Course: Logical Foundations of Cyber-Physical Systems

Course Syllabus

André Platzer
platzer@kit

KIT Department of Informatics, Karlsruhe Institute of Technology

Cyber-physical systems (CPSs) combine cyber effects (computation and/or communication) with physical effects (motion or other physical processes modeled by differential equations). Cars, aircraft, and robots are prime examples, because they move physically in space in a way that is determined by discrete computerized control algorithms. Designing these algorithms to control CPSs is challenging due to their tight coupling with physical behavior. At the same time, it is vital that these algorithms be correct, since we rely on CPSs for safety-critical tasks like keeping aircraft from colliding. In this course we will strive to answer the fundamental question posed by Jeannette Wing:

> "How can we provide people with cyber-physical systems they can bet their lives on?"

The cornerstone of this course design is differential dynamic logic (dL) for hybrid programs (HPs), which capture relevant dynamical aspects of CPSs in a simple programming language with a simple semantics. One important aspect of HPs is that they directly allow the programmer to refer to real-valued variables representing real quantities and specify their dynamics as part of the HP. One important aspect of dL is that it directly proves properties of HPs by logical decomposition.

This course will give you the required skills to formally analyze the CPSs that are all around us – from power plants to pacemakers and everything in between – so that when you contribute to the design of a CPS, you are able to understand important safety-critical aspects and feel confident designing and analyzing system models. It will provide an excellent foundation for students who seek industry positions and for students interested in pursuing research.

Contents

1 Course Information

Home `https://lfcps.org/course/lfcps.html`

Lectures 3+1 SWS

Credit 5 ECTS

Textbook You are expected to follow the accompanying textbook (or its free electronic version), which also comes with videos that enable you to review lectures [1, 2]: André Platzer. *Logical Foundations of Cyber-Physical Systems*. Springer, 2018. DOI 10.1007/978-3-319-63588-0

Grading Grading will be based on a final exam.

Exams closed internet, closed book, one double-sided sheet of hand-written notes permitted. The dates are on the course web page.

Tools we will also make use of the hybrid systems verification tool KeYmaera X `http://keymaeraX.org/`

Key topics Cyber-Physical Systems · Dynamic Logic · Models and Controls · Axiomatics · Differential Equations Reasoning · Hybrid Games · Formal Verification

The course counts toward the Vertiefungsfach Theoretische Grundlagen or the Vertiefungsfach Softwaretechnik or the Wahlbereich Informatik in the KIT Master of Science in Informatics curriculum.

Please carefully read the entire syllabus to make yourself familiar with the contents and expectations and policies in this course. It is also your go-to reference later.

2 Learning Objectives

The learning objectives of Logical Foundations of Cyber-Physical Systems are organized along the dimensions: *modeling and control*, *computational thinking* [3], and *CPS skills*.

2.1 Modeling and Control

In the area of *modeling and control*, successful students will

- **understand the core principles behind CPS.** A solid understanding of these core principles is important for anyone who wants to integrate cyber and physical components to solve problems that no part could solve alone.
- **develop models and controls.** In order to understand, design, and analyze CPS, it is important to be able to develop models for the various relevant aspects of a CPS design and to design controllers for the intended functionalities based on appropriate specifications, including modeling with differential equations.
- **identify the relevant dynamical aspects.** It is important to be able to identify which types of phenomena of a CPS have a relevant influence for the purpose of understanding a particular property of a particular system. These allow us to judge, for example, where it is important to manage adversarial effects, or where a nondeterministic model is sufficient.

2.2 Computational Thinking

In the area of *computational thinking*, successful students should be able to

- **identify safety specifications and critical properties.** In order to develop correct CPS designs, it is important to identify what “correctness” means, how a design may fail to be correct, and how to make it correct.
- **understand abstraction in system designs.** The power of abstraction is essential for the modular organization of CPS, and for the ability to reason about separate parts of a system independently. Because of the overwhelming practical challenges and numerous levels of detail, abstraction is even more critical than it already is in conventional software design.
- **express pre- and post-conditions and invariants for CPS models.** Pre- and post-conditions allow us to capture under which circumstance it is safe to run a CPS or a part of a CPS design, and what safety entails. They allow us to achieve what abstraction and hierarchies achieve at the system level: decompose correctness of a full CPS into correctness of smaller pieces. Invariants achieve a similar decomposition by establishing which relations of variables remain true no matter how long and how often the CPS runs.
- **reason rigorously about CPS models.** Reasoning is required to ensure correctness and find flaws in a CPS design. Both informal reasoning and formal reasoning

in a logic are important objectives for being able to establish correctness, which, for CPS, includes also rigorous reasoning about properties of differential equations.

2.3 CPS Skills

In the area of *CPS skills*, successful students will be able to

- **understand the semantics of a CPS model.** What may be easy in a classical isolated program becomes very demanding when that program interfaces with effects in the physical world. Understanding the meaning of a CPS model with fewer dynamical aspects and knowing how it will execute is fundamental to reasoning.
- **develop an intuition for operational effects.** Intuition for the joint operational effect of a CPS is crucial, e.g., about what the effect of a particular discrete computer control algorithm on a continuous plant will be.
- **understand opportunities and challenges in CPS and verification.** While the beneficial prospects of CPS for society are substantial, it is crucial to also develop an understanding of their inherent challenges and of approaches for minimizing the impact of potential safety hazards. Likewise, it is important to understand the ways in which formal verification can best help improve the safety of system designs.

3 Programming Language

With a suitably generalized programming language, the behavior of a CPS can be described by a program. This course develops the programming language of *hybrid programs* (HPs) to capture relevant dynamical aspects of cyber-physical systems in a simple programming language with a simple semantics. The most distinctive features of HPs are that they prominently feature *differential equations* and *nondeterminism.* HPs support differential equations as continuous models of the physical system dynamics so that you can directly write down a differential equation in the middle of a program to describe the behavior of physics. Nondeterminism is another feature required for the adequacy of CPS models, e.g. for capturing choices in the system execution or uncertainty about the environment. When describing a robot controller, for example, we cannot know for sure what decisions other agents in the environment reach and need to be prepared to handle multiple choices in the execution. The course leverages *differential dynamic logic* (dL) as a specification and verification language for rigorous reasoning about hybrid programs that makes program properties explicit and localizes reasoning about their correctness.

4 FAQ

This section elaborates the expected background and purpose of this course.

4.1 Who Should Take This Course?

You should definitely take this logic course if

- you ever want to program robots that operate near humans so that you need to understand how to do that safely, or
- you ever want to develop computer control systems for cars, or
- you ever want to write programs that control aircraft or drones, or
- you ever want to help computers control power plants or the smart grid, or
- you want to do embedded systems or cyber-physical systems, or
- you are interested in learning how computation interfaces with the real world, or
- you are simply fascinated by combining mathematics and computer science, or
- you want to see logic matter in reality.

4.2 What are Students Expected to Know Before This Course?

The course assumes prior exposure to basic computer programming and that you have seen basic differentiation. The course covers the basic required mathematical and logical background of cyber-physical systems but you will be expected to follow the companion textbook [1] as needed.

If you are afraid of programming or afraid of mathematics, then you will find this course more challenging. The course is specifically designed *not* to require particularly advanced background, but you should feel comfortable picking the required concepts up as we go. We will explain what you need to know in the course and provide pointers to reading material. Coming into this course, you should definitely already know what a derivative is and be comfortable using derivatives in mathematical arguments. Throughout the course you need to develop an intuitive understanding of differential equations for modeling common physical processes. We will frequently need this ordinary differential equation system (ODE)

$$x' = v, v' = a \tag{1}$$

which can be understood as saying that the time-derivative of x is v, and the time-derivative of v is a. In other words, this differential equation means that the derivative of the position x is the velocity v, and the derivative of the velocity v is the acceleration a. Understanding ODE (1) will suffice for the first part of the course. As the course progresses, we will learn how to do elegant reasoning about even ODEs whose solutions are nasty, which provides a good opportunity to reinforce your understanding of ODEs.

4.3 How To Take This Course

Since not all material is covered in full in lecture, and reading presents a complementary way of internalizing material at your own pace, you are *strongly encouraged* to subsequently read the corresponding textbook chapters. Some students also learn better when first going through the textbook chapter at their own pace before the lecture.

5 Schedule

The tentative schedule of lectures follows the chapters of the textbook [1] with some adaptations for semester timing reasons and to follow student interest:

1. Cyber-Physical Systems: Introduction
2. Differential Equations & Domains
3. Choice & Control
4. Safety & Contracts
5. Dynamical Systems & Dynamic Axioms
6. Truth & Proof
7. Control Loops & Invariants
8. Events & Responses
9. Reactions & Delays
10. Differential Equations & Differential Invariants
11. Differential Equations & Proofs
12. Ghosts & Differential Ghosts
13. Differential Invariants & Proof Theory
14. Hybrid Systems & Games
15. Winning Strategies & Regions
16. Winning & Proving Hybrid Games
17. Game Proofs & Separations
18. Axioms & Uniform Substitutions
19. Verified Models & Verified Runtime Validation
20. Virtual Substitution & Real Equations
21. Virtual Substitution & Real Arithmetic

6 Policies

6.1 Course Culture

This course is open to anyone who is excited about cyber-physical systems and wants to learn all they need to become proficient in the subject matter. With its cross-disciplinary appeal, this course attracts students from different majors, different backgrounds, and different prior experiences, who all bring valuable and unique perspectives to the interdisciplinary aspects of cyber-physical systems. Listening to the contributions and opinions of your fellow students provides a huge opportunity for you to learn how others approach and overcome the challenges of the world.

We desire an open and inclusive course culture, where diversity in all its aspects is embraced. Everybody is different, everybody is special, and it is our collective responsibility to ensure that everybody is welcome in this course. If you experience or observe behavior that makes you feel unsafe, unwelcome, or discriminated against, please let the instructors know so they can help.

6.2 Evaluation Criteria

The most important criterion is always correctness. Buggy programs are useless, and likely to get a low score, because the corresponding CPSs are likely to do serious damage. Elegance and clear structure is beneficial if not necessary for achieving correctness. A secondary criterion is the performance of your robot controller in terms of reaching its goal and interacting with its environment.

Grading is based on the correctness of the answer and the presentation of your reasoning. Strive for clarity and conciseness, but show how you arrived at the answer. Stating an answer without explanation does not count as an answer. If you cannot solve a problem, explaining your approach and why you failed is encouraged. Such answers will be given partial credit.

6.3 Laptops and Phones in Lecture

As research on learning shows, unexpected noises and movement automatically divert and capture people's attention, which means you are affecting everyone's learning experience if your phone, laptop, etc. makes noise or is visually distracting during class.

Therefore, please silence all mobile devices during class and stow them away. You are welcome to use tablets or laptops for note-taking only, but if possible, please use laptops only in the back of the classroom so as not to distract others.

Research also shows that concepts are best internalized when actively working with the material and taking notes. You have a full textbook available [1], but you are *strongly encouraged* to write summaries of the most important material in your own words.

6.4 Extra Points for Proof Exploits: KeYmaera X Integrity

All feedback about how to improve the course material and KeYmaera X is always very welcome and is part of your participation grade. There is one form of feedback that is particularly helpful: feedback that concerns soundness.

Soundness is crucial and fundamental, but of special significance for the high stakes of cyber-physical systems. What good would a safety analysis of a broken cyber-physical system do if the analysis procedure itself is broken?

To reflect that, we are soliciting *Proof Exploits.* By which we mean proofs that exploit soundness-critical flaws in the lecture notes or soundness-critical bugs in KeYmaera X. Each new soundness-critical bug that you are the first person to report is worth 20 points of extra credit. For full credit you should also demonstrate with a proof exploit how that bug can be exploited to produce a proof of `false` or of `1=0`. A proof exploit is a formal proof on paper or a test case for KeYmaera X demonstrating how the flaw can be exploited to exhibit a proof of `false`, which, since `false` is rarely true, cannot have a proof in any sound verification procedure.

Needless to say that this is not just a great way for you to earn extra credit but also a really solid preparation for questions scrutinizing what rules and axioms and proof attempts are sound and which ones aren't. This reflection is an invaluable skill when it comes down to analyzing CPSs.

Hint: You are allowed to be *arbitrarily* creative in your proof exploits and do things that you are not ordinarily supposed to do in a verification tool.

References

[1] André Platzer. *Logical Foundations of Cyber-Physical Systems*. Springer, Cham, 2018. doi:10.1007/978-3-319-63588-0.

[2] André Platzer. Videos for: Logical foundations of cyber-physical systems, 2019. URL: http://video.lfcps.org/.

[3] Jeannette M. Wing. Computational thinking. *Commun. ACM*, 49(3):33–35, 2006. doi:10.1145/1118178.1118215.