Chapter 33
A Model-Based Toolchain to Verify Spatial Behavior of Cyber-Physical Systems

Peter Herrmann
Norwegian University of Science and Technology (NTNU), Norway

Jan Olaf Blech
RMIT University, Australia

Fenglin Han
Norwegian University of Science and Technology (NTNU), Norway

Heinz Schmidt
RMIT University, Australia

ABSTRACT

A method preserving cyber-physical systems to operate safely in a joint physical space is presented. It comprises the model-based development of the control software and simulators for the continuous physical environment as well as proving the models for spatial and real-time properties. The corresponding toolchain is based on the model-based engineering tool Reactive Blocks and the spatial model checker BeSpaceD. The real-time constraints to be kept by the controller are proven using the model checker UPPAAL.

1. INTRODUCTION

In safety critical domains like aviation, automotive and robotics, autonomous cyber-physical systems interact with each other in the same physical space. To avoid damage and injuries, the control software of the systems has to guarantee spatiotemporal properties like collision avoidance or the cooperation of several units that carry a heavy workpiece together. A popular way for the creation of functionally correct and safe system software is the application of integrated modeling and verification tools like

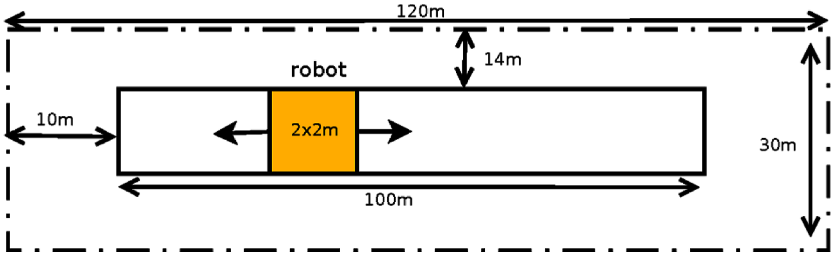
DOI: 10.4018/978-1-7998-1754-3.ch033

MATLAB/Simulink (Tyagi, 2012). Our contribution is the combination of such a tool with efficient provers allowing to verify that the coordinated behavior of multiple controlled cyber-physical systems fulfills relevant spatial safety properties. We introduce a toolchain combining the model-based engineering tool-set Reactive Blocks[1] (Kraemer, Slåtten, & Herrmann, 2009) with the verification tool BeSpaceD (Blech & Schmidt, 2013). In particular, we use a development workflow starting with the collection of requirements for a cyber-physical system and its architecture followed by the steps listed below:

1. Spatiotemporal properties of components are described in the input language of BeSpaceD;
2. A model of the system controller is created in Reactive Blocks. We compose it with a simulator model of the continuous system parts which is created using the BeSpaceD model developed in step 1;
3. The built-in model checker of Reactive Blocks is used to check the combined controller and simulator model for general design errors (Kraemer, Slåtten, & Herrmann, 2009);
4. If the checks in step 3 are passed, the software model is transformed to the input language of BeSpaceD;
5. Assuming certain maximum reaction times of the discrete controller, it is verified with BeSpaceD that the model resulting from the transformation in step 3 fulfills the spatiotemporal properties defined in step 1;
6. The model checker UPPAAL (Bengtsson, et al., 1996) is applied to prove that the real-time properties assumed in the proofs of step 5 are indeed kept by the Reactive Blocks model created in step 2 (Han & Herrmann, 2013), (Han, Herrmann, & Le, 2013);
7. By using the code generator from Reactive Blocks (Kraemer & Herrmann, 2007), (Kraemer, Herrmann, & Bræk, 2006) executable Java code of the controller and, if desired, of the simulator of the continuous behavior is created. The generated code can be deployed on the system components running the control software of the embedded system.

Our approach has to guarantee that a model developed with Reactive Blocks indeed fulfills the desired safety properties if the verifications in steps 5 and 6 succeed. Formally, that proof is merely trivial: Be S the logical formula corresponding to a system model in Reactive Blocks according to (Kraemer & Herrmann, 2010), P the conjoined spatial behavioral properties to be fulfilled by S, and $R(t)$ a statement describing that the controller always guarantees a maximum reaction time t. Using BeSpaceD, we verify in step 5 that the system fulfills the safety properties if t is kept, i.e., $S \wedge R(t) \Rightarrow P$. In step 6, we prove

Figure 1. Layout of the moving robot


120m
14m
robot
10m
2x2m
30m
100m