Principle Component Analysis

W. Hoyt Thomas

02/21/2020

I. Abstract

Principle Component Analysis was performed on three different camera angles of simple harmonic motion in four different scenarios. The goal was to explore the usefulness of the method.

II. Introduction

In data science, often it is important to consider the time and computing power necessary to complete analysis. The challenge lies in determining the most efficient way to get desired results from collected data. Principle Component Analysis (PCA) is a valuable tool to deal with redundant data and ease the strain put on computers. Through the use of singular value decomposition, PCA tells us where the data is varying the most and which dimension it is in.

To demonstrate the usefulness of PCA, four different scenarios have been analyzed. Each scenario features a demonstration of simple harmonic motion from three different angles: a paint can attached to a spring. The first scenario just shows the harmonic motion, in the second scenario the camera shakes to introduce noise, in the third scenario the paint can swings in the x dimension in addition to the z direction harmonic motion, and in the fourth scenario the paint can rotates along with swinging in the x direction and harmonic motion in z direction.

III. Theoretical Background

The Singular Value Decomposition (SVD) provides the theoretical basis for the PCA. The SVD is one way of diagonalizing of matrix or reducing a matrix to its *ideal* basis. The SVD of matrix, A, is defined as:

$$A = U\Sigma V^* \text{ (EQ 1)}$$

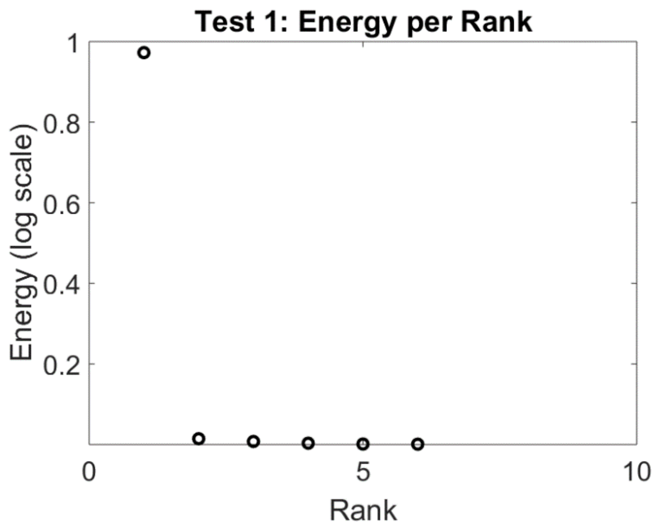

Figure 1: Energy contained in each singular value from A_1 in Test 1

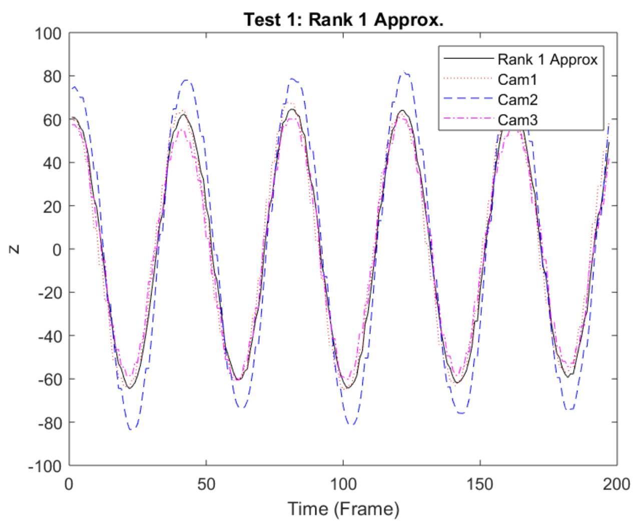

Figure 2: Test 1 Rank 1 Approximation compared to the three camera angles

The ideal basis is given by the singular values, which are the diagonal values in the sigma matrix. Singular values tell us how much variance is in each of the dimensions, the number of non-zero singular values is the rank of A and are ordered from largest to smallest. The principle components are contained in the columns of U or the left singular vectors. It is also possible to represent the relative energy of each rank according to the singular values.

IV. Algorithm Development

The first step was to construct the positional matrix of the paint cam at each frame for the three camera angles. This was achieved by using the Computer Vision toolbox in Matlab, the pointTracker tool was the core of the tracking done. PointTracker needs a set of points to track, to decide which points to track, detectMinEigenfeatures() was used to determine corner points which should be easier to track.

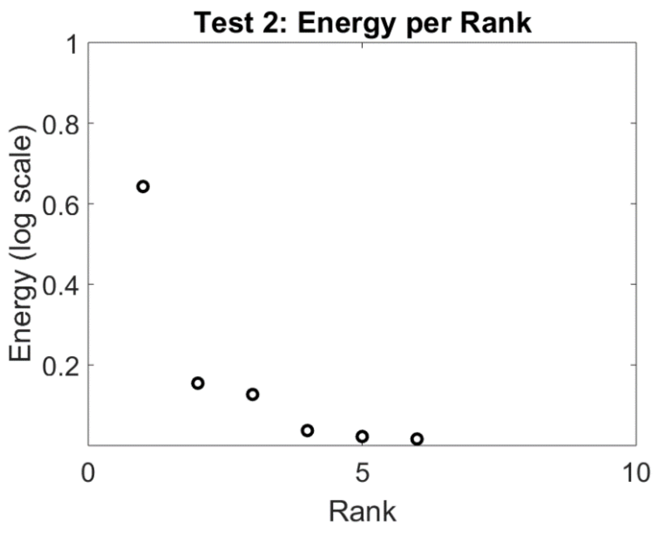

Figure 3: Energy contained in each singular value from A_2 in Test 2

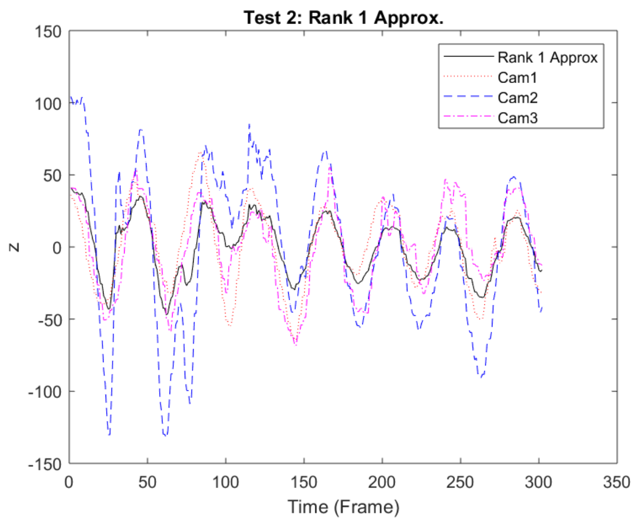

Figure 4: Test 2 Rank 1 Approximation compared to the three camera angles

Once points were determined they were passed to PointTracker and iterated through the length of each video. This was repeated for all twelve videos, for Scenario 2 (Noisy Data), multiple points had to be tested to locate a set PointTracker was able to locate through the noise. With paint can locations for each camera, the positional matrix could be constructed. For each camera the X and Y positions were separated and transposed into row vectors with the mean of each row subtracted from each value in the row. The data from different cameras was also sliced such that paint can started at the same point for all three angles. The X and Y vectors from each angle were then vertically concatenated to get A_test# [NumframesX6]. This matrix was then passed to the svd() command to decompose the matrix. This process was repeated for all four scenarios.

To visualize the amount of variance contained in each rank of matrix, energies were plotted on a logscale (Fig. 1,3,5,7). To visualize the accuracy of the rank 1 approximation, it was plotted against the original tracking from each camera angle (Fig. 2,4,6,8).

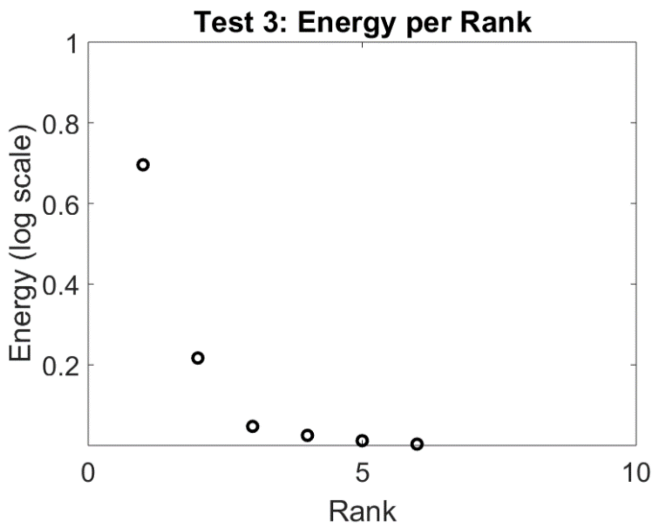

Figure 5: Energy contained in each singular value from A_3 in Test 3

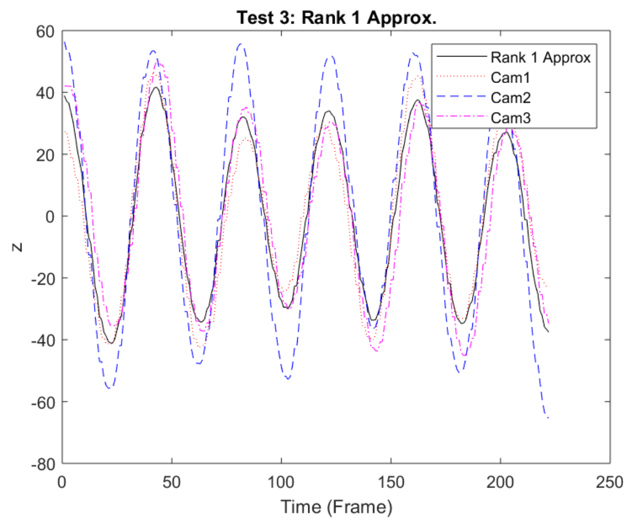

Figure 6: Test 3 Rank 1 Approximation compared to the three camera angles

V. Computational Results

Scenario 1: Ideal Case

Figure 1 showed that about 97% of the variance was contained in the first rank of the principle components and Figure 2 showed the Rank 1 approximation is good estimator of the z location of the paint can.

Scenario 2: Noisy Case

Figure 3 showed about 70% of the variance was contained in the first rank of the principle components, with about 10-15% in the second and third ranks. The rank 1 approximation (Figure 4) is decent estimator of the z location of the paint can but does not fit as well as in scenario 1.

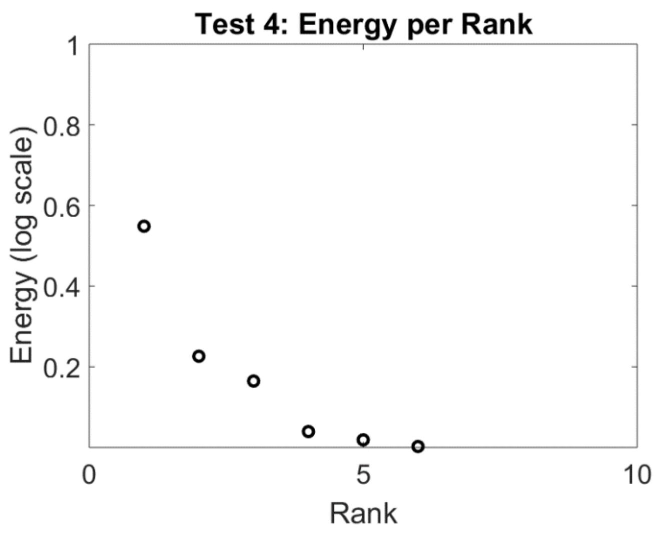

Figure 7: Energy contained in each singular value from A_4 in Test 4

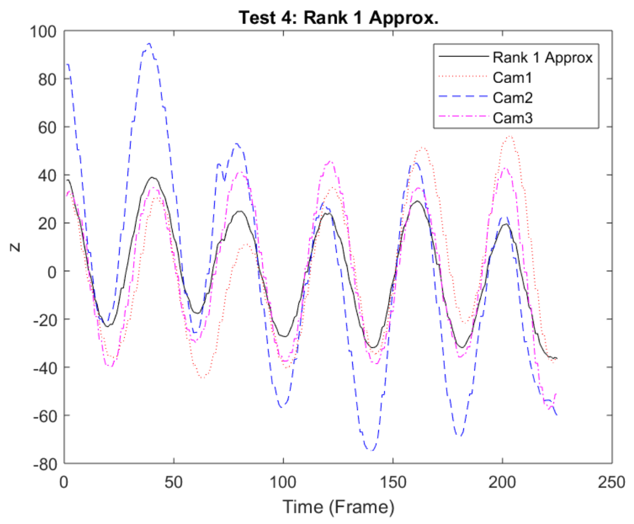

Figure 8: Test 4 Rank 1 Approximation compared to the three camera angles

<u>Scenario 3: Horizontal Displacement</u>

Figure 5 showed about 75% of the variance was contained in the first rank of the principle components, and another 20% in the second rank. The rank 1 approximation (Figure 6) does a decent job of estimating the z displacement of the paint can.

<u>Scenario 4: Horizontal Displacement & Rotation</u>

Figure 8 showed about 60% of the variance was contained in the first rank of the of principle components, with about 20% contained in the second and third ranks. The rank 1 approximation (Figure 8) does estimates the z-displacement with the worst accuracy of the four scenarios.

VI. Conclusion

PCA is a powerful tool to determine the essence of data and extract the important patterns present. The Rank 1 approximation accuracy decreased in accuracy after the first scenario, which follows intuition as other motion was added to the paint can.

Appendix A

detectMinEigenFeatures(I, ‘ROI’, [X Y Width Height])- Finds corner features of object in image
Input Arguments: I – greyscale image, ‘ROI’ – Rectangular region of interest X,Y are top right pixels of region followed by Width and Height of region

vision.PointTracker()- tracks points in images
Must be initialized before use

initialize(tracker,points,I)
Input Arguments: tracker – vision.PointTracker, points - points to be tracked, I – image
svd(A)- computes singular value decomposition of matrix A

Appendix B

```
%AMATH 482 HW3
%2/21/20
%Hoyt Thomas
clear all; close all; clc;

%Test I
load('cam1_1.mat');
load('cam2_1.mat');
load('cam3_1.mat');

X = vidFrames1_1(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[320,220,70,100]);
pointImage = insertMarker(I,point.Location,'+','Color','white');

tracker = vision.PointTracker();
initialize(tracker,point.Location(1,:),I);

points_arr_1_1 = zeros(226,2);
numframes = size(vidFrames1_1,4);
for j = 1:numframes
    color_x = vidFrames1_1(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_1_1(j,:) = points;
end
%%
X = vidFrames2_1(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[260,270,100,100]);
pointImage = insertMarker(I,point.Location,'+','Color','white');

tracker = vision.PointTracker();
initialize(tracker,point.Location(7,:),I);

points_arr_2_1 = zeros(284,2);
numframes = size(vidFrames2_1,4);
for j = 1:numframes
    color_x = vidFrames2_1(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_2_1(j,:) = points;
end
```

```
%%
X = vidFrames3_1(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[310,250,70,100]);
pointImage = insertMarker(I,point.Location,'+','Color','white');

tracker = vision.PointTracker();
initialize(tracker,point.Location(71,:),I);

points_arr_3_1 = zeros(232,2);
numframes = size(vidFrames3_1,4);
for j = 1:numframes
    color_x = vidFrames3_1(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_3_1(j,:) = points;
end

%%
L_11 = length(points_arr_1_1);
t_11 = (1:L_11);
L_12 = length(points_arr_2_1);
t_12 = (1:L_12);
L_13 = length(points_arr_3_1);
t_13 = (1:L_13);

figure(1);
plot(t_11, points_arr_1_1(:,2), 'r')
hold on;
plot(t_12, points_arr_2_1(:,2), 'b')
hold on;
plot(t_13, points_arr_3_1(:,1), 'g')

%%
x_1_1 = points_arr_1_1(30:226,1) -
mean(points_arr_1_1(30:226,1));
y_1_1 = points_arr_1_1(30:226,2) -
mean(points_arr_1_1(30:226,2));
x_2_1 = points_arr_2_1(38:234,1) -
mean(points_arr_2_1(38:234,1));
y_2_1 = points_arr_2_1(38:234,2) -
mean(points_arr_2_1(38:234,2));
x_3_1 = points_arr_3_1(29:225,2) -
mean(points_arr_3_1(29:225,2));
y_3_1 = points_arr_3_1(29:225,1) -
mean(points_arr_3_1(29:225,1));
t_1 = (1:197);
```

```
figure(2);
plot(t_1, y_1_1, 'r', 'DisplayName', 'Cam1')
hold on;
plot(t_1, y_2_1, 'b', 'DisplayName', 'Cam2')
hold on;
plot(t_1, y_3_1, 'g', 'DisplayName', 'Cam3')
legend

A_1 = [(x_1_1).'; (y_1_1).'; (x_2_1).'; (y_2_1).'; (x_3_1).';
(y_3_1).'];

%%
[U,S,V] = svd(A_1,'econ');
sig = diag(S);

plot(sig.^2/sum(sig.^2),'ko','Linewidth',2)
axis([0 10 10^-(18) 1])
ylabel('Energy (log scale)')
xlabel('Rank')
set(gca,'Fontsize',16,'Xtick',0:5:25)
title('Test 1: Energy per Rank')
%%
X1_rank1 = U(:,1)*S(1,1)*V(:,1).';

plot(t_1, X1_rank1(2,:), 'k-', 'DisplayName', 'Rank 1 Approx' );
hold on;
plot(t_1, y_1_1, 'r:', 'DisplayName', 'Cam1')
hold on;
plot(t_1, y_2_1, 'b--', 'DisplayName', 'Cam2')
hold on;
plot(t_1, y_3_1, 'm-.', 'DisplayName', 'Cam3')
xlabel('Time (Frame)')
ylabel('z')
title('Test 1: Rank 1 Approx.')
legend

%% Test II
load('cam1_2.mat');
load('cam2_2.mat');
load('cam3_2.mat');

X = vidFrames1_2(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[320,280,70,100]);
pointImage = insertMarker(I,point.Location,'+','Color','white');
```

```
tracker = vision.PointTracker();
initialize(tracker,point.Location(1,:),I);

points_arr_1_2 = zeros(314,2);
numframes = size(vidFrames1_2,4);
for j = 1:numframes
    color_x = vidFrames1_2(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_1_2(j,:) = points;
end

%%

X = vidFrames2_2(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[285,350,70,100]);
pointImage = insertMarker(I,point.Location,'+','Color','white');

tracker = vision.PointTracker();
initialize(tracker,point.Location(52,:),I);

points_arr_2_2 = zeros(356,2);
numframes = size(vidFrames2_2,4);
for j = 1:numframes
    color_x = vidFrames2_2(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_2_2(j,:) = points;
end


%%
X = vidFrames3_2(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[340,240,100,70]);
pointImage = insertMarker(I,point.Location,'+','Color','white');

tracker = vision.PointTracker();
initialize(tracker,point.Location(1,:),I);

points_arr_3_2 = zeros(327,2);
numframes = size(vidFrames3_2,4);
for j = 1:numframes
    color_x = vidFrames3_2(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_3_2(j,:) = points;
end

%%
L_21 = length(points_arr_1_2);
t_21 = (1:L_21);
L_22 = length(points_arr_2_2);
t_22 = (1:L_22);
L_23 = length(points_arr_3_2);
t_23 = (1:L_23);

plot(t_21, points_arr_1_2(:,2), 'r')
hold on;
plot(t_22, points_arr_2_2(:,2), 'b')
hold on;
plot(t_23, points_arr_3_2(:,1), 'g')

%%
x_1_2 = points_arr_1_2(13:314,1) -
mean(points_arr_1_2(13:314,1));
y_1_2 = points_arr_1_2(13:314,2) -
mean(points_arr_1_2(13:314,2));
x_2_2 = points_arr_2_2(36:337,1) -
mean(points_arr_2_2(36:337,1));
y_2_2 = points_arr_2_2(36:337,2) -
mean(points_arr_2_2(36:337,2));
x_3_2 = points_arr_3_2(15:316,2) -
mean(points_arr_3_2(15:316,2));
y_3_2 = points_arr_3_2(15:316,1) -
mean(points_arr_3_2(15:316,1));
t_2 = (1:302);

figure(2);
plot(t_2, y_1_2, 'r', 'DisplayName', 'Cam1')
hold on;
plot(t_2, y_2_2, 'b', 'DisplayName', 'Cam2')
hold on;
plot(t_2, y_3_2, 'g', 'DisplayName', 'Cam3')
legend

A_2 = [(x_1_2).'; (y_1_2).'; (x_2_2).'; (y_2_2).'; (x_3_2).';
(y_3_2).'];
%%
[U,S,V] = svd(A_2,'econ');
sig = diag(S);

plot(sig.^2/sum(sig.^2),'ko','Linewidth',2)
```

```
axis([0 10 10^-(18) 1])
ylabel('Energy (log scale)')
xlabel('Rank')
set(gca,'Fontsize',16,'Xtick',0:5:25)
title('Test 2: Energy per Rank')
%%
X2_rank1 = U(:,1)*S(1,1)*V(:,1).';

plot(t_2, X2_rank1(2,:), 'k-', 'DisplayName', 'Rank 1 Approx' );
hold on;
plot(t_2, y_1_2, 'r:', 'DisplayName', 'Cam1')
hold on;
plot(t_2, y_2_2, 'b--', 'DisplayName', 'Cam2')
hold on;
plot(t_2, y_3_2, 'm-.', 'DisplayName', 'Cam3')
xlabel('Time (Frame)')
ylabel('z')
title('Test 2: Rank 1 Approx.')
legend

%% Test III
load('cam1_3.mat');
load('cam2_3.mat');
load('cam3_3.mat');

X = vidFrames1_3(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[310,270,70,100]);
pointImage = insertMarker(I,point.Location,'+','Color','white');

tracker = vision.PointTracker();
initialize(tracker,point.Location(1,:),I);

points_arr_1_3 = zeros(239,2);
numframes = size(vidFrames1_3,4);
for j = 1:numframes
    color_x = vidFrames1_3(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_1_3(j,:) = points;
end

%%
X = vidFrames2_3(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[226,290,70,100]);
pointImage = insertMarker(I,point.Location,'+','Color','white');
```

```
tracker = vision.PointTracker();
initialize(tracker,point.Location(22,:),I);

points_arr_2_3 = zeros(281,2);
numframes = size(vidFrames2_3,4);
for j = 1:numframes
    color_x = vidFrames2_3(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_2_3(j,:) = points;
end

%%
X = vidFrames3_3(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[345,208,100,70]);
pointImage = insertMarker(I,point.Location,'+','Color','white');

tracker = vision.PointTracker();
initialize(tracker,point.Location(3,:),I);

points_arr_3_3 = zeros(237,2);
numframes = size(vidFrames3_3,4);
for j = 1:numframes
    color_x = vidFrames3_3(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_3_3(j,:) = points;
end

%%
L_31 = length(points_arr_1_3);
t_31 = (1:L_31);
L_32 = length(points_arr_2_3);
t_32 = (1:L_32);
L_33 = length(points_arr_3_3);
t_33 = (1:L_33);

plot(t_31, points_arr_1_3(:,2), 'r')
hold on;
plot(t_32, points_arr_2_3(:,2), 'b')
hold on;
plot(t_33, points_arr_3_3(:,1), 'g')
%%
x_1_3 = points_arr_1_3(18:239,1) -
mean(points_arr_1_3(18:239,1));
```

```
y_1_3 = points_arr_1_3(18:239,2) -
mean(points_arr_1_3(18:239,2));
x_2_3 = points_arr_2_3(5:226,1) - mean(points_arr_2_3(5:226,1));
y_2_3 = points_arr_2_3(5:226,2) - mean(points_arr_2_3(5:226,2));
x_3_3 = points_arr_3_3(8:229,2) - mean(points_arr_3_3(8:229,2));
y_3_3 = points_arr_3_3(8:229,1) - mean(points_arr_3_3(8:229,1));
t_3 = (1:222);

figure(2);
plot(t_3, y_1_3, 'r', 'DisplayName', 'Cam1')
hold on;
plot(t_3, y_2_3, 'b', 'DisplayName', 'Cam2')
hold on;
plot(t_3, y_3_3, 'g', 'DisplayName', 'Cam3')
legend

A_3 = [(x_1_3).'; (y_1_3).'; (x_2_3).'; (y_2_3).'; (x_3_3).';
(y_3_3).'];
%%
[U,S,V] = svd(A_3,'econ');
sig = diag(S);

plot(sig.^2/sum(sig.^2),'ko','Linewidth',2)
axis([0 10 10^-(18) 1])
ylabel('Energy (log scale)')
xlabel('Rank')
set(gca,'Fontsize',16,'Xtick',0:5:25)
title('Test 3: Energy per Rank')
%%
X3_rank1 = U(:,1)*S(1,1)*V(:,1).';

plot(t_3, X3_rank1(2,:), 'k-', 'DisplayName', 'Rank 1 Approx' );
hold on;
plot(t_3, y_1_3, 'r:', 'DisplayName', 'Cam1')
hold on;
plot(t_3, y_2_3, 'b--', 'DisplayName', 'Cam2')
hold on;
plot(t_3, y_3_3, 'm-.', 'DisplayName', 'Cam3')
xlabel('Time (Frame)')
ylabel('z')
title('Test 3: Rank 1 Approx.')
legend

%% Test IV
load('cam1_4.mat');
load('cam2_4.mat');
load('cam3_4.mat');
```

```
X = vidFrames1_4(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[365,270,70,100]);
pointImage = insertMarker(I,point.Location,'+','Color','white');

tracker = vision.PointTracker();
initialize(tracker,point.Location(2,:),I);

points_arr_1_4 = zeros(392,2);
numframes = size(vidFrames1_4,4);
for j = 1:numframes
    color_x = vidFrames1_4(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_1_4(j,:) = points;
end

%%
X = vidFrames2_4(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[225,210,75,150]);
pointImage = insertMarker(I,point.Location,'+','Color','white');

tracker = vision.PointTracker();
initialize(tracker,point.Location(60,:),I);
%60
points_arr_2_4 = zeros(405,2);
numframes = size(vidFrames2_4,4);
for j = 1:numframes
    color_x = vidFrames2_4(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_2_4(j,:) = points;
end


%%

X = vidFrames3_4(:,:,:,1);
I = rgb2gray(X);
point = detectMinEigenFeatures(I,'ROI',[345,180,100,70]);
pointImage = insertMarker(I,point.Location,'+','Color','white');

tracker = vision.PointTracker();
initialize(tracker,point.Location(60,:),I);
%49, 55, 60!, 101
```

```
points_arr_3_4 = zeros(394,2);
numframes = size(vidFrames3_4,4);
for j = 1:numframes
    color_x = vidFrames3_4(:,:,:,j);
    frame = rgb2gray(color_x);
    [points] = tracker(frame);
    points_arr_3_4(j,:) = points;
end


%%
L_41 = length(points_arr_1_4);
t_41 = (1:L_41);
L_42 = length(points_arr_2_4);
t_42 = (1:L_42);
L_43 = length(points_arr_3_4);
t_43 = (1:L_43);

plot(t_41, points_arr_1_4(:,2), 'r')
hold on;
plot(t_42, points_arr_2_4(:,2), 'b')
hold on;
plot(t_43, points_arr_3_4(:,1), 'g')

%%
x_1_4 = points_arr_1_4(32:256,1) -
mean(points_arr_1_4(32:256,1));
y_1_4 = points_arr_1_4(32:256,2) -
mean(points_arr_1_4(32:256,2));
x_2_4 = points_arr_2_4(1:225,1) - mean(points_arr_2_4(1:224,1));
y_2_4 = points_arr_2_4(1:225,2) - mean(points_arr_2_4(1:224,2));
x_3_4 = points_arr_3_4(32:256,2) -
mean(points_arr_3_4(32:256,2));
y_3_4 = points_arr_3_4(32:256,1) -
mean(points_arr_3_4(32:256,1));
t_4 = (1:225);

figure(2);
plot(t_4, y_1_4, 'r', 'DisplayName', 'Cam1')
hold on;
plot(t_4, y_2_4, 'b', 'DisplayName', 'Cam2')
hold on;
plot(t_4, y_3_4, 'g', 'DisplayName', 'Cam3')
legend

A_4 = [(x_1_4).'; (y_1_4).'; (x_2_4).'; (y_2_4).'; (x_3_4).';
(y_3_4).'];
```

```
%%
[U,S,V] = svd(A_4,'econ');
sig = diag(S);

plot(sig.^2/sum(sig.^2),'ko','Linewidth',2)
axis([0 10 10^-(18) 1])
ylabel('Energy (log scale)')
xlabel('Rank')
set(gca,'Fontsize',16,'Xtick',0:5:25)
title('Test 4: Energy per Rank')
%%
X4_rank1 = U(:,1)*S(1,1)*V(:,1).';

plot(t_4, X4_rank1(2,:), 'k-', 'DisplayName', 'Rank 1 Approx' );
hold on;
plot(t_4, y_1_4, 'r:', 'DisplayName', 'Cam1')
hold on;
plot(t_4, y_2_4, 'b--', 'DisplayName', 'Cam2')
hold on;
plot(t_4, y_3_4, 'm-.', 'DisplayName', 'Cam3')
xlabel('Time (Frame)')
ylabel('z')
title('Test 4: Rank 1 Approx.')
legend
```