Motion_Planning_Hw6

深蓝学院《移动机器人运动规划》第九期，第6章课程作业。该代码实现了基于nonlinear MPC的轨迹跟踪控制，并解决了系统控制时延的问题。

HOW TO RUN

```
./install_tools.sh
catkin_make -j1
source devel/setup.bash
roslaunch mpc_car simulation.launch
```

HOW TO TURN PARAMETERS

```
./src/mpc_car/config/mpc_car.yaml -> mpc parameters
./src/car_simulator/config/car_simulator.yaml -> initial states (in simulation)
```

Theory

Linearize Nonlinear Model

将非线性模型泰勒展开后，整理成状态传递方程的形式如下：

$$
\begin{aligned}
\mathbf{\dot{x}} &= \mathbf{A_c x} + \mathbf{B_c u} + \mathbf{g_c} \\
\frac{\mathbf{x_{k+1}} - \mathbf{x_k}}{T_s} &= \mathbf{A_c x_k} + \mathbf{B_c u_k} + \mathbf{g_c} \\
\mathbf{x_{k+1}} &= (\mathbf{I} + T_s \mathbf{A}_c)\mathbf{x_k} + T_s \mathbf{B_c u_k} + T_s \mathbf{g_c} \\
\mathbf{x_{k+1}} &= \mathbf{A_k x_k} + \mathbf{B_k u_k} + \mathbf{g_k}
\end{aligned}
$$

选取经典的自行车非线性控制模型，因此矩阵内部各项具体表达如下：

$$
\begin{bmatrix} \dot{p_x} \\ \dot{p_y} \\ \dot{\phi} \\ \dot{v} \end{bmatrix} =
\underbrace{\begin{pmatrix} 0 & 0 & -\bar{v}\sin\bar{\phi} & \cos\bar{\phi} \\ 0 & 0 & \bar{v}\cos\bar{\phi} & \sin\bar{\phi} \\ 0 & 0 & 0 & \frac{\tan\bar{\delta}}{L} \\ 0 & 0 & 0 & 0 \end{pmatrix}}_{\boldsymbol{A_c}}
\underbrace{\begin{bmatrix} p_x \\ p_y \\ \phi \\ v \end{bmatrix}}_{\boldsymbol{x}} +
\underbrace{\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & \frac{\bar{v}}{L}\frac{1}{\cos^2\bar{\delta}} \\ 1 & 0 \end{pmatrix}}_{\boldsymbol{B_c}}
\underbrace{\begin{bmatrix} a \\ \delta \end{bmatrix}}_{\boldsymbol{u}} +
\underbrace{\begin{pmatrix} \bar{v}\bar{\phi}\sin\bar{\phi} \\ -\bar{v}\bar{\phi}\cos\bar{\phi} \\ -\frac{\bar{v}}{L}\frac{\bar{\delta}}{\cos^2\bar{\delta}} \\ 0 \end{pmatrix}}_{\boldsymbol{g_c}}
$$

Constrained Quadratic Problem

由于主要任务为跟踪预设轨迹，构建代价函数时主要考虑偏离轨迹和航向角而产生的代价，而控制量仅作约束处理，因此约束二次型优化问题描述如下：

$$
\begin{aligned}
minJ = \sum_{i=0}^{N}(x-x_r)^2 + (y-y_r)^2 + rho*(phi-phi_r)^2 \\
s.t. -0.1 <= v_k <= v_{max} \\
|a_k| <= a_{max} \\
|\delta_k| <= \delta_{max} \\
|\delta_{k+1} - \delta_k| <= \dot{\delta}_{max} T_S
\end{aligned}
$$

为对照代码中输入的矩阵，给出矩阵形式的表达：

$$J(\mathbf{z}, \mathbf{x_0}) = (\mathbf{X} - \mathbf{X_{ref}})^{\mathbf{T}} \bar{\mathbf{Q}} (\mathbf{X} - \mathbf{X_{ref}})$$

代入状态空间传递方程：

$$J(\mathbf{z}, \mathbf{x_0}) = (\bar{\mathbf{A}}_\mathbf{d}\mathbf{x_0} + \bar{\mathbf{B}}_\mathbf{d}\mathbf{z} + \bar{\mathbf{g}}_\mathbf{c} - \mathbf{X_{ref}})^{\mathbf{T}} \bar{\mathbf{Q}} (\bar{\mathbf{A}}_\mathbf{d}\mathbf{x_0} + \bar{\mathbf{B}}_\mathbf{d}\mathbf{z} + \bar{\mathbf{g}}_\mathbf{c} - \mathbf{X_{ref}})$$

其中X为$(\mathbf{x}_1^T, \mathbf{x}_2^T, \mathbf{x}_3^T, ..., \mathbf{x}_N^T)^T$,$\mathbf{x}_i$为某一离散时刻的状态向量。

控制序列$\mathbf{z}$为待优化变量，对其求J的梯度，由此可得到OSQP求解器需要的P和q系数矩阵:

$$
\begin{aligned}
\nabla_\mathbf{z} J &= \mathbf{Pz} + \mathbf{q} \\
&= \mathbf{2}\bar{\mathbf{B}}_\mathbf{d}^\mathbf{T} \bar{\mathbf{Q}} \bar{\mathbf{B}}_\mathbf{d}\mathbf{z} + \mathbf{2}\bar{\mathbf{B}}_\mathbf{d}^\mathbf{T} \bar{\mathbf{Q}} (\bar{\mathbf{A}}_\mathbf{d}\mathbf{x_0} + \bar{\mathbf{g}}_\mathbf{c} - \mathbf{X_{ref}})
\end{aligned}
$$

其中X_{ref}为跟踪的参考轨迹序列

Delay Compensation

对于带有控制时延的系统，需联合上一次优化得到的控制序列，将观测状态(t时刻观察到的)补偿至未来的$t+\tau$时刻的状态量，才能作为MPC求解过程的初始状态$\bar{\mathbf{x}}_\mathbf{0}$。

$$\bar{\mathbf{x}}_\mathbf{0} = \mathbf{x}(t+\tau)$$

主要基于非线性方程 $\dot{\mathbf{x}} = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t-\tau)$，采用四阶龙格库塔法求得$\mathbf{x}(t+\tau)$

AN EXAMPLE