1. Problem description

1.1 background

We are given a description of the dynamics of a robot manipulator in the form of the equation

$$M(\theta)\ddot{\theta}+C(\theta,\dot{\theta})\dot{\theta}+N(\theta,\dot{\theta})=\tau \qquad (1.11)$$

Where $\theta \in \mathbb{R}^n$ is the set of configuration variables for the robot and $\tau \in \mathbb{R}^n$ denotes the torques applied at the joints. We are also given a joint trajectory $\theta_d(\bullet)$ which we wish to track. For simplicity, we assume that θ_d is specified for all time and that it is at least twice differentiable.

If we have a perfect model of the robot and $\theta(0)=\theta_d(0), \dot{\theta}(0)=\dot{\theta}_d(0)$, then we may solve our problem by choosing

$$\tau = M(\theta_d)\ddot{\theta}_d + C(\theta_d,\dot{\theta}_d)\dot{\theta}_d + N(\theta_d,\dot{\theta}_d)$$

Since both θ and θ_d satisfy the same differential equation and have the same initial conditions, it follows from the uniqueness of the solutions of differential equations that $\theta(t)=\theta_d(t)$ for all $t \geq 0$. This an example of an open-loop control law: the Current state of the robot is not used in choosing the control inputs.

There are several approaches for designing stable robot control laws. Using the structural properties of robot dynamics, we will be able to prove stability of these control laws for all robot shaving those properties. Hence, we do not need to design control laws for a specific robot; as long as we show that stability of a particular control algorithm requires only those properties given in Lemma4.2 on page 209, then our control law will work for general open-chain robot manipulators. Of course, the performance of a given control law depends heavily on the particular manipulator, and hence the control laws presented here should only be used as a starting point for synthesizing a feedback compensator.

1.2 Computed torque

Consider the following refinement of the open-loop control law presented above: given the current position and velocity of the manipulator, cancel all nonlinearities and apply exactly the torque needed to overcome the inertia of the actuator,

$$\tau = M(\theta)\ddot{\theta}_d + C(\theta,\dot{\theta})\dot{\theta} + N(\theta,\dot{\theta})$$

Substituting this control law into the dynamic equations of the manipulator, we see that

$$M(\theta)\ddot{\theta} = M(\theta)\ddot{\theta}_d$$

and since $M(\theta)$ is uniformly positive definite in θ, we have

$$\ddot{\theta} = \ddot{\theta}_d \qquad (1.21)$$

Hence, if the initial position and velocity of the manipulator matches the desired position and velocity, the manipulator will follow the desired trajectory. As before, this control law will not correct for any initial condition errors which are present.

The tracking properties of the control law can be improved by adding state feedback. The linearity of equation(1.21) suggests the following control law:

$$\tau = \mathrm{M}(\theta)(\ddot{\theta}_d - K_v \dot{e} - K_p e) + C(\theta, \dot{\theta})\dot{\theta} + N(\theta, \dot{\theta}) \qquad (1.22)$$

Where $e = \theta - \theta_d$, and K_v and K_p are constant gain matrices. When substituted into equation (1.11), the error dynamics can be written as:

$$M(\theta)(\ddot{e} + K_v \dot{e} + K_p e) = 0$$

Since $M(\theta)$ is always positive definite, we have

$$\ddot{e} + K_v \dot{e} + K_p e = 0 \qquad (1.23)$$

This is a linear differential equation which governs the error between the actual and desired trajectories. Equation (1.22) is called the computed torque control law.

The computed torque control law consists of two components. We can write equation (1.22) as

$$\tau = \underbrace{M(\theta)\ddot{\theta}_d + C\dot{\theta} + N}_{\tau_{ff}} + \underbrace{M(\theta)(-K_v \dot{e} - K_p e)}_{\tau_{fb}}$$

Then term τ_{ff} is the feedforward component. It provides the amount of torque necessary to drive the system along its nominal path. The term τ_{fb} is the feedback component. It provides correction torques to reduce any errors in the trajectory of the manipulator.

Since the error equation (1.23) is linear, it is easy to choose K_v and K_p so that the overall system is stable and $e \to 0$ exponentially as $t \to \infty$. Moreover, if we choose K_v and K_p as diagonal:

$$K_v = \begin{pmatrix} k_{v11} & 0 & 0 \\ 0 & k_{v22} & 0 \\ 0 & 0 & k_{v33} \end{pmatrix} \qquad K_p = \begin{pmatrix} k_{p11} & 0 & 0 \\ 0 & k_{p22} & 0 \\ 0 & 0 & k_{p33} \end{pmatrix}$$

Then we get independent exponentially stable systems.

Let us proof the stability of the computed torque control law whose K_v and K_p are diagonal:

If $K_p, K_v \in \mathbb{R}^{n \times n}$ are positive definite ,symmetric matrices

Proof： The error dynamics can be written as a first-order linear system :

$$\frac{d}{dt}\begin{bmatrix} e \\ \dot{e} \end{bmatrix} = \begin{pmatrix} 0 & I \\ -K_p & -K_v \end{pmatrix}\begin{bmatrix} e \\ \dot{e} \end{bmatrix}$$

$$A = \begin{pmatrix} 0 & I \\ -K_p & -K_v \end{pmatrix}$$

It suffices to show that each of the eigenvalues of A has negative real part .Let $\lambda \in \mathbb{C}$ be an eigenvalue of A with corresponding eigenvector $\upsilon = (\upsilon_1, \upsilon_2) \in \mathbb{C}^{2n}, \upsilon \neq 0$

Then,

$$\lambda\begin{bmatrix} \upsilon_1 \\ \upsilon_2 \end{bmatrix} = \begin{pmatrix} 0 & I \\ -K_p & -K_v \end{pmatrix}\begin{bmatrix} \upsilon_1 \\ \upsilon_2 \end{bmatrix} = \begin{bmatrix} \upsilon_2 \\ -K_p\upsilon_1 - K_v\upsilon_2 \end{bmatrix}$$

It follows that if $\lambda = 0$ then $\upsilon = 0$,and hence $\lambda = 0$ is not an eigenvalue of A. Further, if $\lambda \neq 0$, then $\upsilon_2 = 0$ implies that $\upsilon_1 = 0$. Thus, $\upsilon_1, \upsilon_2 \neq 0$ and we may assume without loss of generality that $\| \upsilon_1 \| = 1$. Using this, we write

$$\begin{aligned} \lambda^2 &= \upsilon_1^* \lambda^2 \upsilon_1 = \upsilon_1^* \lambda \upsilon_2 \\ &= \upsilon_1^*(-K_p\upsilon_1 - K_v\upsilon_2) = -\upsilon_1^* K_p \upsilon_1 - \lambda \upsilon_1^* K_v \upsilon_1 \end{aligned}$$

Where * denotes complex conjugate transpose. Since $\alpha = \upsilon_1^* K_p \upsilon_1 > 0$ and $\beta = \upsilon_1^* K_v \upsilon_1 > 0$, we have

$$\lambda^2 + \alpha\lambda + \beta = 0 \qquad \alpha, \beta > 0$$

And hence the real part of λ is negative.

但是考虑到机器人的三个臂的耦合，所以一般的情况 K_v 和 K_p 是全阵。即：

(but consider the coupling between the three arms, the general situation is that K_v and K_p are full matrics)

$$K_v = \begin{pmatrix} k_{v11} & k_{v12} & k_{v13} \\ k_{v21} & k_{v22} & k_{v23} \\ k_{v31} & k_{v32} & k_{v33} \end{pmatrix} \qquad K_p = \begin{pmatrix} k_{p11} & k_{p12} & k_{p13} \\ k_{p21} & k_{p22} & k_{p23} \\ k_{p31} & k_{p32} & k_{p33} \end{pmatrix}$$

下面来讨论在全阵情况下的稳定性问题

(now let us discuss the stability when K_v and K_p are full matrix)

考虑下面的特征矩阵

(consider the follow Characteristic Matrix)

$$A = \begin{pmatrix} 0 & I \\ -K_p & -K_v \end{pmatrix}$$

Where

$$K_v = \begin{pmatrix} k_{v11} & k_{v12} & k_{v13} \\ k_{v21} & k_{v22} & k_{v23} \\ k_{v31} & k_{v32} & k_{v33} \end{pmatrix} \qquad K_p = \begin{pmatrix} k_{p11} & k_{p12} & k_{p13} \\ k_{p21} & k_{p22} & k_{p23} \\ k_{p31} & k_{p32} & k_{p33} \end{pmatrix}$$

So

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -k_{p11} & -k_{p12} & -k_{p13} & -k_{v11} & -k_{v12} & -k_{v13} \\ -k_{p21} & -k_{p22} & -k_{p23} & -k_{v21} & -k_{v22} & -k_{v23} \\ -k_{p31} & -k_{p32} & -k_{p33} & -k_{v31} & -k_{v32} & -k_{v33} \end{bmatrix}$$

We get the characteristic equation of A:

$$Z^6 + a_1 Z^5 + a_2 Z^4 + a_3 Z^3 + a_4 Z^2 + a_5 Z + a_6 = 0$$

Where

$a_1 =$ kv33+kv22+kv11

$a_2 =$ -kv23*kv32+kp11+kp22-kv21*kv12+kv11*kv22+kp33+kv11*kv33+kv22*kv33-kv31*kv13

a3=kp11*kv22-kv11*kv23*kv32-kv13*kp31-kv21*kp12-kv31*kv13*kv22-kv12*kp21+kv21*kv13*kv32+kv11*kp33-kv31*kp13+
kp11*kv33+kv11*kv22*kv33-kv21*kv12*kv33+kv11*kp22+kp22*kv33+kv31*kv12*kv23+kv22*kp33-kp23*kv32-kv23*kp32

a4=-kp11*kv23*kv32-kv11*kp23*kv32-kv21*kv12*kp33-kp13*kp31+kv21*kv13*kp32+kv21*kp13*kv32+kp22*kp11+kv23*kv12*kp31
+kv31*kv12*kp23+kv31*kp12*kv23-kv22*kv13*kp31+kp22*kp33-kv31*kv13*kp22-kv31*kp13*kv22-kv12*kp21*kv33-kp12*kp21-kv21*kp12*kv33+kv13*kp21*kv32+kv11*kv22*kp33+kp33*kp11
-kp23*kp32-kv11*kv23*kp32+kv11*kp22*kv33+kp11*kv22*kv33

a5=-kv22*kp13*kp31-kv12*kp21*kp33-kp11*kv23*kp32-kv21*kp12*kp33+kv13*kp21*kp32+kv23*kp12*kp31
+kp23*kv12*kp31+kv21*kp13*kp32+kp11*kp22*kv33-kp22*kv13*kp31+kp13*kp21*kv32+kv31*kp12*kp23
-kp11*kp23*kv32-kp12*kp21*kv33-kv11*kp23*kp32-kv31*kp13*kp22+kp11*kv22*kp33+kv11*kp22*kp33

a6=kp23*kp12*kp31+kp11*kp22*kp33-kp22*kp13*kp31+kp13*kp21*kp32-kp11*kp23*kp32-kp12*kp21*kp33

To determine the stability of a polynomial , one can simply compute the roots of the polynomial.

So using Routh Criterion, then a_i>0,i=1,...,n

If the above condition is satisfied, then we determine the control algorithm are stable.

通过我的数值求解，在全阵的情况下的稳定情况是很少的：
下面是截取的一组特征值，可以看到大部分情况下都是不稳定的，所以在优化目标函数过程中必须要将不稳定的部分去除。

```
evaluex =   -16.5378     5.8292     5.8292    -2.9461    -0.3716    -0.7789
evaluex =   -28.3576    -5.9326    -5.9326     2.8237    -0.7346     0.9206
evaluex =   -28.2471    -2.9038    -2.9038    -0.3336    -0.3336     1.0525
evaluex =   -34.5607    17.1064    -8.1646     0.7414    -0.1984    -1.3434
evaluex =   -24.8227    -9.7191    -0.9422    -0.9422    -0.2106    -0.2106
evaluex =   -28.7259     6.0426    -7.2991     0.4536    -1.3840    -0.4590
evaluex =   -28.1506     1.1907     1.1907    -0.0246    -1.7298    -1.7298
evaluex =   -41.2884   -16.0734    -5.7253    -0.2879    -0.2879     0.1309
evaluex =   -31.8587   -14.4796     4.6232     1.7373     0.0134    -0.4491
evaluex =   -34.6612     9.6515   -11.0541     0.1651    -0.5390    -1.3423
evaluex =   -35.3750   -13.7301     3.8697    -1.9434     0.1045    -0.6924
```

2. A mathematic description to the problem

Consider the following differential equation group:

$$\ddot{e}+K_v\dot{e}+K_pe=0$$

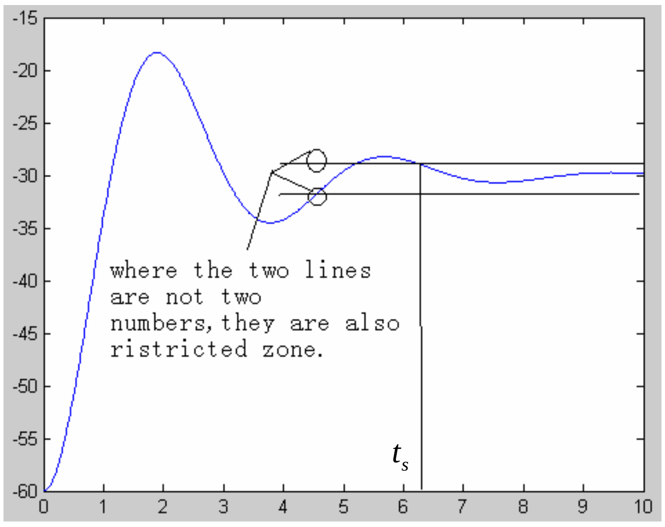
Where

$$K_v=\begin{pmatrix} k_{v11} & k_{v12} & k_{v13} \\ k_{v21} & k_{v22} & k_{v23} \\ k_{v31} & k_{v32} & k_{v33} \end{pmatrix} \qquad K_p=\begin{pmatrix} k_{p11} & k_{p12} & k_{p13} \\ k_{p21} & k_{p22} & k_{p23} \\ k_{p31} & k_{p32} & k_{p33} \end{pmatrix}$$

So the differential equation group

$$\begin{pmatrix} \ddot{e}_1 \\ \ddot{e}_2 \\ \ddot{e}_3 \end{pmatrix}+\begin{pmatrix} k_{v11} & k_{v12} & k_{v13} \\ k_{v21} & k_{v22} & k_{v23} \\ k_{v31} & k_{v32} & k_{v33} \end{pmatrix}\begin{pmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{pmatrix}+\begin{pmatrix} k_{p11} & k_{p12} & k_{p13} \\ k_{p21} & k_{p22} & k_{p23} \\ k_{p31} & k_{p32} & k_{p33} \end{pmatrix}\begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix}=0$$

So if we can solve e, then we can get the settle time of the system t_s.



由于这是三个互相耦合的微分方程组合的微分方程组，所以寻求其解析解很难，于是采用数值解来解决这个问题。

所以优化的目标函数将是调整时间 t_s，约束条件包括：（1）超调约束（2）力矩约束（3）不稳定参数约束

于是这个优化问题将表述如下：

Object function min t_s

s.t.

(1) overshoot constraint

(2) torque constraint

(3) unsteady parameters constraint

optimization algorithm：Differential Evolution (DE)

3. A introduction to DE

Differential Evolution (DE) is a relatively new EA proposed by Price and Storn. The algorithm is based on the use of a special crossover-mutation operator, based on the linear combination of three different individuals and one subject-to-replacement parent. The selection process is performed via deterministic tournament selection between the parent and the child created by it. However, as any other EA, DE lacks a mechanism to deal with constrained search spaces.

Differential Evolution (DE) has to figure out the first generation. The way to create the first generation is using the random function. Assume that the *i*th pop is $X_i = \left(x_{i,1}, x_{i,2}, \cdots, x_{i,n}\right)$, n is the number of the variables. The first generation is $S = \left\{X_1, X_2, \cdots, X_{Np}\right\}$, normally this is generated in the following way:

$$x_{i,j} = x_{i,j\min} + rand()(x_{i,j\max} - x_{i,j\min})$$

3.1 Mutation

to enlarge the range of the variables. In DE, mutation is introduced. assume that the mutation vector is $V_i^{k+1} = \left(v_{i,1}^{k+1}, v_{i,2}^{k+1}, \cdots, v_{i,n}^{k+1}\right)$, generated in the following way:

$$V_{i,j}^{k+1} = x_{r1,j}^k + F \cdot \left(x_{r2,j}^k - x_{r3,j}^k\right)$$

3.2 Crossover

with the same proper above, in DE Crossover is also introduced. assume that the proper vector is $U_i^{k+1} = \left(u_{i,1}^{k+1}, u_{i,2}^{k+1}, \cdots, u_{i,n}^{k+1}\right)$, generated by both v and u in the following method:

$$u_{i,j}^{k+1} = \begin{cases} v_{i,j}^{k+1}, \eta_j \le C_R \mathrm{or} j = q_j \\ x_{i,j}^k, \mathrm{else} \end{cases}$$

In the equation, q_i is a random number in $\left[1, n\right]$, to make sure that in every crossover at least one component of the vectors has to be changed ;$\eta_j \in \left[0,1\right]$ is a control parameter choosen for the ith component of the vectors, $C_R \in \left[0,1\right]$ has to be determined at the beginning.

4. Research on the distinction between the full matrix and diagonal matrix

一般在处理 K_v and K_p 问题时，为了得到互相解耦的微分方程以减少复杂度，就将 K_v and

K_p 处理为对角阵，但是我们知道一个串联机器人的三个臂是互相耦合的，每个臂的运动都对其他臂的运动产生影响，所以最合理的处理方法应该将 K_v and K_p 作为两个全阵来考虑。
可以想象的是对角阵是全阵的一种特殊情况，那么对角阵的情况下优化的结果将不是全局的最优结果。所以我们要得到全局的最优解，我们必须考虑一般的情况。

下面就对两种情况做比较
考虑超调 5%
力矩小于 8
使用 DE 算法

	Best solution(Diag)	Best solution(Full)	(diag)	(full)
1	2.3833 3.3421	[8.38415344 9.66491588 2.82934740 8.48853407 4.03860291 0.46114840 8.46650245 2.30018339 9.91270916 9.75393670 6.20485862 1.71557669 4.86861018 8.46452498 1.19715496 2.47733172 9.85699738 9.88032570]	4.4100	1.90000000
2	1.5217 1.9587	[1.72673409 1.44809618 1.19204083 1.71647225 1.91543135 1.20000000 1.63146158 1.10856461 1.12551764 1.59470321 1.28321770 1.26734351 2.52512542 1.55333279 1.09185410 1.51563279 1.20000000 1.86734762]	3.7310	1.40000000
3	1.3129 1.0761	[2.65988912 17.19640978 4.69860201 1.84740403 22.05042745 8.26414227 1.63362331 5.15625052 0.66985597 5.75128478 2.16823081 2.56862454 5.20386660 2.63099331 7.46930910 6.66251992 9.85351572 7.14483642]	3.4900	0.50000000
4	1.1431 1.0448	[1.06984902 1.72165141 1.62312409 9.31282423 2.64859416 6.83273190 6.71091474 5.91628608 7.81480318 2.74264273 9.97740450 1.92199301 3.13046826 7.45543587 3.20369254 5.61417119 4.85298320 4.62482400]	3.1510	0.40000000

可以看出对角阵的情况下不是全局的最优解。所以我们有必要考虑完整的情况。