上海交通大学

2020-2021 学年第 1 学期

《机器人学》实验报告

姓名：　　沈臻

学号：　　518021910149

班级：　　F1803202

学院：　　电子信息与电气工程学院

专业：　　自动化

任课教师：　　王贺升教授

日期：2020 年 11 月 29 日

目录

上海交通大学 实验报告

一、实验 1

1.问题分析

给定 Puma560 机械臂构型，使用正运动学在关节空间随机采样得到机械臂的工作空间，并画出-90 < q < 90 范围内工作空间。其中：a2 = 100; d3 = 20; a3 = 10; d4 = 100。

2.实验思路

将问题分为三个子任务：建立 Puma560 机械臂构型、在关节空间随机采样、画出-90 < q < 90 范围内工作空间。

在建立 Puma560 机械臂构型时，使用标准型建立机械臂构型。得到机械臂的 D-H 参数表如表 1 所示。使用该 D-H 参数表建立标准型 Puma560 机械臂构型。

表 1 标准型 Puma560 机械臂 D-H 参数表

j	theta	d	a	alpha	offset
1	q1	0	0	-π/2	0
2	q2	0	100	0	0
3	q3	20	10	-π/2	0
4	q4	100	0	π/2	0
5	q5	0	0	-π/2	0
6	q6	0	0	0	0

在关节空间随机采样时，使用蒙特卡罗法，通过大量随机采样寻找关节空间。使用随机数 rand( )随机生成 10000 组关节空间值赋予每一个关节变量，使用正运动学计算得到末端执行器的位置。

在画出-90 < q < 90 范围内工作空间时，使用蒙特卡罗法对关节空间随机采样。采用公式 q = q_min + (q_max - q_min) .* rand(1 , 6) 计算随机的关节变量组。其中 q_max = π/2，q_min = -π/2。使用正运动学变换 fkine(q)得到齐次变换矩阵。使用 transl(T)函数得到末端

执行器的位置。由 10000 点末端执行器位置-90 < q < 90 范围内得到大致的工作空间。

3.结果分析

运行 problem1.m 文件。建立的 Puma560 机械臂构型如图 1 所示。等待一段时间后可以得到绘制的工作空间云图，得到的工作空间如图 2、图 3 所示。

可以看到 Puma560 机械臂是一个 6 自由度的机械臂，有 6 个转动关节。-90 < q < 90 范围的工作空间是一个不规则的立体形状，这是因为对每个关节角加以限制得到的，与未加限制时相比其工作空间应该少了一部分。

相关代码见附录 1 所示。

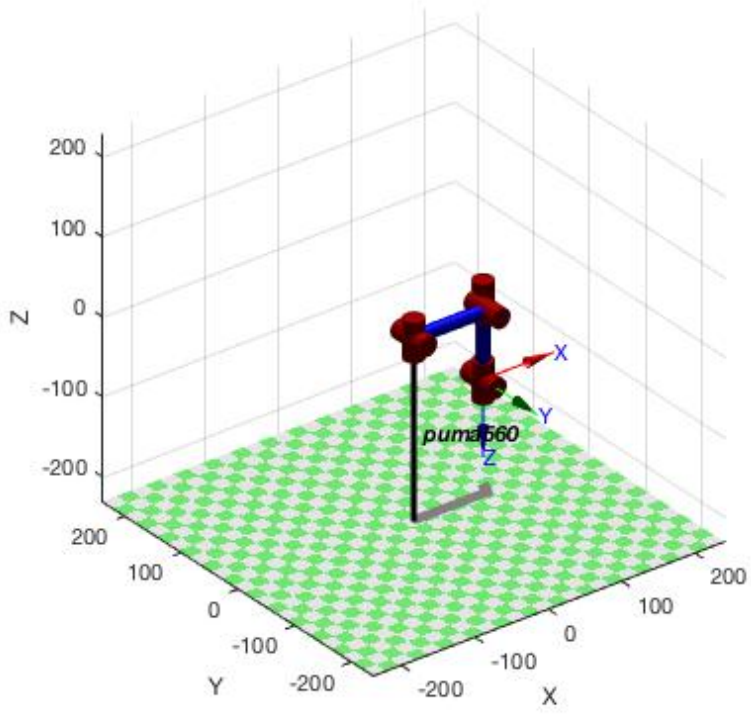

图 1 Puma560 机械臂构型

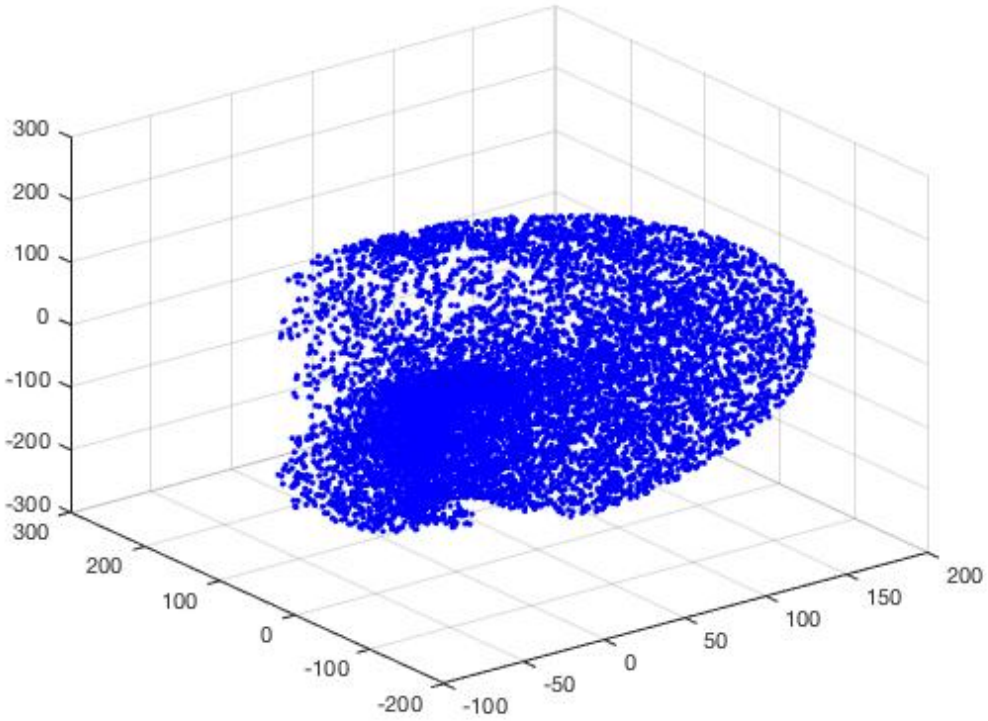

图 2 -90 < q < 90 范围的工作空间主视图

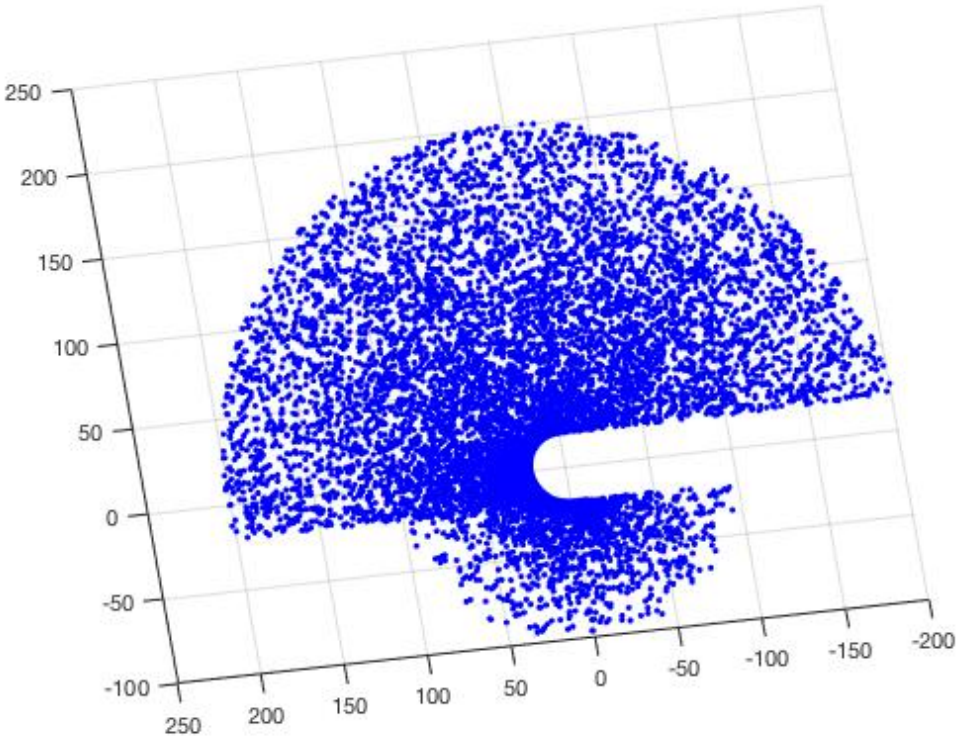

图 3 -90 < q < 90 范围的工作空间俯视图

上海交通大学 实验报告

二、实验 2

1.问题分析

工作空间的 origin=[100,0,50]（中心）位置处有一块 size=[200,30,150]的障碍物。使用逆运动学规划一条从起点 pini = [100 100 10]'到终点 pend = [100 -100 10]'的路径。画出机械臂从起点到终点的工作空间路径及关节角度变化曲线。

2.实验思路

将问题分为四个子任务：建立 Puma560 机械臂构型、选取路径点得到相应的关节空间矩阵、三次插值法求得较为平滑的路径及相应的关节空间值、轨迹和关节角度变化绘制。

在建立 Puma560 机械臂构型时，使用标准型建立机械臂构型，D-H 参数值与实验 1 相同。使用官网提供的函数 plotcube.m 绘制表示障碍物的长方体。

在选取路径点得到相应的关节空间矩阵时，通过起点与终点的轨迹规划，大致确定机械臂的运动轨迹。但如果直接使用起点与终点的轨迹规划，则会撞到障碍物。因此重新选取一个中间点(80 , 0 , 10)，使机械臂可以绕过障碍物。该点与(100 , 100 , 10)和(100 , -100 , 10)一起构成所规划路径的三个关键点。通过函数 ikine6s(T)可以求出 Puma560 机械臂在每个点处的关节角度值，共得到三个关键点的关节角度值。

在求所规划的路径及相应的关节角度值时，使用三次差值法对已求得的三个关键点的关节角度值进行插值，求得较多个离散点。用这些离散点近似为一条连续的平滑曲线，即为机械臂从起点到终点的工作空间路径。

绘制机械臂的工作空间路径，对插值后获得的离散点对应的关节角度值应用正运动学，使用函数 fkine(q)和 transl(T)求得其在工作空间中的位置，可以近似为一条平滑的曲线。绘制关节角度变化，使用快速绘图函数 qplot( )可以直接绘制 6 个关节角度的曲线

3.结果分析

机械臂从起点到终点的工作空间路径的不同视图如图 4(a)(b)(c)所示。机械臂各关节角度的变化曲线如图 5 所示。

由从起点到终点的工作空间路径可见机械臂成功地避开了显示为红色长方体的障碍。由各关节角度的变化曲线可以看到三次插值法成功地将关节角度的变化近似为一条平滑的曲线，实现了平滑的路径规划。

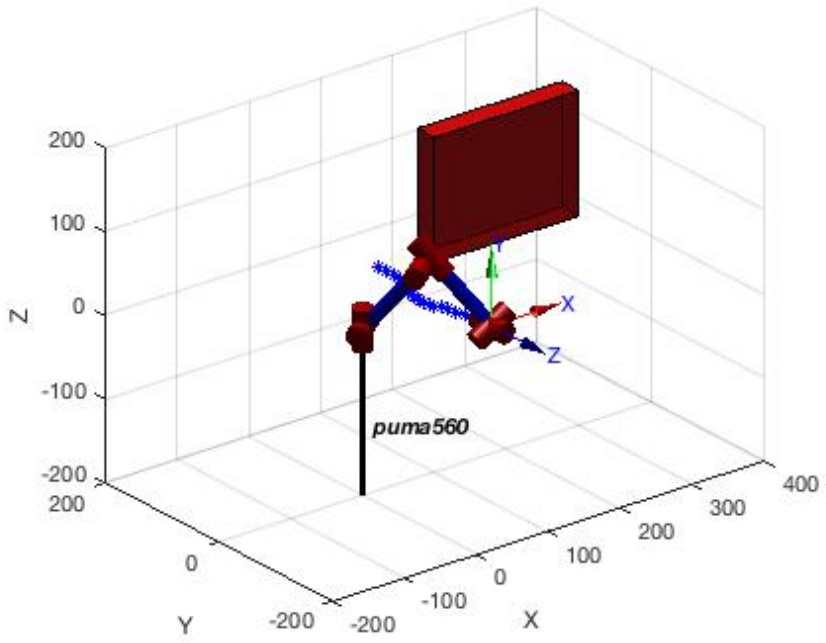

图 4(a) 机械臂从起点到终点的工作空间路径主视图

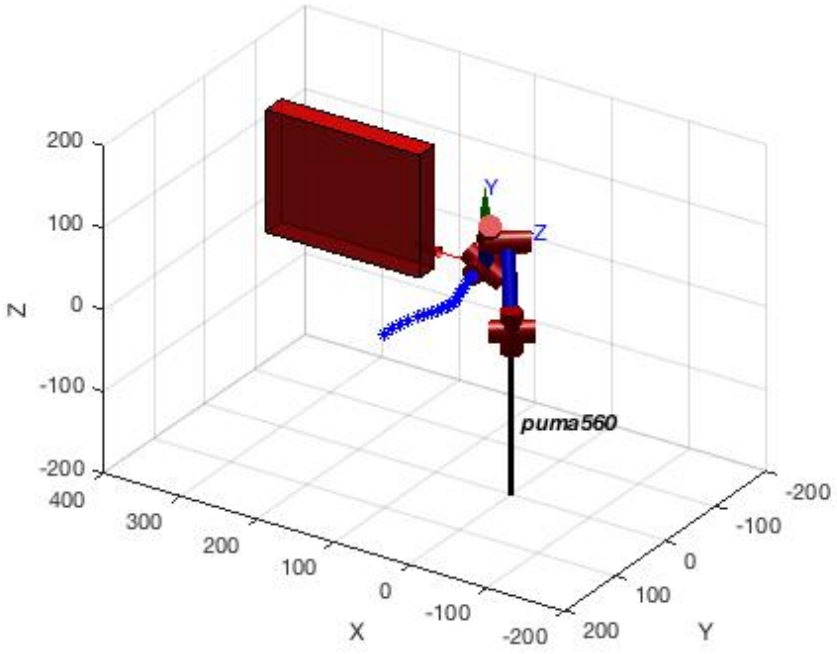

图 4(b) 机械臂从起点到终点的工作空间路径侧视图

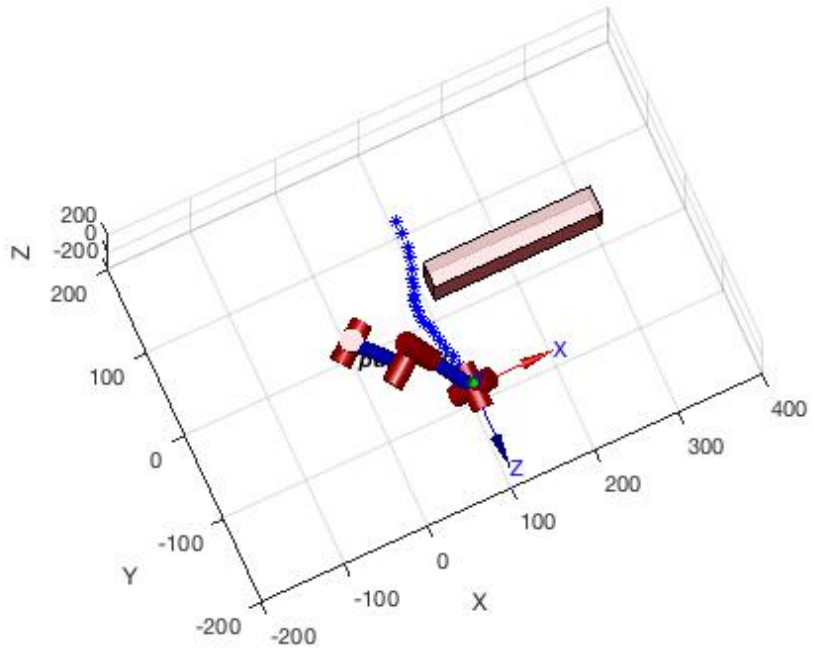

图 4(c) 机械臂从起点到终点的工作空间路径俯视图

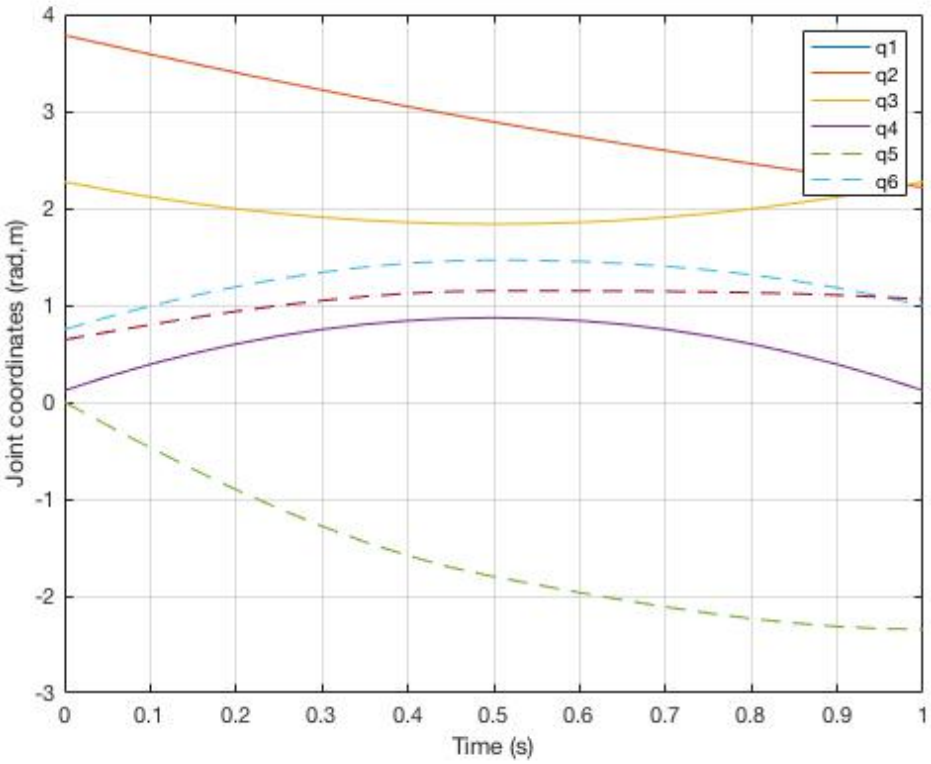
q1
q2
q3
q4
q5
q6
Joint coordinates (rad,m)
Time (s)
4
3
2
1
0
-1
-2
-3
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

图 5 各关节角度变化

三、附录

1.实验 1 代码

```
clc;
clear;

%% puma560

%%标准 D-H 模型

%      theta    d          a         alpha     offset
SL1=Link([0      0          0         -pi/2       0     ],'standard');
SL2=Link([0      0          100       0           0     ],'standard');
SL3=Link([0      20         10        -pi/2       0     ],'standard');
SL4=Link([0      100        0         pi/2        0     ],'standard');
SL5=Link([0      0          0         -pi/2       0     ],'standard');
SL6=Link([0      0          0         0           0     ],'standard');


%定义关节范围

SL1.qlim =[-pi/2, pi/2];
SL2.qlim =[-pi/2, pi/2];
SL3.qlim =[-pi/2, pi/2];
SL4.qlim =[-pi/2, pi/2];
SL5.qlim =[-pi/2, pi/2];
SL6.qlim =[-pi/2, pi/2];

q1_min = -pi/2; q1_max = pi/2;
q2_min = -pi/2; q2_max = pi/2;
q3_min = -pi/2; q3_max = pi/2;
q4_min = -pi/2; q4_max = pi/2;
q5_min = -pi/2; q5_max = pi/2;
q6_min = -pi/2; q6_max = pi/2;

p560=SerialLink([SL1 SL2 SL3 SL4 SL5 SL6],'name','puma560');


%可视化 puma 机械臂
```

```
p560
figure(1)
qz =[0 , 0 , 0 , 0 , 0 , 0];
p560.plot(qz);


%用蒙特卡罗法画出工作空间

q_min = [q1_min q2_min q3_min q4_min q5_min q6_min];
q_max = [q1_max q2_max q3_max q4_max q5_max q6_max];

for i = 0:1:10000
    q = q_min + (q_max - q_min).*rand(1,6);
    T = p560.fkine(q);
    p = transl(T);
    plot3(p(1),p(2),p(3),'b.');
    grid on;
    hold on;
end
```

2.实验 2 代码

(1)主函数:

```
clc;
clear;

%% puma560
%%标准 D-H 模型
%      theta    d          a         alpha     offset
SL1=Link([0      0          0         -pi/2       0     ],'standard');
SL2=Link([0      0          100       0           0     ],'standard');
SL3=Link([0      20         10        -pi/2       0     ],'standard');
SL4=Link([0      100        0         pi/2        0     ],'standard');
SL5=Link([0      0          0         -pi/2       0     ],'standard');
SL6=Link([0      0          0         0           0     ],'standard');

p560=SerialLink([SL1 SL2 SL3 SL4 SL5 SL6],'name','puma560');

%可视化 puma 机械臂
p560
figure(1)
qz=[0,0,0,0,0,0];
p560.plot(qz);
grid on;
```

```
hold on;


%设置障碍
figure(1)
plotcube([200,30,150],[100,0,50],.8,[1,0,0]);
xlim([-200 400]);
ylim([-200 200]);
zlim([-200 200]);
grid on;
hold on;

%选取起点、中间点和终点，取得关节空间矩阵
T1=transl(100, 100, 10) * trotx(pi);
T2=transl(80, 0, 10) * trotx(pi/2);
T3=transl(100, -100, 10) * trotx(pi/2);
q1=p560.ikine6s(T1);
q2=p560.ikine6s(T2);
q3=p560.ikine6s(T3);
Q=[q1;q2;q3]

%三次插值求路径中 21 点的关节空间值
q_c=[];
for i=1:6
    ty=Q(:,i);
    ty=ty';
    tx=1:3;
    tx1=1:0.1:3;
    q_c(i,:)=interp1(tx,ty,tx1,'PCHIP');
end
q_c=q_c';
q=[q_c];
figure(1)
p560.plot(q);

%画轨迹
figure(1)
JTA=transl(p560.fkine(q));
t=[0:0.05:1]';
plot3(JTA(:,1).',JTA(:,2).',JTA(:,3).','b*');
hold on;

figure(2)
plot(t, q(:,2));
```

```
qplot(t,q);
```

(2)绘制长方体的函数 plotcube.m：

```
function plotcube(varargin)
% PLOTCUBE - Display a 3D-cube in the current axes
%
%   PLOTCUBE(EDGES,ORIGIN,ALPHA,COLOR) displays a 3D-cube in the
current axes
%   with the following properties:
%   * EDGES : 3-elements vector that defines the length of cube edges
%   * ORIGIN: 3-elements vector that defines the start point of the
cube
%   * ALPHA : scalar that defines the transparency of the cube faces
(from 0
%           to 1)
%   * COLOR : 3-elements vector that defines the faces color of the
cube
%
% Example:
%   >> plotcube([5 5 5],[ 2  2  2],.8,[1 0 0]);
%   >> plotcube([5 5 5],[10 10 10],.8,[0 1 0]);
%   >> plotcube([5 5 5],[20 20 20],.8,[0 0 1]);

% Default input arguments
inArgs = { ...
  [10 56 100] , ... % Default edge sizes (x,y and z)
  [10 10  10] , ... % Default coordinates of the origin point of the
cube
  .7          , ... % Default alpha value for the cube's faces
  [1 0 0]       ... % Default Color for the cube
  };

% Replace default input arguments by input values
inArgs(1:nargin) = varargin;

% Create all variables
[edges,origin,alpha,clr] = deal(inArgs{:});
XYZ = { ...
  [0 0 0 0]  [0 0 1 1]  [0 1 1 0] ; ...
  [1 1 1 1]  [0 0 1 1]  [0 1 1 0] ; ...
  [0 1 1 0]  [0 0 0 0]  [0 0 1 1] ; ...
  [0 1 1 0]  [1 1 1 1]  [0 0 1 1] ; ...
  [0 1 1 0]  [0 0 1 1]  [0 0 0 0] ; ...
  [0 1 1 0]  [0 0 1 1]  [1 1 1 1]   ...
```

```
  };

XYZ = mat2cell(...
  cellfun( @(x,y,z) x*y+z , ...
    XYZ , ...
    repmat(mat2cell(edges,1,[1 1 1]),6,1) , ...
    repmat(mat2cell(origin,1,[1 1 1]),6,1) , ...
    'UniformOutput',false), ...
  6,[1 1 1]);


cellfun(@patch,XYZ{1},XYZ{2},XYZ{3},...
  repmat({clr},6,1),...
  repmat({'FaceAlpha'},6,1),...
  repmat({alpha},6,1)...
  );

view(3);
```