Imperial College London

Teaching Predictive Control Using Specification-based Summative Assessments

Ian McInerney[a], Eric C. Kerrigan[ab]

[a]Department of Electrical and Electronic Engineering, Imperial College London
[b]Department of Aeronautics, Imperial College London

Imperial College London

What is Predictive Control?

- Optimization-based controller
- Includes system/operational constraints in the controller formulation
- Being applied to many different systems in many industries

Imperial College London

Teaching Predictive Control

- Recent survey shows industry prefers a first control course that contains
  - Optimal control
  - Optimal state feedback
  - Authentic simulation/implementation scenarios

- Very large set of possible topics to cover
  - Linear/Nonlinear MPC
  - Economic MPC
  - Stochastic MPC
  - Robust MPC

Imperial College London

Course Overview

- Offered to MSc and final-year MEng students
- Students usually have one prior control course and no optimization experience
- 10-week course including
  - A 2-hour lecture each week
  - 2 laboratory activities
  - MATLAB Grader checkpointing assessments
  - Specification-based control design assessments
- Conducted fully remotely in 2021

Imperial College London

Lectures

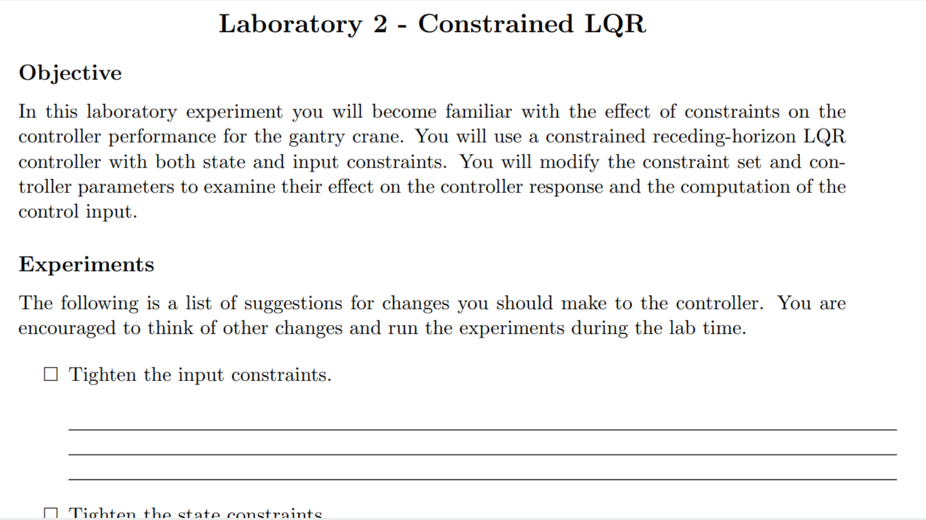
- Initially taught linear MPC, transitioned to nonlinear MPC in 2021
- Topics taught:

Background (2021 only)	**Linear**	**Nonlinear**
State-space modelling	Soft constraints	Soft constraints
Discretization	Disturbance rejection	Constraint tightening
ODE Solvers	Move blocking	Real-time iteration
Numerical/automatic Differentiation	Stability & robustness	External constraint handling

Imperial College London

Laboratory Activity – In-person

- Uses a laboratory scale gantry crane with MATLAB/Simulink
- Focuses on the real-world effects of controller tuning



Laboratory 2 - Constrained LQR

Objective

In this laboratory experiment you will become familiar with the effect of constraints on the controller performance for the gantry crane. You will use a constrained receding-horizon LQR controller with both state and input constraints. You will modify the constraint set and controller parameters to examine their effect on the controller response and the computation of the control input.

Experiments

The following is a list of suggestions for changes you should make to the controller. You are encouraged to think of other changes and run the experiments during the lab time.

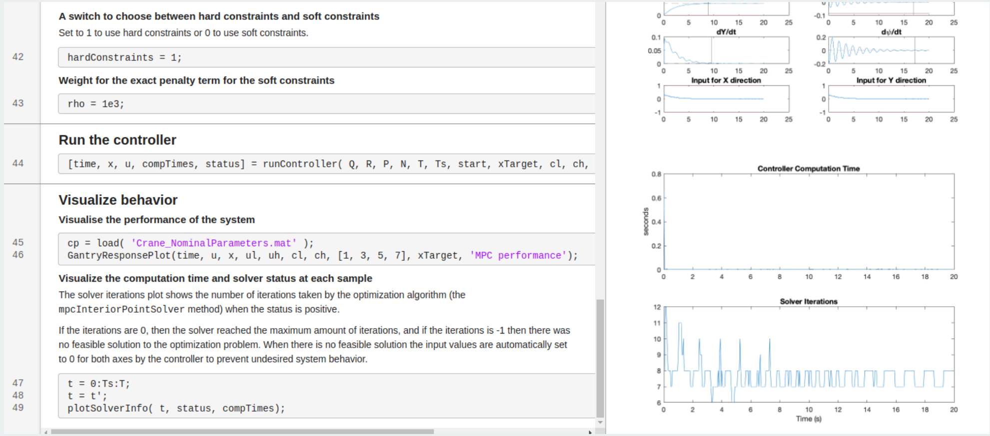
☐ Tighten the input constraints.

☐ Tighten the state constraints.

Imperial College London

Laboratory Activity – Remote

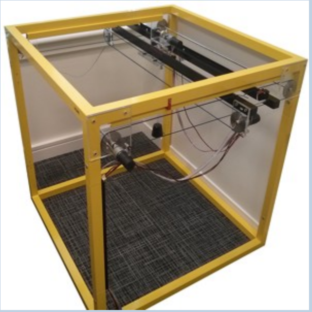
- MATLAB Live Script with simulation model of gantry crane
- Same activities as in-person labs
- Usable in MATLAB Online



Imperial College London

Course Assessments

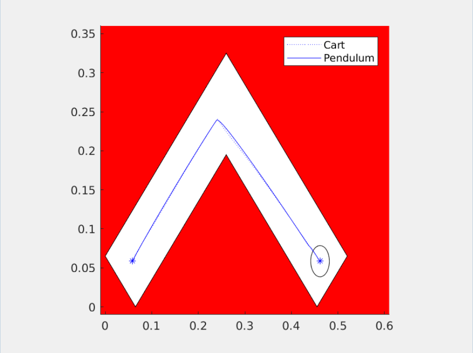
Coursework 1: Derive Physics Model



MATLAB Grader Courseworks: MPC/Background Concepts

$$\begin{aligned} \underset{u,x}{\text{minimize}}\ & \|x_N\|_P^2 + \sum_{k=0}^{N-1} \left(\|x_k\|_Q^2 + \|u_k\|_R^2 \right) \\ \text{subject to}\ & x_{k+1} = Ax_k + Bu_k,\ k = 0, \ldots N-1 \\ & Fu_k \leq c_u,\ k = 0, \ldots N-1 \\ & Dx_k \leq c_x,\ k = 1, \ldots N \end{aligned}$$

Final Coursework: Design Project



Imperial College London

First Coursework

- Derive state-space model for an overhead gantry crane
- Complete a partially filled-in MATLAB Live Script
- Use the symbolic toolbox to derive and linearize the model

```
dLdq1 = diff(L, q_1)
```

Equations of motion

The dynamics of the n-DOF system will be described by a set of n coupled second order differential equations. We convert our second order equations to a set of first order equations to solve them numerically.

Euler-Lagrange equations corresponding to coordinate X and theta (q1 q2)

```
eq1 = diff(dLdq1p) - dLdq1 == f - diff(X) * T
eq2 = diff(dLdq2p) - dLdq2 == 0
```

Notice the RHS of eq1:

We know that our input to the gantry crane system is a PWM signal to the DC motors. Here we will assume that this signal is proportional to force exerted on the cart. Therefore the RHS of eq1 contains a term f for such a force. Furthermore, there is a damping term T acting on velocity.

For you to complete from here on

You should proceed by finding the explicit expressions for X" and theta", thus allowing you to form a state space model.

Since the equation is implicit, to express theta" from eq1, you first need to replace it by an auxiliary variable. Then you can use solve(equation, variable).

```
%Declare symbolic auxiliary variables for solving eq1 and eq2
syms d2theta d2X

%{
 your code here
%}
```

Express theta" from eq2:

```
%{
 your code here
%}
```

Imperial College London

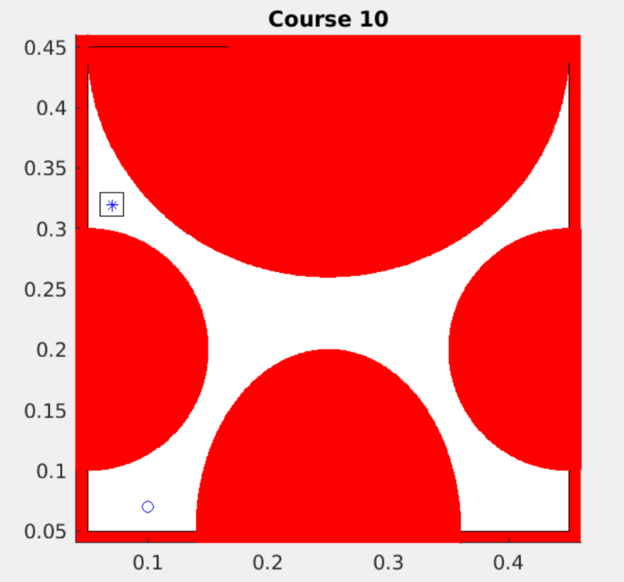
MATLAB Grader Courseworks

- Focus on how to implement MPC-related topics in code
  - Forming MPC matrices
  - ODE solvers
  - Quadrature methods
  - Using fmincon and automatic differentiation
- Self-paced during the course

Imperial College London

Specification-based Summative Assessment

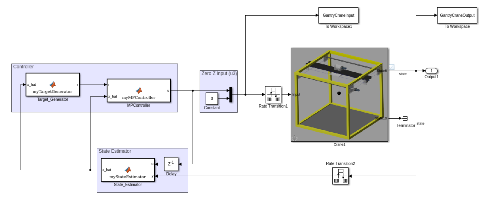
- Give students high-level controller specification
  - Move crane from start to target
  - Remain inside an area
  - Avoid obstacles
- Students design & submit controllers
- Test submitted controllers against many possible area/obstacle combinations



Imperial College London

Assessment Flow

1. Create Controller



2. Upload to MATLAB Grader

Your Function

```matlab
%% Modify the following function for your setup function
function [ param ] = mySetup(c, startingPoint, targetPoint, eps_r, eps_t)

    % This is a sample static K matrix for the controller
    param.K = [1, 0, 0, 0, 0, 0, 0, 0;
               0, 0, 1, 0, 0, 0, 0, 0];

    % This is a sample way to send reference points
    param.xTar = targetPoint(1);
    param.yTar = targetPoint(2);

end % End of mySetup


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%% Modify the following function for your target generation
function r = myTargetGenerator(x_hat, param)

    %% Do not delete this line
    % Create the output array of the appropriate size
    r = zeros(10,1);
    %%

    % Make the crane go to (xTar, yTar)
    r(1,1) = param.xTar;
    r(3,1) = param.yTar;

end % End of myTargetGenerator
```

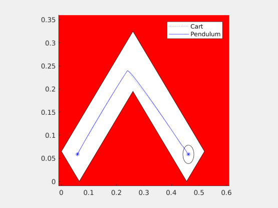
3. Syntax Checking

Assessment:

- Test output sizes (Pretest)
- Test controller for errors (Pretest)

3. Test Controller

In Simulation

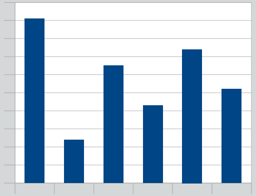


On Hardware

4. Analyze Performance

```
Running the linear simulation
Linear simulation complete
Student setup function ran in 3.243000e-02s
Setting up the nonlinear simulation
Running the nonlinear simulation
Nonlinear Simulation completed in 26.007973
Simulated time: 60.000000
Settling Times:
    State: 4.5
    Input: NaN
Constraint Violations:
    Total violations: 1
    Cart violates:
    Pendulum violates: 2
```

5. Calculate Marks



Diversity in Student Solutions

(a) 2018 course (out of 30 students)

Cost Function	
Quadratic cost	30
Stabilizing terminal penalty	15
Constraints	
Soft constraints	14
Multiple constraint sets	29
State Estimator	
Kalman filter	15
Other state estimator	5
Other Features	
Offset-free tracking	18
Move blocking	1

(b) 2021 course (out of 26 students)

Setup - Path Planning	
MATLAB *nlmpc*	2
MATLAB *fmincon*	4
Other path planning	3
Setup - Nonlinearities	
Nonlinear cost	2
Nonlinear ellipses	9
Controller - Other Features	
Constraint tightening	15
Soft constraints	4
State estimator	5

Controller - Optimizer	
MATLAB *nlmpc*	4
MATLAB *fmincon*	20
Real-time iteration	3
Controller - Model	
Nonlinear/Time-varying	9
Linear	15
Controller - Cost	
Nonlinear	5
Quadratic	23

Imperial College London

Student Reaction

- Engaged with and enjoyed the in-person laboratory activities
- Seemed to spend too much time working on final coursework

Imperial College London

Lessons Learned

- Properly defining specification is difficult and can only be done once per year
- Student code can produce unexpected outputs/results that crash simulators/MATLAB