cf. Section I) are considered and do not jeopardize stability.

As mentioned earlier, worst-case stability analysis of feedback control under the assumption of non-deterministic timing is still subject to research. We find that even with progress in this direction it will be infeasible to dynamically verify scheduled real-time systems with QoC-dependent adaptation of QoS. Therefore, we propose a hybrid execution model where the system switches to a pre-computed, time-triggered

schedule whenever a pessimistic QoC-model anticipates a potential violation of the minimal QoC in the next control step. To yield worst-case guarantees, this model based on Section III-B assumes worst QoS and disturbance.

We call this switching to a static schedule (i.e., deterministic input and output timing) and a verified controller setting (i.e., stability and WCET) our safety net. While in this mode, the QoC will recover verifiably.

This combination of an optimistic QoC-aware and a deterministic safety mode is an ideal supplement to established approaches, such as mixed-criticality scheduling. In contrast to traditional scheduling approaches, it is the potential violation of the QoC that indicates a change in criticality whereas control activities are otherwise categorized as low-criticality jobs. Control-theoretical approaches exist to implement controllers with graded assurance levels, for example, the Simplex architecture as used in [19]: regularly, a controller that performs well in the average case (here: optimistic mode) but does not offer worst-case guarantees is used. If an unstable situation is imminent, a safety mechanism switches to a safety controller (here: safety mode) that offers strict worst-case guarantees but performs worse in the average case.

Outlook: We are currently working on an efficient implementation of the QoC-model and the safety net based on LitmusRT [20]. Here, we investigate the potential for an offline analysis of all possible QoC conditions and to thereof derive a lookup table to eliminate the computational overhead. A further promising candidate that we currently investigate for runtime QoC evaluation are machine-learning approaches. For the safety mechanism, we are working on the control theoretical question of a verifiable design such that it does not activate too often, but still provides provable safety guarantees.

Solving worst-case QoC verification for uncertain timing will permit a relaxation of the deterministic safety mode. Then, only timing bounds instead of timing instants will have to be guaranteed, which greatly simplifies the implementation.

IV. Summary

Real-time control systems face a fundamental design conflict between real-time system and controller design: to improve flexibility and efficient resource usage design goals are shifting from deterministic execution towards good enough QoS properties with weaker guarantees. However, degraded temporal properties, in particular, any variation in sensor or actuator timing, can jeopardize Quality of Control guarantees.

To solve this conflict between efficiency and QoC guarantees, we propose a holistic approach with two modes: an optimistic mode uses dynamic scheduling and adapts the QoS of the control application to the lowest value permitted by current and future QoC. Verification of this mode is generally infeasible. Thus, we provide worst-case guarantees instead by switching to a safety mode if the minimum permissible QoC is about to be violated. This safety mode uses time-triggered, deterministic scheduling to facilitate QoC verification.

We consider our approach a vital step towards the use of runtime dynamics and adaptivity in safety-critical control systems. Its key features are models to capture the non-trivial relation of QoC and QoS for both average-case and worst-case.

Future work will be directed towards the realization of the proposed approach, especially theory and implementation of the safety mechanism, QoC prediction and worst-case analysis.

References

[1] G. Buttazzo and A. Cervin, "Comparative assessment and evaluation of jitter control methods," in *Proc. of the 15th Intl. Conf. on Real-Time and Network Systems (RTNS '07)*, 2007, pp. 163–172.

[2] B. Wittenmark, J. Nilsson, and M. Törngren, "Timing problems in real-time control systems," in *Proc. of the American Control Conf.*, New York, NY, USA, 1995, pp. 2000–2004.

[3] A. Ray, "Output feedback control under randomly varying distributed delays," *Guidance, Control, and Dynamics*, vol. 17, no. 4, pp. 701–711, 1994.

[4] H. Kopetz, *Real-Time Systems: Design Principles for Distributed Embedded Applications*, 1st ed. Kluwer Academic Publishers, 1997.

[5] S. Vestal, "MetaH support for real-time multi-processor avionics," in *Proc. of 5th Intl. Work. on Parallel and Distributed Real-Time Systems and 3rd Work. on Object-Oriented Real-Time Systems*. IEEE, Apr 1997, pp. 11–21.

[6] A. Burns and R. Davis, "Mixed criticality systems – a review," Department of Computer Science, University of York, Tech. Rep. 9th ed., 2016.

[7] D. Simon, A. Seuret, and O. Sename, "On real-time feedback control systems: Requirements, achievements and perspectives," in *Systems and Computer Science (ICSCS), 2012 1st Intl. Conf. on*, Aug. 2012.

[8] A. Cervin and J. Eker, "Feedback scheduling of control tasks," in *Proc. of the 39th IEEE Conf. on Decision and Control (CDC '00)*, vol. 5. New York, NY, USA: IEEE Press, 2000, pp. 4871–4876.

[9] G. Bernat, A. Burns, and A. Liamosi, "Weakly hard real-time systems," *IEEE Trans. on Computers*, vol. 50, no. 4, pp. 308–321, Apr. 2001.

[10] T. Klaus, F. Franzmann, M. Gaukler, A. Michalka, and P. Ulbrich, "Poster Abstract: Closing the Loop: Towards Control-aware Design of Adaptive Real-Time Systems," in *Proc. of the 37th Real-Time Systems Symp. (RTSS '16)*. IEEE, 2016, pp. 363–363.

[11] M. Gaukler, A. Michalka, P. Ulbrich, and T. Klaus, "A new perspective on quality evaluation for control systems with stochastic timing," in *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control - HSCC '18*. ACM Press, 2018.

[12] A. Cervin, J. Eker, B. Bernhardsson, and K.-E. Årzén, "Feedback-feedforward scheduling of control tasks," *Real-Time Systems*, vol. 23, no. 1-2, pp. 25–53, 2002.

[13] G. Buttazzo, M. Velasco, and P. Marti, "Quality-of-control management in overloaded real-time systems," *IEEE Trans. on Computers*, vol. 56, no. 2, pp. 253–266, 2007.

[14] F. Flavia, J. Ning, F. Simonot-Lion, and S. YeQiong, "Optimal on-line (m,k)-firm constraint assignment for real-time control tasks based on plant state information," in *IEEE Intl. Conf. on Emerging Technologies and Factory Automation (ETFA '08)*. IEEE, 2008, pp. 908–915.

[15] B. Lincoln and A. Cervin, "JITTERBUG: a tool for analysis of real-time control performance," in *Proc. of the 41st IEEE Conf. on Decision and Control (CDC '02)*. IEEE, 2002, pp. 1319–1324.

[16] M. Gaukler and P. Ulbrich, "Worst-case analysis of digital control loops with uncertain input/output timing," in *ARCH19. International Workshop on Applied veRification for Continuous and Hybrid Systems*, 2019, accepted for publication.

[17] S. Bak and T. T. Johnson, "Periodically-scheduled controller analysis using hybrid systems reachability and continuization," in *2015 IEEE Real-Time Systems Symposium*. IEEE, dec 2015.

[18] L. Hetel, C. Fiter, H. Omran, A. Seuret, E. Fridman, J.-P. Richard, and S. I. Niculescu, "Recent developments on the stability of systems with aperiodic sampling: An overview," *Automatica*, vol. 76, pp. 309–335, Feb. 2017.

[19] D. Seto, B. H. Krogh, L. Sha, and A. Chutinan, "Dynamic control system upgrade using the simplex architecture," *IEEE Control Systems*, vol. 18, no. 4, pp. 72–80, Aug. 1998.

[20] J. M. Calandrino, H. Leontyev, A. Block, U. C. Devi, and J. H. Anderson, "LITMUS^{RT}: A testbed for empirically comparing real-time multiprocessor schedulers," in *Proceedings of the 27th Real-Time Systems Symposium (RTSS '06)*, 2006, pp. 111–126.