深蓝学院
shenlanxueyuan.com

第六章作业思路

主讲人 刘武

作业要求

1. Implement MPC of tracking reference trajectory in C++;
2. Implement MPC with delays in C++;
3. Implement MPCC in C++ (optional);

第一题

深蓝学院
shenlanxueyuan.com

初始化 Ad, Bd, gd

```
// TODO: set initial value of Ad, Bd, gd
Ad_.setIdentity(); // Ad for instance
                   // ...
                   // set size of sparse matrices

Bd_.setZero();
gd_.setZero();
```

第一题

深蓝学院 shenlanxueyuan.com

Linear Time-Varying MPC and Nonlinear MPC

- Linearization and discretization:

$$\begin{bmatrix} \dot{p}_x \\ \dot{p}_y \\ \dot{v} \\ \dot{\phi} \end{bmatrix} = \underbrace{\begin{pmatrix} 0 & 0 & -\bar{v}\sin\bar{\phi} & \cos\bar{\phi} \\ 0 & 0 & \bar{v}\cos\bar{\phi} & \sin\bar{\phi} \\ 0 & 0 & 0 & \frac{\tan\bar{\delta}}{L} \\ 0 & 0 & 0 & 0 \end{pmatrix}}_{\boldsymbol{A}_c} \underbrace{\begin{bmatrix} p_x \\ p_y \\ v \\ \phi \end{bmatrix}}_{\boldsymbol{x}} + \underbrace{\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & \frac{\bar{v}}{L}\frac{1}{\cos^2\bar{\delta}} \\ 1 & 0 \end{pmatrix}}_{\boldsymbol{B}_c} \underbrace{\begin{bmatrix} a \\ \delta \end{bmatrix}}_{\boldsymbol{u}} + \underbrace{\begin{pmatrix} \bar{v}\bar{\phi}\sin\bar{\phi} \\ -\bar{v}\bar{\phi}\cos\bar{\phi} \\ 0 \\ -\frac{\bar{v}}{L}\frac{\bar{\delta}}{\cos^2\bar{\delta}} \end{pmatrix}}_{\boldsymbol{g}_c}$$

$$\boldsymbol{x}_{k+1} = (\boldsymbol{I} + T_s\boldsymbol{A}_c)\boldsymbol{x}_k + T_s\boldsymbol{B}_c\boldsymbol{u}_k + T_s\boldsymbol{g}_c$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{g}_k$$

linearization 函数

第一题

linearization函数

```
void linearization(const double& phi, const double& v, const double& delta) {
    // x_{k+1} = Ad * x_{k} + Bd * u_k + gd
    // TODO: set values to Ad_, Bd_, gd_
    // ...
    Ad_ << 1, 0, -dt_ * v * std::sin(phi), dt_ * std::cos(phi),
        0, 1, dt_ * v * std::cos(phi), dt_ * std::sin(phi),
        0, 0, 1, dt_ * std::tan(delta) / ll_,
        0, 0, 0, 1;
    Bd_ << 0, 0,
        0, 0,
        0, dt_ * v / (ll_ * std::cos(delta) * std::cos(delta)),
        dt_ * 1.0, 0;

    gd_ << dt_ * v * phi * std::sin(phi),
        -dt_ * v * phi * std::cos(phi),
        -dt_ * v * delta / (ll_ * std::cos(delta) * std::cos(delta)),
        0;
    return;
}
```

第一题

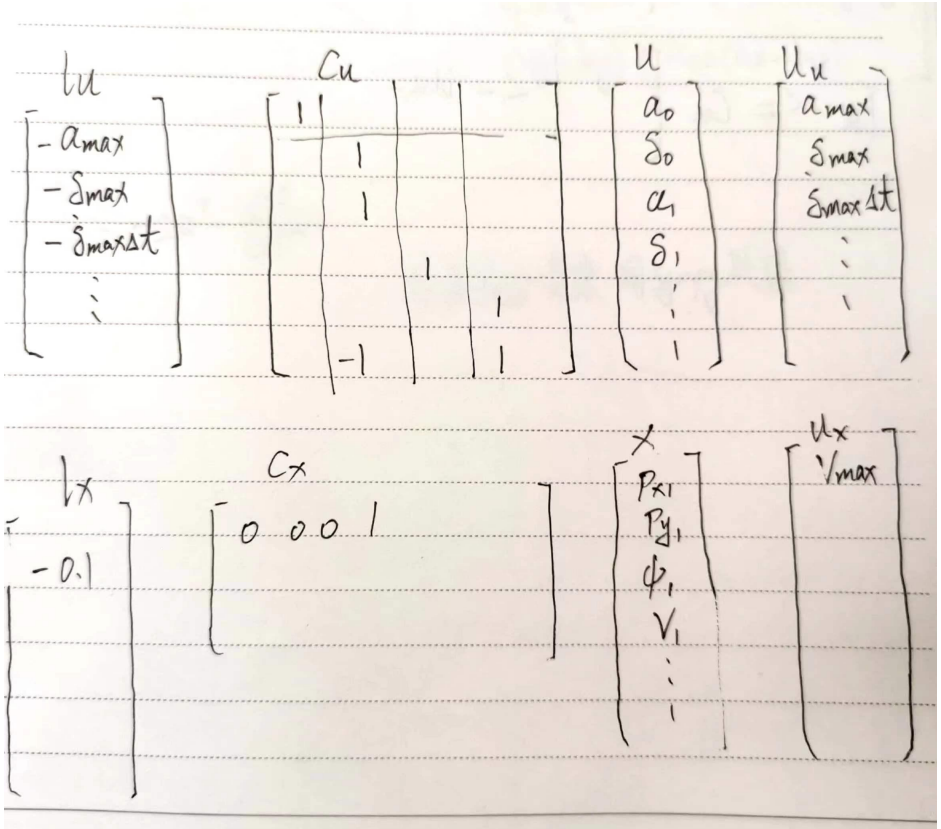
深蓝学院
shenlanxueyuan.com

a, delta, ddelta,v的约束性限制

https://blog.csdn.net/shoufei403/article/details/108672152

```
/**
 * osqp interface:
 * minimize     0.5 x^T P_ x + q_^T x
 * subject to   l_ <= A_ x <= u_
 **/
Eigen::SparseMatrix<double> P_, q_, A_, l_, u_;

/* *
 *              /  x1  \
 *              |  x2  |
 *  lx_ <=  Cx_ |  x3  |  <= ux_
 *              | ...  |
 *              \  xN  /
 * */
Eigen::SparseMatrix<double> Cx_, lx_, ux_; // p, v constrains
/* *
 *              /  u0  \
 *              |  u1  |
 *  lu_ <=  Cu  |  u2  |  <= uu_
 *              | ...  |
 *              \ uN-1 /
 * */
```

第一题




```
for (int i = 0; i < N_; ++i) {
    // TODO: set stage constraints of inputs (a, delta, ddelta)
    // -a_max <= a <= a_max for instance:
    Cu_.coeffRef(i * 3 + 0, i * m + 0) = 1;
    lu_.coeffRef(i * 3 + 0, 0) = -a_max_;
    uu_.coeffRef(i * 3 + 0, 0) = a_max_;

    Cu_.coeffRef(i * 3 + 1, i * m + 1) = 1;
    lu_.coeffRef(i * 3 + 1, 0) = -delta_max_;
    uu_.coeffRef(i * 3 + 1, 0) = delta_max_;

    Cu_.coeffRef(i * 3 + 2, i * m + 1) = 1;
    lu_.coeffRef(i * 3 + 2, 0) = -ddelta_max_ * dt_;
    uu_.coeffRef(i * 3 + 2, 0) = ddelta_max_ * dt_;
    if (0 != i) {
        Cu_.coeffRef(i * 3 + 2, (i - 1) * m + 1) = -1;
    }

    // TODO: set stage constraints of states (v)
    // -v_max <= v <= v_max
    // Cx_.coeffRef( ...
    // lx_.coeffRef( ...
    // ux_.coeffRef( ...
    Cx_.coeffRef(i * 1, i * n + 3) = 1;
    lx_.coeffRef(i, 0) = -0.1;
    ux_.coeffRef(i, 0) = v_max_;
}
```

第一题

深蓝学院 shenlanxueyuan.com

BB AA gg设置

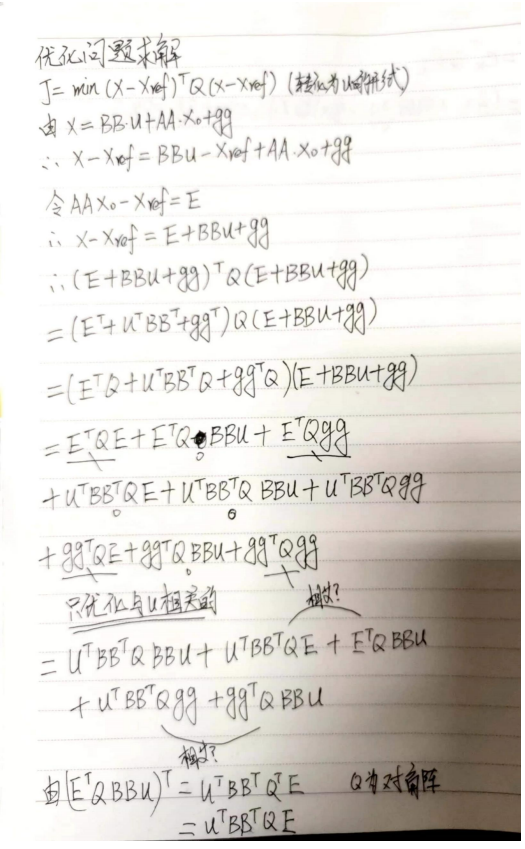
```
// calculate big state-space matrices
/* *                BB                AA
 * x1    /       B    0  ... 0 \    /   A \
 * x2    |      AB    B  ... 0 |    |  A2 |
 * x3  = |    A^2B   AB  ... 0 |u + | ... |x0 + gg
 * ...   |     ...  ...  ... 0 |    | ... |
 * xN    \A^(n-1)B  ...  ... B /    \ A^N /
 *
 *     X = BB * U + AA * x0 + gg
 * */
if (i == 0) {
    BB.block(0, 0, n, m) = Bd_;
    AA.block(0, 0, n, n) = Ad_;
    gg.block(0, 0, n, 1) = gd_;
} else {
    // TODO: set BB·AA·gg
    // ...

    BB.block(i * n, m * i, n, m) = Bd_;
    BB.block(i * n, 0, n, i * m) = Ad_ * BB.block((i - 1) * n, 0, n, i * m);

    AA.block(i * n, 0, n, n) = Ad_ * AA.block((i - 1) * n, 0, n, n);

    gg.block(i * n, 0, n, 1) = Ad_ * gg.block((i - 1) * n, 0, n, 1) + gd_;
}
```

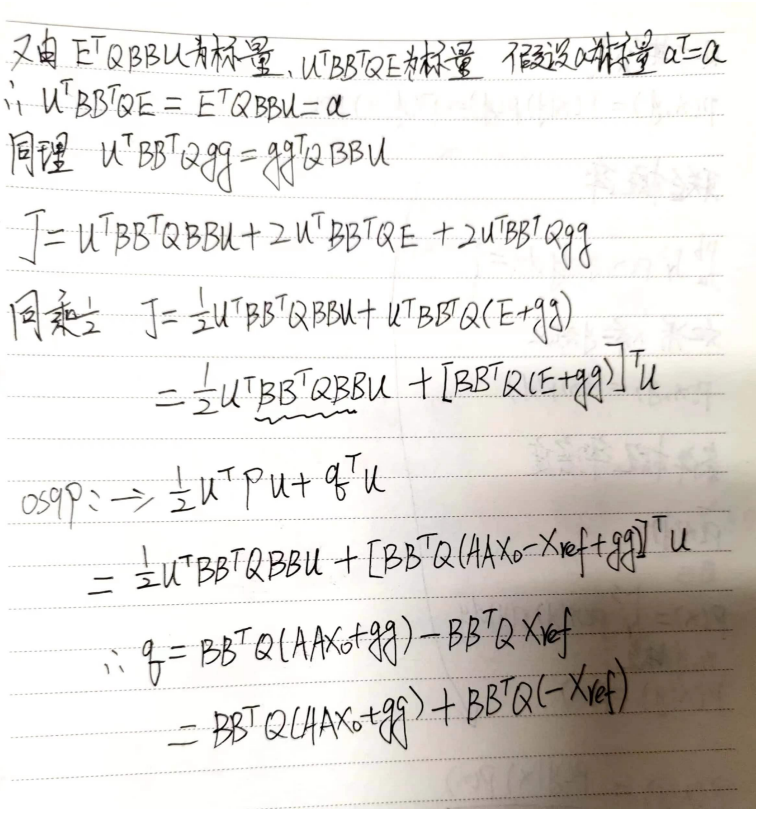
第一题

qx设置

```
        // ...

        qx.coeffRef(i * n, 0) = -xy(0);
        qx.coeffRef(i * n + 1, 0) = -xy(1);
        qx.coeffRef(i * n + 2, 0) = -rho_ * phi;
        if (i == N_ - 1) {
            qx.coeffRef(i * n, 0) *= rhoN_;
            qx.coeffRef(i * n + 1, 0) *= rhoN_;
            qx.coeffRef(i * n + 2, 0) *= rhoN_;
        }
        // qx.coeffRef(i * n + 3, 0) = -v;

        s0 += desired_v_ * dt_;
        s0 = s0 < s_.arcL() ? s0 : s_.arcL();
    }

    qx = Qx_ * qx;
```

第一题

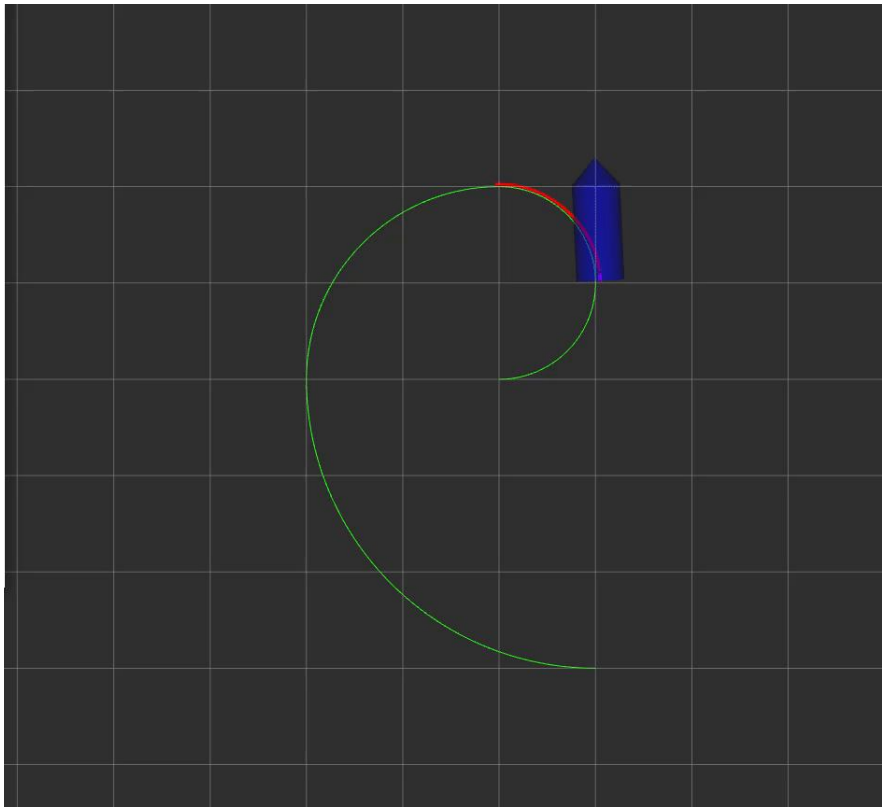

优化问题求解
$J=\min (X-X_{ref})^T Q(X-X_{ref})$ (转化为qp问题形式)
由 $X=BBU+AA\cdot X_0+gg$
$\therefore X-X_{ref}=BBu-X_{ref}+AA\cdot X_0+gg$
令 $AAX_0-X_{ref}=E$
$\therefore X-X_{ref}=E+BBU+gg$
$\therefore (E+BBU+gg)^T Q(E+BBU+gg)$
$=(E^T+U^TBB^T+gg^T)Q(E+BBU+gg)$
$=(E^TQ+U^TBB^TQ+gg^TQ)(E+BBU+gg)$
$=E^TQE+E^TQBBU+E^TQgg$
$+U^TBB^TQE+U^TBB^TQBBU+U^TBB^TQgg$
$+gg^TQE+gg^TQBBU+gg^TQgg$
只优化与U相关的
$=U^TBB^TQBBU+U^TBB^TQE+E^TQBBU$
$+U^TBB^TQgg+gg^TQBBU$
由 $(E^TQBBU)^T=U^TBB^TQ^TE$ Q为对角阵
$=U^TBB^TQE$


又由 E^TQBBU 为标量、U^TBB^TQE 为标量 假设 α 为标量 $\alpha^T=\alpha$
$\therefore U^TBB^TQE=E^TQBBU=\alpha$
同理 $U^TBB^TQgg=gg^TQBBU$
$J=U^TBB^TQBBU+2U^TBB^TQE+2U^TBB^TQgg$
同乘$\frac{1}{2}$ $J=\frac{1}{2}U^TBB^TQBBU+U^TBB^TQ(E+gg)$
$=\frac{1}{2}U^TBB^TQBBU+[BB^TQ(E+gg)]^TU$
$\Rightarrow \frac{1}{2}U^TPU+q^TU$
$=\frac{1}{2}U^TBB^TQBBU+[BB^TQ(HAX_0-X_{ref}+gg)]^TU$
$\therefore q=BB^TQ(AAX_0+gg)-BB^TQX_{ref}$
$=BB^TQ(HAX_0+gg)+BB^TQ(-X_{ref})$

第二题


```
VectorX compensateDelay(const VectorX& x0) {
    VectorX x0_delay = x0;
    // TODO: compensate delay
    // ...
    double dt = 0.001;
    for (double t = delay_; t > 0; t -= dt) {
        int i = std::ceil(t / dt_);
        VectorU input = historyInput_[history_length_ - i];
        step(x0_delay, input, dt);
    }
    return x0_delay;
}
```

结果



在线问答

深蓝学院
shenlanxueyuan.com

Q&A

深蓝学院
shenlanxueyuan.com

感谢各位聆听！

Thanks for Listening