用SVD进行人脸数据分析

1 在matlab中加载并画出数据集

1.1 任务

1. 加载数据
2. 画出36张人脸集合图

1.2 理解

所给数据X为1*38的元胞，每个元胞为32256*64的矩阵，矩阵的每一列为一张人脸矩阵向量化后的结果，一个元胞包含一个人的64张脸。第一个任务首先需要加载数据集X，再提取出前36个人的第一张人脸，画在一幅图上。

1.3 实现

1. 代码实现思路
   - 加载数据
   - 提取前36个元胞，将36个矩阵的第一列转换为192*168的矩阵
   - 将36个矩阵拼接成目标矩阵
2. 代码如下：

```
clear all;
close all;

%加载数据
load('YaleBExtend.mat');

%得到6*6的人脸矩阵face_all
face1=[];
for i=1:6
    face1=[face1,reshape(X{1,i}(:,1),192,168);];
end
```

```
face2=[];
for i=7:12
    face2=[face2,reshape(X{1,i}(:,1),192,168);];
end
face3=[];
for i=13:18
    face3=[face3,reshape(X{1,i}(:,1),192,168);];
end
face4=[];
for i=19:24
    face4=[face4,reshape(X{1,i}(:,1),192,168);];
end
face5=[];
for i=25:30
    face5=[face5,reshape(X{1,i}(:,1),192,168);];
end
face6=[];
for i=31:36
    face6=[face6,reshape(X{1,i}(:,1),192,168);];
end
face_all=[face1;face2;face3;face4;face5;face6];

%作图
figure(1);imagesc(face_all), colormap gray;
```

1.4 结果

---

2 用SVD处理数据，计算特征脸

2.1 任务

1. 画出平均脸
2. 提取并可视化前四张特征脸

2.2 理解

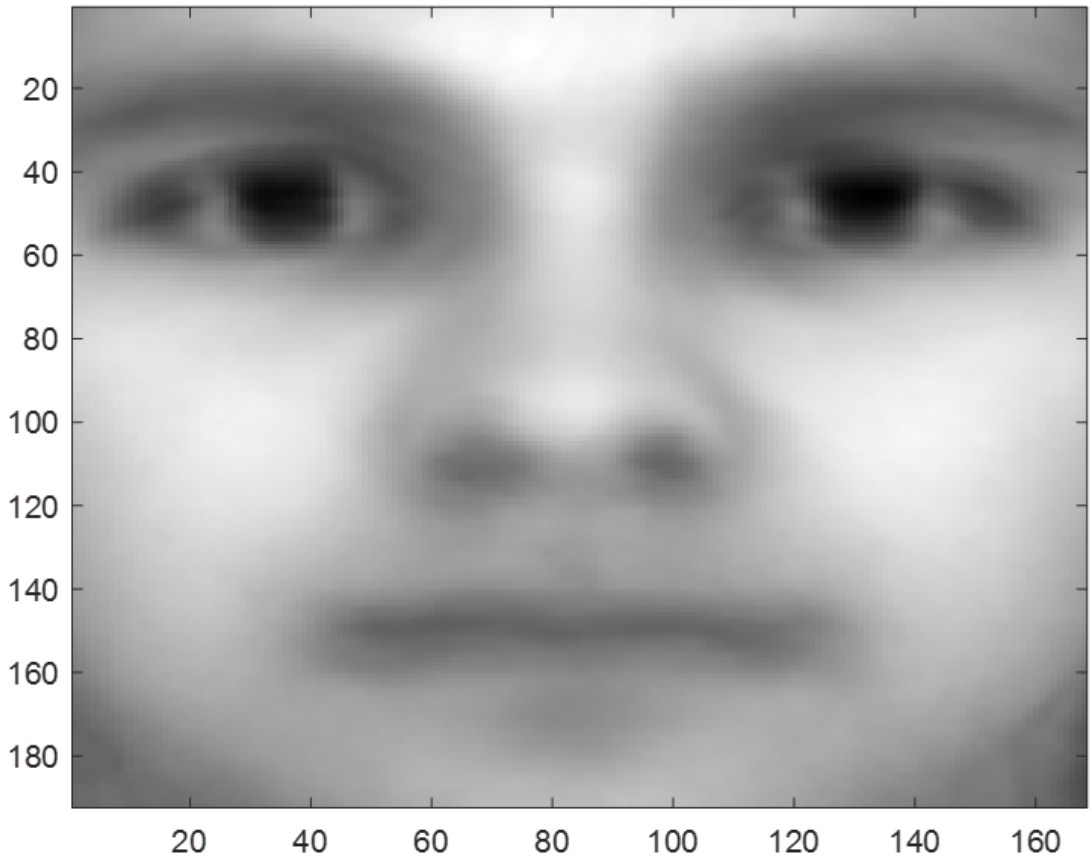
要提取特征脸需要对X做SVD，在做SVD之前首先要零均值化，防止过拟合和方便数据分析。对人脸数据矩阵求每行求平均得到平均向量，可画出平均脸；提取SVD后的前四个左奇异向量，可画出前四张特征脸。

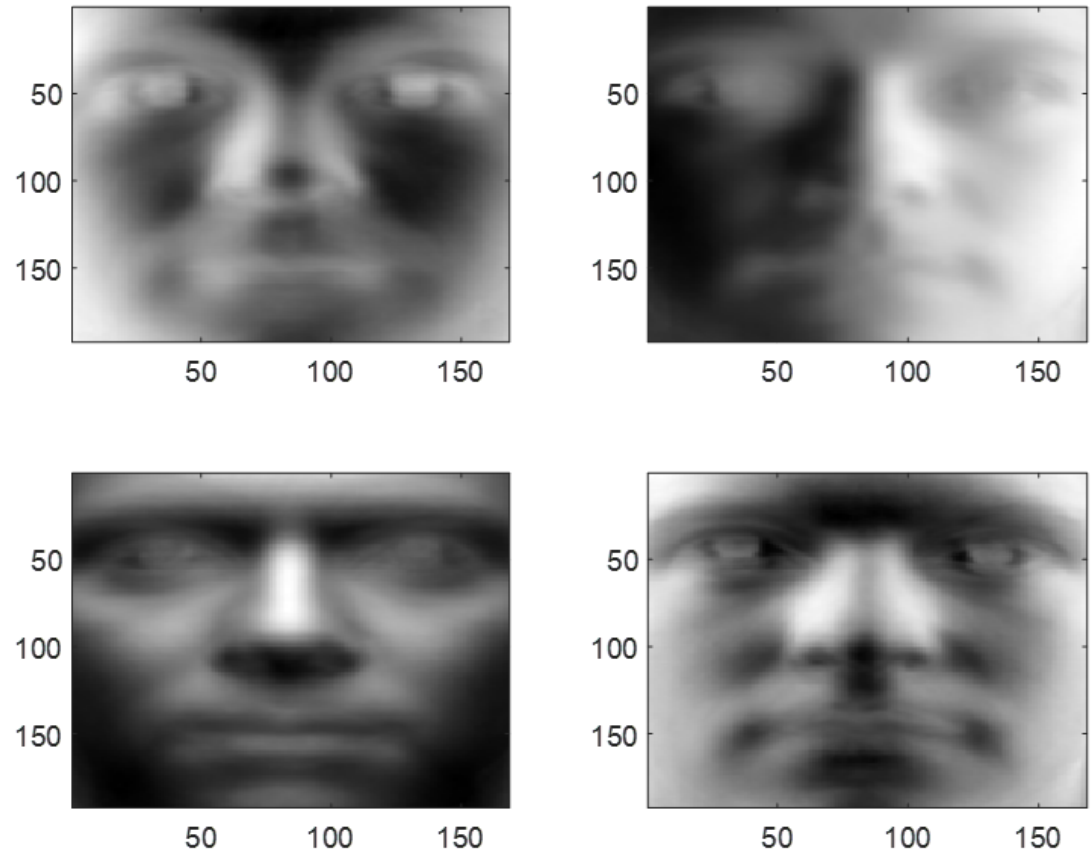
2.3 实现

1. 代码实现思路
    - 将元胞X整合为32256*2286的数据矩阵data
    - 求得均值向量meanface
    - 取均值做SVD
    - 作图
2. 代码如下：

```
%任务二

%得到前36个人的人脸数据矩阵
data=[];
for k=1:36
    data=[data,X{1,k}];
end

%得到平均脸maenface向量
meanface=mean(data,2);
figure(2);imagesc(reshape(meanface,192,168)), colormap gray;

%去均值，再做SVD
data_sub=data-repmat(mean(data,2),1,2286);
[U,S,V]=svd(data_sub,'econ');
figure(3);
subplot(2,2,1);imagesc(reshape(U(:,1),192,168)), colormap gray;
subplot(2,2,2);imagesc(reshape(U(:,2),192,168)), colormap gray;
subplot(2,2,3);imagesc(reshape(U(:,3),192,168)), colormap gray;
subplot(2,2,4);imagesc(reshape(U(:,4),192,168)), colormap gray;
```

2.4 结果


20
40
60
80
100
120
140
160
180
20
40
60
80
100
120
140
160


50
100
150
50
100
150
50
100
150
50
100
150
50
100
150
50
100
150

3 人脸近似

3.1 任务

- 利用SVD得到的奇异向量做低秩近似
- 分别用一张测试集中的人脸和其他任意图片测试

3.2 理解

需要利用SVD得到的奇异向量，对数据集外的人脸进行近似。这里利用投影$\hat{\mathbf{x}}_{\text{test}} = \mathbf{U}\mathbf{U}^T\mathbf{x}_{\text{test}}$完成用学习到的特征脸估计测试集数据，取不同截断数的左奇异矩阵$\mathbf{U}$来代表不同秩下的近似。

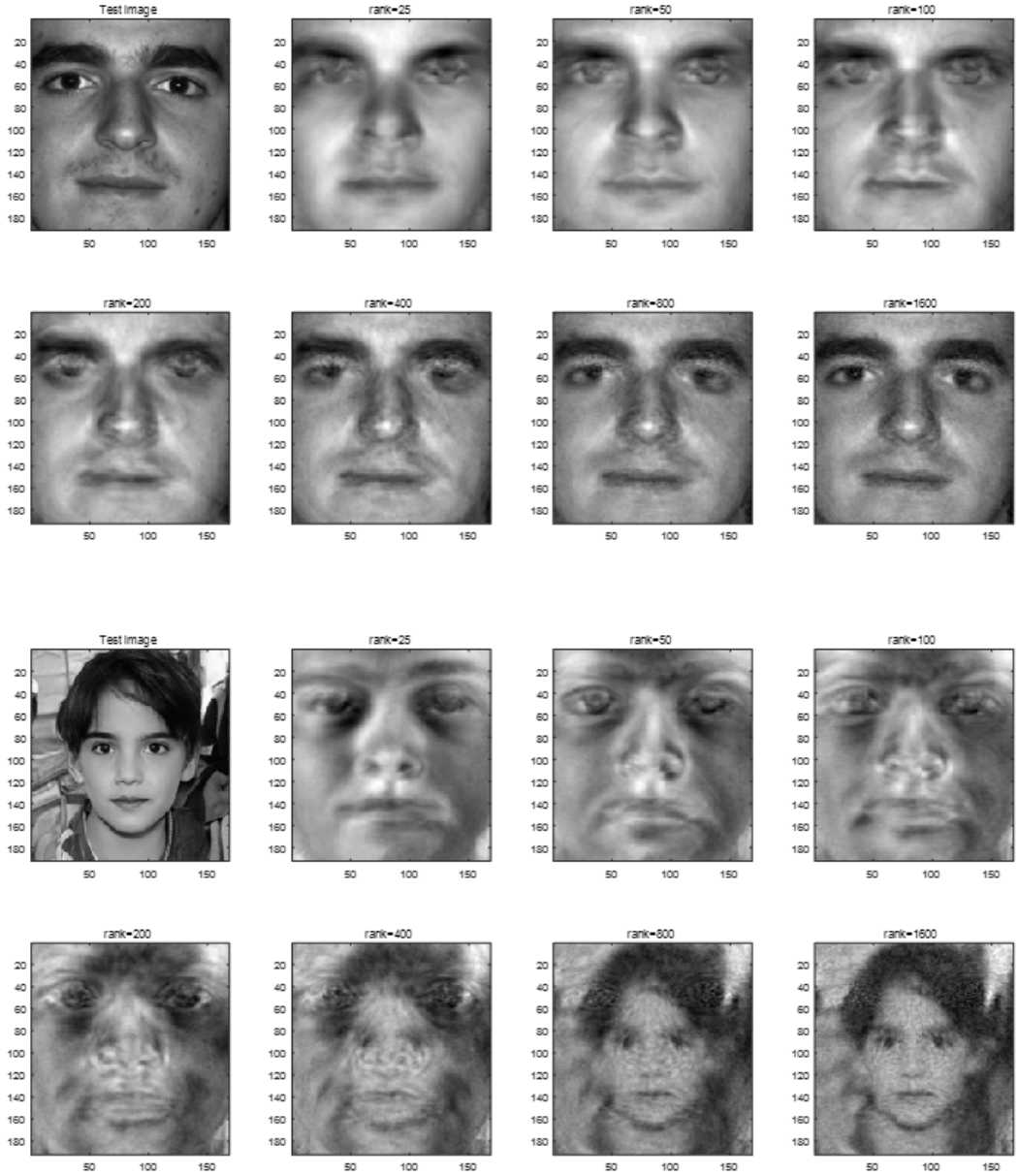
3.3 实现

1. 代码实现思路
    - 取出测试人脸向量，与不同秩下的奇异矩阵投影，得到估计数据
    - 作出图像对比
2. 代码如下

```
%用第38个人的脸测试
X_38=X{1,38};
face_test1=X_38(:,1);
face_fit1=U(:,1:25)*U(:,1:25)'*face_test1;%rank=25
face_fit2=U(:,1:50)*U(:,1:50)'*face_test1;%rank=50
face_fit3=U(:,1:100)*U(:,1:100)'*face_test1;%rank=100
face_fit4=U(:,1:200)*U(:,1:200)'*face_test1;%rank=200
face_fit5=U(:,1:400)*U(:,1:400)'*face_test1;%rank=400
face_fit6=U(:,1:800)*U(:,1:800)'*face_test1;%rank=800
face_fit7=U(:,1:1600)*U(:,1:1600)'*face_test1;%rank=1600
figure(4);
subplot(2,4,1);imagesc(reshape(face_test1,192,168)), colormap
gray;title('Test image');
subplot(2,4,2);imagesc(reshape(face_fit1,192,168)), colormap
gray;title('rank=25');
subplot(2,4,3);imagesc(reshape(face_fit2,192,168)), colormap
gray;title('rank=50');
subplot(2,4,4);imagesc(reshape(face_fit3,192,168)), colormap
gray;title('rank=100');
subplot(2,4,5);imagesc(reshape(face_fit4,192,168)), colormap
```

```
gray;title('rank=200');
subplot(2,4,6);imagesc(reshape(face_fit5,192,168)), colormap
gray;title('rank=400');
subplot(2,4,7);imagesc(reshape(face_fit6,192,168)), colormap
gray;title('rank=800');
subplot(2,4,8);imagesc(reshape(face_fit7,192,168)), colormap
gray;title('rank=1600');

%自选图片测试
load('testimage.mat');
face2_test=im2double(testimage);
face2_fit1=U(:,1:25)*U(:,1:25)'*face2_test;%rank=25
face2_fit2=U(:,1:50)*U(:,1:50)'*face2_test;%rank=50
face2_fit3=U(:,1:100)*U(:,1:100)'*face2_test;%rank=100
face2_fit4=U(:,1:200)*U(:,1:200)'*face2_test;%rank=200
face2_fit5=U(:,1:400)*U(:,1:400)'*face2_test;%rank=400
face2_fit6=U(:,1:800)*U(:,1:800)'*face2_test;%rank=800
face2_fit7=U(:,1:1600)*U(:,1:1600)'*face2_test;%rank=1600
figure(5);
subplot(2,4,1);imagesc(reshape(face2_test,192,168)), colormap
gray;title('Test image');
subplot(2,4,2);imagesc(reshape(face2_fit1,192,168)), colormap
gray;title('rank=25');
subplot(2,4,3);imagesc(reshape(face2_fit2,192,168)), colormap
gray;title('rank=50');
subplot(2,4,4);imagesc(reshape(face2_fit3,192,168)), colormap
gray;title('rank=100');
subplot(2,4,5);imagesc(reshape(face2_fit4,192,168)), colormap
gray;title('rank=200');
subplot(2,4,6);imagesc(reshape(face2_fit5,192,168)), colormap
gray;title('rank=400');
subplot(2,4,7);imagesc(reshape(face2_fit6,192,168)), colormap
gray;title('rank=800');
subplot(2,4,8);imagesc(reshape(face2_fit7,192,168)), colormap
gray;title('rank=1600');
```

3.4 结果



---

4 人脸识别及分类预处理

4.1 任务

- 将两个不同的人的脸投影特征空间并可视化

4.2 理解

可以用特征脸作为坐标系统，构成特征脸空间，将不同的人脸投影到该特征空间，即可观察出不同人脸的差别。理论上，特征脸选取越多，差异越明显。

4.3 实现

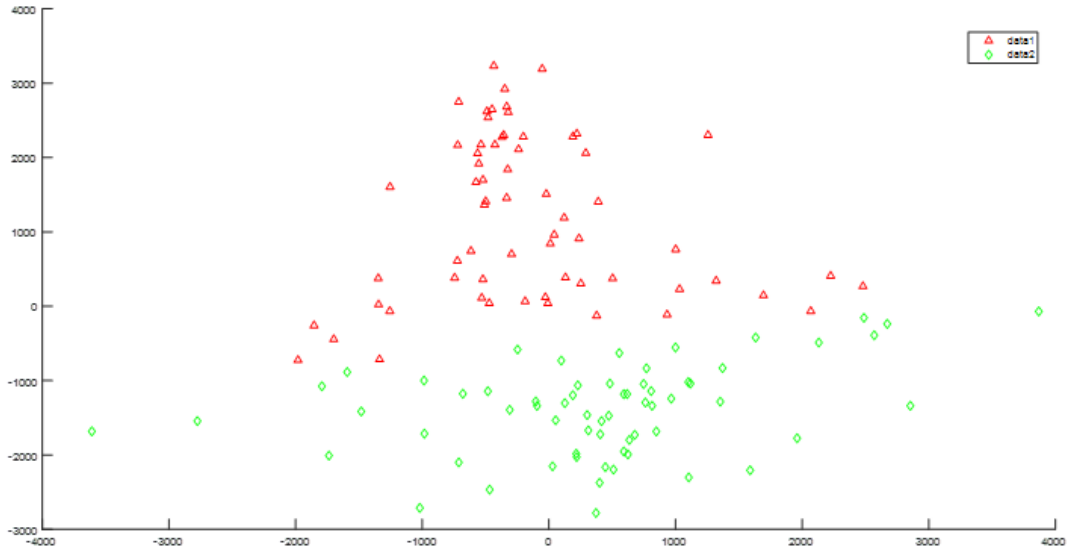
1. 代码实现思路
    - 考虑用向量内积做投影，每个人64张脸即得到64个坐标
    - 作出图像可视化
2. 代码如下

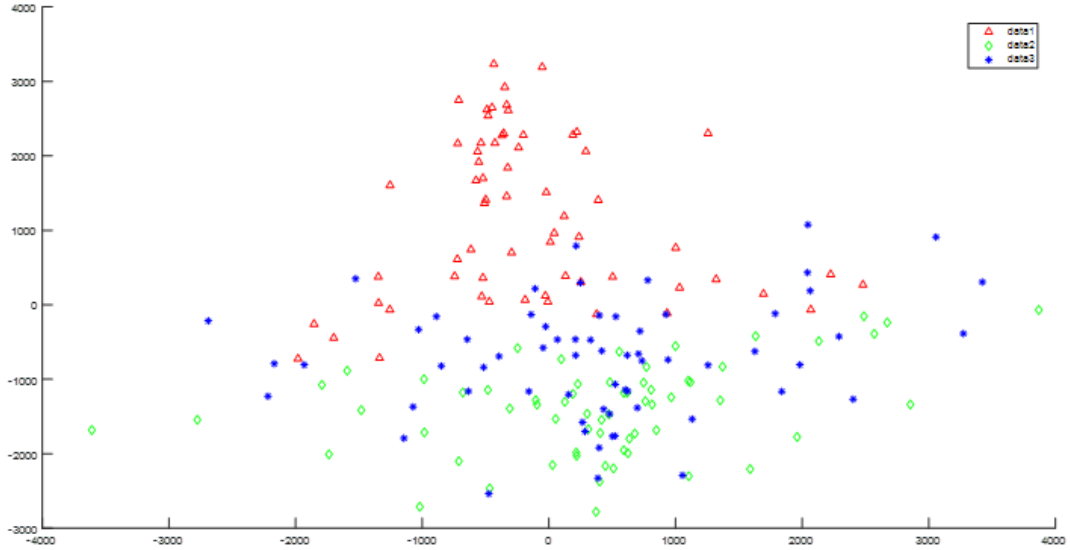
```
%任务四
%二维空间可视化
X_2=X{1,2};%第二个人所有脸的数据
X_3=X{1,3};%第三个人所有脸的数据
X_7=X{1,7};%第七个人所有脸的数据
pc5_2=[];pc6_2=[];%第二个人脸在pc5和pc6上的投影
pc5_7=[];pc6_7=[];%第七个人脸在pc5和pc6上的投影
pc5_3=[];pc6_3=[];%第三个人脸在pc5和pc6上的投影
for i=1:64
    pc5_2(i)=X_2(:,i)'*U(:,5)/norm(U(:,5));
    pc6_2(i)=X_2(:,i)'*U(:,6)/norm(U(:,6));
end
for j=1:64
    pc5_7(j)=X_7(:,j)'*U(:,5)/norm(U(:,5));
    pc6_7(j)=X_7(:,j)'*U(:,6)/norm(U(:,6));
end
for k=1:64
    pc5_3(k)=X_3(:,k)'*U(:,5)/norm(U(:,5));
    pc6_3(k)=X_3(:,k)'*U(:,6)/norm(U(:,6));
end
figure(6);
scatter(pc5_2,pc6_2,'r','^');
hold on;
scatter(pc5_7,pc6_7,'g','d');
hold on;
scatter(pc5_3,pc6_3,'b','*');

%三维空间可视化
kpc4_2=[];kpc5_2=[];kpc6_2=[];%第二个人脸在pc5和pc6上的投影
kpc4_7=[];kpc5_7=[];kpc6_7=[];%第七个人脸在pc5和pc6上的投影
kpc4_3=[];kpc5_3=[];kpc6_3=[];%第三个人脸在pc5和pc6上的投影
```

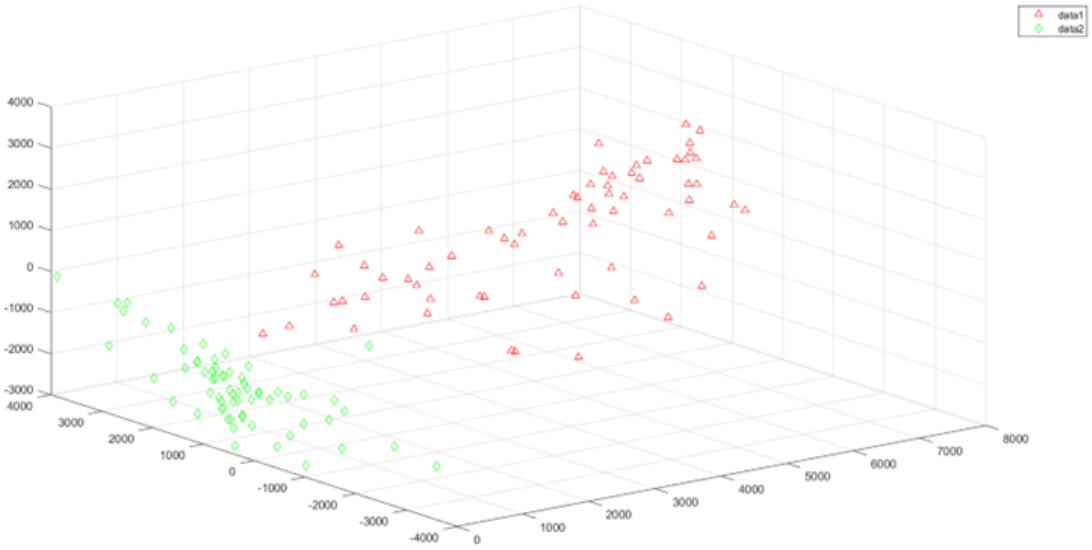
```
for i=1:64
    kpc5_2(i)=X_2(:,i)'*U(:,5)/norm(U(:,5));
    kpc6_2(i)=X_2(:,i)'*U(:,6)/norm(U(:,6));
    kpc4_2(i)=X_2(:,i)'*U(:,4)/norm(U(:,4));
end
for j=1:64
    kpc5_7(j)=X_7(:,j)'*U(:,5)/norm(U(:,5));
    kpc6_7(j)=X_7(:,j)'*U(:,6)/norm(U(:,6));
    kpc4_7(i)=X_7(:,i)'*U(:,4)/norm(U(:,4));
end
for k=1:64
    kpc5_3(k)=X_3(:,k)'*U(:,5)/norm(U(:,5));
    kpc6_3(k)=X_3(:,k)'*U(:,6)/norm(U(:,6));
    kpc4_3(i)=X_3(:,k)'*U(:,4)/norm(U(:,4));
end
figure(9);
scatter3(kpc4_2,kpc5_2,kpc6_2,'r','^');
hold on;
scatter3(kpc4_7,kpc5_7,kpc6_7,'g','d');
hold on;
scatter3(kpc4_3,kpc5_3,kpc6_3,'b','*');
```

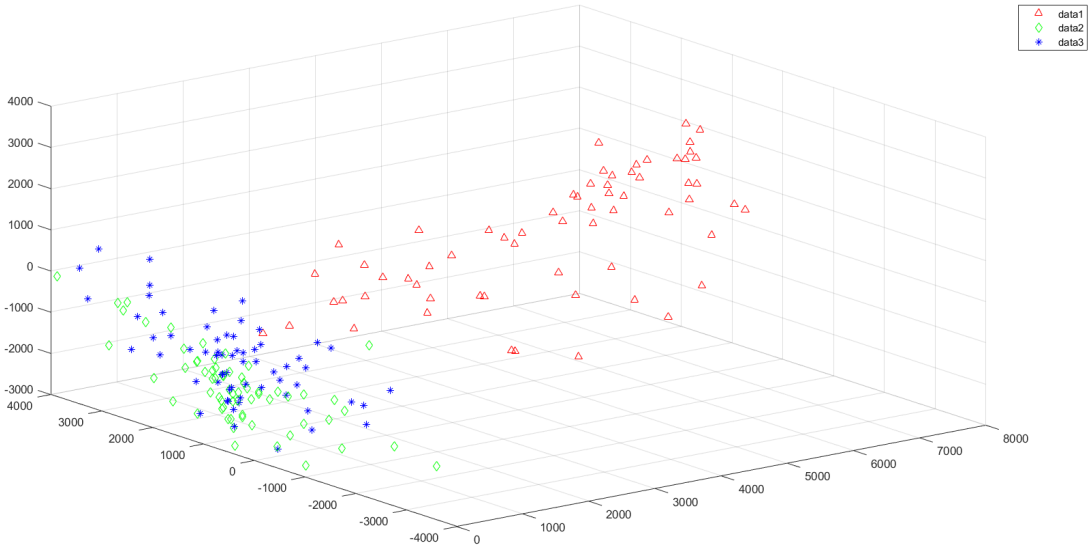
4.3 结果

可以发现，对于两个不同的人的脸，利用特征空间的分类处理是有效的；对于三个不同人的脸，会出现两个人的混叠，无法完成识别作用。


data1
data2
4000
3000
2000
1000
0
-1000
-2000
-3000
-4000
-3000
-2000
-1000
0
1000
2000
3000
4000


data1
data2
data3
4000
3000
2000
1000
0
-1000
-2000
-3000
-4000
-3000
-2000
-1000
0
1000
2000
3000
4000


data1
data2
4000
3000
2000
1000
0
-1000
-2000
-3000
-4000
0
1000
2000
3000
4000
5000
6000
7000
8000


data1
data2
data3
4000
3000
2000
1000
0
-1000
-2000
-3000
-4000
0
1000
2000
3000
4000
5000
6000
7000
8000