1. Trajectory Planning for Automatic machines and Robots
   1. polynomial function

大多数简单情况，可定义初始时刻 和终止时刻 ,以及位置、速度和加速度条件，确定运动。从数学的角度，问题是找到函数



满足给定条件。该问题可考虑找到一个多项式



这里确定 个系数 使得初始和终止约束条件都满足。多项式的自由度 取决于约束条件的个数和运动期望的“平滑度”。多项式的自由度通常是奇数的，例如3,5,7等。

除了轨迹的初始和结束条件，同时可以指定一般时刻 关于时间的微分（速度，加速度，加加速度，...）。这些条件可以指定为 关于时间的 阶微分



矩阵形式可以表述为



其中 已知的 的矩阵， 为给定的 需要满足的条件。 是需要计算的未知参数构成的向量。方程的解为



* 1. Linear trajectory

最简单的点 到 的运动，定义为



初始时刻和结束时刻分别为 ，则满足初始条件和结束条件为



即



这里 是运动时间。因此解得



这里 是位移。在时间间隔内速度恒定，即



* 1. Parabolic trajectory

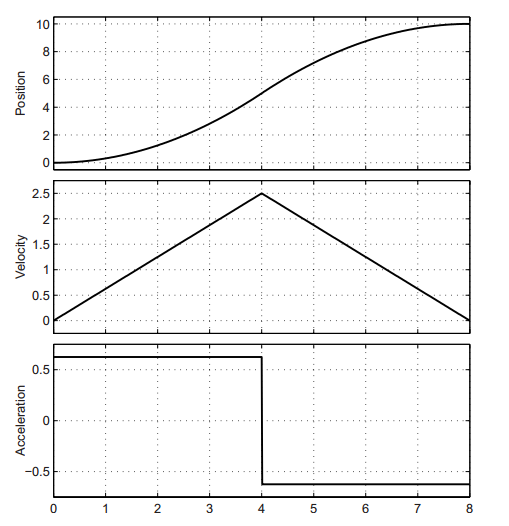


图 1

关于拐点对称的抛物线轨迹

该轨迹被广泛称为重力轨迹或加速度恒定轨迹。如下图所示，假设拐点时刻为 。不妨首先考虑关于拐点对称的情形。运动轨迹可定义为 ， 。注意到， 。

第一段，加速段。轨迹定义为



参数由位置和速度条件计算



有上述方程可求得



因此，对 ，轨迹定义为



拐点速度为



第二段，减速段。同理，根据位置和速度条件，可求得对于 的轨迹表达式



* + 1. Trajectory with asymmetric constant acceleration

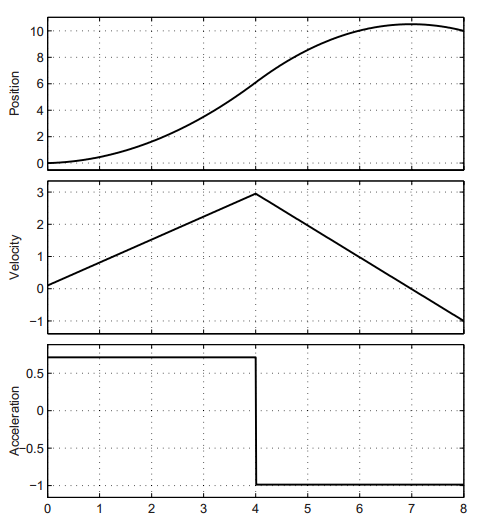


图 2

具有对称恒定加速度的抛物线轨迹

考虑更一般的情况，拐点不一定在时间中点时刻。这时轨迹可定义为两个多项式





根据位置和速度在起始和结束时刻的四个条件，以及位置和速度两个连续性条件可得



再定义 ，求得多项式系数为



速度和加速度可由位移轨迹多项式求导得到。

* 1. Cubic trajectory

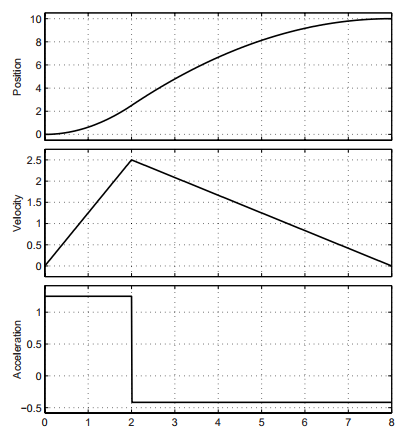


图 3

三次多项式轨迹

一旦指定了 时刻的位置和速度的值 ，那么有四个条件需要满足，所以必须使用三次多项式



根据给定条件，可求得四个系数分别为



通过使用这个结果，可很容易计算出n个点的序列的速度连续的轨迹。整个运动可分为n-1段，每段连接了时刻 和 对应的点 和 ，并且分别具有起始速度和终点速度 。上式可以用来定义这些曲线段的4（n-1）个参数 。

Example2.6:（a）

（b）

由（a）和（b）的条件确定的轨迹分别如下图所示

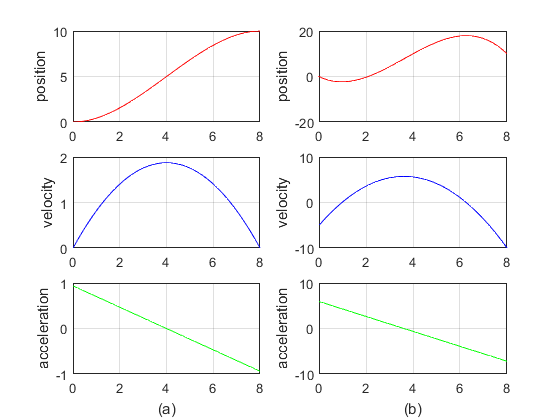


图 4

Example 2.6

matlab仿真代码参见examplesCode文件夹下的example2\_6.m文件

%example2.6

clc

clear

%轨迹定义条件

%时间

t0=0;

t1=8;

%位置和速度（a）

q0=0;

q1=10;

v0=0;

v1=0;

%利用公式（1-22）求系数

h=q1-q0;

T=t1-t0;

a0=q0;

a1=v0;

a2=(3\*h-(2\*v0+v1)\*T)/(T\*T);

a3=(-2\*h+(v0+v1)\*T)/(T\*T\*T);

%轨迹生成

t=t0:0.1:t1;

%位置

q=a0+a1\*power((t-t0),1)+a2\*power((t-t0),2)+a3\*power((t-t0),3);

%速度

v=a1+2\*a2\*power((t-t0),1)+3\*a3\*power((t-t0),2);

%加速度

acc=2\*a2+6\*a3\*power((t-t0),1);

%绘图

subplot(3,2,1)

plot(t,q,'r');

ylabel('position')

grid on

subplot(3,2,3)

plot(t,v,'b');

ylabel('velocity')

grid on

subplot(3,2,5)

plot(t,acc,'g');

xlabel('(a)');

ylabel('acceleration')

grid on

%时间

t0=0;

t1=8;

%位置和速度（b）

q0=0;

q1=10;

v0=-5;

v1=-10;

%利用公式（1-22）求系数

h=q1-q0;

T=t1-t0;

a0=q0;

a1=v0;

a2=(3\*h-(2\*v0+v1)\*T)/(T\*T);

a3=(-2\*h+(v0+v1)\*T)/(T\*T\*T);

%轨迹生成

t=t0:0.1:t1;

%位置

q=a0+a1\*power((t-t0),1)+a2\*power((t-t0),2)+a3\*power((t-t0),3);

%速度

v=a1+2\*a2\*power((t-t0),1)+3\*a3\*power((t-t0),2);

%加速度

acc=2\*a2+6\*a3\*power((t-t0),1);

%绘图

subplot(3,2,2)

plot(t,q,'r');

ylabel('position')

grid on

subplot(3,2,4)

plot(t,v,'b');

ylabel('velocity')

grid on

subplot(3,2,6)

plot(t,acc,'g');

xlabel('(b)');

ylabel('acceleration')

grid on

Example 2.7 :由多个点定义的运动轨迹



利用式确定系数，然后求导得到速度和加速的表达式。example 2.7确定的轨迹如下图所示。

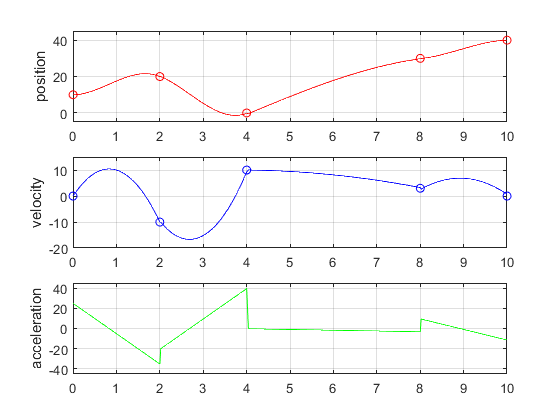


图 5

给定多个点确定的轨迹

matlab仿真代码参见examplesCode文件夹下的example2\_7.m文件

%example2.7

clc

clear

%轨迹定义条件

t\_array=[0,2,4,8,10];

q\_array=[10,20,0,30,40];

v\_array=[0,-10,10,3,0];

%计算轨迹

%初始位置

t=t\_array(1);

q=q\_array(1);

v=v\_array(1);

%计算各段轨迹

for k=1:length(t\_array)-1

%计算各段多项式的系数

h(k)=q\_array(k+1)-q\_array(k);

T(k)=t\_array(k+1)-t\_array(k);

a0(k)=q\_array(k);

a1(k)=v\_array(k);

a2(k)=(3\*h(k)-(2\*v\_array(k)+v\_array(k+1))\*T(k))/(T(k)\*T(k));

a3(k)=(-2\*h(k)+(v\_array(k)+v\_array(k+1))\*T(k))/(T(k)\*T(k)\*T(k));

%生成各段轨迹密化的数据点

%局部时间坐标

tau=t\_array(k):T(k)/100:t\_array(k+1);

%全局时间坐标，由局部时间坐标组成

t=[t,tau(2:end)];

%局部位置坐标

qk=a0(k)+a1(k)\*power(tau-tau(k),1)+a2(k)\*power(tau-tau(k),2)+a3(k)\*power(tau-tau(k),3);

%全局位置坐标

q=[q,qk(2:end)];

%速度

vk=a1(k)+2\*a2(k)\*power(tau-tau(k),1)+3\*a3(k)\*power(tau-tau(k),2);

v=[v,vk(2:end)];

%加速度

acck=2\*a2(k)+6\*a3(k)\*power(tau-tau(k),1);

if(k==1)

acc=2\*a2(k);

end

acc=[acc,acck(2:end)];

end

%绘图

subplot(3,1,1);

plot(t\_array,q\_array,'or');

hold on;

plot(t,q,'r');

axis([0,10,-5,45]);

ylabel('position')

grid on;

subplot(3,1,2);

plot(t\_array,v\_array,'ob');

hold on;

plot(t,v,'b');

axis([0,10,-20,15]);

ylabel('velocity')

grid on;

subplot(3,1,3);

plot(t,acc,'g');

axis([0,10,-45,45]);

ylabel('acceleration')

grid on;

通过一系列点 定义轨迹时，中间点的速度并不总是满足指定的速度。这时中间点可根据直观判断确定合适的速度值，例如



这里 ， 是函数的符号。

Example2.8:把Example2.7的给定点按照确定中间点的速度值，那么确定的轨迹如下图(虚线)所示

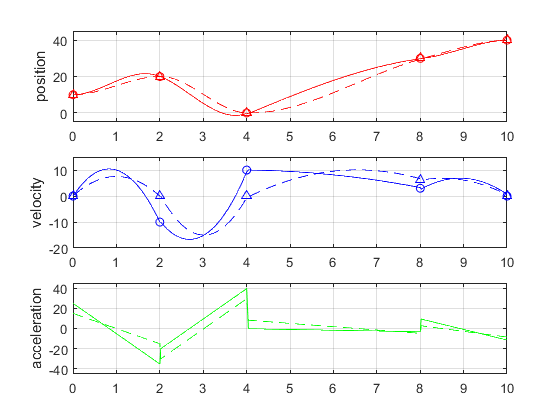


图 6

中间点的速度不一定等于指定速度

matlab仿真代码参见examplesCode文件夹下的example2\_8.m文件

%example2.8

clc

clear

%轨迹定义条件

t\_array=[0,2,4,8,10];

q\_array=[10,20,0,30,40];

v\_array=[0,-10,10,3,0];

%计算轨迹

%初始位置

t=t\_array(1);

q=q\_array(1);

v=v\_array(1);

v\_array2=v\_array;

for k=1:length(t\_array)-1

%按照式（1-23）式确定中间点的速度值

if(k>1)

dk1=(q\_array(k)-q\_array(k-1))/(t\_array(k)-t\_array(k-1));

dk2=(q\_array(k+1)-q\_array(k))/(t\_array(k+1)-t\_array(k));

if((dk2>=0 && dk1>=0) || (dk2<=0 && dk1<=0))

v\_array2(k)=1.0/2.0\*(dk1+dk2);

else

v\_array2(k)=0;

end

end

end

%计算各段轨迹

for k=1:length(t\_array)-1

%计算各段多项式的系数

h(k)=q\_array(k+1)-q\_array(k);

T(k)=t\_array(k+1)-t\_array(k);

a0(k)=q\_array(k);

a1(k)=v\_array2(k);

a2(k)=(3\*h(k)-(2\*v\_array2(k)+v\_array2(k+1))\*T(k))/(T(k)\*T(k));

a3(k)=(-2\*h(k)+(v\_array2(k)+v\_array2(k+1))\*T(k))/(T(k)\*T(k)\*T(k));

%生成各段轨迹密化的数据点

%局部时间坐标

tau=t\_array(k):T(k)/100:t\_array(k+1);

%全局时间坐标，由局部时间坐标组成

t=[t,tau(2:end)];

%局部位置坐标

qk=a0(k)+a1(k)\*power(tau-tau(k),1)+a2(k)\*power(tau-tau(k),2)+a3(k)\*power(tau-tau(k),3);

%全局位置坐标

q=[q,qk(2:end)];

%速度

vk=a1(k)+2\*a2(k)\*power(tau-tau(k),1)+3\*a3(k)\*power(tau-tau(k),2);

v=[v,vk(2:end)];

%加速度

acck=2\*a2(k)+6\*a3(k)\*power(tau-tau(k),1);

if(k==1)

acc=2\*a2(k);

end

acc=[acc,acck(2:end)];

end

%绘图

subplot(3,1,1);

h2=plot(t,q,'--r');

legend(h2,'第二种方式')

hold on;

plot(t\_array,q\_array,'^r');

axis([0,10,-5,45]);

ylabel('position')

grid on;

subplot(3,1,2);

plot(t\_array,v\_array2,'^b');

hold on;

plot(t,v,'--b');

axis([0,10,-20,15]);

ylabel('velocity')

grid on;

subplot(3,1,3);

plot(t,acc,'--g');

axis([0,10,-45,45]);

ylabel('acceleration')

grid on;