xCPS: A tool to eXplore Cyber Physical Systems

Shreya Adyanthaya[*], Hadi Alizadeh Ara[*], João Bastos[*], Amir Behrouzian[*],
Róbinson Medina Sánchez[*], Joost van Pinxten[*], Bram van der Sanden[*],
Umar Waqas[*], Twan Basten[*†], Henk Corporaal[*], Raymond Frijns[*], Marc Geilen[*],
Dip Goswami[*], Sander Stuijk[*], Michel Reniers[*], Jeroen Voeten[*†]

[*]Eindhoven University of Technology
Eindhoven, The Netherlands

[†]TNO ESI
Eindhoven, The Netherlands

http://www.es.ele.tue.nl/cps/

ABSTRACT

Cyber-Physical Systems (CPS) play an important role in the modern high-tech industry. Designing such systems is a challenging task due to the multi-disciplinary nature of these systems, and the range of abstraction levels involved. To facilitate hands-on experience with such systems, we develop a cyber-physical platform that aids in research and education on CPS. This paper describes this platform, which contains all typical CPS components. The platform is used in various research and education projects for bachelor, master, and PhD students. We discuss the platform and a number of projects and the educational opportunities they provide.

Keywords

Cyber-Physical Systems; embedded systems; education

1. INTRODUCTION

Cyber-Physical Systems (CPS) are becoming increasingly ubiquitous in society. CPS can be found in day-to-day situations such as traffic-light networks, smart homes, advanced automotive systems and energy systems, but also in almost any high-tech domain, such as medical imaging, electron microscopy, professional printing, and chip fabrication. These systems tightly integrate computation and communication and physical processes, where embedded computers and networks control those physical processes. Consequently, these systems pose several challenges due to their complexity and multidisciplinary requirements in all system development phases: design, analysis and implementation.

Designing a CPS is a challenging task, since exploring different design alternatives requires designers to understand and identify different abstraction layers and domains and how they connect. For instance, close to the physical-layer local continuous feedback controllers are used to ensure correct functioning of actuators while at a higher level, supervisory controllers orchestrate the interactions among local controllers and globally control the entire system.

Even at the analysis stage of a design choice, predicting and evaluating performance and checking system requirements becomes highly complex because of interactions between different disciplines and system layers. Consider that in high-performance systems, control signals are required at very high frequencies. Therefore, computation and communication delays can no longer be dismissed and must be taken into account when designing the controllers.

Finally, looking at the implementation phase, new methodologies are needed that guarantee that the design and the predicted analysis is actually satisfied at all different layers of the system. Research in this area is focussed on automated synthesis techniques that provide guarantees (like safety) and efficiency (like maximal throughput).

All these challenges require not only research and innovation, but also a change of view in the domain and field, such that future engineers and researchers have a solid understanding and practical experience to address these challenges. Academic research and education play a crucial role by developing and disseminating methods and solutions to address the various issues described above.

In this paper we present eXplore Cyber-Physical Systems (xCPS); a platform of industrial complexity for research and education on CPS. It embeds all the typical CPS components and is used as a vehicle for CPS education and research. It allows for experimentation, research and development of components relating to different disciplines, as well as multi-disciplinary aspects of a complex system. Furthermore, it gives students and researchers the chance to obtain a global view for all the steps in the development, from designing [9], analysing [15], and implementing [10] CPS.

Section 2 introduces the xCPS platform, Section 3 discusses education and research opportunities in different subjects using ongoing projects. Section 4 discusses the connections between the projects and the opportunities for integration and multidisciplinary approaches. Section 5 discusses related work and Section 6 concludes.

2. THE xCPS PLATFORM

The xCPS platform (Figure 1) is a small scale machine mimicking a production line capable of assembling and disassembling objects. The cylindrical assembly pieces of xCPS (Figure 2) come in two complementary shapes, and three colours (silver, black or red). The cylindrical shape simplifies the transport and assembly process and different coloured pieces

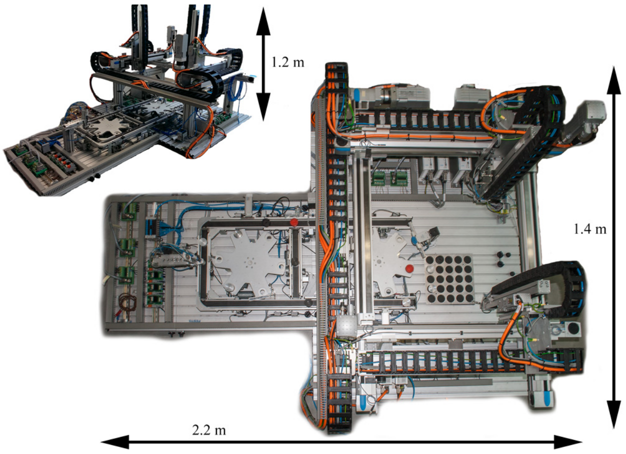

Figure 1: Top and side view of the xCPS platform.

enable the creation of different jobs the platform is able to execute.

For reasons of cost and effort students cannot easily get access to perform experiments on industrial machines. Assumptions made by students, in the absence of measurements, cannot be validated and may be unrealistic for practical industrial platforms. Systems such as xCPS provide students and researchers with a realistic platform to perform measurements and analysis on, enabling development of novel techniques that are closer and more applicable to practice.

2.1 Platform Overview

The system layout (Figure 3) illustrates the main components of the platform. The platform consists of one storage area, six conveyor belts, two indexing tables, two gantry arms, and several actuators and sensors. The storage area is a grid where 25 components can be stored.

There are six actuators called 'stoppers' in strategic positions along the assembly process that can obstruct the movement of pieces, effectively creating buffers accumulating pieces on the conveyor belts. Switches make it possible to change the route of individual objects. A turner can flip pieces. The platform also contains two actuators for assembly and disassembly of pieces. The pick & place actuator can clamp a part and combine it with a complementary part. A separator can disassemble two combined pieces. The xCPS platform is equipped with 15 sensors that can detect the presence of an object in the surrounding area. These sensors cannot distinguish the type of an object or its colour. To detect the type, colour, and location of an object, a camera can be added to the set-up.

There are many possible use cases of the xCPS platform. One can for example simulate an assembly process where individual parts arrive on the first conveyor belt, are subsequently assembled and then put back into the storage. The gantry arms and storage area can be used as a sorting machine, where the storage is sorted according to selected criteria, but can also be used for two-player games such as tic-tac-toe and four-in-a-row. Each of these use cases carry different objectives and learning goals.

Figure 2: Assembly pieces in the xCPS.

2.2 Hardware Abstraction Layer

The mechanical hardware of the xCPS platform (i.e., belts, arms, picker) is controlled by electrical motors, servos and pneumatic actuators. These actuators are controlled by signals from several data acquisition and control input/output cards inside a general-purpose computer platform.

The Hardware Abstraction Layer (HAL) for the xCPS platform is currently a C-based Application Programming Interface (API) that consists of several functions which act as device drivers for the actuators and sensors. Given this abstraction, it is for example possible to control the gantry arms by sending them commands to move to 3-dimensional (x, y, z) locations. Such location commands are used in a high-level controller to hide the required complex series of signals beneath the HAL.

The HAL is essential to use the xCPS platform as an educational platform. This abstraction layer allows students to access and execute the physical system without detailed knowledge of the underlying realization. When desired, students can study the underlying layers and get hands-on experience in programming and controlling a real CPS.

2.3 Virtualised platform

Especially during the early development stages, it is often too expensive or time-consuming to develop a physical prototype of each component in a system. The API introduced in Section 2.2 captures the higher-level behaviour of the mechanical hardware and can therefore be defined before a real prototype has even been developed. Such API's are used to simulate the mechanical hardware (Simulator-in-the-Loop) from early development stages, to facilitate rapid prototyping and early design of control algorithms and software applications. On the other hand, Hardware-in-the-Loop simulations enable testing of individual hardware components in circumstances that are hard to reproduce in a real environment. Students can use such simulations and visualizations when the real xCPS platform itself is not available or when it is not supporting certain functionality yet. Such visualizations may show the physical environment, or focus on specific aspects of the activities, such as Gantt chart visualizations that focus on the ordering of activities over time. We have created a model in Parallel Object-Oriented Specification Language (POOSL) [17] that can simulate an environment (for instance non-deterministic or a predefined sequence of events) to test the robustness of controllers or software implementations for xCPS [12].

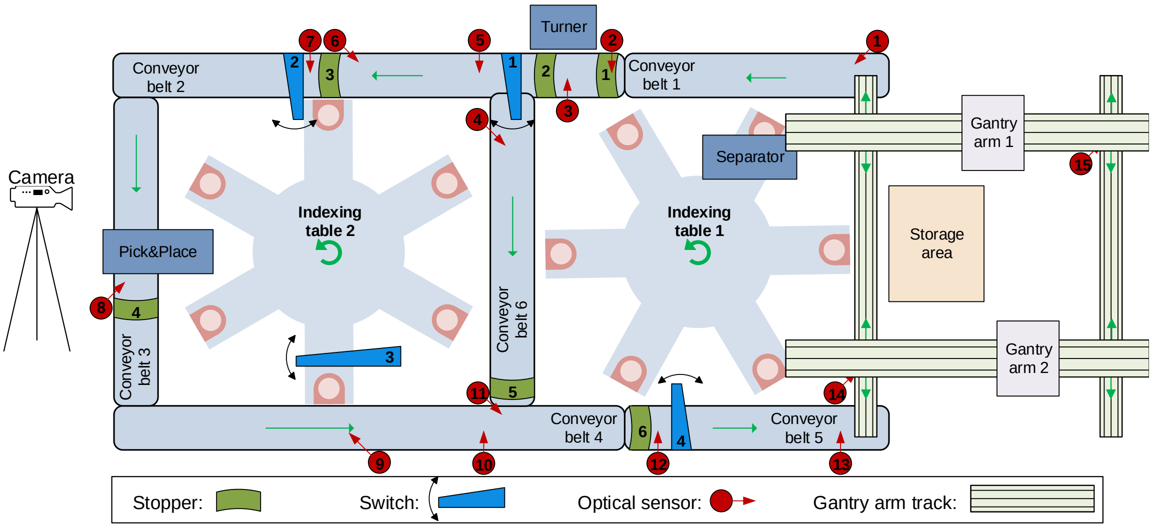

Figure 3: xCPS system layout.

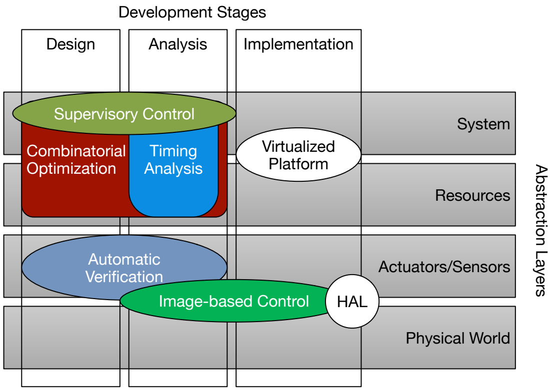

Figure 4: Positioning of the research areas that are explored using xCPS.

3. TEACHING AND RESEARCH OPPORTUNITIES IN THE xCPS PLATFORM

Exposing students and researchers to a platform such as the xCPS, gives them an opportunity to grasp the complexity of CPS. The overall layering of the system, from software-level to servo-level, creates an environment where multidisciplinary and cross-layers work is explored. Moreover, the different use cases for the xCPS allow for a wide and diverse range of areas of expertise to be explored. Figure 4 depicts the different development stages (Design, Analysis, Implementation), abstraction layers (System, Resources, Servo, Physical) and research areas that are investigated. In the next sections we address a number of these areas and explain why and how they can be used in education and research.

3.1 Supervisory Control

Supervisory control coordinates and orchestrates actuation at the system level to achieve system goals (objects get assembled / disassembled). High-level control actions at supervisory control level are translated to instructions sent to lower level controllers. The supervisor has to ensure that the system is free of deadlocks and eliminates any unsafe behaviour, such as gantry arms colliding.

High-tech CPS, such as the xCPS platform, typically consist of many sensors and actuators. Due to the number of components and the high level of interaction among them, proper modelling of the system and requirements is far from trivial. Requirements have to be at the right abstraction level, and the model should allow for local changes when the system configuration evolves. Due to the large state space, there are challenges in the synthesis step to automatically derive a supervisory controller that ensures satisfaction of the requirements.

In a bachelor level project [6] a student has modelled part of the platform to develop a supervisory controller following the synthesis-based model-based systems engineering process [19, 4]. In this process the uncontrolled system and the control requirements are modelled independently and in a modular way, using small, loosely coupled models based on the formal model of extended finite automata (efas) [14, 20]. The CIF 3 tool set [19] has been used to carry out the modelling.

A high-level, abstract, model has been used that partitions the conveyor belts into segments to simplify the control of the product flow. Each segment will be allowed to hold at most one object. Examples of used models for a stopper (*Stopper1*), a sensor (*Sensor_Optical1*), and an area (*Area1*) are given in Figure 5. Solid edges and dashed edges represent controllable and uncontrollable events in these automata, respectively. Controllable events are those which may be prevented by the supervisory controller; uncontrollable events may not be prevented by the supervisory controller.

Requirements are formulated with respect to correct functioning of the actuators, and enforcing a correct product flow. A complete set of requirements (and requirement models) is presented in [6]. An example of such a requirement is that the number of items never exceeds the finite capacity of the areas on the conveyor belt and the indexing table. Figure 6 shows another example of a requirement model. The requirement states that the camera may only scan a

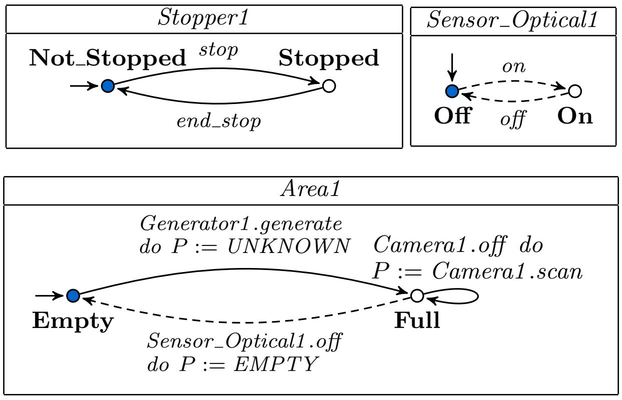

Figure 5: Example CIF 3 uncontrolled system models [6].

$$\begin{array}{lcl} \rightarrow \{\mathit{Camera1.on}\} & \Rightarrow & \mathit{Stopper1.Stopped} \downarrow \\ & \wedge & \mathit{Sensor\_Optical1.On} \downarrow \\ & \wedge & \mathit{Area1.P} = \mathit{UNKNOWN} \end{array}$$

Figure 6: Example of a requirement model in CIF 3 [6].

work piece (notation → {*Camera1.on*} denotes the possibility to execute event *on* in automaton *Camera1*) when there actually is a non-moving (*Stopper1.Stopped* ↓ denotes that automaton *Stopper1* is in a location with name *Stopped*) workpiece in front of it (*Sensor_Optical1.On* ↓) that has not been scanned before (*Area1.P* = *UNKNOWN*, where *Area1.P* refers to the value of the variable *P* in automaton *Area1*).

Based on the models for the uncontrolled system and the requirements, we tried monolithic synthesis to automatically derive a supervisory controller. Due to the large state-space, this did not succeed. Instead modular supervisory controllers have been developed by hand. Complete models are available in [6]. In order to validate the correctness of these manually developed supervisory controllers, hybrid (combined discrete and continuous) models are added to represent timing aspects and physical position information of the products. For this hybrid model, a visualization (Figure 7) was added to validate the controlled system, i.e., the system with the developed supervisory controllers. CIF 3 provides means to simulate and visualize hybrid models.

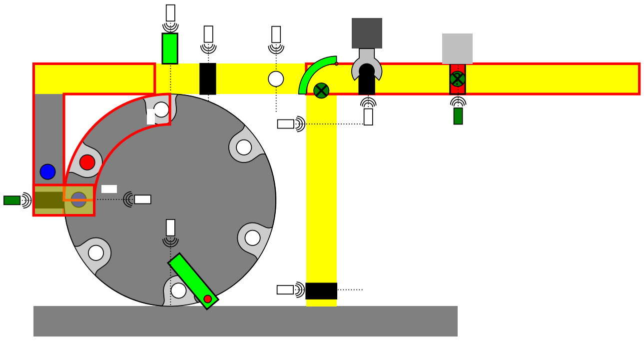

Figure 7: Visualization of the product flow in CIF 3.

Current research in a PhD project uses xCPS as a case study to develop modular synthesis techniques to overcome or reduce the algorithmic scalability challenges.

3.2 Combinatorial optimization

Supervisory control ensures maximum flexibility of actions while ensuring safe operating boundaries. However, it does not provide any scheduling mechanism that optimizes the sequence of actions to achieve certain goals. For example, the goal of xCPS may be to assemble components *as fast as possible*, or to play tic-tac-toe with *the lowest number of moves*. Developing real-time algorithms that generate close-to-optimal schedules can be challenging, due to the computational complexity associated with such scheduling or optimization problems.

As it is nearly impossible to make all aspects of a machine optimal for all use cases, system designers need to make design-time decisions to optimize for certain typical use-cases and operating conditions. Moreover, multiple different objectives play a role and they are often conflicting, for instance productivity, quality and cost. Most designs have some degree of freedom in the high-level configuration of the system. The designers can then exploit the reconfigurability towards use-cases for different customers.

Besides design time decisions and system level configuration, CPS often also require online decision making or scheduling. The quality of such scheduling has an impact on system productivity or processing time. Due to the discrete and often non-linear nature of the system, such scheduling problems are usually of very high computational complexity and solved to near-optimality by combinatorial optimization algorithms. The computational complexity of the optimization problem is typically highly correlated with the way the models are constructed.

The signals that control the various sensors and actuators in the xCPS platform are produced by control tasks that are mapped onto general purpose platforms. These platforms suffer from low predictability resulting in variations in task execution times. This uncertain timing behaviour is also apparent in the xCPS platform: not only in the data processing but also in the pick & place actuator that may need a varying number of attempts to pick the pieces using suction in vacuum. There is an additional need for *robust combinatorial optimization* techniques that generate sequences that can guarantee performance goals even in the presence of timing variations. Robust combinatorial optimization is an ongoing PhD research.

The xCPS platform can be used to simulate real industrial machines such as chip fabrication machines and large-scale production printers. This kind of industrial machines perform different actions taking different amounts of time: changing lithography masks or adjusting the print head height for thicker/thinner paper respectively. We can simulate such behaviour by constraining the execution of certain physical tasks on the xCPS platform. For example, the red item takes ten seconds to process, while the black and silver components can be processed in five seconds. Some machines also take some time to re-adjust between processing different (kinds of) items; e.g. a reconfiguration penalty of five seconds may be imposed when processing a black item directly after a silver item. The order in which these actions have been scheduled can heavily influence the performance. This behaviour is known as *sequence-dependent setup* and is a crucial aspect in scheduling problems such as production printing [22] and wafer scanners [2].

The xCPS platform is used to teach students different aspects of performance modelling and combinatorial optimization. The students can learn which aspects of the optimization problem are essential to achieve a high quality model. For example, choosing which decision variables to take into account in what way, which solution space search strategy works better and determining the selection of derived constraints to reduce the search space. These aspects are usually key in getting good solutions to the optimization problem in a limited time budget.

xCPS facilitates exploration of the strong and the weak points of different modelling and optimization approaches. For example, we can use Synchronous Data Flow (SDF) [11], genetic algorithms [7], (meta-)heuristics and constraint solving based approaches to find schedules for the xCPS platform. Comparing these approaches on an industrial high-tech system is very challenging due to the complexity and/or even impossible due to intellectual property restrictions. The xCPS platform shows key ingredients that are of interest for an industrial system or can be easily extended to incorporate such components. Optimizing machines with e.g. setup times, buffering, pipelining, and/or timing variation, in a reduced complexity setting allows students to explore the combinatorial challenges of modern high-tech systems.

3.3 Timing Analysis

Manufacturing systems have key-performance indicators based on throughput and latency, e.g., number of products per hour while satisfying certain deadlines in the system. Latency metrics are defined as the time distance between two specified events in the system, for instance between the start of processing and the output of a product. These metrics must be considered in the design-space exploration, scheduling decisions, or for validation purposes. Therefore, timing analysis of CPS is an important subject to study or research, which can be facilitated with the xCPS platform.

For xCPS, we look at performance analysis at the system level and at the resource level. The particular challenge is to capture the timing behaviour of tasks in the product flow taking into account the tasks themselves, the resources they use, as well as their mutual dependencies. For example, the product flow relies on synchronizing events between different actions of actuators or sensors. This can be modelled as tasks with dependencies. Such systems also have resource dependencies when resources are shared between multiple tasks. Besides task and resources dependencies, these systems depend on the pipelining of multiple products to increase performance. This causes additional dependencies between different products, for example while one product is being assembled, other pieces can be already introduced in the system.

We are exploring the use of data flow models-of-computation, such as Synchronous Data Flow graphs [11] or extensions of this model such as Scenario-Aware Data Flow (SADF) [18], since they can naturally capture (cyclic) task dependencies, resource dependencies and pipelined behaviour. Task durations (execution times) and synchronizations are also natural ingredients of data flow models. Moreover, data flow models have good analyzability properties for which we employ our research data flow analysis tool (SDF3) [15] to determine throughput and latency.

Consider a part of xCPS that takes bottom and top objects, assembles them and outputs the assembled object. Top and bottom objects take different paths in the machine to reach the assembling station. Each path consists of a different sequence of actions, with some resources being shared between paths. The durations of the individual actions are known. They can for instance be measured. We can model this system using SADF, in which each path (bottom or top) can be modelled by a specific SDF graph and the possible orderings of bottoms and tops can be expressed as a formal language accepted by a finite state automaton on infinite words. An optimal or safe order can be derived using combinatorial optimization or controller synthesis techniques as discussed in other sections. Data flow analysis results can then be used to find and resolve bottlenecks within xCPS. This project is being pursued by a PhD student.

Another PhD student considers the use of data flow models in design-space exploration. A data flow model is created that abstractly represents the whole system. It can then be used to explore different platform configurations (e.g., number of resources, types of resources) and find the optimal platform(s) based on one or more optimization objectives (timing performance indicators or resource cost). For instance one may explore the cost/performance trade-offs from using faster or slower conveyor belts or processing units. From these trade-offs one could for instance determine the lowest number of processing units to meet a certain throughput requirement. In the xCPS platform, the use-case is the design of input/output of products using the pneumatic arms (Gantry arms 1 and 2 in Figure 3). Currently, the objective is to model the input/output flow using SADF [18] to estimate product flow performance and explore different platform designs considering speed of the arms (using different profile settings) or the use of both or only one arm.

The above analysis techniques use worst-case task execution timings, assumed to be known beforehand. Predicting the timing behaviour of the xCPS system heavily depends on the accuracy of the timing models. Dealing with unpredictable platforms with large, stochastic variations, raises the need for timing models to also take timing variations into account giving rise to the PhD study of *robust timing analysis*. Industrial CPS often have timing information limited only to measurements. We need efficient techniques to obtain these measurements and combine them into reasonable estimates of execution time distributions. Another challenge is that most performance measures are derived from the completion times of operations. Obtaining these completion time distributions needs complex stochastic analysis involving analytically hard *max* operations on distributions [1]. We have studied the statistical analysis [2] of this behaviour. With that information robustness analysis can be employed to compute metrics which quantify the robustness of the system towards achieving desired performance. An example of a stochastic metric is the expected value of the number of pieces assembled by xCPS per hour.

3.4 Automatic Verification

The abstraction used in Sections 3.2 and 3.3, considers each individual action in the system as a task with an *assumed* fixed (possibly worst case) or stochastic execution time. In reality, the execution of an action or task is the result of the dynamics of the actuator executing the task possibly in feedback control, affected by errors and physical disturbances. In this section we zoom into the details of one of such actions. We will explore the use of abstraction techniques for infinite-state and hybrid automata to *verify* that the assumptions of the higher level models adhere to the physical world. Therefore, we employ automata-based abstraction and verification techniques to verify the validity of the timed models.

To study the behaviour under all possible control inputs, we need a suitable model of the system. In control theory, it is common to derive a state space model of the dynamical systems, in which the states represent the relevant physical quantities of the system by real valued numbers. This model can however not be directly used for automata-based verification techniques such as model-checking, due to the fact that the continuous variables give the system an (uncountably) infinite number of states. We overcome this difficulty by making finite-state abstractions of the infinite state model [16]. We present a simplified example that leads to a model with only a few states to be able to visualize it and illustrate the concepts involved. In practical problems, the finite state models can have many more states and need model-checker tool support to check its properties automatically.

In our example we consider the task in the system to be the placement of an object in the desired place by the movement of the conveyor belt. We consider the position of the object on the conveyor belt to be the (continuous) state, x of the system. For this belt there is only one possible actuator action: turning on the conveyor belt for a fixed amount of time. The action makes the object move, and hence makes the state x change to a new state x'. Due to small physical variations however, there is some disturbance, variation, in x'. The finite abstraction consists in a partitioning of the state space (object positions) across the conveyor belt that allows us to distinguish being positioned in the desired place (at a pusher, switch or turner) from being positioned in a wrong place. The variations in the state due to disturbances leads to non-deterministic behaviours in the abstract state-space. However, with sufficiently small disturbances and a well-chosen partitioning, it may be possible to find a sequence of activations of the belt that is guaranteed to bring the object to a desired place.

Figure 8 shows finite state models of the belt controlled by three different controllers having different activation times for the belt, after abstraction. In constructing these models it is assumed that the conveyor has an unknown speed error with a known bound. Abstract states in which the object is positioned in the desired spot are shown in green. The states in yellow represent positions on the belt before the target and the red ones are states in which the object may have passed the target position. Figure 8(a) shows that there exists a behaviour in which the object never stops in the desired positions (passes a green state). The activations are too long and the object may go at once from a position before the target to a position after the target (according to the abstracted model). In Figure 8(b) a smaller activation period is used. We can see that the object will eventually stop in a green state after three or four activations. However, neither three activations nor four activations are guaranteed to bring the object to the right place and there is no sensor feedback that tells us how often to activate. Finally, Figure 8(c) shows a different activation period, which is suitable for a feed forward controller; after two activations the object is guaranteed to be at the desired position. As we have now verified the behaviour of this controller we may choose to implement this controller and we know the fixed execution time of the task. The xCPS platform is used as the case study of a PhD course project in this subject.

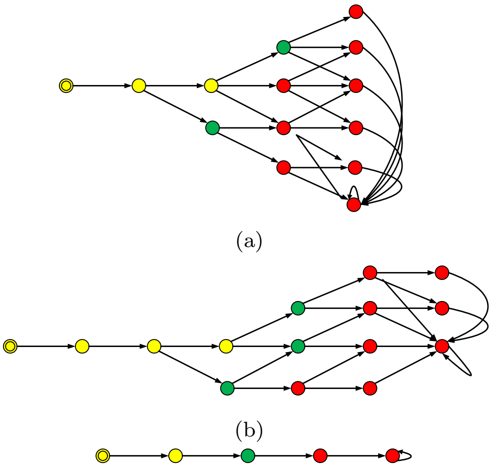
(a)
(b)
(c)

Figure 8: Finite state models of the belt for three different controllers

3.5 Image-based sensing and control

Image-based controllers use sensor data that is extracted from camera images by an image processing algorithm. As the computational power of embedded devices increases, these controllers are becoming more common in CPS such as robotics and Advanced Driver Assistance Systems (ADAS). In ADAS, an image sensor is used to detect, track and classify objects. This information is used to either help the driver control the vehicle or to autonomously control it [10]. In robotics, image based sensing is used to track a reference (objects of interest) and guide a robot towards it [5].

Image based control involves multiple challenges. Firstly, the image processing algorithm should have both a reliable quality and a predictable (a priori bounded) execution time. Secondly, the controller should be capable of coping with the latency induced by the image processing algorithm and provide guarantees for performance and stability. Finally, trade-offs between resource usage and quality of control play an important role since the resources in the embedded platforms are limited.

We are studying and exploring these challenges in the xCPS platform by trying to control the assembly/disassembly process with a camera through multiple student projects. In

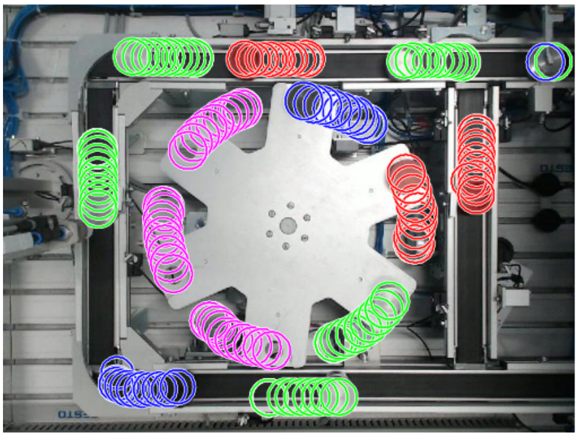

Figure 9: The image processing algorithm working on the pre-recorded video stream, currently tracking 14 pieces. The tracks shown are from the first 12 frames of the video. The circles of the colours correspond to the object types. Purple denotes an empty object location on the indexing table.

a bachelor level project an image processing algorithm has been implemented to obtain the position of all the objects on the conveyor belts and the indexing tables [21] (see Figure 9). To this end, the image is divided in two parts; one containing the conveyor belts and one the indexing tables. The objects on both parts of the image are located using a Hough Transform for circles, together with different preprocessing algorithms. The implemented software achieves a detection ratio of 93% with an execution time of one second. Experimentation reveals that despite a high ratio, the algorithm leads to a slow assembly/disassembly process because of its relatively long execution time. A successor master thesis student is investigating strategies to reduce the latency of the algorithm.

Trade-offs between resource usage and quality of control are also being studied. A master thesis project is analysing these trade-offs arising from alternative image processing algorithms: one slower and more accurate and one faster but inaccurate. The goal of this project is to improve the quality of control by developing strategies for switching between the two algorithms, instead of using a single one.

Finally, the development of resource-efficient control systems is being investigated in a PhD research project. In order to deal with the long execution time in the sensing, a pipelined configuration is used to keep a high sampling frequency. Such an implementation potentially requires a large amount of resources. The research involves cross-layer codesign between embedded implementations of data-intensive image sensing algorithms and the quality of control strategies for sensors with high sample rates and high latency, a typical concern in CPS.

4. DISCUSSION

Different aspects of design and analysis have been highlighted in the previous sections. Examples of various activities have been given and it has been shown how many of these aspects are interdependent and interrelated. Especially in design of CPS, learning how to effectively master these interrelationships to come to an overall optimal system level design is of crucial value.

At the highest abstraction level, the supervisory control constructs the state space of all safe and deadlock free configurations (built from abstract states and coarse grain actions). Combinatorial optimization techniques are used to explore the state space of safe configurations to find the optimal behaviours and optimal (run-time) scheduling strategies. Given a schedule, timing analysis techniques provide the time-based performance metrics such as throughput or latency. At the lowest abstraction level, continuous feedback controllers implement the abstract actions and verification and synthesis techniques guarantee that the higher level abstract models of the actions are sound, i.e., adhere to the physical world behaviour.

xCPS as a platform facilitates research and learning on all these aspects. Furthermore, the various activities on the platform are often multidisciplinary in nature and encourage collaboration between students with different backgrounds, in Computer Science, Computer Engineering, Electrical Engineering and Mechanical Engineering, with the side-effect of also developing non-technical skills challenged by such cooperations.

5. RELATED WORK

Recently there has been a lot of interest in the development of CPS platforms to be used both for research and teaching. It is important to realize that the xCPS platform is more general than the automatic control platforms which their main goal is to verify the functionality of certain control laws. Axelsson et al. [3] introduce MOPED, a mobile open platform for experimental design of CPS. The platform consists of a model car chassis, controlled by a set of three control units running the automotive software standard AUTOSAR. It is designed to be highly representative of real automotive systems in terms of software, while simplifying other aspects. The platform is extensible, to allow both students and researchers to add new functionality and interfaces. A robot car platform is developed by González-Nalda et al. [8], where a Raspberry Pi board is used as processing unit and there is interaction with the robot using WiFi.

In xCPS, image-based controllers are used as part of some controllers. Mosterman et al. [13] introduce a manufacturing facility that solves the Towers of Hanoi puzzle, where image processing is also used in the control loop. Compared to xCPS, there is also more focus on feedback and feed-forward control for the actuators. The aim of platform in [13] is to be used by students in project-based learning whereas xCPS provides a flexible platform imitating industrial scale manufacturing systems.

CPS education is also provided in the form of a massive open online course (MOOC) [9]. Here, the iRobot Roomba autonomous vacuum cleaner is used as a platform for CPS research. The platform is capable of driving, sensing the surroundings, executing scripts and communicating with an external controller. Similar to xCPS, a hardware abstraction layer is provided for programming. This layer allows simulation of sensors and actuators within a simulator, and linking executable models to the actual sensor and actuator drivers. Although xCPS employs several similar techniques, the applications focus on imitating manufacturing systems, which show challenges not only on the implementation level, but also how to cope with the challenges in different models and abstraction levels.

6. CONCLUSIONS

This paper presents the xCPS research and education platform, that is designed to be representative of CPS with industrial-size complexity. The platform contains various types of sensors and actuators that allow different types of challenges to be investigated and CPS related competencies to be explored. From an educational and research point of view, students working with the platform not only become familiar with the different types of subsystems, their design trade-offs and models, methods, analysis and tools, but they also experience how the parts interrelate and the importance of cross-layer design considerations that are typical of cyber-physical systems. More information about the xCPS platform is available at `www.es.ele.tue.nl/cps`.

Acknowledgments

This research is supported by the Dutch Technology Foundation STW under the Robust CPS program (projects 12693, 12694, and 12697) and the NEST project (10346), by the ARTEMIS joint undertaking under the ALMARVI project (621439), and by ASML, Océ, Technolution, TNO-ESI, and FEI.

7. REFERENCES

[1] S. Adyanthaya et al. Robustness analysis of multiprocessor schedules. In *Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIV), 2014 Int. Conf. on*, pages 9–17, 2014.

[2] S. Adyanthaya, M. Geilen, T. Basten, R. Schiffelers, B. Theelen, and J. Voeten. Fast multiprocessor scheduling with fixed task binding of large scale industrial cyber physical systems. In *Digital System Design (DSD), 2013 Euromicro Conf. on*, pages 979–988, Sept 2013.

[3] J. Axelsson, A. Kobetski, Z. Ni, S. Zhang, and E. Johansson. Moped: A mobile open platform for experimental design of cyber-physical systems. In *Software Engineering and Advanced Applications (SEAA), 2014 40th EUROMICRO Conf. on*, pages 423–430, Aug 2014.

[4] J. Baeten, J. van de Mortel-Fronczak, and J. Rooda. Integration of supervisory control synthesis in model-based systems engineering. *Proc. of Special Int. Conf. on Complex Systems: Synergy, of Control, Communications and Computing*, pages 167–178, 2011.

[5] L. Feng. *Development and applications of a vision-based unmanned helicopter.* National University of Singapure, 2010.

[6] R. Fransen. Modeling a flow shop workstation using CIF. Bachelor thesis, 2015. CST 2015.077.

[7] D. Goldberg. *Genetic Algorithms.* Pearson Education, 2006.

[8] P. Gonzalez-Nalda, I. Calvo, I. Etxeberria-Agiriano, A. Garcia-Ruiz, S. Martinez-Lesta, and D. Caballero-Martin. The challenge of building a cyber physical system as an educational experience. In *Information Systems and Technologies (CISTI), 2014 9th Iberian Conference on*, pages 1–6, June 2014.

[9] J. C. Jensen, E. A. Lee, and S. A. Seshia. Virtualizing cyber-physical systems: Bringing CPS to online education. 2013.

[10] K. Jo, J. Kim, D. Kim, C. Jang, and M. Sunwoo. Development of Autonomous Car - Part II: A Case Study on the Implementation of an Autonomous Driving System Based on Distributed Architecture. *IEEE Trans. on Industrial Electronics*, 0046:1–1, 2015.

[11] E. Lee, D. G. Messerschmitt, et al. Synchronous data flow. *Proc. of the IEEE*, 75(9):1235–1245, 1987.

[12] Y. Li., J. Voeten, and R. Frijns. A model-driven design approach for an industrial-scale mechatronic system. Master thesis, Eindhoven University of Technology, 2011.

[13] P. Mosterman, J. Zander, and Z. Han. The towers of hanoi as a cyber-physical system education case study. *Proc. of the First Workshop on Cyber-Physical Systems Education at CPSWeek*, 2013.

[14] M. Skoldstam, K. Åkesson, and M. Fabian. Modeling of discrete event systems using finite automata with variables. In *Decision and Control, 2007 46th IEEE Conf. on*, pages 3387–3392, Dec 2007.

[15] S. Stuijk, M. Geilen, and T. Basten. SDF^3: SDF For Free. In *Application of Concurrency to System Design, 6th International Conference Proceedings*, pages 276–278. IEEE Computer Society Press, Los Alamitos, CA, USA, June 2006.

[16] P. Tabuada. *Verification and control of hybrid systems: a symbolic approach.* Springer Science & Business Media, 2009.

[17] B. Theelen, O. Florescu, M. Geilen, J. Huang, P. van der Putten, and J. Voeten. Software/Hardware Engineering with the Parallel Object-Oriented Specification Language. *Formal Methods and Models for Codesign (MEMOCODE), 2007 5th IEEE/ACM Int. Conf. on*, pages 139–148, May 2007.

[18] B. D. Theelen, M. Geilen, T. Basten, J. Voeten, S. V. Gheorghita, and S. Stuijk. A scenario-aware data flow model for combined long-run average and worst-case performance analysis. In *Formal Methods and Models for Codesign (MEMOCODE), 2006 4th IEEE/ACM Int. Conf. on*, pages 185–194, 2006.

[19] D. A. van Beek, W. J. Fokkink, D. Hendriks, A. Hofkamp, J. Markovski, J. M. van de Mortel-Fronczak, and M. A. Reniers. CIF 3: Model-based engineering of supervisory controllers. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 8413 of *Lecture Notes in Computer Science*, pages 575–580. Springer Berlin Heidelberg, 2014.

[20] B. van der Sanden, M. Reniers, M. Geilen, T. Basten, J. Jacobs, J. Voeten, and R. Schiffelers. Modular model-based supervisory controller design for wafer logistics in lithography machines. In *Proc. of MODELS 2015*, 2015. Accepted for publication.

[21] S. Vegt. A Fast and Robust Algorithm for the Detection of Circular Pieces in a Cyber Physical System. Bachelor thesis, Eindhoven University of Technology, 2015. available at www.es.ele.tue.nl/esreports/esr-2015-02.pdf.

[22] U. Waqas, M. Geilen, J. Kandelaars, L. Somers, T. Basten, S. Stuijk, P. Vestjens, and H. Corporaal. A re-entrant flowshop heuristic for online scheduling of the paper path in a large scale printer. In *Design, Automation & Test in Europe Conference & Exhibition, Proc. of the 2015*, DATE '15, pages 573–578, San Jose, CA, USA, 2015. EDA Consortium.