Remote Teaching with the Cyber-Physical Mobility Lab

Armin Mokhtarian $^{1,1}$, Patrick Scheffe $^2$, Bassam Alrifaee $^2$, and Stefan Kowalewski $^2$

$^1$RWTH Aachen University - Embedded Software Laboratory
$^2$Affiliation not available

October 31, 2023

Abstract

Self-driving laboratories receive increasing acceptance as they are used in research and education. For research, they mainly serve as a platform on which algorithms can be tested under realistic conditions before getting deployed into the real world. In education, they can add value by creating a reference to reality as they offer an application for theoretical knowledge. In combination with the fact that practical work is well appreciated among students, such setup helps to motivate and thus enhance the learning experience. However, not every institution has the capabilities to create a self-driving lab on its own. Instead, open-source and remotely accessible laboratories, like our Cyber-Physical Mobility Lab (CPM Lab) can be used to engage with the domain of networked and autonomous vehicles. By means of two of our courses, we demonstrate the capabilities of the CPM Lab. These two courses can either be used as groundwork to develop own lectures in this domain, or used directly since the course materials are open-source.
Abstract: Self-driving laboratories receive increasing acceptance as they are used in research and education. For research, they mainly serve as a platform on which algorithms can be tested under realistic conditions before getting deployed into the real world. In education, they can add value by creating a reference to reality as they offer an application for theoretical knowledge. In combination with the fact that practical work is well appreciated among students, such setup helps to motivate and thus enhance the learning experience. However, not every institution has the capabilities to create a self-driving lab on its own. Instead, open-source and remotely accessible laboratories, like our Cyber-Physical Mobility Lab (CPM Lab) can be used to engage with the domain of networked and autonomous vehicles. By means of two of our courses, we demonstrate the capabilities of the CPM Lab. These two courses can either be used as groundwork to develop own lectures in this domain, or used directly since the course materials are open-source.

Keywords: Self-Driving Laboratories, Networked and Autonomous Vehicles, Control Education, Computer Science Education

SUPPLEMENTARY MATERIAL

The construction plans and a virtual machine of the CPM Lab and the course materials are available at our website https://cpm.embedded.rwth-aachen.de/. The software of the CPM Lab is available at https://github.com/embedded-software-laboratory/cpm_lab. A remote access to the CPM Lab is provided at: www.cpm-remote.de

1. INTRODUCTION

Practical work is one of the fundamental tools used in education. During university studies, such insights are important so students can apply theory practically and fortify their knowledge. Furthermore, real-world tasks can make learning more meaningful to students and help increase the motivation to gain knowledge about specific topics. Project-based learning can combine practical work with real-world tasks to improve student engagement and performance (Robinson (2012)).

However, in the domain of networked and autonomous driving, access to practical work is rather difficult, since it often requires an expensive setup with vehicles and infrastructure. On the other hand, working solely with simulations reduces the connection to the real world due to unrealistic or incomplete modeling. This arising gap between simulation only and real-world development can be filled with model-scale laboratories. They combine the advantages of field testing, such as more realistic driving physics, with lower maintenance and acquisition costs.

Our open-source Cyber-Physical Mobility Lab is such a laboratory that tries to make research and education more accessible. True to the motto: See your ideas develop into reality, we strive to provide a platform for the development of networked and autonomous vehicles. Nevertheless, the access is limited due to various reasons. For one, the lab considered as a resource can be unavailable, if it is already in use. Its capacity is limited as only one development team at a time can use it. Further, due to its stationary build in Aachen (Germany), travel to the lab is required. This fact alone creates a high inhibition threshold for users around the world to engage with the CPM Lab. On top of that, other non-functional limitations complicate the usage of such a laboratory. Among them is the ongoing COVID-19 pandemic, which caused our laboratory to remain unused for a long period. This emphasized the necessity of our already running project CPM Remote, where we developed a remote access to provide a more efficient and globally accessible interface to the CPM Lab.

This new setup raises opportunities and challenges for teaching, as the fact that the student is no longer physically present in the lab is a boon and a bane. One of the challenges is that the interface to the lab, as well as the course needs to be designed in a way such that the relation to the real world is still present.

* This research is supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) within the Post Graduate Program GRK 1856 Integrated Energy Supply Modules for Roadbound E-Mobility. We acknowledge the financial support for this project by the Exploratory Teaching Space of the RWTH Aachen University (Germany)
This paper describes our laboratory setup with its remote access and specifies two courses of our Chair to provide examples of use. Since our lab as well as the remote access is open source, this paper can be used by other educational institutions to develop their own courses using the CPM Lab.

2. RELATED WORK

The increasing interest in digital laboratories is not only driven by the ongoing pandemic, but also by the fact that they reduce cost and increase efficiency. Laboratories like the CPM Lab are used worldwide and in various fields like education, robotics, networked and autonomous vehicles, and artificial intelligence. Examples of comparable labs are the Robotarium (Pickem et al. (2017)), Duckietown (Paull L. et al. (2017)), Scaled Smart City (Stager et al. (2018)) and F1tenth (F1/10) (O’Kelly et al. (2020)). These often consist of a standalone usable simulator and the lab itself in which experiments can be executed in the physical world. Even though these labs differ in the application domain, concepts, usage, and availability for third parties, they share the goal of making research and education possibilities publicly available. This is done either by making it affordable, respectively easy to set up, or by providing access to their lab via remote access.

All of the above-mentioned laboratories have already created courses based on their platform. F1tenth, for example, points out how a hands-on course can combine various advantages to teach the foundations of autonomous systems (Agnihotri et al. (2020)). A comparable course is provided by the Institut of Automatic Control at the RWTH Aachen University with their IRT-Buggy (Reiter et al. (2014)). However, in this chapter, we focus on describing the Robotarium and Duckietown, as they have more conceptual overlap with our CPM Lab in terms of domain and accessibility.

Duckietown is an open-source project initiated by MIT in 2016. Self-driving vehicles named Duckiebots drive on roads through the Duckietown. Roads can have curves, lines, intersections, street signs and are decorated with rubber ducks. To use this project, tracks, Duckiebots, and other elements have to be bought and set up. Different kits are available starting at a few hundred USD going up to 11,000.00 USD. One is the ‘starter pack’ for the Duckietown Massive Online Open Course (duckietown.org (2021)). This course is available online for free, but the hardware needs to be bought in advance. Furthermore, classroom kits are available which consist of learning materials, hardware resources, and support material. Those cover all needs for teaching a Duckietown lecture.

The main teaching objectives of their courses are computer vision, robot operations, object detection, and onboard localization. Today, some classes are adapted by the ETH Zürich, Université de Montréal, and the Toyota Technological Institute in Chicago. Already more than 700 students in 10 different countries took part in Duckietown classes and were able to collaborate since they all share the same platform. This emphasizes the need and acceptance for such a platform. However, within the field of autonomous driving, Duckietown focuses on perception, while the CPM Lab targets networked vehicle control. This paper aims to provide an approach to fill this gap.

3. CYBER-PHYSICAL MOBILITY LAB

The CPM Lab (Kloock et al. (2020a)) is designed in a scale of 1:18. As an open-source platform for the domain of networked and autonomous vehicles, the CPM Lab acts as a platform to conduct experiments of networked algorithms. The algorithm under test runs on a development computer that interfaces with the other components.

Figure 2 illustrates the components of the CPM Lab:

(1) **20 model-scale vehicles (μCars)**

Figure 1 shows one of the μCars. The networked and autonomous vehicles with Ackermann steering geometry scale a passenger car in 1:18. The chassis is the XRAY M18 Pro LiPo platform (XRAY (2010)), which comes with a motor for propulsion and a servo motor for steering. We transformed the vehicle by adding an inertial measurement unit, an odometer and microcontrollers for control and communication (Scheffe et al. (2020)).

(2) **Indoor positioning system (IPS)**

The IPS – consisting of a camera and the main com-
computer—measures the position and orientation of all vehicles (Kloock et al. (2020b)). These measurements provide global localization, and therefore correspond to a global navigation satellite system (GNSS) signal in real experiments.

(3) **External computation devices**

The development computers 1 to N that execute the control algorithms control the vehicles 1 to N.

(4) **Main computer**

The main computer controls and monitors experiments.

(5) **Map**

The driving area of the CPM Lab is 4.5 m × 4 m. The map which displays the road structure is printed on foil and can be switched.

(6) **Router**

The communication between the components is realized with LAN and WLAN.

The vehicles can be operated in three modes, depending on the control signal received from the external computation devices, as follows:

(1) **Direct control mode**

The external computation device sends dimensionless inputs to the vehicle. The vehicle applies them directly to the motor and servo motor.

(2) **Path tracking mode**

The external computation device sends a sequence of points that determine a path and a reference speed to the vehicle. The vehicle controls motor and servo motor to follow the reference speed and path.

(3) **Trajectory tracking mode**

The external computation device sends a sequence of points, velocities and timestamps to the vehicle. The vehicle controls motor and servo motor to follow the reference trajectory.

The control signal as well as measurement data is communicated via the Data Distribution Service (DDS) standard (Pardo-Castellote (2003)). DDS is a networking middleware that implements a publish-subscribe pattern for decentralized communication. Besides its robust and responsive nature, communication via DDS has two major benefits. First, a high-level controller (HLC) can be developed in any programming language as long as it has a DDS interface. Second, our simulation of the CPM Lab uses the same interface as the real system, which enables a seamless transition from simulation to experiment. Therefore, the simulation enables rapid prototyping of control algorithms, as work in progress can be tested rapidly and continuously in simulation.

The construction plans as well as the software are open source and linked in the beginning of this paper. These assets allow a deeper understanding of the lab’s infrastructure, and, with the required resources, enable rebuilding the CPM Lab at another location. Considering that not everyone has access to these resources, the simulation is a great place to start testing algorithms for networked and autonomous vehicles.

### 3.1 Simulation Environment

In the simulation environment, the IPS and the μCars are replaced with their simulated counterparts. The components communicate via DDS locally on the simulating computer. The simulation of the CPM Lab can be used in the ready-to-run virtual machine, or it can be set up with a single installer script in Ubuntu (see our website and software repository linked in the beginning of this paper). An HLC that runs in simulation can be seamlessly transferred to the real CPM Lab without any adjustments. Furthermore, it is possible to extend an experiment with simulated vehicles, which lets simulated and real vehicles interact. The CPM Lab treats both as equal, allowing experiments in even larger scale.

While the simulation is great for rapid prototyping, its ability to reflect reality accurately is limited. One of the advantages of the CPM Lab is that algorithms for networked and autonomous vehicles can be evaluated in real experiments. If the resources to recreate the CPM Lab are not available, conducting these experiments would require traveling to Aachen or accessing the CPM Lab remotely.

### 3.2 CPM Remote

![Fig. 3. Development cycle with CPM Remote.](image-url)
In order to reduce the inhibition threshold for engaging with the CPM Lab for the development of networked and autonomous vehicles, we extended the aforementioned virtual machine. We embedded the software into a web server and developed a full-stack web application that allows operating the CPM Lab. The whole setup process including installation is replaced by a quick registration on the CPM Remote web page: www.cpm-remote.de (Mokhtarian and Alrifae (2022)). New users can explore this web IDE with the aid of a guide-through tutorial which explains the first steps based on a HelloWorld example. Hence, with just a few clicks, everybody is capable of running an experiment in the CPM Lab within minutes. Further, the inbuilt editor allows quick modifications which can be tested immediately since the simulation environment including the visualization is integrated as well.

Using the CPM Remote platform only requires a browser and performs regardless of the user’s hardware, as the computation is outsourced to our servers. This is supposed to enable target groups with less powerful hardware, or even just a tablet, to use the CPM Lab.

Figure 3 illustrates the development cycle of CPM Remote. The provided Code and Play interface allows fast prototyping. Once a user wants to submit an experiment, in order to be executed in the CPM Lab, s/he needs to specify some parameters the like duration of the experiment. Afterward, a validation process is started on our server to make sure the experiment is collision-free. If that’s the case, the experiment gets deployed in the lab and the results, which include a video and other recorded data, are returned to the user. Currently, we are working on a real-time interface to the lab for users who need faster prototyping, as the students of our chair. Furthermore we developed scenarios for benchmarking in networked and autonomous driving and plan to host a public competition named CPM Olympics (Mokhtarian et al. (2022)).

4. TEACHING WITH THE CPM LAB

The CPM Lab was designed as a platform for applying networked control system (NCS) theory in research and education. The following sections present two courses that we teach with the CPM Lab.

4.1 Control in Networked Vehicles

Control and Perception in Networked and Autonomous Vehicles (CPNAV) is a course that consists of a lecture part, which presents theory, and a lab part, where students apply this theory into practice. It is designed as a one-semester graduate course, but can be condensed with reduced content to a one-week intensive PhD course. The course materials are available from our website linked in the beginning of this paper.

In the course, we teach on the basis of the Sense – Plan – Act paradigm (Gat et al. (1998)). The lecture topics are mapped to core components of networked and autonomous vehicles in Figure 4. In Perception, an introduction to machine learning for visual sensing is given. In Control and Optimization, students learn about model-based trajectory planning methods for vehicles. In Network and Distribution, we advance from the perspective on one vehicle to the interaction of multiple vehicles. Centralized and distributed approaches are presented. In Software Architectures and Testing Concepts, possibilities to evaluate networked and autonomous systems are discussed. The participants of the course CPNAV are graduate students of either Computer Science, Automation Technology or Computational Engineering Science. Therefore, their specialty typically lies in different areas. One might have a deeper understanding of control, but lack in programming, and vice versa. Aware of the fact that students have strengths in different topics, we group them to interdisciplinary teams such that each team can cover every topic. Successful participation in the course requires the students to communicate between disciplines. Besides presenting classically, we encourage team work in group discussions, cooperative games on NCS, flipped classroom group presentations and practical exercises in MATLAB.

The core topics of the lecture are deepened in accompanying lab exercises, where students develop an HLC for vehicle platooning. The students develop the HLC in MATLAB and test it in the CPM Lab, which encompasses the following steps. First, they develop a model structure for longitudinal vehicle motion of the vehicle in path tracking mode. Afterwards, they identify the parameters for the longitudinal vehicle model in experiments. Using the resulting model, the students develop a controller using model predictive control (MPC) for distance control of a single vehicle. Following the course content, the control of a single vehicle is extended to networked control. By creating a centralized model of the platoon from the model of one vehicle, the students program a centralized controller for vehicle platooning with MPC. The final part is to distribute the control problem to each vehicle in the platooning with priority-based non-cooperative distributed model predictive control (Alrifae et al. (2016, 2015, 2013)).

As in the lecture, the students benefit from interdisciplinary teams in the lab exercise. Aside from the benefit of having an expert for different topics, this also trains interdisciplinary communication. The concept of eXtreme Programming (Beck (2000)), with core elements like evo-
olutionary development in small increments, always having working code or pair programming, serves as a guide for software development in the lab exercises. The lab exercises are an example of project-based learning (Robinson (2012)), where the success of the project serves as feedback for the students’ work.

The students take oral exams, in which they present their lab work with a video of their networked control algorithm in action, before we examine their knowledge on the lecture topics.

The lecture and lab are designed for physical interaction, but we have taught the course digitally as well. Even though the students had no access to the lab during the winter semester of 2020/2021, they could develop an HLC in simulation with a provided virtual machine. The final platooning controller was submitted by the students and run in the CPM Lab by us.

4.2 Software-Project Practical

Bachelor computer science students of the RWTH Aachen University have to apply to a ‘Software-Project Practical’ course. Within the curriculum, this course is suggested for the 5th semester as it requires basic knowledge in programming. The focus of this course lies in the application of theoretical knowledge, as well as the organization and implementation of a software project. At our chair, the course is structured in a customer/developer relationship rather than student/teacher. Five groups of six members each, are supposed to follow the agile development method Scrum (Schwaber (1997)) in order to organize their software project.

The project’s goal is to develop a package delivery service. Up to 20 vehicles are supposed to collect packages in the most efficient way possible. Here, the students are responsible for the problem definition, the solution as well as the metrics to evaluate their performance. First, they implement a generator which creates random delivery points (packages) over time. The vehicles then have to continuously update its route and decide in which order they should collect the assigned packages. This results in the traveling salesman problem (Flood (1956)) which is a NP-Hard problem and thus not to be solved optimally in real-time. Finally, metrics have to be determined and recorded in order to evaluate the performance of the algorithms. During the semester, our students have 10 weeks for the active development of this project. This results in 10 milestones (sprints), some of which are described in the following.

(1) Software architecture
Many of the following processes run in parallel. Furthermore, modules like the routing algorithm should be exchangeable so different algorithms can be compared to each other. The resulting requirements should be addressed in the software architecture design.

(2) Package generation
Packages shall spawn in random temporal intervals at random coordinates. It needs to be ensured that there are packages available at any time. However, too many packages would facilitate the problem as there would always be a package close by the vehicle. Hence, a modular and appropriate generator needs to be designed.

(3) Distribution of packages
The first important heuristic assigns packages to vehicles. Here, the students are also free to decide how many and how often new packages are assigned to vehicles.

(4) Traveling salesman problem
Once a vehicle has a number of assigned packages, it needs to decide for an order to collect them. The complexity of this resulting TSP problem increases with the number of packages that are assigned to a vehicle.

(5) Trajectory planning
Each vehicle needs to plan its own trajectory. Several factors like the planning horizon impact the computation cost of this algorithm.

(6) Collision detection
An observer which detects (future) collisions has to consider all planned trajectories.

(7) Collision avoidance
A reasonable approach to adjust the planned trajectories is crucial in order to avoid expensive computations.

(8) Evaluation
Metrics like driven meters, collected packages (per vehicle), average waiting time (per package) and so forth need to be recorded and evaluated.

Figure 5 illustrates some of the milestones on an exemplary map with a crossing and a ring road. The colored points show the mapping to the vehicles. A number indicates the order by which these points are collected. Furthermore, additional information, like the planned trajectory (blue lines), can be visualized in order to facilitate the devel-
opment. At the end of the semester, the students present their package delivery service with a live demonstration in the CPM Lab.

5. CONCLUSION

This paper described how an open-source and remotely accessible laboratory can be used in education. We showed at the example of two courses that the CPM Lab provides the capability to host digital courses on mobile networked systems. The courses encourage project-based learning of core topics in networked control. With the provided course materials, the documentation and open-source, remote availability of the CPM Lab, it offers an opportunity for any educator to design a course where the students apply new concepts in a laboratory environment from their home.

REFERENCES


