High Assurance Controller of Self-balancing Robot

Capstone Project

Sponsored by: Matt Clark and Michal Podhradsky, Galois: galois.com

Background and Motivation

High assurance autonomous cyber-physical systems provide the highest degree of challenge as tools and practices to design critical software with high assurance are in their infancy. Studies have shown that although some tools exist within this domain to make high assurance software, often graduating engineers do not understand how to implement these tools in a realistic design process.[1] This capstone project is exploratory, as Galois is looking for a demonstration of a workflow for a verification of inner control loops in cyber-physical systems.

The classic inverted pendulum problem is a long standing control problem that requires active balancing and non-linear control.[2] Verification of non-linear control systems are typically performed using several methods leveraging:

1. Stability, Observability, and Controllability using a set of linearized models[3]
2. Lyapunov stability of non-linear models[4]
3. Extensive simulation

These verification methods often rely on model based abstractions of the true dynamics. These model based abstractions do not segregate between uncertainty introduced by the design implementation, errors in software implementation, and environmental uncertainties. Often, sampling time, hardware variations are coupled with true environmental uncertainty, thereby putting more strain on the robustness requirements of the system.

The challenge of this project is to build on top of the results of a previous capstone team, which designed and built an inverted pendulum two-wheeled robot, designed a PID

---

[1] Davis, Jennifer A., et al. "Study on the barriers to the industrial adoption of formal methods." *International Workshop on Formal Methods for Industrial Critical Systems*. Springer, Berlin, Heidelberg, 2013. http://loonwerks.com/publications/pdf/cofer2013fmics.pdf

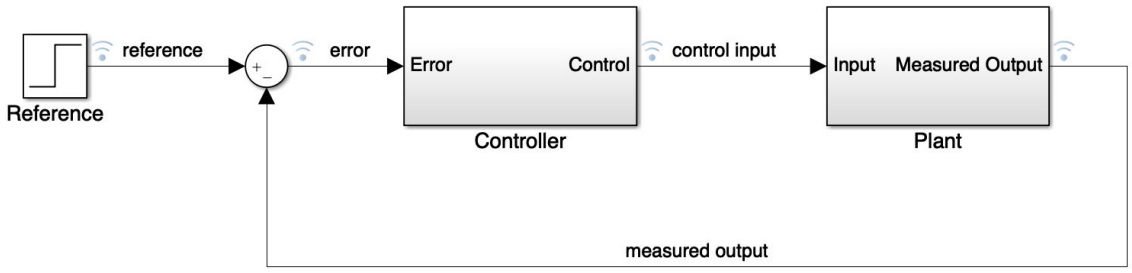
[2] Pathak, Kaustubh, Jaume Franch, and Sunil Kumar Agrawal. "Velocity and position control of a wheeled inverted pendulum by partial feedback linearization." *IEEE Transactions on robotics* 21.3 (2005): 505-513 http://www.academia.edu/download/4133309/p61.pdf

[3] Lavretsky, Eugene. "Adaptive control: Introduction, overview, and applications." *Lecture notes from IEEE Robust and Adaptive Control Workshop*. 2008. https://www.cds.caltech.edu/archive/help/uploads/wiki/files/140/IEEE_WorkShop_Slides_Lavretsky.pdf

[4] Ibanez, Carlos Aguilar, O. Gutierrez Frias, and M. Suarez Castanon. "Lyapunov-based controller for the inverted pendulum cart system." *Nonlinear Dynamics* 40.4 (2005): 367-374. http://www.wseas.us/e-library/conferences/venice2004/papers/472-109.pdf

controller to keep it upright, and explored high assurance controller design. This assignment will require you to leverage both the Rust programming language[5] and the STM32F3 Discovery (or similar) board.

In this capstone, Galois would like the team to focus on the controller design and verification only, leveraging both the hardware design and the simulation environment produced by the last capstone team. To accomplish this, this team must isolate the specific code used to provide the PID control, such that the control and the environment can be developed and verified separately. For verification, it is often valuable to create a model of the system you are trying to verify. It will be helpful for at least part of your team to understand the basics of digital feedback control. Several references may be helpful for this context.[6] [7] The below picture depicts a basic control diagram where the Plant is the inverted pendulum robot in the environment, represented dynamical equations. We would like this team to focus on the modeling, verification, and implementation of a Rust library of controllers for the inverted pendulum robot. The focus will be to allow the rest of the system to remain designed in c while creating a separate Rust library for the controller only. Showing traceability between the controller design in a modeling paradigm and the implementation in Rust will be key. I.e. how can formal verification ensure that the control design performed using either a modeling language (like Matlab) or through direct mathematical calculations has been implemented in the Rust code.



Objectives and Learning Outcomes

The team that works on this project will, with the support of Galois engineers, achieve the following objectives and learning outcomes. *Outcomes that are italicized are optional and*

[5] Rust type safe language, https://www.rust-lang.org/en-US/

[6] Stevens, Brian L., Frank L. Lewis, and Eric N. Johnson. Aircraft control and simulation: dynamics, controls design, and autonomous systems. John Wiley & Sons, 2015.

[7] Aström, Karl Johan, and Richard M. Murray. Feedback systems: an introduction for scientists and engineers. Princeton university press, 2010. https://www.cds.caltech.edu/~murray/amwiki/index.php?title=Main_Page

will be supported/encouraged if there are team members with an interest in the topic and relevant skills.

- Gain experience blending formal verification with simulation using Octave and Kind2. Gain expertise in embedded Rust
- Gain experience with STM32 microcontrollers
- Gain experience with advanced and nonlinear control
- Develop environment interface models and hardware abstractions for the Octave / Matlab simulation environment. Model the functional blocks in Kind2 / Lustre and compare closed loop simulation with invariant checks.
- Develop a nonlinear controller in Octave / Matlab m-files and Rust and formally verify as many properties as possible.
- *Develop a suite of verified nonlinear controllers and compare their performance with linear variants.*
- Galois is hiring, and a successful capstone project could lead to an internship/permanent position.

Milestones

The team is expected to meet these milestones. *The stretch goals that are italicized are optional*. Each milestone has to be extensively documented so it can be reproduced **without** any input from the student team. The documentation is a hard requirement, and the milestone cannot be considered to be achieved without being properly documented. Galois will cover the cost of hardware, all required software is open source.

- extend the existing code for the inverted pendulum robot to include Rust control library described above.
- modify Lustre/Kind2 Rust code generator to generate “embedded-friendly” Rust, which can be directly imported in your library
- Develop and identify a model of a nominal self-balancing robot in Octave or Matlab. Reusing existing models is encouraged.
- Isolate the “Software under Test” as a severable block to analyze only the linear / non-linear controller for specific properties
- *Attempt to leverage Rust quickcheck[8] to verify the properties developed in Kind2 in the source files after the control design is implemented*
- Develop and identify a dynamics model of the robot in Octave or Matlab
- *Alternately, Rust code into a LEAN theorem definitions may be helpful.*[9]
- Develop and verify a nonlinear controller and compare its performance with the PID controller through both simulation and in the real system.

[8] https://docs.rs/quickcheck/0.7.2/quickcheck/
[9] https://github.com/Kha/electrolysis

Student Skills

A team that tackles this ECE project will need team members that have expertise in, or are willing to build upon intermediate expertise in, each of the following:

- Embedded systems programming and debugging in Rust
- Control loop theory including advanced and nonlinear control
- Learn the concepts of applied formal methods and their realization via Kind2 and quickcheck.