Introduction

Tasks

- Write a function that converts bound constraints to general stage constraints.
- Write a function that converts stage constraints to trajectory constraints.
- Write a function that computes the constraints for a QP arising in tracking problems.
- Implement a receding horizon controller with constraints and non-zero target state.

Framework provided:

myStageConstraints.m	Template for submission
myTrajectoryConstraints.m	Template for submission
myConstraintMatrices.m	Template for submission
myMPController.m	Template for submission
SimscapeCrane_MPChard.slx	Nonlinear Simscape model for testing controller
SSmodelParams.mat	Physical parameters for the linear ODE
Params_Simscape.mat	Physical parameters for nonlinear simulation
GantryResponsePlot.m	Utility function
MatlabSimulation.m	Simulates the linear gantry crane model
testMyMPC.m	Function to test your RHC law performance
genCraneODE.p	Answer function for a previous assignment
genPrediction.p	Answer function for a previous assignment
genCostMatrices.p	Answer function for a previous assignment

Approach

Suppose you have a horizon length N and a model of a discrete-time system of the form

$$x_{k+1} = Ax_k + Bu_k,\ k = 0, 1, \ldots, N-1 \tag{1}$$

where the state $x_k \in \mathbb{R}^n$ and input $u_k \in \mathbb{R}^m$ at sample instant k. Suppose also that a target state x_e is given and that x_e is an equilibrium point if the input applied to the system is zero, i.e. $x_e = Ax_e$.

- Edit **myStageConstraints.m** to compute matrices $\tilde{D}$ (`Dt`), $\tilde{E}$ (`Et`) and vector $\tilde{b}$ (`bt`) such that

  $$\tilde{D}x_k + \tilde{E}u_k \leq \tilde{b},\ k = 0, 1 \ldots, N-1 \tag{2}$$

  is equivalent to

  $$\check{u} \leq u_k \leq \hat{u},\ k = 0, 1 \ldots, N-1 \tag{3a}$$
  $$\check{c} \leq Dx_k \leq \hat{c},\ k = 1, 2, \ldots, N \tag{3b}$$

  where $\check{u}$ (`ul`) and $\check{c}$ (`cl`) are lower bounds, $\hat{u}$(`uh`) and $\hat{c}$ (`ch`) are upper bounds and the constrained vector $Dx_k \in \mathbb{R}^c$ is defined by the matrix D.

  Your code should ensure that the number of constraints in (2) is equal to the number of constraints in (3).

- Edit **myTrajectoryConstraints.m** to compute matrices $\mathbf{D}$ (`DD`), $\mathbf{E}$ (`EE`) and $\mathbf{b}$ (`bb`) such that (2) is equivalent to

$$\mathbf{D}\tilde{\mathbf{x}} + \mathbf{E}\mathbf{u} \leq \mathbf{b} \tag{4}$$

where $\tilde{\mathbf{x}} := [x_0^T \; x_1^T \; \cdots \; x_{N-1}^T]^T$ and $\mathbf{u} := [u_0^T \; u_1^T \; \cdots \; u_{N-1}^T]^T$. Note that $\mathbf{b}$ is *not* a function of x_0.

Your code should ensure that the number of constraints in (2) is equal to the number of constraints in (4).

- Edit **myConstraintMatrices.m** to compute the matrices F, J and L in the quadratic program (QP) given by

$$\mathbf{u}^*(x_0, x_e) := \arg\min_{\mathbf{u}} \; \|x_N - x_e\|_P^2 + \sum_{k=0}^{N-1} \left(\|x_k - x_e\|_Q^2 + \|u_k\|_R^2\right) \text{ s.t. (1)–(2)} \tag{5a}$$

$$= \arg\min_{\mathbf{u}} \frac{1}{2}\mathbf{u}^T H \mathbf{u} + (x_0 - x_e)^T G^T \mathbf{u} \text{ s.t. } F\mathbf{u} \leq \mathbf{b} + Jx_0 + Lx_e \tag{5b}$$

where P, Q and R are given positive semi-definite matrices and $\mathbf{x} = \Phi x_0 + \Gamma \mathbf{u}$ as in the previous assignment. The notation $\|z\|_M^2 := z^T M z$ for any square matrix M.

- We will use the MATLAB function `mpcqpsolver` to solve the QPs arising in this course. Read the documentation for `mpcqpsolver` and `mpcqpsolverOptions`. Please pay special attention to input/output conventions and the way inequality constraints are posed in `mpcqpsolver`, since these are different to (5b). See **testMyMPC.m** for details on putting H into an appropriate form for `mpcqpsolver` as discussed in the MATLAB documentation.

- Edit **myMPController.m** so that it provides the value of the RHC law κ for the target state x_e and the current estimate of the state $\hat{x} =: x_0$, i.e.

$$\kappa(\hat{x}, x_e) := u_0^*(\hat{x}, x_e) \tag{6a}$$

where

$$\mathbf{u}^*(\hat{x}, x_e) =: \left[u_0^*(\hat{x}, x_e)^T \; u_1^*(\hat{x}, x_e)^T \; \cdots \; u_{N-1}^*(\hat{x}, x_e)^T\right]^T. \tag{6b}$$

Please see how the function is called from the test script **testMyMPC.m**. Pay special attention to which matrices are inside and which are outside the function.

- Once all of the above functions are in working order you will be able to use **testMyMPC.m** for assessing the performance of your controller. Open **SimscapeCrane_MPChard.slx**, navigate to the block *SimscapeCrane_MPChard/MPC* and replace the code inside the MATLAB function block with your own. Investigate the effect of using different cost weights, sample period, horizon length, initial conditions, target state, constraints, etc. This will be helpful for the final assignment.

Submission

filename	Description.
myStageConstraints.m	Computes the matrices and vector in (2)
myTrajectoryConstraints.m	Computes the matrices and vector in (4)
myConstraintMatrices.m	Computes the constraint matrices for the QP (5b)
myMPController.m	Computes the optimal RHC input (6a) with a non-zero target equilibrium state by solving the QP (5b)

Edit or upload the MATLAB template for the four files above into Cody Coursework. All your code should be written for the general case and not just for the gantry crane.