Model predictive speed and steering control

Model predictive speed and steering control

Model predictive speed and steering control

code:

PythonRobotics/model_predictive_speed_and_steer_control.py at master · AtsushiSakai/PythonRobotics (https://github.com/AtsushiSakai/PythonRobotics/blob/master/PathTracking/model_predictive_speed_and_steer_control/model_predictive_speed_and_steer_control.py)

This is a path tracking simulation using model predictive control (MPC).

The MPC controller controls vehicle speed and steering base on linealized model.

This code uses cvxpy as an optimization modeling tool.

- Welcome to CVXPY 1.0 — CVXPY 1.0.6 documentation (http://www.cvxpy.org/)

MPC modeling

State vector is:

$$z = [x, y, v, \phi]$$

x: x-position, y:y-position, v:velocity, φ: yaw angle

Input vector is:

$$u = [a, \delta]$$

a: accellation, δ: steering angle

The MPC cotroller minimize this cost function for path tracking:

$$min\ Q_f(z_{T,ref} - z_T)^2 + Q\Sigma(z_{t,ref} - z_t)^2 + R\Sigma {u_t}^2 + R_d\Sigma(u_{t+1} - u_t)^2$$

z_ref come from target path and speed.

subject to:

- Linearlied vehicle model

$$z_{t+1} = Az_t + Bu + C$$

- Maximum steering speed

$$|u_{t+1} - u_t| < du_{max}$$

- Maximum steering angle

$$|u_t| < u_{max}$$

- Initial state

$$z_0 = z_{0,ob}$$

- Maximum and minimum speed

$$v_{min} < v_t < v_{max}$$

- Maximum and minimum input

$$u_{min} < u_t < u_{max}$$

This is implemented at

PythonRobotics/model_predictive_speed_and_steer_control.py at f51a73f47cb922a12659f8ce2d544c347a2a8156 · AtsushiSakai/PythonRobotics (https://github.com/AtsushiSakai/PythonRobotics/blob/f51a73f47cb922a12659f8ce2d544c347a2a8156/PathTracking/model_predictive_speed_and_steer_control/model_predictive_s L301)

Vehicle model linearization

Vehicle model is

$$\dot{x} = vcos(\phi)$$

$$\dot{y} = vsin((\phi)$$

$$\dot{v} = a$$

$$\dot{\phi} = \frac{vtan(\delta)}{L}$$

ODE is

$$\dot{z} = \frac{\partial}{\partial z} z = f(z, u) = A'z + B'u$$

where

$$A' = \begin{bmatrix} \frac{\partial}{\partial x} vcos(\phi) & \frac{\partial}{\partial y} vcos(\phi) & \frac{\partial}{\partial v} vcos(\phi) & \frac{\partial}{\partial \phi} vcos(\phi) \\ \frac{\partial}{\partial x} vsin(\phi) & \frac{\partial}{\partial y} vsin(\phi) & \frac{\partial}{\partial v} vsin(\phi) & \frac{\partial}{\partial \phi} vsin(\phi) \\ \frac{\partial}{\partial x} a & \frac{\partial}{\partial y} a & \frac{\partial}{\partial v} a & \frac{\partial}{\partial \phi} a \\ \frac{\partial}{\partial x} \frac{vtan(\delta)}{L} & \frac{\partial}{\partial y} \frac{vtan(\delta)}{L} & \frac{\partial}{\partial v} \frac{vtan(\delta)}{L} & \frac{\partial}{\partial \phi} \frac{vtan(\delta)}{L} \end{bmatrix}$$
$$= \begin{bmatrix} 0 & 0 & cos(\bar{\phi}) & -\bar{v}sin(\bar{\phi}) \\ 0 & 0 & sin(\bar{\phi}) & \bar{v}cos(\bar{\phi}) \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{tan(\bar{\delta})}{L} & 0 \end{bmatrix}$$

$$B' = \begin{bmatrix} \frac{\partial}{\partial a} vcos(\phi) & \frac{\partial}{\partial \delta} vcos(\phi) \\ \frac{\partial}{\partial a} vsin(\phi) & \frac{\partial}{\partial \delta} vsin(\phi) \\ \frac{\partial}{\partial a} a & \frac{\partial}{\partial \delta} a \\ \frac{\partial}{\partial a} \frac{vtan(\delta)}{L} & \frac{\partial}{\partial \delta} \frac{vtan(\delta)}{L} \end{bmatrix}$$
$$= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & \frac{\bar{v}}{Lcos^2(\bar{\delta})} \end{bmatrix}$$

You can get a discrete-time mode with Forward Euler Discretization with sampling time dt.

$$z_{k+1} = z_k + f(z_k, u_k)dt$$

Using first degree Tayer expantion around zbar and ubar

$$z_{k+1} = z_k + (f(\bar{z}, \bar{u}) + A'z_k + B'u_k - A'\bar{z} - B'\bar{u})dt$$
$$z_{k+1} = (I + dtA')z_k + (dtB')u_k + (f(\bar{z}, \bar{u}) - A'\bar{z} - B'\bar{u})dt$$

So,

$$z_{k+1} = Az_k + Bu_k + C$$

where,

$$A = (I + dtA')$$
$$= \begin{bmatrix} 1 & 0 & cos(\bar{\phi})dt & -\bar{v}sin(\bar{\phi})dt \\ 0 & 1 & sin(\bar{\phi})dt & \bar{v}cos(\bar{\phi})dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{tan(\bar{\delta})}{L}dt & 1 \end{bmatrix}$$

$$B = dtB'$$
$$= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ dt & 0 \\ 0 & \frac{\bar{v}}{Lcos^2(\bar{\delta})}dt \end{bmatrix}$$

$$C = (f(\bar{z}, \bar{u}) - A'\bar{z} - B'\bar{u})dt$$
$$= dt(\begin{bmatrix} \bar{v}cos(\bar{\phi}) \\ \bar{v}sin(\bar{\phi}) \\ \bar{a} \\ \frac{\bar{v}tan(\bar{\delta})}{L} \end{bmatrix} - \begin{bmatrix} \bar{v}cos(\bar{\phi}) - \bar{v}sin(\bar{\phi})\bar{\phi} \\ \bar{v}sin(\bar{\phi}) + \bar{v}cos(\bar{\phi})\bar{\phi} \\ 0 \\ \frac{\bar{v}tan(\bar{\delta})}{L} \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ \bar{a} \\ \frac{\bar{v}\bar{\delta}}{Lcos^2(\bar{\delta})} \end{bmatrix})$$
$$= \begin{bmatrix} \bar{v}sin(\bar{\phi})\bar{\phi}dt \\ -\bar{v}cos(\bar{\phi})\bar{\phi}dt \\ 0 \\ -\frac{\bar{v}\bar{\delta}}{Lcos^2(\bar{\delta})}dt \end{bmatrix}$$

This equation is implemented at

PythonRobotics/model_predictive_speed_and_steer_control.py at eb6d1cbe6fc90c7be9210bf153b3a04f177cc138 · AtsushiSakai/PythonRobotics (https://github.com/AtsushiSakai/PythonRobotics/blob/eb6d1cbe6fc90c7be9210bf153b3a04f177cc138/PathTracking/model_predictive_speed_and_steer_control/model_predictive_sp L102)

Reference

- Vehicle Dynamics and Control | Rajesh Rajamani | Springer (http://www.springer.com/us/book/9781461414322)
- MPC Course Material - MPC Lab @ UC-Berkeley (http://www.mpc.berkeley.edu/mpc-course-material)