Anne Koziolek, Ina Schaefer, Christoph Seidl (Hrsg.): Software Engineering 2021,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2021 67

Skill-Based Verification of Cyber-Physical Systems

Alexander Knüppel[1] Inga Jatzkowski[2] Marcus Nolte[3] Tobias Runge Knüppel[4] Thomas Thüm[5] Ina Schaefer[6]

Abstract: This work has been accepted at the 23rd International Conference on Fundamental Approaches to Software Engineering, held as part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2020 [Kn20].

The increase of complexity in modeling cyber-physical systems poses a challenge for formally ensuring their functional correctness. Lack of expert knowledge and scalability are two limiting factors that prohibit a seamless integration into today's software engineering processes. To address this challenge, we propose to adopt and formalize the notion of *skill graphs*, an abstract and easy-to-use modeling notion for representing automated vehicle driving maneuvers. For formally verifying that skill graphs are well-formed and comply with a given set of safety requirements, we incorporate *hybrid programs* into our formalization. Hybrid programs constitute a program notion for cyber-physical systems on the basis of differential dynamic logic, which enables deductive and compositional verification following the idea of Hoare-style reasoning. That is, simpler verified skill graphs can be combined to exhibit complex maneuvers while validity is retained (i.e., without the need of re-verification). To showcase the benefits of our theoretical considerations, we implemented our framework in an open-source tool named Skeditor and conducted a case study exhibiting an automatic *vehicle follow mode*.

Keywords: Deductive verification; design by contract; formal methods; theorem proving; hybrid program; cyber-physical systems

Nowadays, cyber-physical systems are ubiquitously present in our lives and have a direct impact on most humans. As the complexity of modern cyber-physical systems increases while also being applied for safety-critical tasks, formal verification methods are required. This poses a major challenge on developers and engineering processes in the early development stages to ensure functional correctness. Hence, an important challenge is to derive modeling and verification techniques that (1) are easy to integrate into the software development cycle by reducing modeling complexity and (2) allow to identify severe requirement and modeling mistakes at design time. To address this challenge, we propose a model-based verification framework called Skeditor and an accompanying IDE that allows to prototype driving maneuvers (e.g., *following a leading a vehicle*) and verify their adherence to a set of formal requirements.

[1] TU Braunschweig a.knueppel@tu-bs.de
[2] TU Braunschweig jatzkowski@ifr.ing.tu-bs.de
[3] TU Braunschweig nolte@ifr.ing.tu-bs.de
[4] TU Braunschweig tobias.runge@tu-bs.de
[5] University of Ulm thomas.thuem@uni-ulm.de
[6] TU Braunschweig i.schaefer@tu-bs.de

doi:10.18420/SE2021_22

To achieve this goal, SKEDITOR combines two concepts, namely *skill graphs* and *hybrid programms*. Skill graphs were introduced by Reschka et al. [Re15] and serve as modeling notation, following the principle of *separation of concerns* by enabling the decomposition of more complex maneuvers into modular, reusable, and inter-related building blocks (i.e., skills). To model the behavior of skills that interact with the physical environment (i.e., controllers), we rely on *hybrid programs* [Pl18], which comprise a program notation for cyber-physical systems together with a deductive calculus to prove controller correctness. One of the most important properties of our formalization is *compositionality*, for which we defined a composition operator for skill graphs. That is, smaller provably correct skill graphs can be combined while correctness is retained without the need of re-verification.

In our evaluation, we modeled and verified a skill graph representing a vehicle follow mode combining two simpler maneuvers, namely *following a leading vehicle* and *following hard shoulder* (i.e. the lateral and longitudinal control tasks). We modeled each of these maneuvers individually and measured the verification effort with KEYMAERA X [Fu15]. Due to the typical overlap in skills for both maneuvers, our results show that the composition reduces the total verification effort by 53% compared to monolithic modeling.

In summary, we provide the first formalization of skill graphs including tool support and show how they can be combined with hybrid programming as a formal underpinning. As demonstrated in our work, SKEDITOR allows developers to prototype driving maneuvers and verify their safety as part of the early software development life cycle, while – due to compositionality – costly re-verification can be minimized.

Bibliography

[Fu15] Fulton, Nathan; Mitsch, Stefan; Quesel, Jan-David; Völp, Marcus; Platzer, André: KeYmaera X: An Axiomatic Tactical Theorem Prover for Hybrid Systems. In: International Conference on Automated Deduction. Springer, pp. 527–538, 2015.

[Kn20] Knüppel, Alexander; Jatzkowski, Inga; Nolte, Marcus; Thüm, Thomas; Runge, Tobias; Schaefer, Ina: Skill-Based Verification of Cyber-Physical Systems. In: International Conference on Fundamental Approaches to Software Engineering. Springer, Cham, pp. 203–223, 2020.

[Pl18] Platzer, André: Logical Foundations of Cyber-Physical Systems, volume 662. Springer, 2018.

[Re15] Reschka, Andreas; Bagschik, Gerrit; Ulbrich, Simon; Nolte, Marcus; Maurer, Markus: Ability and Skill Graphs for System Modeling, Online Monitoring, and Decision Support for Vehicle Guidance Systems. In: 2015 IEEE Intelligent Vehicles Symposium (Vol. IV). IEEE, pp. 933–939, 2015.