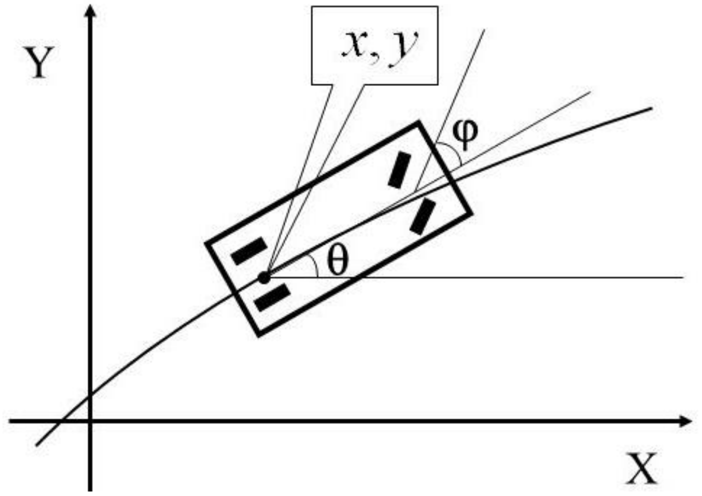
CONTROL EXERCISE

Point-to-Point control of a simple car using Model Predictive Control

(Matlab and C++ implementations using CasADi and ACADO)

By
Srivatsan Srinivasan

Simple Car Model & Euler-Forward Discretization



$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix}v\cos\psi\\ v\sin\psi\\ \frac{v}{L}\tan\delta\end{bmatrix} \text{ or } \begin{bmatrix}x_{k+1}\\ y_{k+1}\\ \psi_{k+1}\end{bmatrix} = \begin{bmatrix}v_k\cos\psi_k\\ v_k\sin\psi_k\\ \frac{v_k}{L}\tan\delta_k\end{bmatrix} + T_s\begin{bmatrix}\dot{x}_k\\ \dot{y}_k\\ \dot{\psi}_k\end{bmatrix}$$

$$where, \qquad X_u(k) = \begin{bmatrix}x_k\\ y_k\\ \psi_k\end{bmatrix} given\, u_k$$

x is the longitudinal displacement (m)

y is the lateral displacement (m)

ψ is the global orientation (rad)

v is the velocity input (m/s)

δ is the steering angle input (rad)

L is the wheel base (m)

T_s is the time step (s)

w is the track width (m)

MPC Setup (Single shooting)

$$Cost: C(X,u) = |X_u - X_{ref}|_Q^2 + |u - u_{ref}|_R^2$$

i.e. $\left(X_u - X_{ref}\right)' * Q * \left(X_u - X_{ref}\right) + \left(u - u_{ref}\right)' * R * \left(u - u_{ref}\right)$

$Subject\ to\ constraints$:

$X_u = F(u_k, X_0, t_k)$

$F(u_0, X_0, t_0) = X_0$ (Known initial state)

$v_{min} < v < v_{max}$ and $u_{min} < u < u_{max}$ (Input limit constraints)

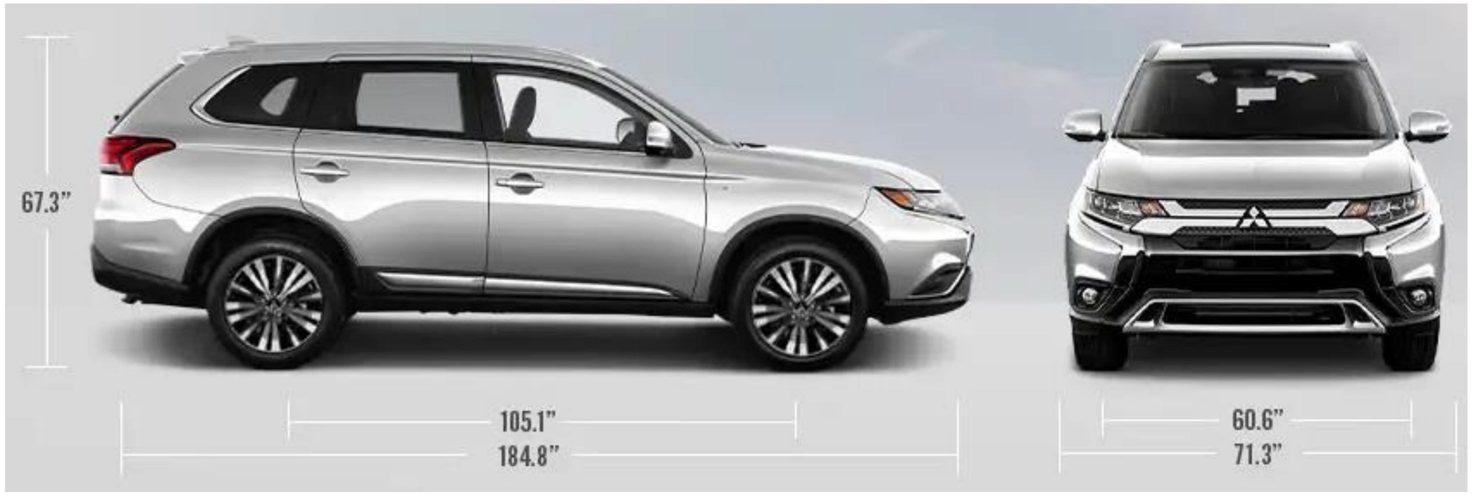
$X_{min} < X < X_{max}$ (State limit constraints)

$$Objective: Min\ J_N(X_0, u) = \sum_0^{N-1} C(\,X_u(k), u(k))$$

where N is the prediction horizon

Garage parking scenario (single shooting)

- Single shooting because the equality between the actual state and prediction state only set at initial condition.
- The rest of the prediction state is a recursion from the initial state.
- The shooting point for the solver of the boundary value problem becomes only the single point (initial condition).



SUV trackwidth and wheelbase

Garage parking scenario (single shooting)

Constraints

$$X_u = F(u_k, X_0, t_k)$$

$F(u_0, X_0, t_0) = [0,0,0]$ (Initial state)

$-5 < v < 15$ and $-1.4 < u < 1.4$ (Input limit constraints)

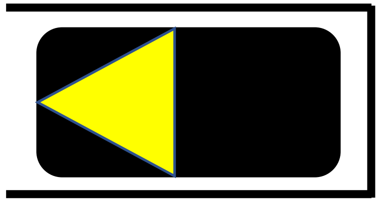
$[-5, -5, -inf] < X < [25, 25, inf]$ (Map constraints)

Parameters

$$N = 50, T_s = 0.1, L = 2.7, w = 1.5, \text{Solver} \rightarrow \text{ipopt}$$

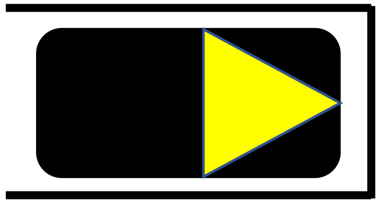
$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} \quad R = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.05 \end{bmatrix}$$

Single shooting performs well

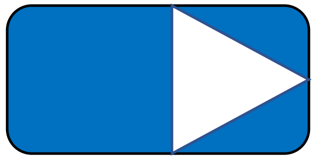


Ref 1: (20,20,pi)

Single shooting struggles but succeeds



Ref 2: (20,20,0)



Init Cond: (0,0,0)

Parallel parking (single shooting)

- **Red bold indicates parameters tuned.**
- **Q and R are kept the same for tuning convenience.**

<u>Constraints</u>

$$X_u = F(u_k, X_0, t_k)$$

$F(u_0, X_0, t_0) = [0,0,0]$ (Initial state)

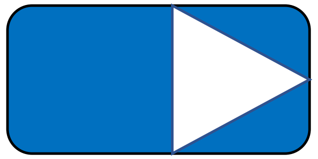
$-5 < v < \mathbf{10}$ and $-1.4 < u < 1.4$ (Input limit constraints)

$[-5, -5, -inf] < X < [25, 25, inf]$ (Map constraints)

<u>Parameters</u>

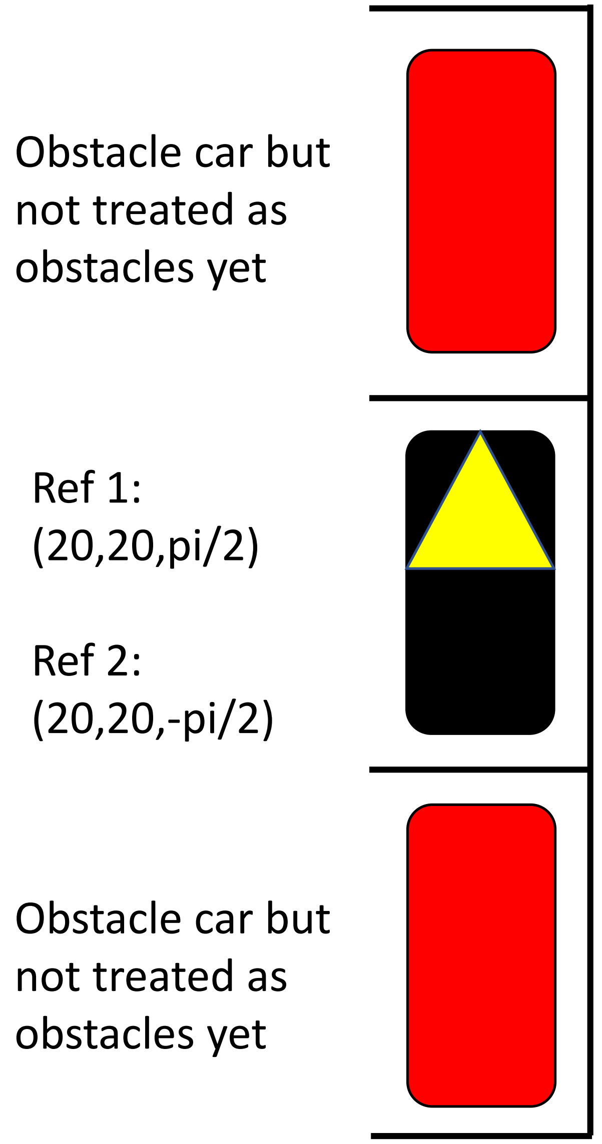
$N = \mathbf{100 - 170}, T_s = 0.1, L = 2.7, w = 1.5,$ Solver $\rightarrow$ ipopt

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} \quad R = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.05 \end{bmatrix}$$



Init Cond: (0,0,0)

Single shooting really struggles to find a good solution even at N=200 to achieve error tolerance.


Obstacle car but not treated as obstacles yet
Ref 1: (20,20,pi/2)
Ref 2: (20,20,-pi/2)
Obstacle car but not treated as obstacles yet

Multiple shooting

$$Cost: C(X,u) = |X_u - X_{ref}|_Q^2 + |u - u_{ref}|_R^2$$

$$Objective: \ Min \ J_N(X,u) = \sum_0^{N-1} C(\,X_u(k), u(k))$$

$Subject\ to\ constraints$:

$X_u(k+1) = F(X_u(k), u(k))$

$X_u(0) = X_0$ (Known/Measured initial state)

$v_{min} < v < v_{max}$ and $u_{min} < u < u_{max}$

$X_{min} < X < X_{max}$

- Now including measurements/feedback into the control loop
- The equality constraint $F(u_0, X_0, t_0) = X_0$ that was set at initial condition in single shooting is now enforced in every step.
- This now gives multiple shooting points that results in faster convergence.

$Additional\ constraint$:

$X_{k+1} - F(X_u(k), u(k)) = 0$ (Equality constraint at every step 'k')

Parallel parking (multiple shooting)

<u>Constraints</u>

$$X_u = F(u_k, X_0, t_k)$$

$F(u_0, X_0, t_0) = [0,0,0]$ (Initial state)

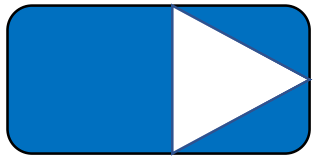
$-5 < v < \mathbf{10}$ and $-1.4 < u < 1.4$ (Input limit constraints)

$[-5, -5, -inf] < X < [25,25, inf]$ (Map constraints)

<u>Parameters</u>

$$N = \mathbf{75 - 120}, T_s = 0.1, L = 2.7, w = 1.5, \text{Solver} \rightarrow \text{ipopt}$$

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} \quad R = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.05 \end{bmatrix}$$



Init Cond: (0,0,0)

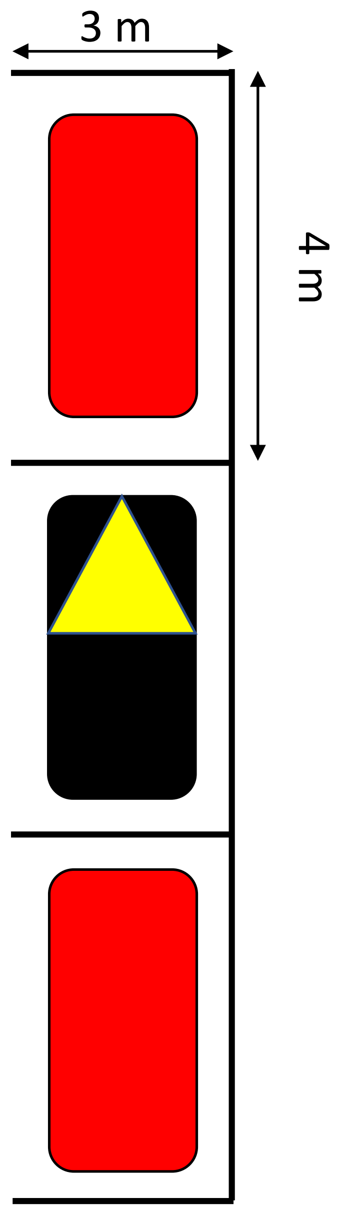
Multiple shooting finds a good solution and also at a lesser computation time.

Obstacle car but not treated as obstacles yet

Ref 1: (20,20,pi/2)

Ref 2: (20,20,-pi/2)

Obstacle car but not treated as obstacles yet



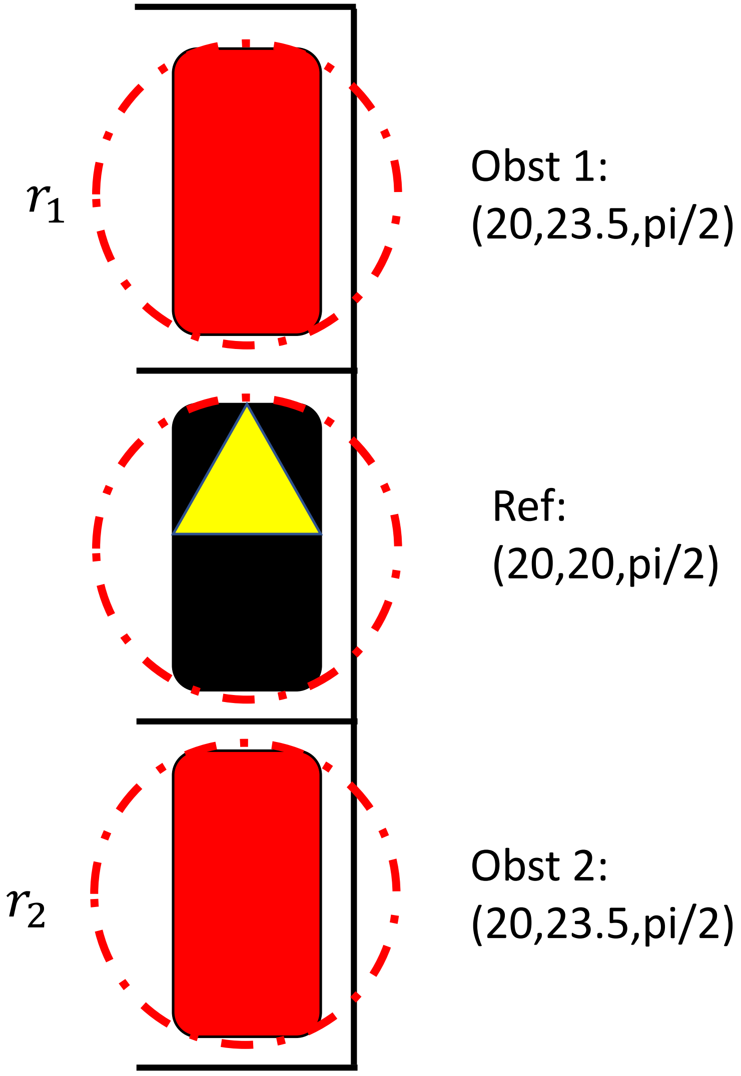
Obstacle avoidance(MS)

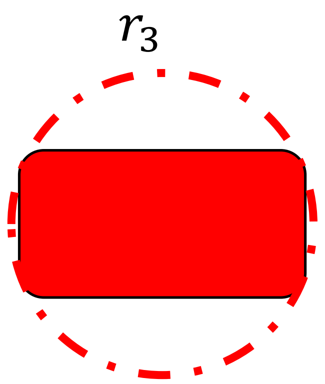
Obstacle cars added as constraints

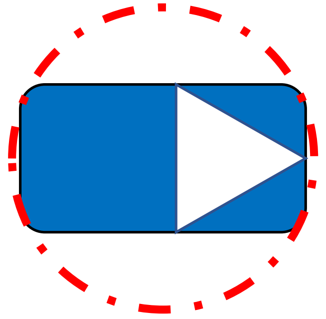
Additional constraints:

For every obstacle: $-\inf < -\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2} + (r + r_{obs}) \le 0$

$N = \mathbf{170 - 200}$







MPC finds a good solution avoiding all obstacles.

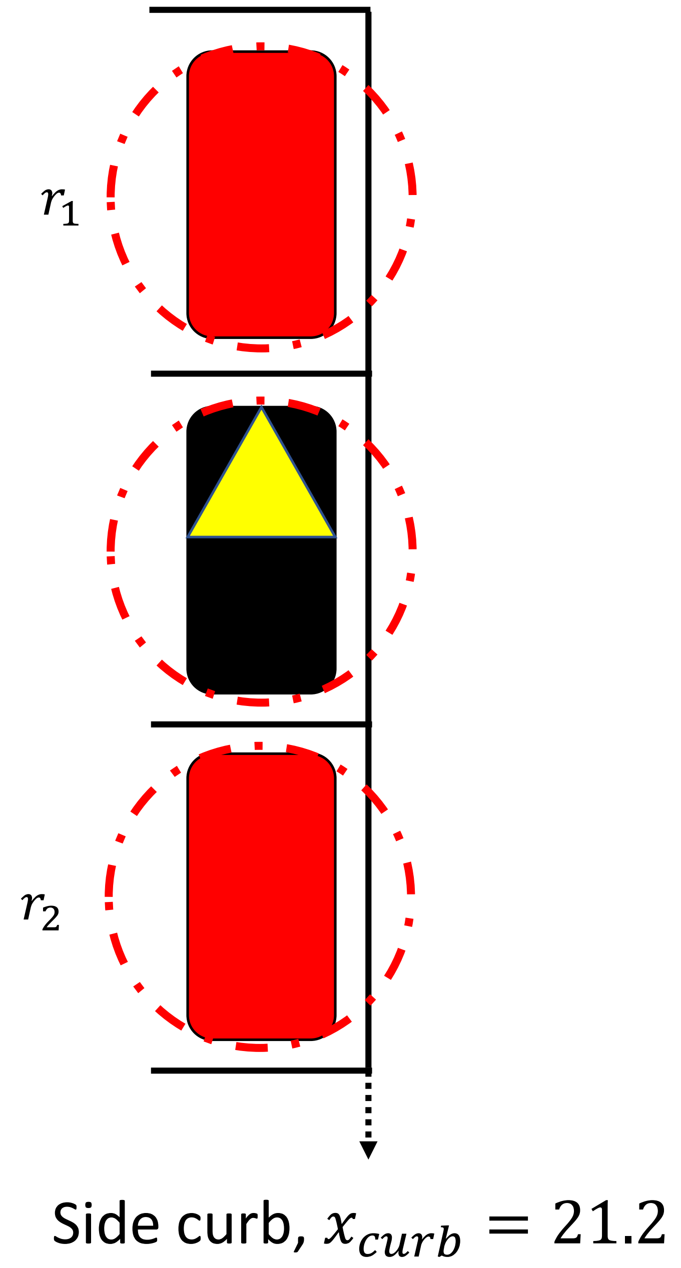
Obstacle avoidance(MS)

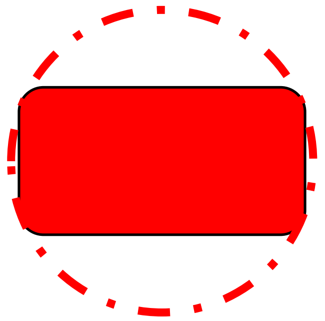
PARKING SIDE CURB ADDED AS CONSTRAINT

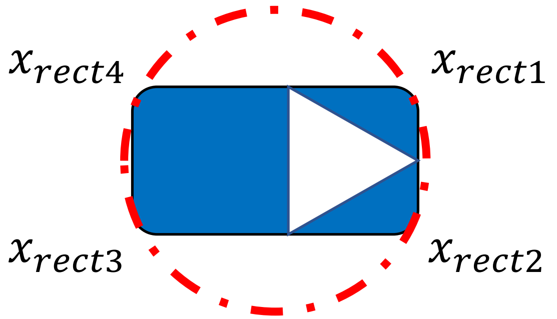
Additional constraints:

For ego car: $0.2 \leq (x_{curb} - x_{rect1,2,3,4}) < inf$

$N = \mathbf{170 - 200}$







Init Cond: (0,0,0)

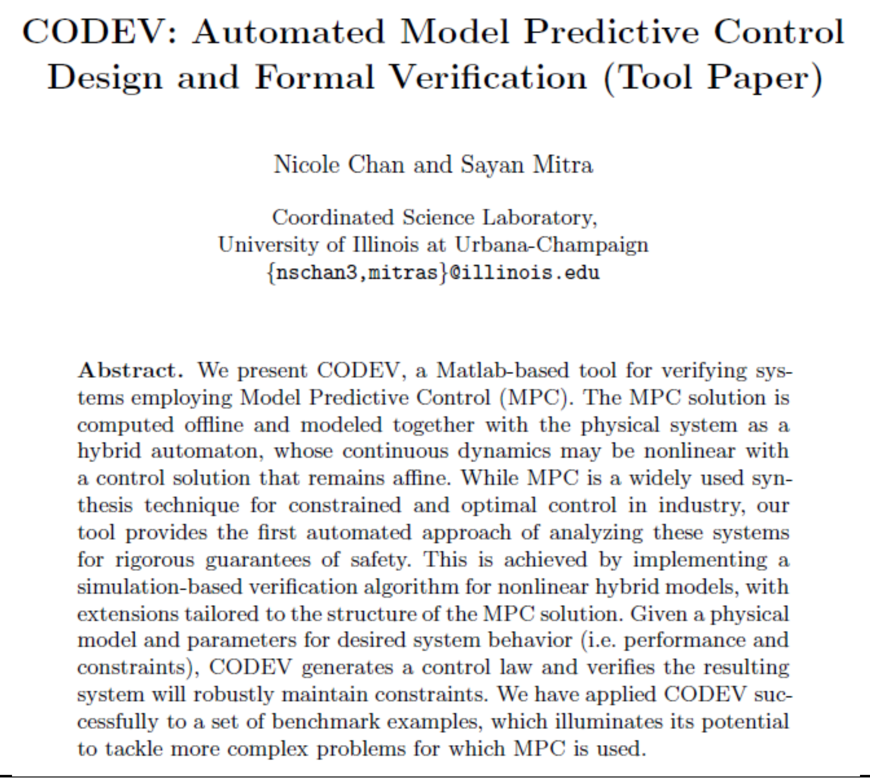
MPC does not find a solution even at N=200.

Possible improvements/additions

- Change the distance in input to change in input at the cost function level $(u(k+1) - u(k))$ instead of $(u - u_{ref})$
- Tuning Q and R further.
- EKF for measurement estimation
- Trying all the other solvers
- Implement trajectory tracking instead of point-point control.
- Comparison with other controllers like Stanley, Pure-Pursuit
- Change vehicle model to more complex models

CODEV

Biggest problem with MPC is tuning. There are lot of tuning factors and there is no way to identify right parameters and provide guarantees without extensive simulation. CODEV (image below for reference) can help.



CODEV: Automated Model Predictive Control Design and Formal Verification (Tool Paper)

Nicole Chan and Sayan Mitra

Coordinated Science Laboratory,
University of Illinois at Urbana-Champaign
{nschan3,mitras}@illinois.edu

Abstract. We present CODEV, a Matlab-based tool for verifying systems employing Model Predictive Control (MPC). The MPC solution is computed offline and modeled together with the physical system as a hybrid automaton, whose continuous dynamics may be nonlinear with a control solution that remains affine. While MPC is a widely used synthesis technique for constrained and optimal control in industry, our tool provides the first automated approach of analyzing these systems for rigorous guarantees of safety. This is achieved by implementing a simulation-based verification algorithm for nonlinear hybrid models, with extensions tailored to the structure of the MPC solution. Given a physical model and parameters for desired system behavior (i.e. performance and constraints), CODEV generates a control law and verifies the resulting system will robustly maintain constraints. We have applied CODEV successfully to a set of benchmark examples, which illuminates its potential to tackle more complex problems for which MPC is used.