Autonomous Mobile Robot Design

Basics of Linear Model Predictive Control

Dr. Kostas Alexis (CSE)
Model Predictive Control concept

- Use a dynamic **model** of the process to **predict** its future evolution, choose the “best” **control** action sequence, and execute only its first step.
Model Predictive Control concept

- At time $t$: solve an **optimal control** problem over a future horizon of $N$ steps

$$\min \sum_{k=0}^{N-1} \ell(y_k - r(t + k), u_k)$$

s.t. $x_{k+1} = f(x_k, u_k)$

$y_k = g(x_k, u_k)$

constraints on $u_k, x_k, y_k$

$x_0 = x(t)$

“minimize the error of the system outputs against their references, as well as the use of the actuators, while account for the vehicle dynamics, as well as constraints on the system inputs, states and measured outputs”.
Receding Horizon Control

- **At time** $t$: solve an **optimal control** problem over a future horizon of $N$ steps

  Penalty on tracking error

  $$
  \min \sum_{k=0}^{N-1} \ell(y_k - r(t + k), u_k)
  $$

  s.t. \quad x_{k+1} = f(x_k, u_k)

  $$
  y_k = g(x_k, u_k)
  $$

  constraints on $u_k, x_k, y_k$

  $x_0 = x(t)$

- Apply only the first optimal move $u^*(t)$, throw the rest of the sequence away

- **At time** $t+1$: Get new measurements, repeat the optimization. And so on…

- Penalty on actuation error
Model Predictive Control concept

- **Analogy of a driver steering a car:**
  - Prediction model is what describes how the vehicle is expected to move
  - System constraints are the set of rules to drive as well as the limitations of the car
  - Disturbances are the driver’s inattention, wind and other reasons of undesirable deviation from the reference trajectory
  - Set point is the desired location
  - Cost function may be the goal of minimum time
  - The receding horizon control strategy would re-plan the route of the car and the corresponding driver actions periodically in time, find the overall set of actions over a time horizon, apply the first and then re-plan for the next step.
Linear MPC – Unconstrained case

- **Linear Prediction Model:**
  \[ x_{k+1} = Ax_k + Bu_k \]
  \[ y_k = Cx_k \]

- **Performance Index:**
  \[ J(z, x_0) = x_N'Px_N + \sum_{k=0}^{N-1} x_k'Qx_k + u_k'Ru_k \]
  \[ z = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} \]
  \[ R = R' \succ 0 \]
  \[ Q = Q' \succ 0 \]
  \[ P = P' \succeq 0 \]

- **Goal:** find sequence
  \[ z^* = \begin{bmatrix} u_0^* \\ u_1^* \\ \vdots \\ u_{N-1}^* \end{bmatrix} \]
  That steers the state to the origin optimally, i.e. minimizing \( J(z, x_0) \)
(Computation of cost function)

\[
J(z, x_0) = x_0'Qx_0 + \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ x_N \end{bmatrix}' \begin{bmatrix} Q & 0 & 0 & \cdots & 0 \\ 0 & Q & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & Q & 0 \\ 0 & 0 & \cdots & 0 & P \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ x_N \end{bmatrix} + \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}' \begin{bmatrix} R & 0 & \cdots & 0 \\ 0 & R & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}
\]

\[
\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} + \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}x_0
\]

\[
J(z, x_0) = (\bar{S}z + \bar{T}x_0)'\bar{Q}(\bar{S}z + \bar{T}x_0) + z'\bar{R}z + x_0'Qx_0
\]

\[
= \frac{1}{2}z'\begin{bmatrix} 2(\bar{R} + \bar{S}'\bar{Q}\bar{S}) & z \\ \bar{H} & \bar{F} \end{bmatrix}z + x_0'\begin{bmatrix} 2\bar{Q} + \bar{T}'\bar{Q}\bar{T} \end{bmatrix}x_0
\]
(Computation of cost function)

\[
J(z, x_0) = x'_0 Q x_0 + \sum_{i=0}^{N-1} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} \begin{bmatrix} R \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} + \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \begin{bmatrix} B \\ AB \\ \vdots \\ A^{N-1}B \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix} + \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix} x_0
\]

\[
J(z, x_0) = (z' x_0)' \overline{Q}(z' x_0) + z' R z + x'_0 Q x_0
\]

\[
= \frac{1}{2} z' \overline{2(R + \overline{Q} \overline{S})} z + x'_0 \overline{2T' Q S} z + \frac{1}{2} x'_0 \overline{2(Q + T' Q T)} x_0
\]
Linear MPC – Unconstrained case

\[ J(z, x_0) = \frac{1}{2} z' H z + x'_0 F' z + \frac{1}{2} x'_0 Y x_0 \]

- The optimum is obtained by zeroing the gradient

\[ \nabla_z J(z, x_0) = H z + F x_0 = 0 \]

- And hence

\[ z^* = \begin{bmatrix} u_0^* \\ u_1^* \\ \vdots \\ u_{N-1}^* \end{bmatrix} = -H^{-1} F x_0 \]  
  (batch least squares)

- Alternative approach: use dynamic programming to find U* (Ricatti Iterations)
Linear MPC – Constrained case

- Optimization problem:
  \[ V(x_0) = \frac{1}{2} x_0' Y x_0 + \min_{z} \frac{1}{2} z' H z + x_0' F' z \]  
  (quadratic)

- Subject to:
  \[ G z \leq W + S x_0 \]  
  (linear)

Convex Quadratic Program (QP)

\[ z = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} \in \mathbb{R}^s, \ s \triangleq Nm \]  
is the optimization vector

- \( H = H' \succ 0 \) and \( H, F, Y, G, W, S \) depends on weights \( Q, R, P \), upper and lower bounds \( u_{\text{min}}, u_{\text{max}}, y_{\text{min}}, y_{\text{max}} \), and model matrices \( A, B, C \)
(Computation of constraint matrices)

- Input Constraints: $u_{\text{min}} \leq u_k \leq u_{\text{max}}, \ k = 0, \ldots, N - 1$

\[
\begin{bmatrix}
1 & 0 & \ldots & 0 \\
0 & 1 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & \ldots & 0 & 1 \\
-1 & 0 & \ldots & 0 \\
0 & -1 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & \ldots & 0 & -1
\end{bmatrix} \quad U \leq \begin{bmatrix} u_{\text{max}} \\ u_{\text{max}} \\ \vdots \\ u_{\text{max}} \\ -u_{\text{min}} \\ -u_{\text{min}} \\ \vdots \\ -u_{\text{min}} \end{bmatrix} \quad z = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}
\]

- Output constraints $y_k = CA^k x_0 + \sum_{i=0}^{k-1} CA^i B u_{k-1-i} \leq y_{\text{max}}, \ k = 1, \ldots, N$

\[
\begin{bmatrix}
CB & 0 & \ldots & 0 \\
CAB & CB & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
C^{A-1}B & \ldots & C^{A-1}B & CB
\end{bmatrix} \quad z \leq \begin{bmatrix} y_{\text{max}} \\ y_{\text{max}} \\ \vdots \\ y_{\text{max}} \end{bmatrix} - \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N} \end{bmatrix} x_0
\]
Linear MPC - Algorithm

- At each sampling step \( t \):
  - Measure/estimate the current state \( x(t) \)
Linear MPC - Algorithm

- At each sampling step $i$:
  - Measure/estimate the current state $x(t)$
  - Get the solution $z^* = \{u_0^*, \ldots, u_{N-1}^*\}$ of the QP

$$
\min_z \frac{1}{2} z'Hz + x'F'z
\quad s.t. \quad Gz \leq W + Sx(t)
$$
Linear MPC - Algorithm

- At each sampling step $t$:

- Measure/estimate the current state $x(t)$

- Get the solution $z^* = \{u_0^*, ..., u_{N-1}^*\}$ of the QP

- Apply only $u(t) = u_0^*$, discard remaining optimal inputs $u_1^*, ..., u_{N-1}^*$
Linear MPC - Algorithm

- At each sampling step $t$:
  - Measure/estimate the current state $x(t)$
  - Get the solution $z^* = \{u_0^*, \ldots, u_{N-1}^*\}$ of the QP:
    \[
    \min_\mathbf{z} \quad \frac{1}{2} \mathbf{z}' \mathbf{H} \mathbf{z} + \mathbf{x}' \mathbf{F}' \mathbf{z} \\
    \text{s.t.} \quad \mathbf{G} \mathbf{z} \leq \mathbf{W} + \mathbf{S} x(t)
    \]
  - Apply only $u(t) = u_0^*$, discard remaining optimal inputs $u_1^*, \ldots, u_{N-1}^*$
  - Feedback
Simple MPC Example

- X-axis only Position dynamics:

\[
\begin{bmatrix}
\dot{x} \\
\ddot{x}
\end{bmatrix} =
\begin{bmatrix}
0 & 1 \\
0 & 0
\end{bmatrix}
\begin{bmatrix}
x \\
\dot{x}
\end{bmatrix} +
\begin{bmatrix}
0 \\
-g
\end{bmatrix} \theta
\]
%% MPC Example: Position Dynamics

```matlab
clear; close all;

g = 9.8065;
A = [0 1; 0 0]; B = [0; -g]; C = eye(2); D = zeros(2,1);
x_ss = ss(A,B,C,D);
Ts = 0.01;
x_ss_d = c2d(x_ss,Ts,'zoh');

x_ss_d.InputName = {'theta'};
x_ss_d.OutputName = {'x_m','vx_m'};
x_ss_d.StateName = {'x','vx'};

N = 40;
N_c = 1;
mpcobj = mpc(x_ss_d,Ts,N,N_c)
```

```matlab
% Specify input constraints
mpcobj.MV.Min = -pi/4;
mpcobj.MV.Max = pi/4;
mpcobj.MV.RateMin = -2*pi;
mpcobj.MV.RateMax = 2*pi;

% Tune the Controller
mpcobj.W.OutputVariables = [1000 200];
mpcobj.W.ManipulatedVariables = .1;
T_steps = floor(3/Ts);
ref = [1 0; 1 0];
sim(mpcobj,T_steps,ref)
```
MPC Example
MPC Example
Examples of MPC for Aerial Robots

- Robust Model Predictive Control – Optimize against disturbances

Robust Explicit Model Predictive Flight Control of Unmanned Rotorcrafts: Design and Experimental Evaluation

Kostas Alexis, Christos Papachristos, Roland Siegwart and Anthony Tzes

This video presents some first video recorded results on our study on Robust Model Predictive Control of Unmanned Rotorcrafts. The designed control framework is shown to provide high quality robustness and performance characteristics in a balanced and tunable way.

Presented Experiments:
1) Trajectory Tracking subject to wind disturbance
2) Slung Load operations with disturbance on the load
3) Avoidance of known obstacles
Experiments (1) and (2) are evaluated using two different rotorcraft configurations (ASLquad and UPAT-TTR).
Examples of MPC for Aerial Robots

- Hybrid Model Predictive Control – Handle rapid change of dynamics

Aerial Robotic Contact-based Inspection Planning and Physical Interaction Control
Kostas Alexis, Georgios Darvianakis, Michael Burri and Roland Siegwart
Examples of MPC for Aerial Robots

- Nonlinear Model Predictive Control – Exploit flight envelope and actuation

Fast Nonlinear Model Predictive Control for Multicopter Attitude Tracking on SO(3)

Mina Kamel, Kostas Alexis, Markus Achtelik and Roland Siegwart

Position tracking without one propeller
Examples of MPC for Aerial Robots

- Linear Model Predictive Control – Fixed-Wing Low-Level Control
MPC Design and Code Generation

- Many tools exist: CVX, YALMIP, MPT, PYOMO, JuMP and more
- CVX-based tool CVXGEN: CVXGEN generates fast custom code for small, QP-representable convex optimization problems, using an online interface with no software installation. With minimal effort, turn a mathematical problem description into a high speed solver.

http://cvxgen.com/docs/index.html
Code Examples

- MATLAB Linear MPC design with CVX

- CVXGEN
  - [http://www.kostasalexis.com/linear-model-predictive-control.html](http://www.kostasalexis.com/linear-model-predictive-control.html)
Study Material

MPC in ROS

Thank you!

Please ask your question!