ENME 6215: Final Project

Computer Vision based sensing for structural health monitoring of vibrating systems

Avikal Sagar

(UNI: as6804)

Master of Science - Columbia University (2023)

Computational & Data-Driven Engineering Mechanics

OBJECTIVES

- The main objective of this project is to develop computer vision algorithms that can measure the relative displacement of targets or points when we feed either images or videos into it (also called **template matching**).

- For image datasets, we use integer-pixel method:

❖ **Normalized Cross-Correlation** - it is the inverse Fourier transform of the convolution of the Fourier transform of two (in our case) images, normalized using the local sums and sigmas.

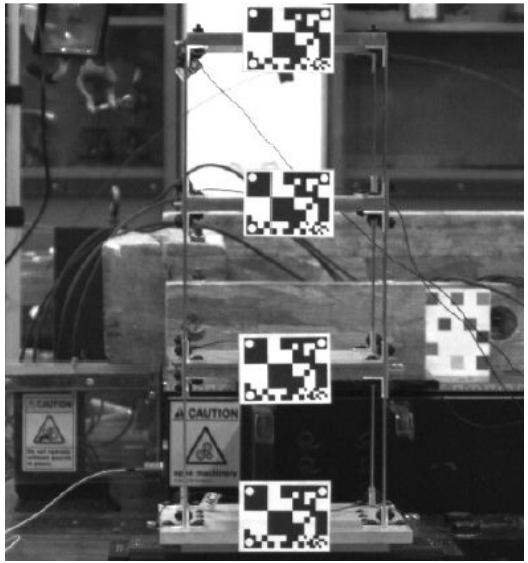
- For video datasets, we use sub-pixel methods:

❖ **Orientation code matching** - itis robust for searching objects in cluttered environments even in the cases of illumination fluctuations resulting from shadowing or highlighting, etc

❖ **Upscaled Cross-Correlation** - first, the Fourier transform of the image is computed, which is then embedded into a larger matrix, and, finally, the inverse transform, representing an upsampled version of the original image, is obtained

EXPERIMENTAL SETUP

- **Structure:** 3-story frame + shaker table
- **Signals:** White noise
- **Number of natural & artificial targets:** 4 each
- **Recorder sampling rate:** 150 fps
- **Video length:** 69 seconds



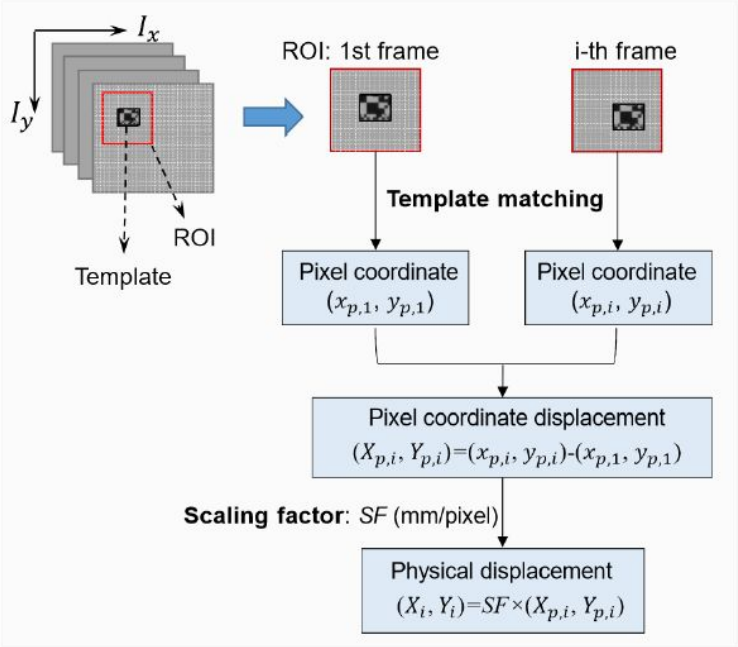
Vibrating structure

Artificial Target

Natural Target (intersection of floor and column)

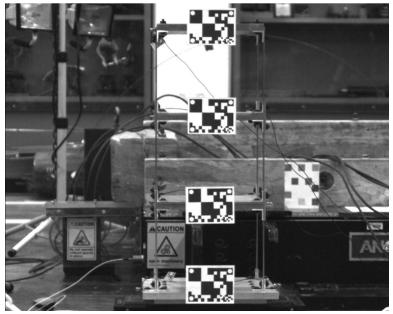
IN THIS PRESENTATION, WE SHALL SEE THE IMPLEMENTATION OF INTEGER-BASED PIXEL METHOD (WITH IMAGES) AND SUB-PIXEL BASED METHODS (FOR VIDEO DATA)

Procedure of 2D vision sensor implementation

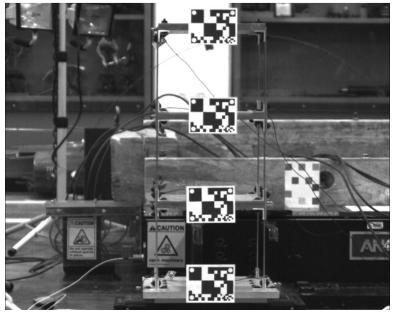


1. INTEGER-PIXEL METHOD - DESCRIPTION

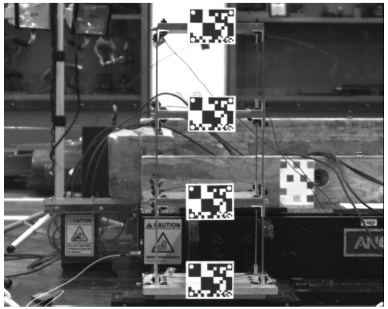
- Dataset: Image (in .png) of the structure and template at every 5 second interval of the video (from 0 to 69 seconds)
- Hence, totally there are 14 images



0th second

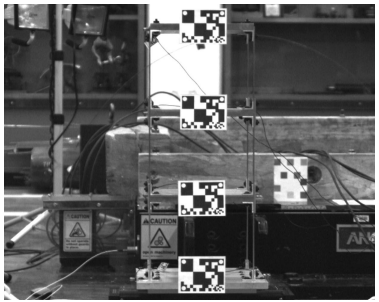


5th second



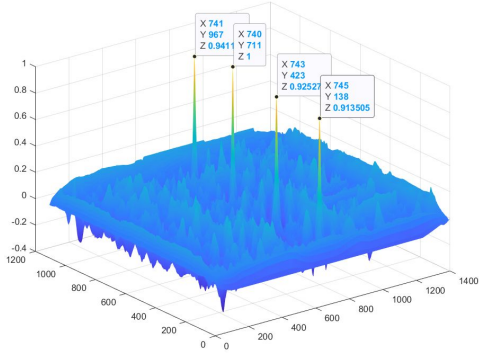
10th second

. . . . . .

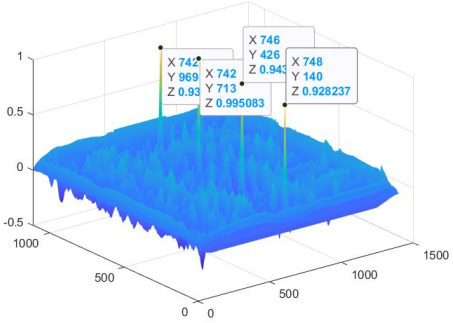


65th second

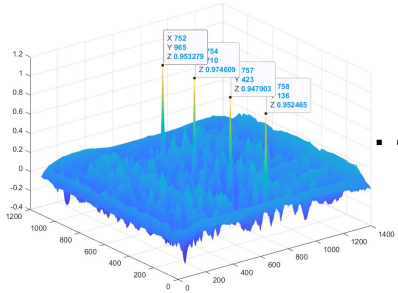
1. INTEGER-PIXEL METHOD - PLOT RESULTS



0th second

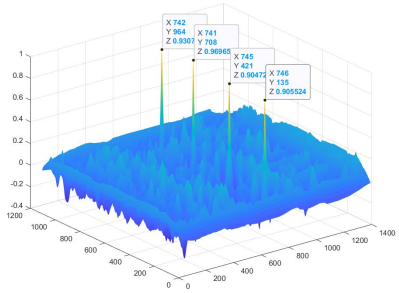


5th second



. . . .

10th second



65th second

Each of the four peaks correspond to the artificial targets that we chose in the MATLAB code. As you can see, at different time instances, the coordinates are slightly different. This shows that:

(1) The white noise induced vibration led to displacement mainly in X-direction with small displacement in Y-direction and negligible displacement in the Z-direction

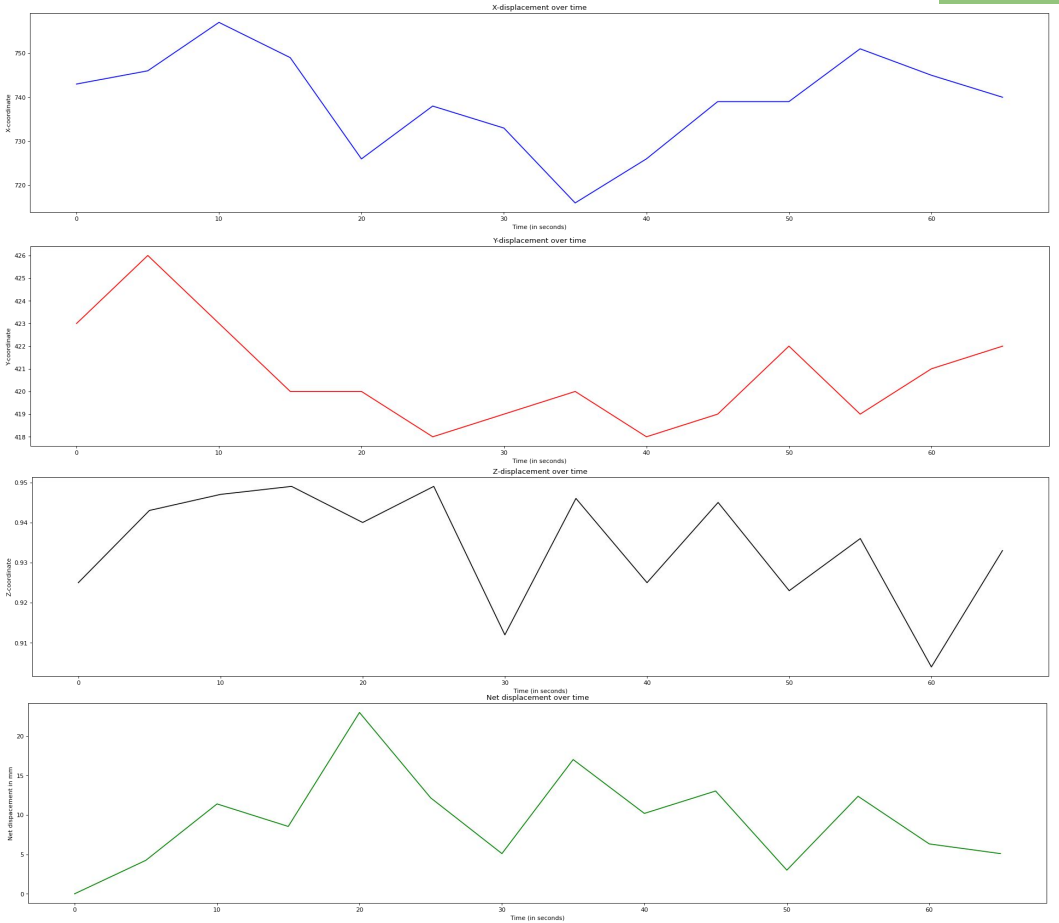
(2) Our algorithm was able to identify even the minute changes in the displacement

1. INTEGER-PIXEL METHOD - TABULAR RESULTS

(Note: this is for artificial target at floor 1. We can repeat it for other floors using same procedure)

A	B	C	D	E	F	G	H
Timeframe (in seconds)	**X**	**Y**	**Z**	**Delta X**	**Delta Y**	**Delta Z**	**Displacement**
0	743	423	0.925	0	0	0	0
5	746	426	0.943	3	3	0.018	4.242678871
10	757	423	0.947	11	-3	0.004	11.40175495
15	749	420	0.949	-8	-3	0.002	8.544003979
20	726	420	0.94	-23	0	-0.009	23.00000176
25	738	418	0.949	12	-2	0.009	12.16552839
30	733	419	0.912	-5	1	-0.037	5.099153753
35	716	420	0.946	-17	1	0.034	17.02942031
40	726	418	0.925	10	-2	-0.021	10.19806065
45	739	419	0.945	13	1	0.02	13.03842015
50	739	422	0.923	0	3	-0.022	3.000080666
55	751	419	0.936	12	-3	0.013	12.36932371
60	745	421	0.904	-6	2	-0.032	6.324636274
65	740	422	0.933	-5	1	0.029	5.09910198

1. INTEGER-PIXEL METHOD - ROUGH TIME SERIES PLOTS



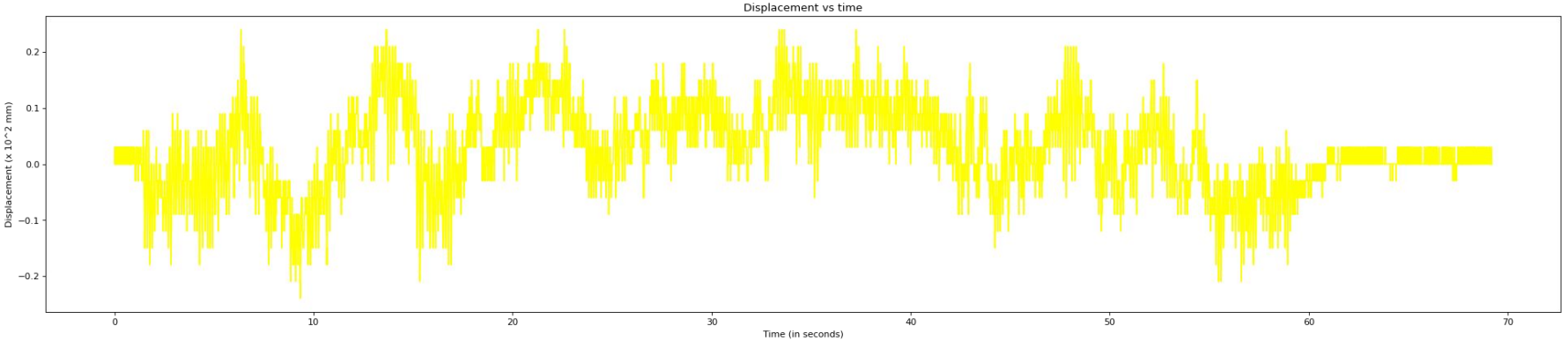
- These are the four very rough plots of the x, y, z and net displacements over time
- Do not worry, we shall see the exact time history curves with sub-pixel methods as they have video data as the input

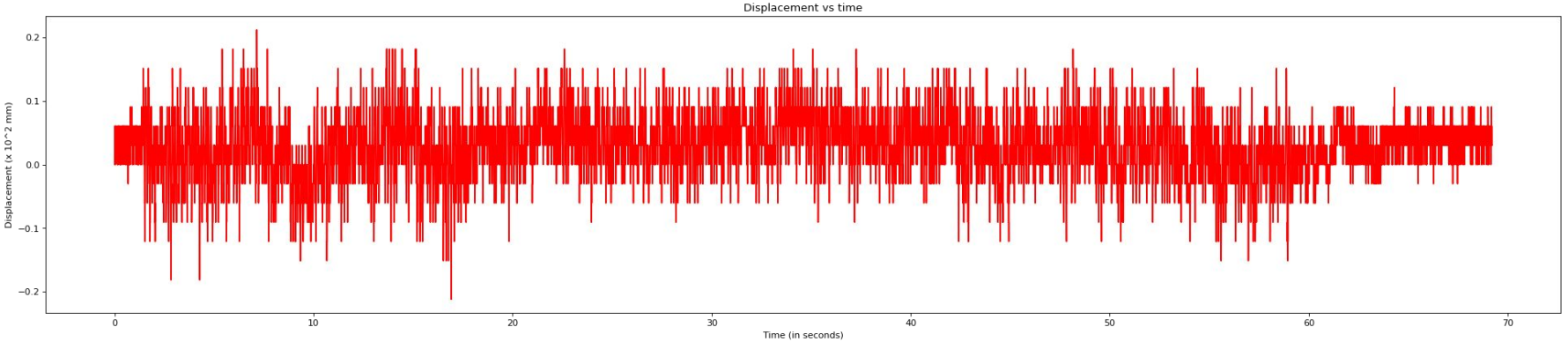
2. SUB-PIXEL METHOD - DESCRIPTION

- Dataset: Entire video of the structure vibrating
- Total number of frames from video: 10382 (Sampling rate * total duration of video)
- Points chosen for understanding real physical distance: diagonally opposite corners of template (124.2 mm)
- Scaling Factor (SF): 1.5146
- Height of video frame in pixels: 512
- Width of video frame in pixels: 10382

2. SUB-PIXEL METHOD - PLOT RESULTS

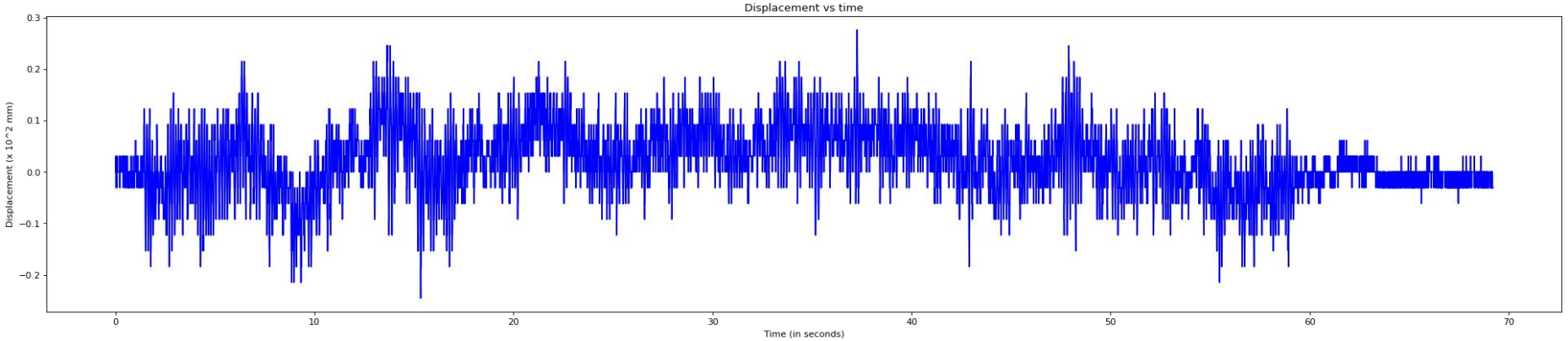
Displacement-time history curves obtained for the four natural targets (base to 3rd floor respectively)

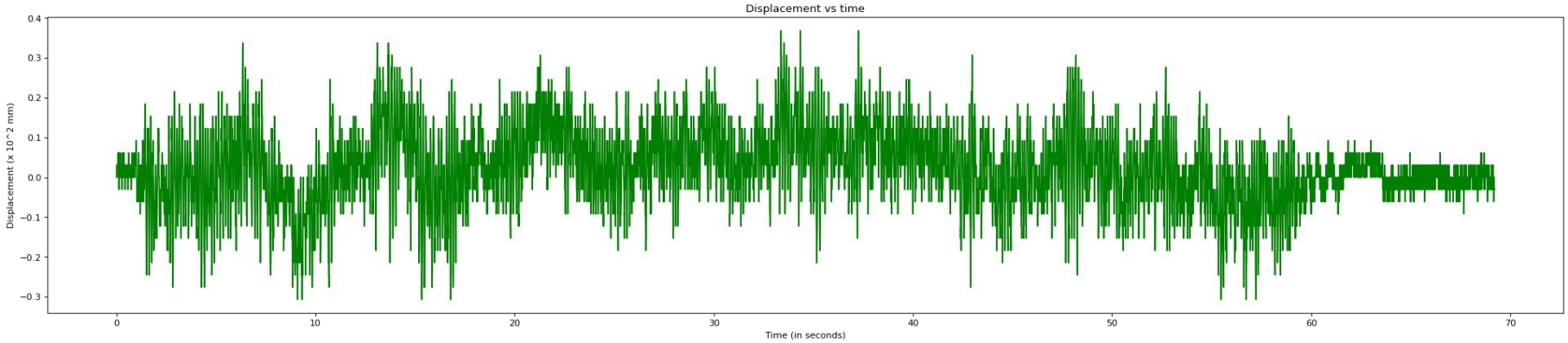




2. SUB-PIXEL METHOD - PLOT RESULTS

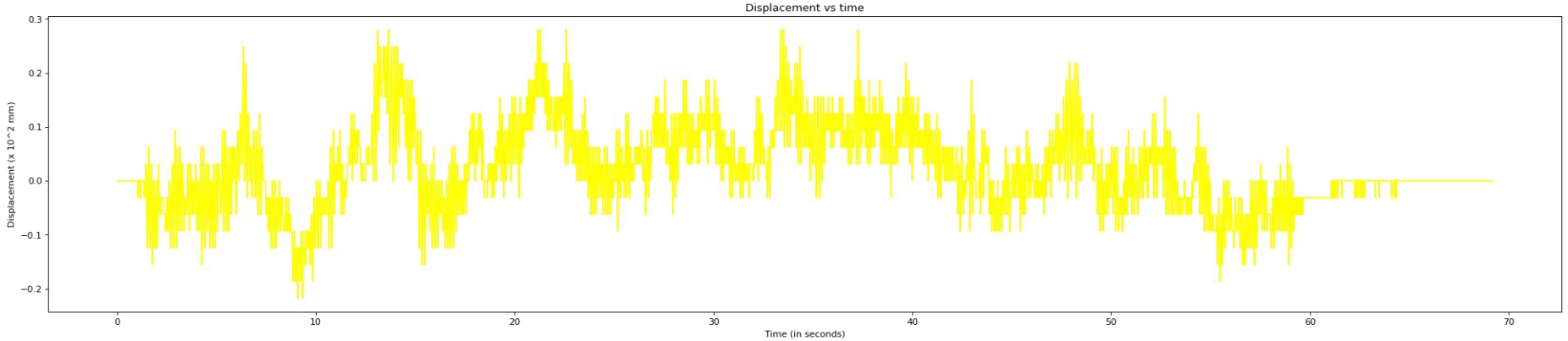
Displacement-time history curves obtained for the four natural targets (base to 3rd floor respectively)

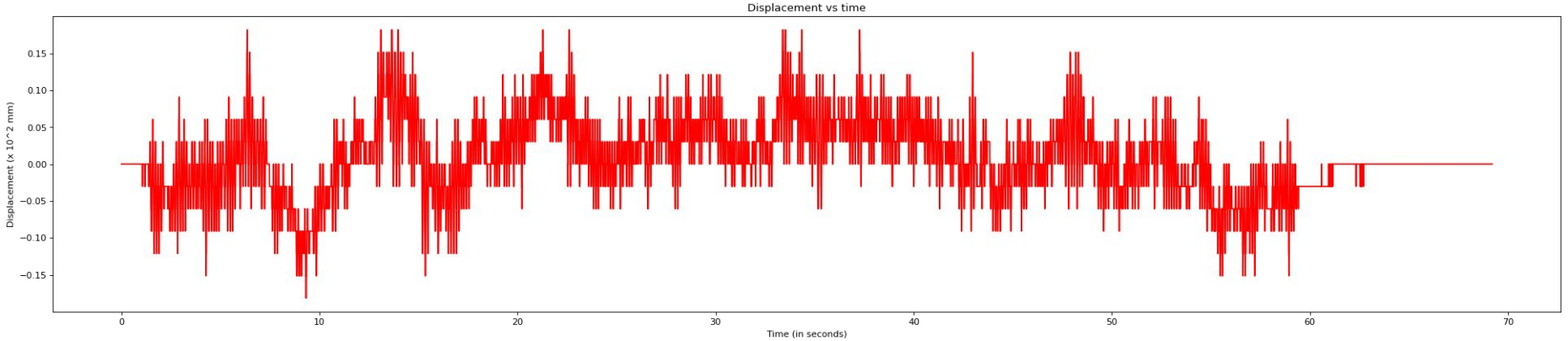




2. SUB-PIXEL METHOD - PLOT RESULTS

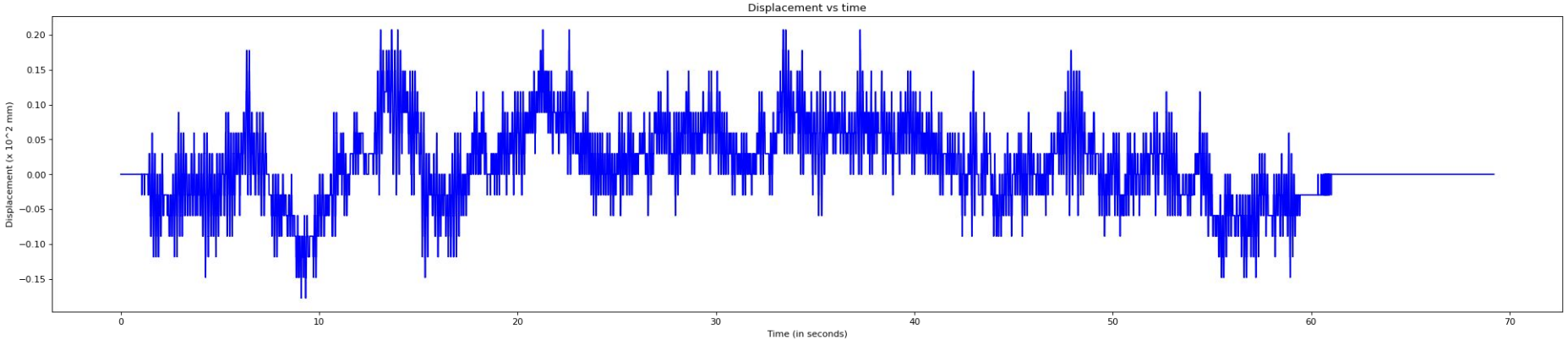
Displacement-time history curves obtained for the four artificial targets (base to 3rd floor respectively)

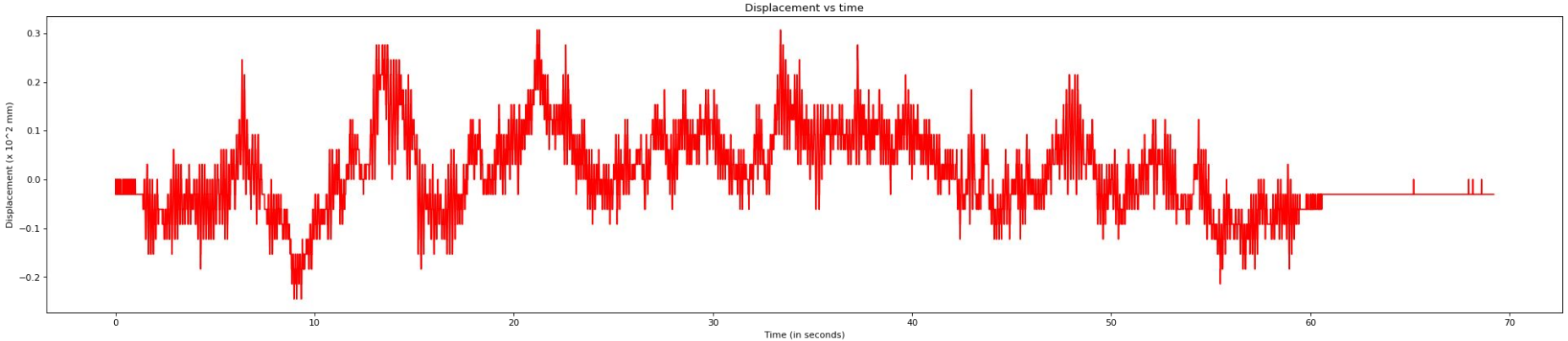




2. SUB-PIXEL METHOD - PLOT RESULTS

Displacement-time history curves obtained for the four artificial targets (base to 3rd floor respectively)



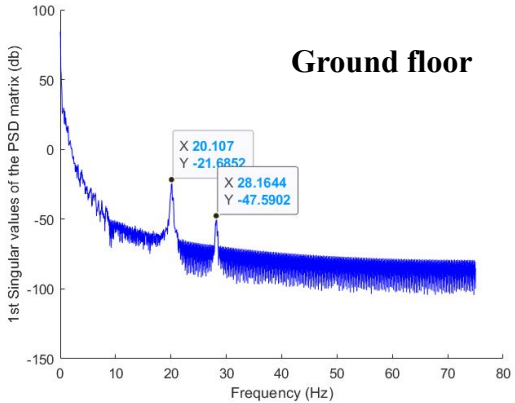


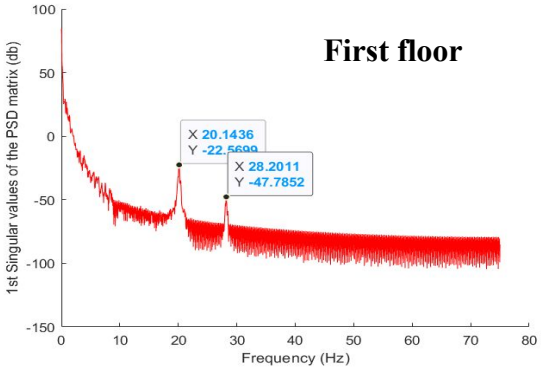
3. RELATIVE ERROR

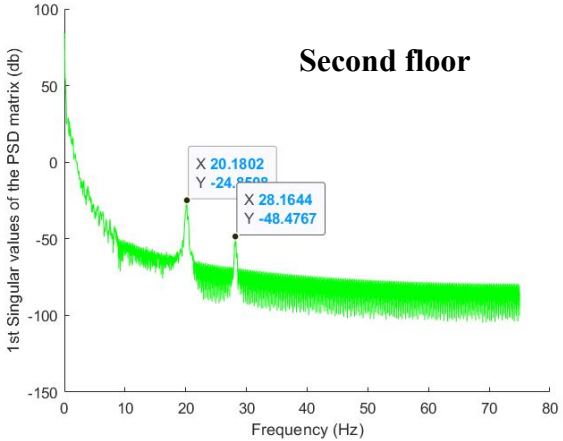
A	B	C	D
Timeframe (in seconds)	**Displacement in mm (from integer-pixel)**	**Displacement in mm (from sub-pixel)**	**Net Error in mm**
0	0	0	0
5	3	6.21	-3.21
10	14	13.41	0.59
15	6	4.17	1.83
20	-17	-14.22	-2.78
25	-5	-2.18	-2.82
30	-10	-12.83	2.83
35	-27	-23.56	-3.44
40	-17	-17.88	0.88
45	-4	-0.75	-3.25
50	-4	-2.72	-1.28
55	8	10.7842	-2.7842
60	2	3.99	-1.99
65	-3	-2	-1
	AVERAGE ERROR = -1.17mm		

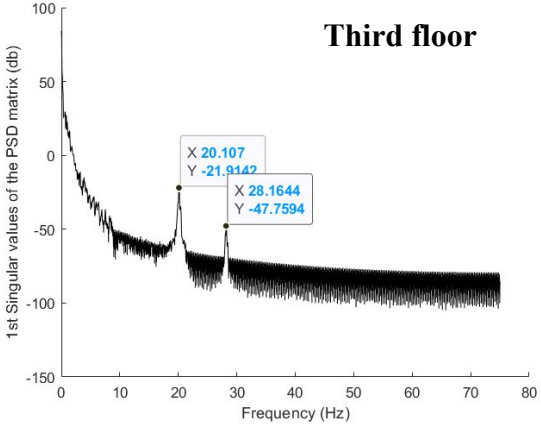
This is the relative error for the displacements obtained of the artificial target at floor 1 by both integer and sub-pixel methods. Same can be repeated for the other floors

4. Fast Fourier Decomposition for artificial targets









5. SUMMARY AND DISCUSSION

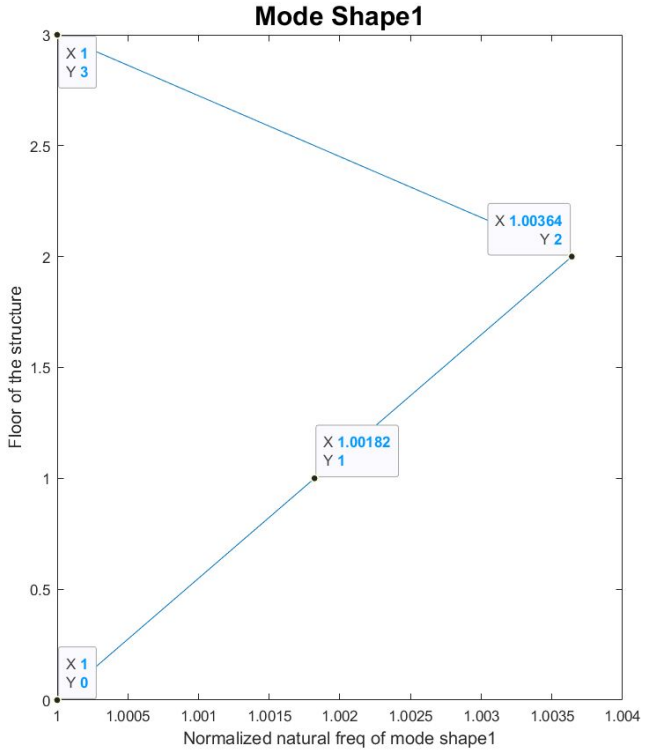
- In sub-pixel method, after observing the time history curves, we observe that the displacements of the natural target and artificial target are nearly identical (with error of +/- 5mm). While comparing the displacements for a particular floor using the integer and sub-pixel method, we observe that the sub-pixel method is more efficient. The average error is -1.17 mm.

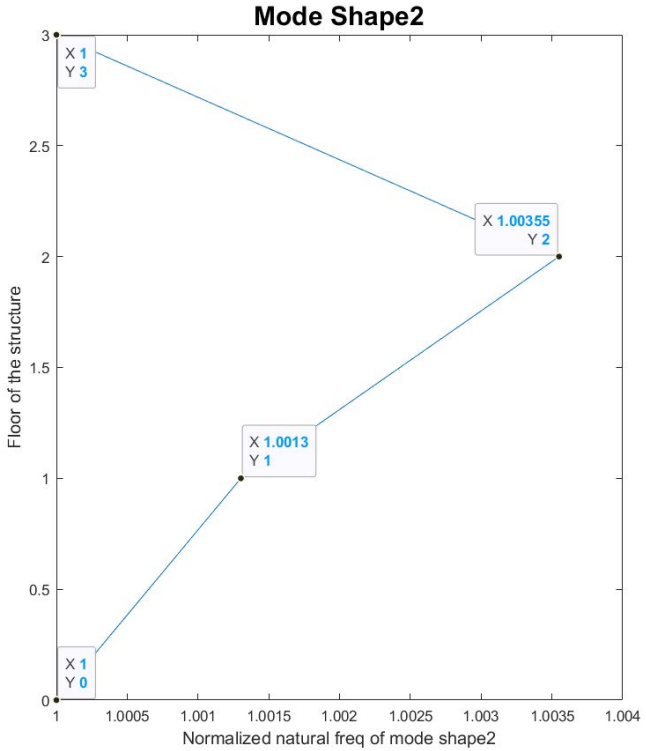
❖ Frequency:

Floor	Mode	Frequencies (Hz)
1	1	20.107
	2	28.165
2	1	20.143
	2	28.201
3	1	20.180
	2	28.164
4	1	20.107
	2	28.164

5. SUMMARY AND DISCUSSION

❖ Mode Shape:





6. APPENDIX: MATLAB CODES

Sub-pixel method (UCC)

```
path = "C:\Users\avika\OneDrive\Desktop\Fall 2022\Prin & App of Sensors\Project\Test1_WhiteNoise1 (1).avi";
obj = VideoReader(path);   %Reads video data from file
N = obj.NumFrames;  %Total number of frames from video frame
h = obj.Height;  %Height of video frame in pixels
w = obj.Width;   % Width of video frame in pixels
Fs = obj.FrameRate;   %Frame rate of video in pixels
dt = 1/Fs;

t = 0:dt:(N-1)*dt;  %Generates tine instants
frame1 = im2double(rgb2gray(read(obj, 1)));
T = imcrop(frame1);
[mt, nt] = size(T);
imshow(frame1);
 [x, y] = ginput(2);
l_known = max(abs(x(2)-x(1)), abs(y(2) - y(1)));
D_known = input('Input the known physical distance (Units: mm)');
SF = D_known/l_known;
px = round(h/20);
py = round(w/20);
ROI_frame1 = frame1(round(y(1) - px): round(y(2) + px), round(x(1) - py): round(x(2) + py), :);
[mr, nr] = size(ROI_frame1);
T(mt + 1: mr,:) = 0;
T(:, nt+1:nr) = 0;

buf2ft = fft2(T);
usfac = 50;

V = zeros(N, 2);
Disp_P = zeros(N, 2);
Disp_S = zeros(N, 2);
for i = 1:N
    framei = im2double(rgb2gray(read(obj, i)));
    ROI_framei = framei(round(y(1) - px): round(y(2) + px), round(x(1) - py): round(x(2) + py));

    buf1ft = fft2(ROI_framei);
    [output, Greg] = dftregistration(buf1ft, buf2ft, usfac);
    V(i,:) = [output(3), output(4)];
    Disp_P(i, :) = V(i, :) - V(1, :);
    Disp_S(i, :) = Disp_P(i, :)*SF;
end

save Displacement.mat Disp_S
```

Integer-based method (NCC)

```
frame_1 = imread('video30.jpg');
frame_2 = imread('video35.jpg');
frame_1gray = rgb2gray(frame_1);
frame_2gray = rgb2gray(frame_2);
T = imcrop(frame_1gray);

c1 = normxcorr2(T, frame_1gray);
c2 = normxcorr2(T, frame_2gray);

figure(1), surf(c1), shading flat
figure(2), surf(c2), shading flat

[ypeak1, xpeak1] = find(c1==max(c1(:)));
[ypeak2, xpeak2] = find(c2==max(c2(:)));
```

6. APPENDIX: MATLAB CODES

Frequency domain decomposition

```
function [Frq,phi]=FDD(Input,Fs)
close all
Acc=xlsread("FDD_UCC_6_floor.xlsx");
display('FDD is in progress, please wait ...')
% --------------------------------------------------------------------
% Compute Power Spectral Density (PSD) matrix.
% CPSD function, with default settings, is used to compute the cross power
% spectral density matrix. More sophisticated methods can also be
% applied for more accuracy.
for I=1:size(Acc,2)
    for J=1:size(Acc,2)
        [PSD(I,J,:),F(I,J,:)]=cpsd(Acc(:,I),Acc(:,J),[],[],[],150.015);
    end
end
Frequencies(:,1)=F(1,1,:);

[Frq,phi,Fp,s1] = Identifier(PSD,Frequencies);
% Save results
save('IdResults.mat','phi','Fp','s1','Frequencies')
% --------------------------------------------------------------------
% Print results
display('-----------------------------------------------------')
display('              Identification Results                 ')
display('-----------------------------------------------------')
% Print frequencies
display('Identified frequencies')
for I=1:size(Frq,1)
    fprintf('Mode: %d; Modal Frequency: %6.4g (Hz)\n',I,Frq(I))
end
% Print Mode shapes
display('Related mode shapes')
for I=1:size(Frq,1)
    fprintf('Mode shape # %d:\n\n',I)
    disp(phi(:,I))
end
end

function [Frq,phi,Fp,s1] = Identifier(PSD,F)
% Compute SVD of the PSD at each frequency
for I=1:size(PSD,3)
    [u,s,~] = svd(PSD(:,:,I));
    s1(I) = s(1);                                   % First eigen values
    s2(I) = s(2,2);                                 % Second eigen values
    ms(:,I)=u(:,1);                                 % Mode shape
end
% Plot first singular values of the PSD matrix
figure
hold on
plot(F,mag2db(s1), color = 'blue')
xlabel('Frequency (Hz)')
ylabel('1st Singular values of the PSD matrix (db)')
end
```

ENME 6215: Final Project

THANK YOU!

Avikal Sagar

(UNI: as6804)

Master of Science - Columbia University (2023)

Computational & Data-Driven Engineering Mechanics