AMATH 482: Homework 3

Alex Omusoru

Due: February 24, 2021

Abstract

Provided with several cases of a spring-mass system, the intent of this report is to determine the motion of the simple harmonic motion. The first case consists of the ideal case of motion along one axis. The second case consists of the ideal case but with camera shaking, so as to add noise to the measurements. The third case consists of a mass released off-center resulting in motion in the x-y plane as well as the vertical direction (z-direction). The fourth and final case consists of horizontal displacement (like the third case) as well as rotation.

1 Introduction and Overview

The goal of this paper had several parts. The first was to extract the motion of a mass from a RGB video, and report the (x, y, z) coordinates of the motion for every case. After that was complete, a Singular Value Decomposition (SVD) was used to further a Principal Component Analysis (PCA) of each cases motion. The PCA was used to determine the general direction of motion, and then the results were compared between the cases to determine the behaviour of simple harmonic motion.

2 Theoretical Background

In this report, the singular value decomposition (SVD) was used in a principal component analysis of the average x and y values of the mass motion. For any matrix A, we define the SVD of that matrix to be:

$$A = U\Sigma V^*$$

The matrices U and V^* are both unitary matrice, while Σ is a diagonal matrix with the diagonal formed of the singular values of A in decreasing order.

The most straightforward application of the SVD is for low-rank approximations. In Principal Component Analysis, we look specifically at the U matrix, as each vector in the range represents a component, or a direction of variance. This allows us to interpret the data with respect to the principal component vectors $u_1, u_2, ...$ rather than x and y.

3 Algorithm Implementation and Development

The **Algorithm: Case by Case Analysis** was applied for each case to determine the motion along the x and y axes, and then to draw the first principal component of the motion.

Algorithm: Case by Case Analysis

```
given video1, video2 and video3
for each video j :
    for i = 1:(number of frames in video j )
        X = ith frame in grayscale

        M = largest value (8-bit integer)
```

Case 1

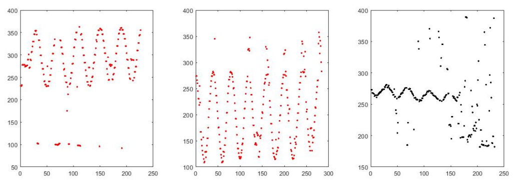

Figure 1: Graph of Y for video 1, 2, and 3

Case 1

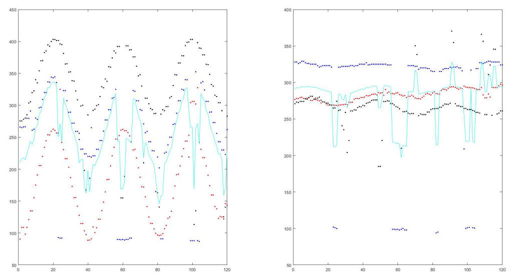

Figure 2: Graph of X and Y with Averages

```
        Mx, My = coordinates of all points with value M

        xj(i) = mean(Mx)
        yj(i) = mean(My)
    end
end

align yj waves and reduce to same number of frames

y_avg = average(all yj)
x_avg = average(all xj)

[U, S, V] = SVD of [x_avg; y_avg]
```

4 Computational Results

For each case, three plots are presented. The first is the side-by-side of the y-values for the three videos. The second is the side-by-side of all three videos overlaid for both x and y, with the cyan line representing the average values for each frame. The final graph is the data in the x,y plane with the first principal component data presented.

5 Summary and Conclusions

Throughout this report, several oversimplifications were made for simplicity, which resulted in skewed data. However, from the results we can conclude that harmonic motion tends to one direction as can be seen from the first principal component, and that with a slight shift in camera position, the principal component could correspond with the x or y axis.

Case 1

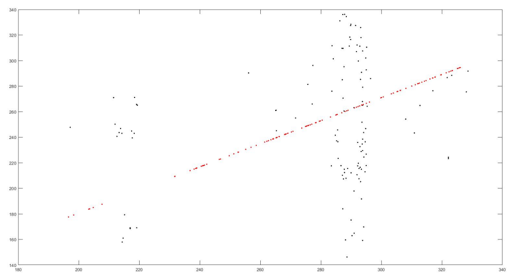

Figure 3: Graph of data and first rank of SVD

Case 2

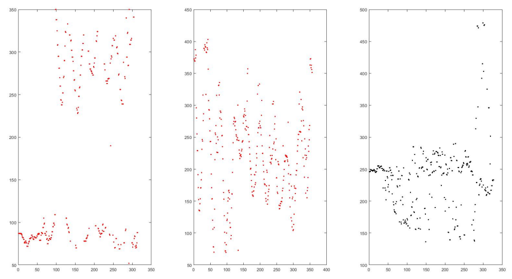

Figure 4: Graph of Y for video 1, 2, and 3

Each case had its own differences, however all the cases showed simple harmonic motion in some form and that accounts for the similarities in SVD decomposition.

Appendix A MATLAB Functions

- `I = rgb2gray(RGB)` converts the truecolor image RGB to the grayscale image I. The rgb2gray function converts RGB images to grayscale by eliminating the hue and saturation information while retaining the luminance. If you have Parallel Computing Toolbox™ installed, rgb2gray can perform this conversion on a GPU.
- `[row,col] = ind2sub(sz,ind)` returns the arrays row and col containing the equivalent row and column subscripts corresponding to the linear indices ind for a matrix of size sz. Here sz is a vector with two elements, where sz(1) specifies the number of rows and sz(2) specifies the number of columns.
- `[U,S,V] = svd(A)` performs a singular value decomposition of matrix A, such that A = U*S*V'.

Appendix B MATLAB Code

Code used for Case 1

```
% Clean workspace
clear all; close all; clc

%% load
load cam1_1.mat;
[height1, width1, rgb1, num_frames1] = size(vidFrames1_1);
```

Case 2

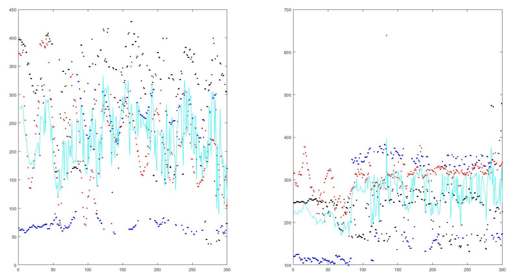

Figure 5: Graph of X and Y with Averages

Case 2

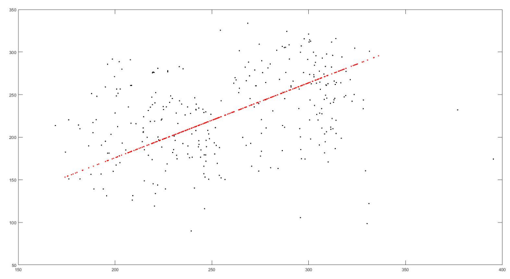

Figure 6: Graph of data and first rank of SVD

```
load cam2_1.mat;
[height2, width2, rgb2, num_frames2] = size(vidFrames2_1);

load cam3_1.mat;
[height3, width3, rgb3, num_frames3] = size(vidFrames3_1);

%% plotting - image 1
for i = 1:num_frames1
    X = rgb2gray(vidFrames1_1(:,:,:,i));

    a = 0;
    M = max(X,[],'all');
    [My, Mx] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x1(i) = median(x_M);
    y1(i) = median(y_M);
end
[~, n] = size(y1);
t1 = 1:n;

figure(1)
subplot(1,3,1)
```

Case 3

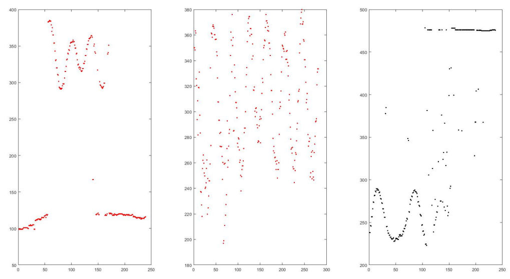

Figure 7: Graph of Y for video 1, 2, and 3

Case 3

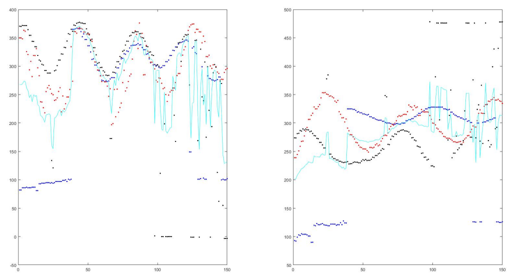

Figure 8: Graph of X and Y with Averages

```
plot(t1, y1, 'r.')

% plotting - image 2

for i = 1:num_frames2
    X = rgb2gray(vidFrames2_1(:,:,:,i));

    M = max(X,[],'all');
    [My, Mx] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x2(i) = mean(x_M);
    y2(i) = mean(y_M);
end
t2 = 1:num_frames2;

subplot(1,3,2)
plot(t2, y2, 'r.')
% plotting - image 3

for i = 1:num_frames3
    X = rgb2gray(vidFrames3_1(:,:,:,i));

    a = 0;
```

Case 3

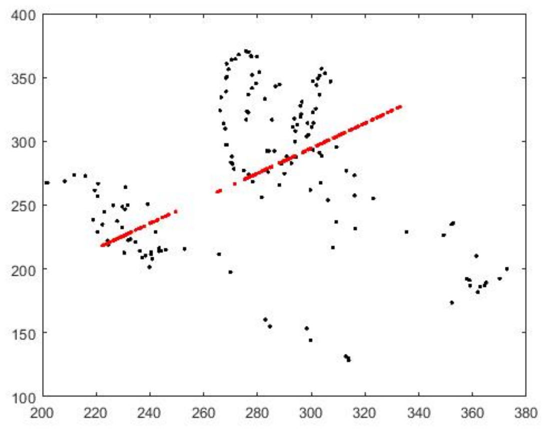

Figure 9: Graph of data and first rank of SVD

Case 4

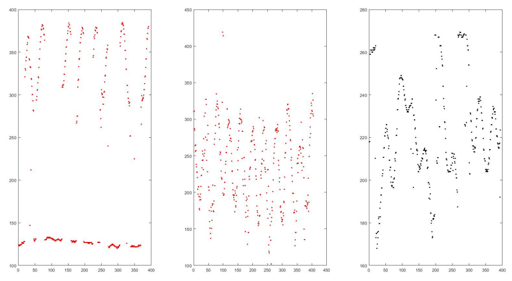

Figure 10: Graph of Y for video 1, 2, and 3

```
    M = max(X,[],'all');
    [Mx, My] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x3(i) = mean(x_M);
    y3(i) = mean(y_M);
end
t3 = 1:num_frames3;

subplot(1,3,3)
plot(t3, x3, 'k.')


y1 = y1 - 10;
y1 = y1(10:129);

y3 = y3 - 10;
y3 = y3(10:129);
```

Case 4

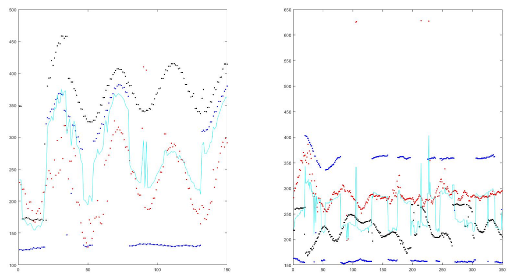

Figure 11: Graph of X and Y with Averages

Case 4

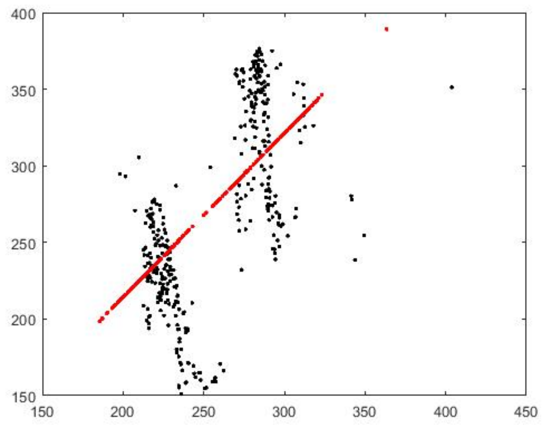

Figure 12: Graph of data and first rank of SVD

```
y2 = y2 - 20;
y2 = y2(20:139);

t = 1:120;

for i = t
    y_avg(i) = y1(i) + y2(i) + y3(i);
    y_avg(i) = y_avg(i)/3;
end


x1 = x1(10:129);
x2 = x2(10:129);
x3 = x3(20:139);

for i = t
    x_avg(i) = x1(i) + x2(i) + x3(i);
    x_avg(i) = x_avg(i)/3;
end
```

```
% plotting
figure(2)
subplot(1, 2, 1)
plot(t, y1, 'b.', t, y2, 'r.', t, y3, 'k.', t, y_avg, 'c');
xlim([0, 120])

subplot(1, 2, 2)
plot(t, x1, 'b.', t, x2, 'r.', t, x3, 'k.', t, x_avg, 'c')


%% SVD
M = [x1; y1; x2; y2; x3; y3];
M = [x_avg; y_avg];
[U, S, V] = svd(M);

rank = 1;
M_rank = U(:,1:rank)*S(1:rank,1:rank)*V(:,1:rank).';

figure(3)
plot(M(1,:), M(2,:), 'k.', M_rank(1,:), M_rank(2,:), 'r.');
```

Code used for Case 2

```
% Clean workspace
clear all; close all; clc

%% load
load cam1_2.mat;
[height1, width1, rgb1, num_frames1] = size(vidFrames1_2);

load cam2_2.mat;
[height2, width2, rgb2, num_frames2] = size(vidFrames2_2);

load cam3_2.mat;
[height3, width3, rgb3, num_frames3] = size(vidFrames3_2);

%% plotting - image 1
for i = 1:num_frames1
    X = rgb2gray(vidFrames1_2(:,:,:,i));

    a = 0;
    M = max(X,[],'all');
    [My, Mx] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x1(i) = median(x_M);
    y1(i) = median(y_M);
end
[~, n] = size(y1);
t1 = 1:n;

figure(1)
```

```
subplot(1,3,1)
plot(t1, y1, 'r.')

% plotting - image 2

for i = 1:num_frames2
    X = rgb2gray(vidFrames2_2(:,:,:,i));

    M = max(X,[],'all');
    [My, Mx] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x2(i) = mean(x_M);
    y2(i) = mean(y_M);
end
t2 = 1:num_frames2;

subplot(1,3,2)
plot(t2, y2, 'r.')

% plotting - image 3

for i = 1:num_frames3
    X = rgb2gray(vidFrames3_2(:,:,:,i));

    a = 0;
    M = max(X,[],'all');
    [Mx, My] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x3(i) = mean(x_M);
    y3(i) = mean(y_M);
end
t3 = 1:num_frames3;

subplot(1,3,3)
plot(t3, x3, 'k.')

% adjustments
y3 = y3 - 15;
y3 = y3(15:314);

y1 = y1 - 15;
y1 = y1(15:314);

y2 = y2(1:300);

t = 1:300;

for i = t
```

```
    y_avg(i) = y1(i) + y2(i) + y3(i);
    y_avg(i) = y_avg(i)/3;
end


x1 = x1(15:314);
x2 = x2(1:300);
x3 = x3(1:300);

for i = t
    x_avg(i) = x1(i) + x2(i) + x3(i);
    x_avg(i) = x_avg(i)/3;
end


% plotting

figure(2)
subplot(1, 2, 1)
plot(t, y1, 'b.', t, y2, 'r.', t, y3, 'k.', t, y_avg, 'c');
xlim([0, 300])

subplot(1, 2, 2)
plot(t, x1, 'b.', t, x2, 'r.', t, x3, 'k.', t, x_avg, 'c')


%% SVD
M = [x_avg; y_avg];
[U, S, V] = svd(M);

rank = 1;
M_rank = U(:,1:rank)*S(1:rank,1:rank)*V(:,1:rank).';

figure(3)
plot(M(1,:), M(2,:), 'k.', M_rank(1,:), M_rank(2,:), 'r.');
```

Code used for Case 3

```
% Clean workspace
clear all; close all; clc

%% load
load cam1_3.mat;
[height1, width1, rgb1, num_frames1] = size(vidFrames1_3);

load cam2_3.mat;
[height2, width2, rgb2, num_frames2] = size(vidFrames2_3);

load cam3_3.mat;
[height3, width3, rgb3, num_frames3] = size(vidFrames3_3);

%% plotting - image 1
for i = 1:num_frames1
    X = rgb2gray(vidFrames1_3(:,:,:,i));
    a = 0;
    M = max(X,[],'all');
    [My, Mx] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x1(i) = median(x_M);
    y1(i) = median(y_M);
end
[~, n] = size(y1);
t1 = 1:n;

figure(1)
subplot(1,3,1)
plot(t1, y1, 'r.')

% plotting - image 2

for i = 1:num_frames2
    X = rgb2gray(vidFrames2_3(:,:,:,i));

    M = max(X,[],'all');
    [My, Mx] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x2(i) = mean(x_M);
    y2(i) = mean(y_M);
end
t2 = 1:num_frames2;

subplot(1,3,2)
plot(t2, y2, 'r.')

% plotting - image 3

for i = 1:num_frames3
    X = rgb2gray(vidFrames3_3(:,:,:,i));

    a = 0;
    M = max(X,[],'all');
    [Mx, My] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x3(i) = mean(x_M);
    y3(i) = mean(y_M);
end
t3 = 1:num_frames3;
```

```
subplot(1,3,3)
plot(t3, x3, 'k.')

% adjustments
y3 = y3 - 8;
y3 = y3(8:157);

y1 = y1 - 18;
y1 = y1(18:167);

y2 = y2(1:150);

t = 1:150;

for i = t
    y_avg(i) = y1(i) + y2(i) + y3(i);
    y_avg(i) = y_avg(i)/3;
end


x1 = x1(18:167);
x2 = x2(1:150);
x3 = x3(8:157);

for i = t
    x_avg(i) = x1(i) + x2(i) + x3(i);
    x_avg(i) = x_avg(i)/3;
end


% plotting
figure(2)
subplot(1, 2, 1)
plot(t, y1, 'b.', t, y2, 'r.', t, y3, 'k.', t, y_avg, 'c');
xlim([0, 150])

subplot(1, 2, 2)
plot(t, x1, 'b.', t, x2, 'r.', t, x3, 'k.', t, x_avg, 'c')


%% SVD
M = [x_avg; y_avg];
[U, S, V] = svd(M);

rank = 1;
M_rank = U(:,1:rank)*S(1:rank,1:rank)*V(:,1:rank).';

figure(3)
plot(M(1,:), M(2,:), 'k.', M_rank(1,:), M_rank(2,:), 'r.');
```

Code used for Case 4

```
% Clean workspace
clear all; close all; clc
```

```
%% load
load cam1_4.mat;
[height1, width1, rgb1, num_frames1] = size(vidFrames1_4);

load cam2_4.mat;
[height2, width2, rgb2, num_frames2] = size(vidFrames2_4);

load cam3_4.mat;
[height3, width3, rgb3, num_frames3] = size(vidFrames3_4);

%% plotting - image 1
for i = 1:num_frames1
    X = rgb2gray(vidFrames1_4(:,:,:,i));

    M = max(X,[],'all');
    [My, Mx] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x1(i) = median(x_M);
    y1(i) = median(y_M);
end
[~, n] = size(y1);
t1 = 1:n;

figure(1)
subplot(1,3,1)
plot(t1, y1, 'r.')

% plotting - image 2

for i = 1:num_frames2
    X = rgb2gray(vidFrames2_4(:,:,:,i));

    M = max(X,[],'all');
    [My, Mx] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x2(i) = mean(x_M);
    y2(i) = mean(y_M);
end
t2 = 1:num_frames2;

subplot(1,3,2)
plot(t2, y2, 'r.')

% plotting - image 3

for i = 1:num_frames3
    X = rgb2gray(vidFrames3_4(:,:,:,i));
    a = 0;
    M = max(X,[],'all');
    [Mx, My] = ind2sub(size(X),find(abs(X) == M));

    x_M = Mx;
    y_M = My;

    x3(i) = mean(x_M);
    y3(i) = mean(y_M);
end
t3 = 1:num_frames3;

subplot(1,3,3)
plot(t3, x3, 'k.')

% adjustments
y3 = y3(1:350);

y1 = y1(1:350);

y2 = y2 - 9;
y2 = y2(9:358);

t = 1:350;

for i = t
    y_avg(i) = y1(i) + y2(i) + y3(i);
    y_avg(i) = y_avg(i)/3;
end


x1 = x1(1:350);
x2 = x2(9:358);
x3 = x3(1:350);

for i = t
    x_avg(i) = x1(i) + x2(i) + x3(i);
    x_avg(i) = x_avg(i)/3;
end


% plotting
figure(2)
subplot(1, 2, 1)
plot(t, y1, 'b.', t, y2, 'r.', t, y3, 'k.', t, y_avg, 'c');
xlim([0, 150])

subplot(1, 2, 2)
plot(t, x1, 'b.', t, x2, 'r.', t, x3, 'k.', t, x_avg, 'c')


%% SVD
M = [x_avg; y_avg];
```

```
[U, S, V] = svd(M);

rank = 1;
M_rank = U(:,1:rank)*S(1:rank,1:rank)*V(:,1:rank).';

plot(M(1,:), M(2,:), 'k.', M_rank(1,:), M_rank(2,:), 'r.');
```