Real-Time Systems

Project Descriptions 2017

1. Administrative Information

- Four persons per team
- Constraints
  - Hardware
  - Processes
  - Supervisors
- If you cannot find a group to join, we will try to find a suitable group for you

Project Selection

- Project teams should be organized and hand in a priority list of their desired projects by **February 28**.
- Email to karlerik@control.lth.se with the topic ”Real-Time Project 2017”
- The list should contain four projects in **priority order**.
- Projects will be assigned March 2-3. Your supervisor(s) will contact you by email and arrange a first meeting.

Suggested Solutions

- 3–4 pages, should be handed in to your supervisor by **March 24**
- You are not allowed to start working in the lab until your suggested solution has been approved
- Structure:
  1. Introduction
  2. Program structure
  3. Operator communication
  4. Control principles (if applicable)
  5. Project plan

Project Requirements

- A **program** meeting the specifications should be demonstrated to your supervisor by **May 12**
- A **project report** (Swedish or English) should be handed in to your supervisor by **May 12**
  - The supervisor may request revisions
  - You are only allowed to hand in your report three times (including the first version)
- The project should be demonstrated on **May 16**, 15:15–17:00
- An oral presentation (10 mins) should be made on **May 16**, 17:15–19:00

Structure of the Report

- Cover page
- Introduction
- Program structure
- Control design
- User interface and HowTo
- Results
- Conclusions

Project Suggestions

Symbols:

- **P** – Pure real-time programming project.
  May not be selected by those who have taken EDA040 Concurrent and Real-Time Programming
  - However, an extended version that also includes controller design and implementation could be accepted.
- **C** – Programming in C required
- **2** – Small project. Only two persons
- **S** – Special project. High risk factor. Only for highly motivated persons

2. Predictive Control Joint Projects

PC4 – Mass-Spring-Damper System
PC5 – Control of an Inverted Pendulum (discrete-time)
PC6 – Control of an Inverted Pendulum (continuous-time)
PC7 – Adaptive Control of the See-saw Process
PC8 – Control of the Helicopter
PC9 – Adaptive Friction Compensation
PC10 – Model Predictive Control Using CVXGEN
PC11 – Autotuning of Robust PID Controllers
PC12 – Control of Ball-and-Beam Process

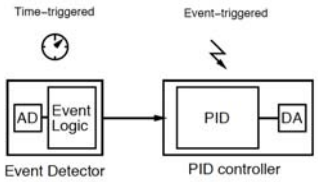
2. Predictive Control Joint Projects

PC13 – MPC Control of the Ball-and Plate Process

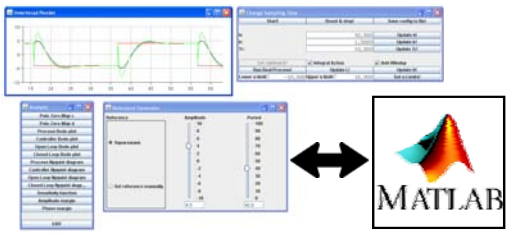
- **CVXGEN or Qpgen**
- **Follow a predefine trajectory**
- **Not the same process as on the photo**

3. Algorithm-Oriented Projects

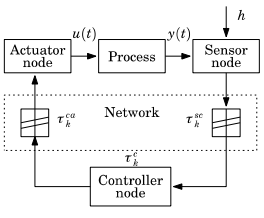
Project 1 – Event-Based Sampling and PID Control (2)



Project 2 – General State Feedback Controller Using MATLAB Compute Engine

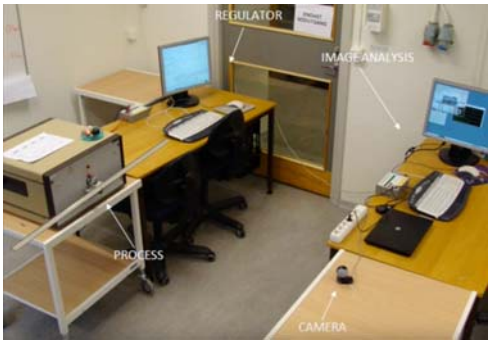


Project 3 – Compensation for Network Delays



$$u_k = -L_k(\tau_k^{sc}) \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix}$$

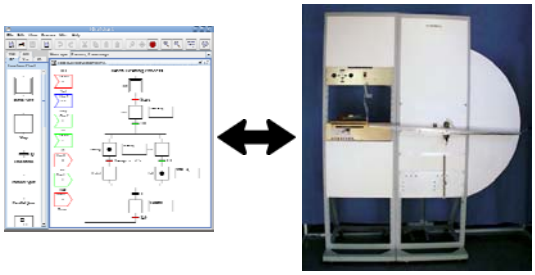
Project 4 – Vision Feedback



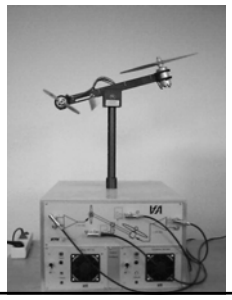
4. Control-Oriented Projects

Project 5 – ”Catch and Throw” Ball and Beam Process

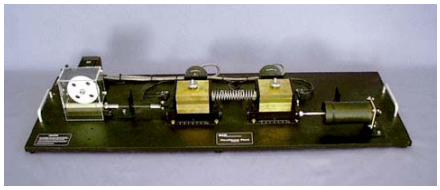
Project 6 – ”Catch and Throw” Ball and Beam Process in JGrafchart



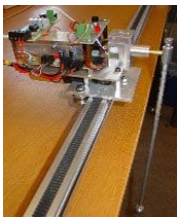
Project 7 – Control of the Helicopter Process



Project 8 – Control of Mass-Spring-Damper Process

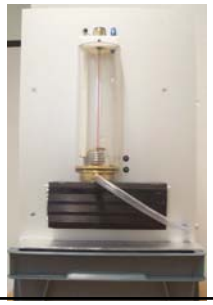


Project 9 – Control of Linear Pendulum



Project 10 – Control of Furuta Pendulum


Project 11 – Multivariable Control of the Batch Tank Process



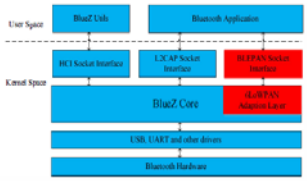
5. Real-Time Programming Projects

Project 12 – Linux for Control (2CP)

- POSIX pthreads
- Gtk (or similar)
- Reimplement, e.g., Lab 1

Project 13 – Wireless Control over Bluetooth (P)

- Bluez stack
- Reimplement, e.g., Lab 1

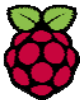


Project 14 – Wireless Control Using Android (P)

- Implement a controller in Java on Android phone

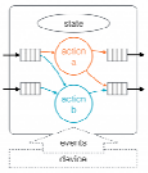
Project 15 – Wireless Control in the Cloud Using 4G/LTE (2CP)

- Execute the controller in the Cloud
  - e.g., Amazon or Ericsson Research DC
- Raspberry Pi with LTE dongle



Project 16 – Control over IoT Using Calvin (2CP)

- Implement and execute some controller in the Calvin IoT framework
  - Data flow (actors) model where individual Actors are written in Python
- Migration between executing the control locally or in a data center



Project 17 – Evaluation of FreeRTOS (CP)

•www.freertos.org
•Texas board with ARM Cortex M4 CPU
•Implement, e.g., a small, multithreaded control application

Project 18 – Evaluation of Zephyr (CP)

•www.zephyrproject.org
•Arduino platform
•Implement, e.g., a small, multithreaded control application

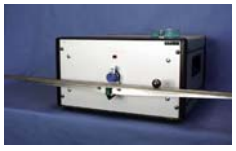
Project 19 – Effects of SCHED_DEADLINE on the Implementation of Control Strategies (PC2)

•Use SCHED_DEADLINE in the Linux kernel
  – EDF-based scheduler
•Implement controller with tracing facilities
  – Evaluate delays, jitter
  – Evaluate control performance

6. Embedded Projects

Project 20 – Embedded Control of the Ball and Beam Process (C)

•Atmel AVR
•State feedback from observer, integral action
•Fixed-point arithmetic



Project 21 – Embedded Control of the Mass-Spring-Damper Process (C)

•Atmel AVR
•State feedback from observer, integral action
•Fixed-point arithmetic

Project 22 – Embedded Control of the Tiny Demo Process

• Tiny servo process with Arduino that can be connected to a laptop where the controller execute in, e.g., Simulink

• Task is to instead implement the controller in the Arduino and develop a GUI on the other computer

Project 23 – Lego Mindstorm Projects

•Various programming options (Java, C, ...)

•Segway or your own idea

Project 24 – Reversing a Lego Mindstorm Trailer Truck

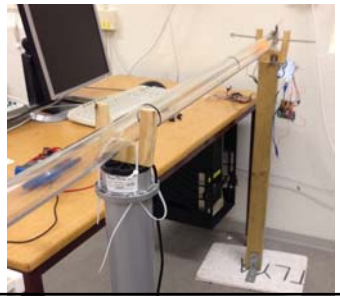
•Various programming options (Java, C, ...)

• Possibly using a camera with OpenCV to measure the angle

Project 25 – Embedded Control of the Balanduino Balancing Robot (CS)

•Arduino platform (Atmel AVR)

Project 26 – Embedded Control of a Ball in Pipe Process (CS)



Project 27 – Control of the Crazyflie Quadcopter (CS)

• www.bitcraze.io/crazyflie-2

• Localization using UWB pulses to/from anchor nodes

• Hover and follow waypoints

Project 28 – Localization of the Parrot BeBop UAV (CS)

- Larger UAV
- Port existing UWB localization system to the BeBop
- Do some simple control

Project 29 – Autonomous Control of the TurtleBot (CS)

- Mobile robot with camera and touch sensors
- ROS
- Implement some ”autonomous lawn-mower functionality”

Project 30 Autonomous Racing: Trajectory Control of a Small Scale Racing Car (CS)

- 1/10 th scale racing car
- Lidar sensor
- ROS
- Develop trajectory controller that minimizes laptime

7. Your Own Ideas