Low-cost Carry-home Mobile Platforms for Project-based Evaluation of Control Theory ⋆

Armin Steinhauser * **Maarten Verbandt** *
Niels van Duijkeren * **Ruben Van Parys** * **Laurens Jacobs** *
Jan Swevers * **Goele Pipeleers** *

* *Department of Mechanical Engineering, KU Leuven, Belgium.*
(e-mail: <firstname>.<surname>@kuleuven.be).

Abstract: This paper presents mobile platforms that were recently designed in support of an introductory control course. Through dedicated assignments, the students are guided to implement and validate all parts of the course on a setup, ranging from basic time-domain system identification, over root locus analysis and loop shaping PID design, to state feedback, state estimation and Kalman filtering. The platforms are flexible, allowing for numerous extensions and variations; cheap, allowing for a large pool of setups from which the students can borrow platforms to take home; and of sufficient quality, allowing the students to get maximal insight in the course material. The setups are easy to set up and administer using the supporting material provided by the authors.

Keywords: Control engineering education; low-cost laboratory; embedded control system.

1. INTRODUCTION

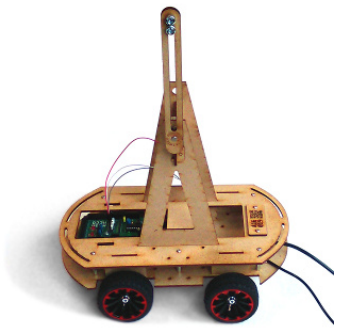
Teaching a control theory course to Mechanical Engineering students can be challenging. Their primary interests lie in application domains such as manufacturing, thermodynamics, robotics, automotive or aerospace, and they are often impeded by the mathematics involved in control: Laplace and Fourier transforms, state-space models and transformations, etc. In addition, many students struggle with the level of abstraction of the course, which prevents them from appreciating the value of control to their field of application. Hands-on lab sessions are a vital complement to such a course in order to reinforce the students' understanding of the theoretical principles, to strengthen their control design capabilities, and to spark true interest and appreciation for control (Leva, 2003; Feisel and Rosa, 2005; Reck, 2016).

The current paper describes test setups we recently designed in support of the introductory control course B-KUL-H04X3A, a mandatory course in the first year of the Master of Mechanical Engineering at the KU Leuven. We established a pool of 40 setups from which the students can loan a setup to perform experiments whenever and wherever they want. Through dedicated assignments, the students are guided to implement and validate all parts of the course on a setup. They will work on these assignments in teams of two and, as the course evaluation will be based on the assignments, with limited support from the teaching staff.

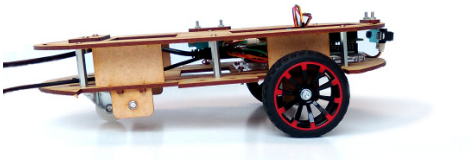
Recognizing the value of hands-on experience, various control theory teaching teams have developed dedicated lab kits over the last decade (Sarik and Kymissis, 2010; Gunasekaran and Potluri, 2012; Ovalle and Cómbita, 2014; Chancharoen et al., 2014; Hill, 2015; Migchelbrink et al., 2015; Reck and Sreenivas, 2015; Krauss, 2016; Bay and Rasmussen, 2016). Each of these kits has been tailored to the particularities of the course and the aspirations of the team. In our case, the requirements are the following: First and foremost, the setups must enable hands-on experience with as many techniques covered in the course as possible, ranging from basic time-domain system identification, over root locus analysis and loop shaping PID design, to state feedback, state estimation and Kalman filtering. In addition, a diverse set of setups is ambitioned in order to obtain a wide variety of assignments as a safeguard to excessive sharing of solutions and plagiarism in the course evaluation. Therefore a flexible platform was opted for that allows for numerous extensions and variations. The first year we start with two design variants, shown in Fig. 1, while next summer more design variations are planned. In addition, the students are encouraged to be creative and come up with alternative assignments themselves.

Envisaging a large pool of setups limits the available budget per device. As a consequence, the cost of the components is carefully traded off against their quality and the overall robustness of the design. As the setups are

⋆ This work benefits from KU Leuven-BOF PFV/10/002 Centre of Excellence: Optimization in Engineering (OPTEC); the Belgian Programme on Interuniversity Attraction Poles, initiated by the Belgian Federal Science Policy Office (DYSCO); Flanders Make projects ROCSIS: Robust and Optimal Control of Systems of Interacting Subsystems and RoFaLC: Robust and Fast Learning Control; and project G0C4515N of the Research Foundation-Flanders (FWO-Flanders). Niels van Duijkeren is a fellow of the TEMPO Initial Training Network (FP7-ITN-TEMPO (607 957)). Ruben Van Parys is a PhD fellow of the Research Foundation-Flanders. The MECO Research Group is an associated research lab of Flanders Make.



(a) Pendulum version



(b) Swivel version

Fig. 1. Pictures of the two mobile platform variants.

intended to last for a number of years, they should be sufficiently durable. Cheap components with excessive friction or backlash are avoided to prevent students from getting bogged down by irrelevant practical issues and too severe hardware limitations. Laser-cut MDF was chosen for the body parts of the platforms. This can be manufactured in house such that the students can repair broken parts and extend the setups themselves. Finally, the setups are supplied with sufficient on-board computational resources to allow their future usage for sophisticated control experiments in a more advanced control course.

Although an extensive survey of existing lab kits (see the aforementioned references) revealed no direct match with our requirements, it did provide us with invaluable inspiration for our design and hinted us to interesting tools and sources. In a similar way, we hope that this paper will be of value to other teams that are in the process of designing experimental setups and dedicated assignments to their control theory course. To this end, all information regarding the platforms, as well as all supporting material is made publicly available on `https://github.com/meco-group/mecotron`.

In the remainder of the paper we first describe the hardware and software design of the setups. Afterwards, we elaborate on their usage in the course and the corresponding student assignments. We close the paper with concluding remarks.

2. HARDWARE DESIGN

This section discusses details on the hardware of the developed platforms. The first part gives an overview of the specified objectives and the overall platform design. The second part elaborates on the interfaced electronic components and the sensors used.

2.1 Design Overview

To allow for a valuable hands-on experience, a major objective is to make the design sufficiently robust and therefore also easily maintainable, e.g. by replacing broken parts without difficulties. Furthermore, to enable various applications, we aim for two different platforms - a pendulum and a swivel version, which consist to a great extent of identical parts. Somewhat contradictory to these objectives, we additionally seek to minimize the costs, such that the cost per device and therefore the hardware design of the mobile platforms plays a critical role. Trading robustness off against cost by careful selection of the hardware components allows us to keep the total cost per platform as low as €100 and €120 for the swivel and the pendulum version, respectively.

Due to the mentioned requirements we choose to manufacture a laser-cut 3 mm MDF frame that is rigid but also inexpensive. Additionally, this enables fast manufacturing of new parts and gives students the opportunity to extend the platforms. An exploded view of the pendulum version is shown in Fig. 2, sharing the majority of the parts with the swivel version. The basis of every platform consists of a bottom (1a) and top (1b) frame that are connected by inserts and spacers. Attached to the bottom frame and the front inserts are two 6 V DC motors (5) - including gearboxes and magnetic encoders - that drive the platform via a pair of RC wheels. The pendulum version additionally includes an A-shaped rack (2) that slides in to hold a pendulum (8). Another pair of wheels is mounted on an axle held by the rear inserts to constrain the platform to linear motions. For the swivel version the rear axle and wheels are replaced by a swivel caster wheel for increased maneuverability. For the purpose of managing a large number of platforms, a numberplate with an identifier and a QR code is fixed to the top frame. The latter is used for automated booking of the platforms in a central database, to keep track of their whereabouts.

2.2 Electronics and Sensors

At the heart of every platform an Arduino MEGA 2560 (3) is used. On top, a custom breakout board - called the MEGA MECO (4) - is stacked that holds a PWM-driven L293D quadruple half-H driver to power the motors and provides easy access to various analog and digital inputs of the Arduino together with a selectable supply voltage for sensors of 3.3 or 5 V. Additionally, a receptacle to connect a Bluetooth module is provided. Unlike using one of the many already existing breakout boards, building our own enables us to tailor the design to our needs, such as placing headers for the motor terminal connectors. Furthermore, the design was kept simple to decrease the chance of defects and the sensitivity to faulty parts as much as possible. The motors of the platforms are equipped with magnetic encoders that are read by two digital inputs of the Arduino, one of them triggering an interrupt on a change of state. Taking the gear ratio of 1:34 into account, this results in 1496 counts per wheel revolution that can be read bi-directionally. Additionally, the pendulum version holds one infrared sensor (6) for distance measurement and an analog-output, multi-turn potentiometer (7) that allows the measurement of the pendulum angle. Using the

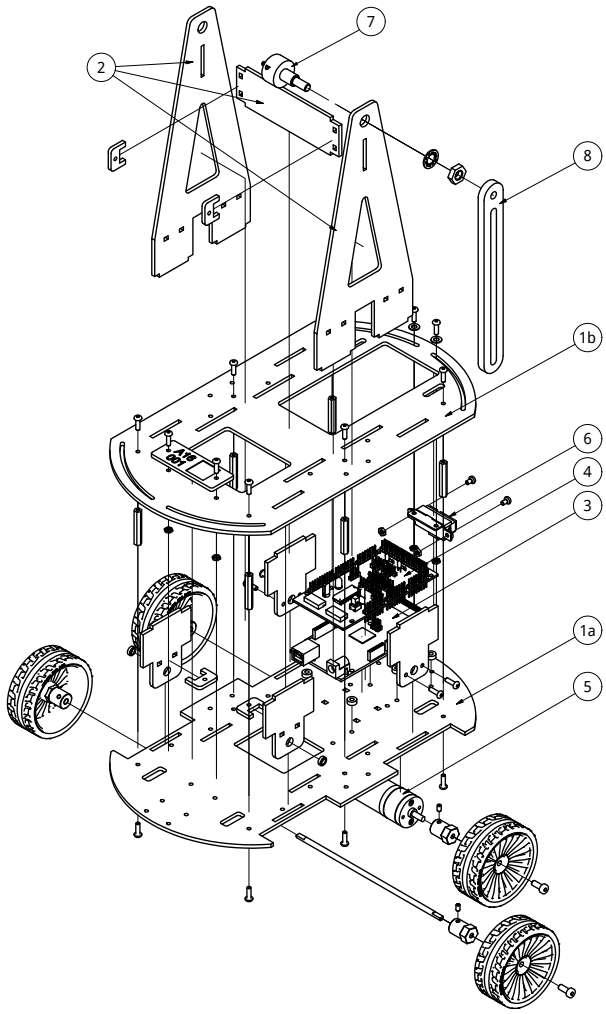


Fig. 2. Exploded view of the pendulum platform with main parts numbered; 1a, 1b: frame, 2: pendulum rack, 3: Arduino MEGA 2560, 4: MEGA MECO, 5: motor incl. gearbox and encoder, 6: infrared sensor, 7: potentiometer, 8: pendulum

Arduino's 10-bit analog to digital converter, this yields a resolution of approximately 0.352 deg. The swivel version, on the other hand, holds two infrared sensors - one in the front and one on the side - to enable localization in an environment. Since the USB connection is limited to supply 5 V at a maximum of 500 mA, an external power supply with 6 V and 12 W is used that can be substituted by a battery pack consisting of 5×1.2 V rechargeable AA-batteries.

3. SOFTWARE DESIGN

This section describes the overall software framework. The first part covers the need to develop a custom framework to operate the platform. Next, the general layout of the software and its key aspects are explained. The last part focuses on accessing and processing data using third-party software.

3.1 Motivation for a New Framework

Interfacing embedded platforms such as the Arduino with a PC is certainly not uncommon. The fact that both

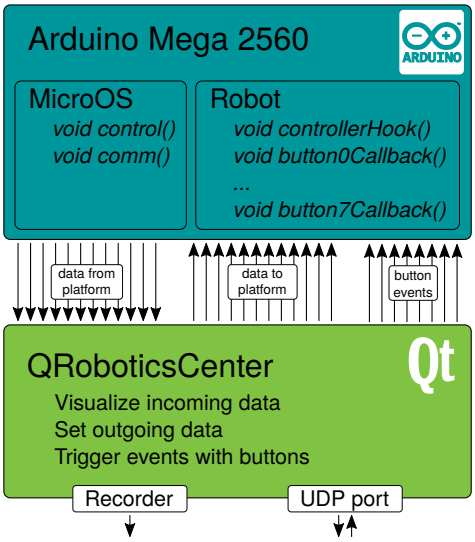


Fig. 3. Overview of the software architecture with the software running on the platform (top) and the PC side (bottom).

Mathworks and National Instruments provide their own way of interfacing Arduino boards supports this claim. However, relying on commercial software introduces extra costs, limits flexibility and usually brings some toolchain specific overhead. These are the main reasons to develop a custom software framework. No extra costs are involved as the software relies on open source projects only, namely Arduino, its many libraries and the QT framework for designing a graphical user interface (GUI) which runs on Windows, Unix and Mac. Having access to all pieces of the software results in a tailored solution that minimizes the overhead.

3.2 Software Architecture

The software architecture is displayed in Fig. 3. The software on the Arduino is based on a custom operating system called *MicroOS*. It governs the scheduling of different tasks running on the Arduino, which are mainly communication and control. Apart from the operating system, a *Robot* class is provided that serves as an interface to all the hardware components such as motors, encoders and other sensors. Moreover, providing one main class students can start from prevents them from getting lost in a bunch of files. Within the *Robot* class, the central function is called *controllerHook()*, in which students implement their control algorithm. This function represents the control task and runs at a configurable rate of 100 Hz.

At the PC side, the used software is called *QRobotics-Center*. Such an application is not strictly necessary since the setup can operate on its own. However, this application facilitates the communication and data exchange between PC and platform. This allows students to focus on the controller design and implementation, rather than dealing with low-level issues. Information from the platform is transmitted via 12 virtual channels, each carrying a scalar value. The GUI (Fig. 4) provides the means to visualize this data and therefore helps the user to understand the platform's behaviour and track down faults. Another 12 virtual channels take care of the communication from

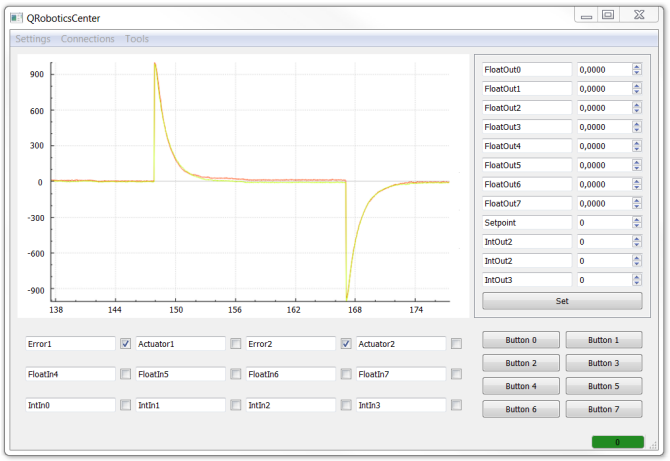


Fig. 4. Overview of the graphical user interface with its visualization capabilities and commands to the platform.

the PC to the platform. These are for instance useful to manually tune the parameters of a PID controller or change the setpoint for a control loop. The GUI also implements eight virtual buttons which are linked directly to the corresponding callback functions in the *Robot* class. These are meant to represent events such as enabling the controller or resetting the encoders.

3.3 Data Acquisition

Visualizing the output from the platform only provides qualitative insight. This however is usually not sufficient, for instance when identifying a model. Therefore the output channels of the setups are linked to a recorder, which writes the incoming data to an XML formatted file at 100 Hz. The preferred third-party software then reads the file and prepares it for processing. For this purpose MATLAB code is provided to parse the recordings and to extract the data of interest. In addition, time-varying input data can be read from a CSV file and transmitted to the platform. This is for instance useful to supply a reference trajectory or an offline computed feed-forward signal.

QRoboticsCenter also provides a direct way of interfacing the setups. The application creates a UDP port from which other programs such as MATLAB or Python can easily read the data. Moreover, external applications are also allowed to write data to this port, resulting in two-way communication. This feature proves itself useful when the control strategy is too resource consuming to run on the Arduino itself.

4. STUDENT ASSIGNMENTS

This section describes the assignments that students perform with the setups. In a first set of basic assignments they identify a model of the motors and design controllers to make the cart track a velocity and position reference. These basic components are then used in two advanced assignments. In the first, the cart is supposed to follow a reference trajectory in two-dimensional space where its position and orientation are determined by a Kalman

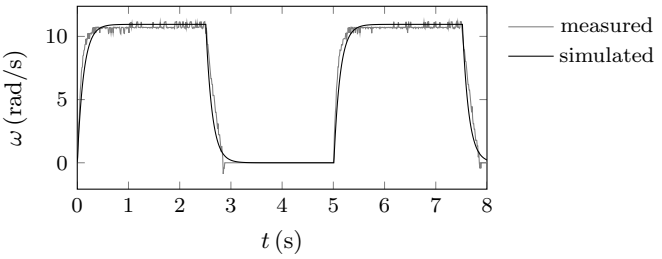


Fig. 5. Validation of the motor identification by means of a step response.

filter. In the second assignment the students control an inverted pendulum mounted on the platform.

4.1 Basic Assignments

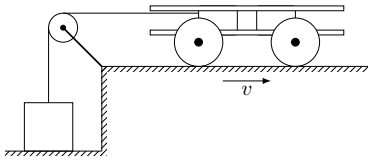
The first basic assignment, corresponding to the course content, is a time-domain identification of the motors. The students derive a realistic, yet practicable low order model, choose a suitable excitation signal, and finally fit the model to the measured response. Students validate the model accuracy through a comparison of simulation and experimental results, such as illustrated in Fig. 5. Furthermore, the students are expected to investigate possible sources of non-linearities that are not covered by the chosen model.

Once a model for the motors is obtained, a controller is designed to track a velocity command. This controller results in a closed-loop system with maximized bandwidth while preserving a reasonable stability margin. The steady state error for a constant reference and force disturbance should be zero. The controller design is achieved by applying loop shaping techniques taught in the course. The students validate their controller experimentally, for instance by conducting an experiment as illustrated in Fig. 6. The cart is supposed to lift a mass with constant velocity. Around $t = 4\,\mathrm{s}$ the mass is lifted from the ground. Despite the constant force acting on the cart, the velocity controller manages to follow the setpoint of $0.1\,\mathrm{m/s}$.

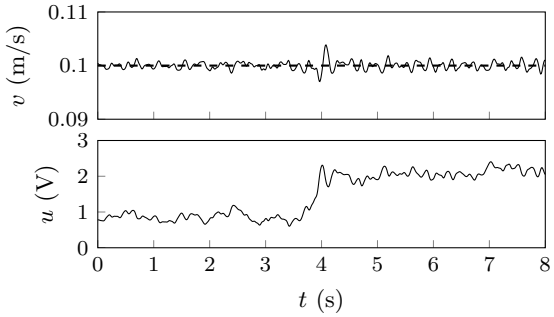
In a next step a position control loop is built around the velocity controlled system. The students first design a proportional position controller based on root locus analysis. Secondly, they improve the performance by designing a more involved controller based on loop shaping. These controllers are initially validated by using position measurements from the motor encoders. As an extension the students are asked to design a state estimator that merges measurements from the encoders and from a frontal infrared sensor measuring the distance to a wall.

4.2 Navigation & Kalman Filtering

One of the advanced assignments is to implement a navigation controller for the swivel platform which tracks a prescribed reference path. The assignment involves the implementation of an extended Kalman filter to estimate the position of the platform, as well as the design of a feedback controller to follow the path. Figure 7 gives an illustration of the scenario. The first step students take is to construct a (non-linear) kinematic model of the vehicle. This model governs the trajectories (x_c, y_c, θ) with



(a) Experiment overview



(b) Cart velocity v and motor voltage u

Fig. 6. Experiment validating the velocity controller in the presence of a constant force disturbance. Around $t = 4\,\mathrm{s}$ the mass is lifted from the ground. After a small transient, the velocity controller keeps track of the setpoint of $v = 0.1\,\mathrm{m/s}$ by increasing the motor voltage u in a correct way.

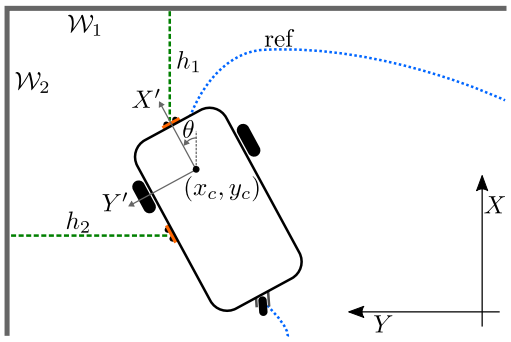


Fig. 7. Illustration of the navigation task for the swivel platform. The orientation of the walls $\mathcal{W}_1$ and $\mathcal{W}_2$ is assumed to be known.

the rotational velocities of the wheels (ω_L, ω_R) as control inputs:

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} := \begin{cases} \dot{x}_c &= v\cos(\theta) \\ \dot{y}_c &= v\sin(\theta) \\ \dot{\theta} &= \omega \end{cases}, \text{ with } \begin{aligned} v &= \frac{(\omega_L + \omega_R)\mathrm{r}_w}{2} \\ \omega &= \frac{(\omega_R - \omega_L)\mathrm{r}_w}{\mathrm{a}_w} \end{aligned},$$

where r_w denotes the radius of the wheels and a_w the wheelbase of the platform. Using the wheel velocities as control inputs assumes that a velocity controller from the basic assignments is already in place.

The second step is to formulate the measurement equations for the distances h_1 and h_2. The measurements of the infrared sensors can be modelled as the closest distance between the position of the sensors $(x_{s,1}, y_{s,1})$, $(x_{s,2}, y_{s,2})$ and the walls $\mathcal{W}_1$, $\mathcal{W}_2$,

$$\mathbf{y} := \begin{cases} h_1 &= \min_w \{\mathrm{dist}([x_{s,1}(\mathbf{x})\ \ y_{s,1}(\mathbf{x})]^\intercal, w) \mid w \in \mathcal{W}_1\} \\ h_2 &= \min_w \{\mathrm{dist}([x_{s,2}(\mathbf{x})\ \ y_{s,2}(\mathbf{x})]^\intercal, w) \mid w \in \mathcal{W}_2\} \end{cases}.$$

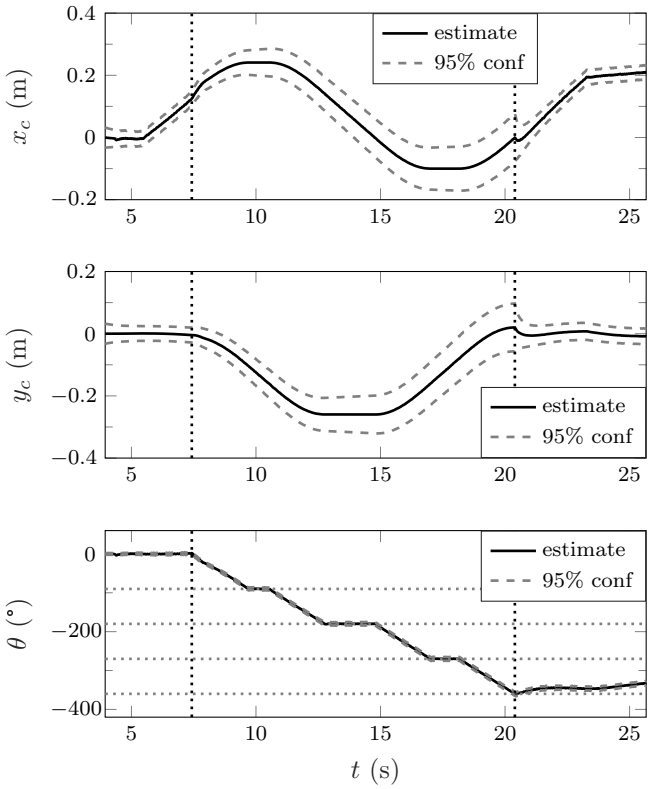


Fig. 8. Results of navigation control with extended Kalman filter. The state estimates for x_c, y_c, θ are shown with 95%-confidence bounds. Vertical dotted lines indicate the start and end of the dead reckoning phase.

Note that this approach implicitly assumes that the cart is correctly oriented with respect to the wall. We choose to use straight, perpendicular walls for which an explicit expression for $\mathbf{y}$ can be obtained.

An extended Kalman filter is implemented on the Arduino, where students are provided with a code template to start from. They are allowed to use a linear algebra package for matrix operations, such as the *BasicLinearAlgebra* library. If the cart is suitably oriented with respect to the walls, measurements are used to correct the state estimate, while otherwise we perform dead reckoning.

The navigation control is implemented with a linear state feedback controller, exploiting linearity of the system dynamics in the local cart frame (X', Y'). Furthermore, the feedback action can be combined with a feed-forward signal to improve the performance. The reference trajectory can be hard-coded on the Arduino, it can come from the connection with QRoboticsCenter, or potentially originate from on-board intelligence programmed by the student. For the results in Fig. 8 we use a reference provided by QRoboticsCenter which feeds the content of a CSV file line-by-line to the Arduino. The figure shows the results of the navigation controller following a square-shaped path. Kalman innovations are only performed on the first and last straight segment of the trajectory. Due to the dead reckoning we observe that the confidence bounds on the estimate widen. When the Kalman innovations are reinstated, the state estimate is corrected and the feedback controller eliminates the tracking error.

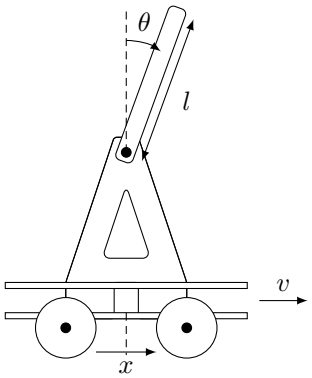


Fig. 9. Simplified representation of a velocity controlled cart with an inverted pendulum.

4.3 Control of an Inverted Pendulum

In this assignment the cart will be used to control an inverted pendulum. By measuring the pendulum angle and by correctly steering the cart's velocity, the open-loop unstable system can be rendered stable. In order to obtain a working controller, the students follow the steps described below.

In a first step an appropriate model for the system, illustrated in Fig. 9, is derived. After deriving a theoretical non-linear model and linearizing it, the students come up with the following transfer function, giving the relationship between the input velocity v and pendulum angle θ

$$\frac{\Theta(s)}{V(s)} = \frac{1}{l}\frac{s}{s^2 - g/l}\,,$$

where l represents the pendulum length and g the gravitational acceleration. Since the system is unstable, simple identification techniques to determine the system parameters are not applicable. However, $\sqrt{g/l}$ can be easily measured as it is the natural frequency of the pendulum around its stable equilibrium. In order to stabilize the system, the students design a state feedback controller. Therefore the system is first written in a state-space form with three states of which one is the cart position. A possible choice of states is $(x_1, x_2, x_3) = (x, \theta - \frac{x}{l}, \dot{\theta} - \frac{\dot{x}}{l})$, for which the state-space description becomes

$$\begin{aligned}\dot{x}_1 &= v\,,\\ \dot{x}_2 &= x_3\,,\\ \dot{x}_3 &= \frac{g}{l}x_2 - \frac{g}{l^2}x_1\,.\end{aligned}$$

Using a pole-placement approach, a state feedback controller is designed that controls the states (x_1, x_2, x_3) to zero. An appropriate place for the closed-loop poles are Butterworth pole locations. The bandwidth is maximized experimentally. Since the controller requires the feedback of all states, a state estimator is designed as well to provide the state information. This estimator can also be designed by pole-placement. In order not to influence the closed-loop dynamics, these poles are typically chosen as two to six times faster than the control poles.

5. CONCLUSION

This paper describes mobile platforms we recently designed in support of an introductory control course. The setups are based on a flexible platform that allows for numerous extensions and variations, and the components are selected such that the platforms are cheap, yet sufficiently robust and of high quality. Through dedicated assignments, the students are guided to implement and validate all parts of the course on a setup. This way, we hope to reinforce the students' understanding of the principles of control, to strengthen their control design capabilities, and to spark true interest and appreciation for control. To support usage by other teaching teams, all information on the setups, as well as all supporting material is made available online on `https://github.com/meco-group/mecotron`.

REFERENCES

Bay, C.J. and Rasmussen, B.P. (2016). Exploring controls education: A re-configurable ball and plate platform kit. In *Proc. of the 2016 American Control Conference (ACC)*, 6652–6657.

Chancharoen, R., Sripakagorn, A., and Maneeratana, K. (2014). An Arduino kit for learning mechatronics and its scalability in semester projects. In *Proc. of the 2014 International Conference on Teaching, Assessment and Learning (TALE)*, 505–510.

Feisel, L.D. and Rosa, A.J. (2005). The role of the laboratory in undergraduate engineering education. *Journal of Engineering Education*, 94(1), 121–130.

Gunasekaran, M. and Potluri, R. (2012). Low-cost undergraduate control systems experiments using microcontroller-based control of a dc motor. *IEEE Transactions on Education*, 55(4), 508–516.

Hill, R.C. (2015). Hardware-based activities for flipping the system dynamics and control curriculum. In *Proc. of the 2015 American Control Conference (ACC)*, 2777–2782.

Krauss, R. (2016). Combining Raspberry Pi and Arduino to form a low-cost, real-time autonomous vehicle platform. In *Proc. of the 2016 American Control Conference (ACC)*, 6628–6633.

Leva, A. (2003). A hands-on experimental laboratory for undergraduate courses in automatic control. *IEEE Transactions on Education*, 46(2), 263–272.

Migchelbrink, M., White, W.N., Gorentz, L., Wagner, J., and Blankenau, B. (2015). Design, build, and test of an autonomous myRIO based segbot. In *Proc. of the 2015 American Control Conference (ACC)*, 2783–2788.

Ovalle, D.M. and Cómbita, L.F. (2014). Teaching basic control concepts with a home-made thermal system. In *Proc. of the 2014 IEEE Global Engineering Education Conference (EDUCON)*, 739–744.

Reck, R.M. (2016). Defining common aspects of undergraduate instructional laboratories for control systems. In *Proc. of the 2016 American Control Conference (ACC)*, 6646–6651.

Reck, R.M. and Sreenivas, R.S. (2015). Developing a new affordable DC motor laboratory kit for an existing undergraduate controls course. In *Proc. of the 2015 American Control Conference (ACC)*, 2801–2806.

Sarik, J. and Kymissis, I. (2010). Lab kits using the arduino prototyping platform. In *Proc. of the 2010 IEEE Frontiers in Education Conference (FIE)*, T3C–1–T3C–5.