|  |  |
| --- | --- |
| %Control of Quadrotor Using Nonlinear Model Predictive Control  % This example shows how to design a trajectory tracking controller for a quadrotor using nonlinear model  % predictive control (MPC).  %  %%%Quadrotor Model  %The quadrotor has four rotors which are directed upwards. From the center of mass of the quadrotor,  % rotors are placed in a square formation with equal distance. The mathematical model for the quadrotor  % dynamics are derived from Euler-Lagrange equations [1].  %The twelve states for the quadrotor are:  getQuadrotorDynamicsAndJacobian11;    %Design Nonlinear Model Predictive Controller  %Create a nonlinear MPC object with 12 states, 12 outputs, and 4 inputs. By default, all the inputs  % are manipulated variables (MVs)    nx = 12;  ny = 12;  nu = 4;  nlobj11 = nlmpc(nx, ny, nu);  %Specify the prediction model state function using the function name. You can also specify functions  % using a function handle.  nlobj11.Model.StateFcn = "QuadrotorStateFcn11";    %Specify the Jacobian of the state function using a function handle. It is best practice to provide an  % analytical Jacobian for the prediction model. Doing so significantly improves simulation efficiency  nlobj11.Jacobian.StateFcn = @QuadrotorStateJacobianFcn11;  %Validate your prediction model, your custom functions, and their Jacobians.  rng(0)  validateFcns(nlobj11,rand(nx,1),rand(nu,1));    %Specify a sample time of 0.1 seconds, prediction horizon of 18 steps, and control horizon of 2 steps.  Ts = 0.1;  p = 18;  m = 2;  nlobj11.Ts = Ts;  nlobj11.PredictionHorizon = p;  nlobj11.ControlHorizon = m;    %Limit all four control inputs to be in the range [0,12].  nlobj11.MV = struct('Min',{0;0;0;0},'Max',{30;30;30;30});    %The default cost function in nonlinear MPC is a standard quadratic cost function suitable for reference  % tracking and disturbance rejection. In this example, the first 6 states are required to follow a given  % reference trajectory. Because the number of MVs (4) is smaller than the number of reference output  % trajectories (6), there are not enough degrees of freedom to track the desired trajectories for all  % output variables (OVs).  nlobj11.Weights.OutputVariables = [0 0 1 1 1 1 0 0 0 0 0 0];    %In this example, MVs also have nominal targets to keep the quadrotor floating, which can lead to conflict  % between the MV and OV reference tracking goals. To prioritize targets, set the average MV tracking  % priority lower than the average OV tracking priority.  nlobj11.Weights.ManipulatedVariables=[0.1 0.1 0.1 0.1];    %Also, penalize aggressive control actions by specifying tuning weights for the MV rates of change.  nlobj11.Weights.ManipulatedVariablesRate = [0.1 0.1 0.1 0.1];    %Closed-Loop Simulation  %Simulate the system for 20 seconds with a target trajectory to follow.    % Specify the initial conditions  x = [0;0;0.1;pi/15;0;0;0;0;0;0;0;0];  % Nominal control that keeps the quadrotor floating  nloptions = nlmpcmoveopt;  nloptions.MVTarget = [4.9 4.9 4.9 4.9];  mv = nloptions.MVTarget;      %%Simulate the closed-loop system using the nlmpcmove function, specifying simulation options using  % an nlmpcmove object.    Duration = 40;  hbar = waitbar(0,'Simulation Progress');  xHistory = x';  lastMV = mv;  uHistory = lastMV;  for k = 1:(Duration/Ts)  % Set references for previewing  t = linspace(k\*Ts, (k+p-1)\*Ts,p);  yref = QuadrotorReferenceTrajectory11(t);  % Compute the control moves with reference previewing.  xk = xHistory(k,:);  [uk,nloptions,info] = nlmpcmove(nlobj11,xk,lastMV,yref',[],nloptions);  uHistory(k+1,:) = uk';  lastMV = uk;  % Update states.  ODEFUN = @(t,xk) QuadrotorStateFcn11(xk,uk);  [TOUT,YOUT] = ode45(ODEFUN,[0 Ts], xHistory(k,:)');  xHistory(k+1,:) = YOUT(end,:);  waitbar(k\*Ts/Duration,hbar);  end  close(hbar)    % Open the Simulink model.  %mdl = 'HameedLaneFollowingNMPC';  %%%% open\_system(mdl)      %%Visualization and Results  %Plot the results, and compare the planned and actual closed-loop trajectories.  plotQuadrotorTrajectory11;    %animateQuadrotorTrajectory11; | %Control of multirotor Using Nonlinear Model Predictive Control  % This example shows how to design a trajectory tracking controller for a multirotor using nonlinear model  % predictive control (MPC).  %  %%%Multirotor Model  %The multirotor has many rotors which are directed upwards. From the center of mass of the multirotor,  % rotors are placed in a square formation with equal distance. The mathematical model for the quadrotor  % dynamics are derived from Euler-Lagrange equations .    %The states for the quadrotor are:  getmultirotorDynamicsAndJacobian;    %Design Nonlinear Model Predictive Controller  %Create a nonlinear MPC object with 13 states, 12 outputs, and 6 inputs. By default, all the inputs  % are manipulated variables (MVs)    nx = 13;  ny = 13;  nu = 6;  nlobj = nlmpc(nx, ny, nu);  %Specify the prediction model state function using the function name. You can also specify functions  % using a function handle.  nlobj.Model.StateFcn = "multirotorStateFcn";    %Specify the Jacobian of the state function using a function handle. It is best practice to provide an  % analytical Jacobian for the prediction model. Doing so significantly improves simulation efficiency  nlobj.Jacobian.StateFcn = @multirotorStateJacobianFcn;    %Validate your prediction model, your custom functions, and their Jacobians.  rng(0)  validateFcns(nlobj,rand(nx,1),rand(nu,1));    %Specify a sample time of 0.1 seconds, prediction horizon of 18 steps, and control horizon of 2 steps.  Ts = 0.1;  p = 18;  m = 2;  nlobj.Ts = Ts;  nlobj.PredictionHorizon = p;  nlobj.ControlHorizon = m;    %Limit all four control inputs to be in the range [0,12].  nlobj.MV = struct('Min',{0;0;0;0;0;0},'Max',{30;30;30;30;30;30});    %The default cost function in nonlinear MPC is a standard quadratic cost function suitable for reference  % tracking and disturbance rejection. In this example, the first 6 states are required to follow a given  % reference trajectory. Because the number of MVs (4) is smaller than the number of reference output  % trajectories (6), there are not enough degrees of freedom to track the desired trajectories for all  % output variables (OVs).  nlobj.Weights.OutputVariables = [0 0 1 1 1 1 0 0 0 0 0 0];    %In this example, MVs also have nominal targets to keep the multirotor floating, which can lead to conflict  % between the MV and OV reference tracking goals. To prioritize targets, set the average MV tracking  % priority lower than the average OV tracking priority.  nlobj.Weights.ManipulatedVariables=[0.1 0.1 0.1 0.1 0.1 0.1];    %Also, penalize aggressive control actions by specifying tuning weights for the MV rates of change.  nlobj.Weights.ManipulatedVariablesRate = [0.1 0.1 0.1 0.1 0.1 0.1];    %Closed-Loop Simulation  %Simulate the system for 20 seconds with a target trajectory to follow.    % Specify the initial conditions  x = [0;0;0.1;pi/15;0;0;0;0;0;0;0;0;0];  % Nominal control that keeps the multirotor floating  nloptions = nlmpcmoveopt;  nloptions.MVTarget = [4.9 4.9 4.9 4.9 4.9 4.9];  mv = nloptions.MVTarget;      %%Simulate the closed-loop system using the nlmpcmove function, specifying simulation options using  % an nlmpcmove object.    Duration = 40;  hbar = waitbar(0,'Simulation Progress');  xHistory = x';  lastMV = mv;  uHistory = lastMV;  for k = 1:(Duration/Ts)  % Set references for previewing  t = linspace(k\*Ts, (k+p-1)\*Ts,p);  yref = multirotorReferenceTrajectory(t);  % Compute the control moves with reference previewing.  xk = xHistory(k,:);  [uk,nloptions,info] = nlmpcmove(nlobj,xk,lastMV,yref',[],nloptions);  uHistory(k+1,:) = uk';  lastMV = uk;  % Update states.  ODEFUN = @(t,xk) multirotorStateFcn(xk,uk);  [TOUT,YOUT] = ode45(ODEFUN,[0 Ts], xHistory(k,:)');  xHistory(k+1,:) = YOUT(end,:);  waitbar(k\*Ts/Duration,hbar);  end  close(hbar)    % Open the Simulink model.  %mdl = 'HameedLaneFollowingNMPC';  %%%% open\_system(mdl)      %%Visualization and Results  %Plot the results, and compare the planned and actual closed-loop trajectories.  plotmultirotorTrajectory;    %animateQuadrotorTrajectory11; |