ASSIGNMENT 2

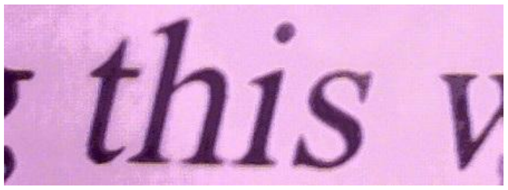
(2018702012)

ALGORITHM:

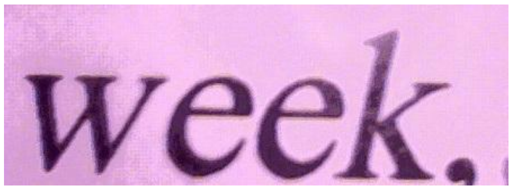
1) Set dirName to the name of the folder containing all the images to be stitched together.
2) Set I_2 to the first image in the folder.
3) Set i = 1 and Repeat steps 4 to 11 until i = Number of images in dirName - 1
4) Set I_1 = next image in the folder.
5) Convert the I_1 and I_2 to RGB.
6) Using SIFT, find corresponding matching points.
7) Estimate Homography matrix that transforms I_2 to I_1.
8) Set I = padarray(I_2, size(I_1), 'both').
9) For each row and column in I, find corresponding point in I_1 and assign the intensity of I_1 at that point to the corresponding location in I.
10) Post process the image to remove unwanted black rectangles.
11) Set I_2 = I.
12) Final stitched image is I.

CHALLENGES:

1. The stitched image is dependent on the order of the input images given.
2. Seldom homography estimated by the Ransac algorithm might not be good enough. Hence output might look bad. This situation can be avoided by increasing the number of iterations or reducing the error threshold.
3. The stitched image is very poor if the two images to be stitched have considerably less matching points. Like the following images



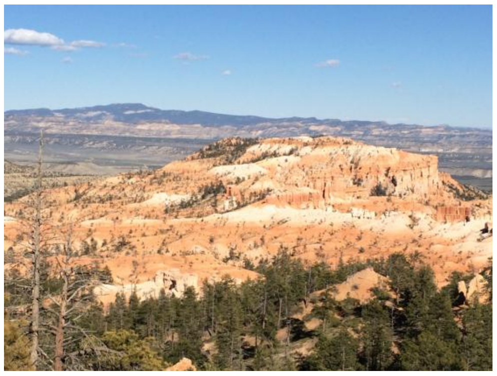
this v

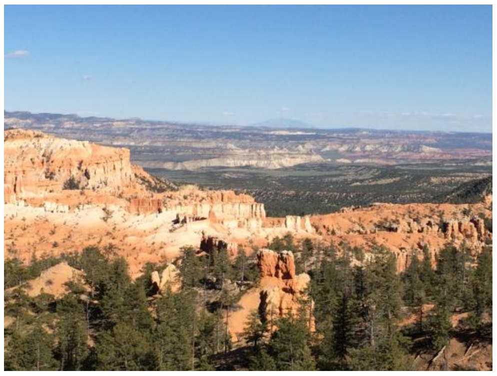


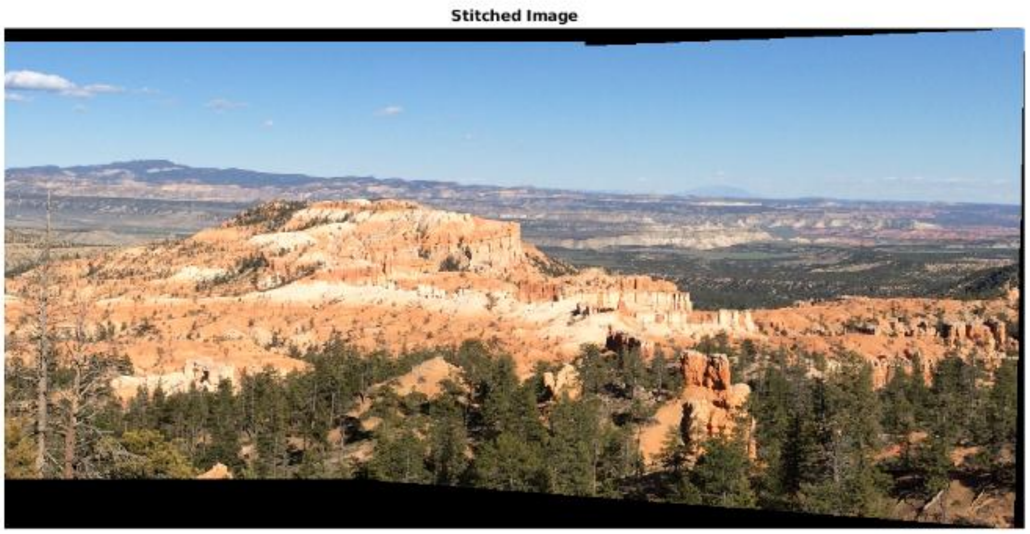
week.

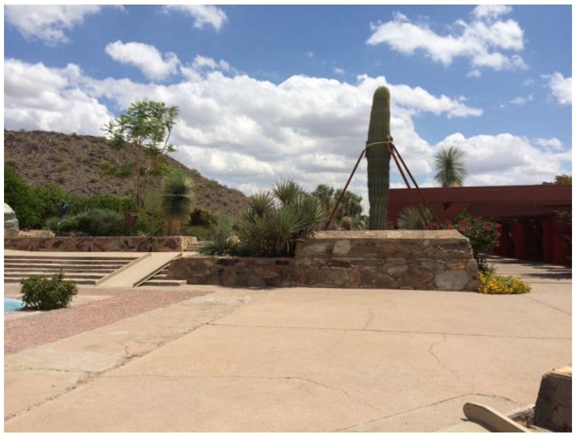
RESULTS (SUCCESS):

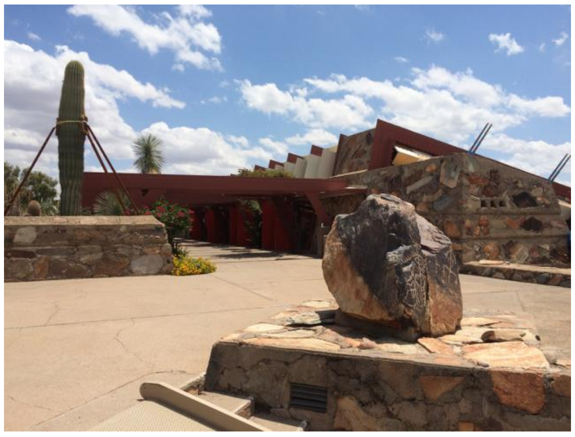
The algorithm gives following results for the given datasets:



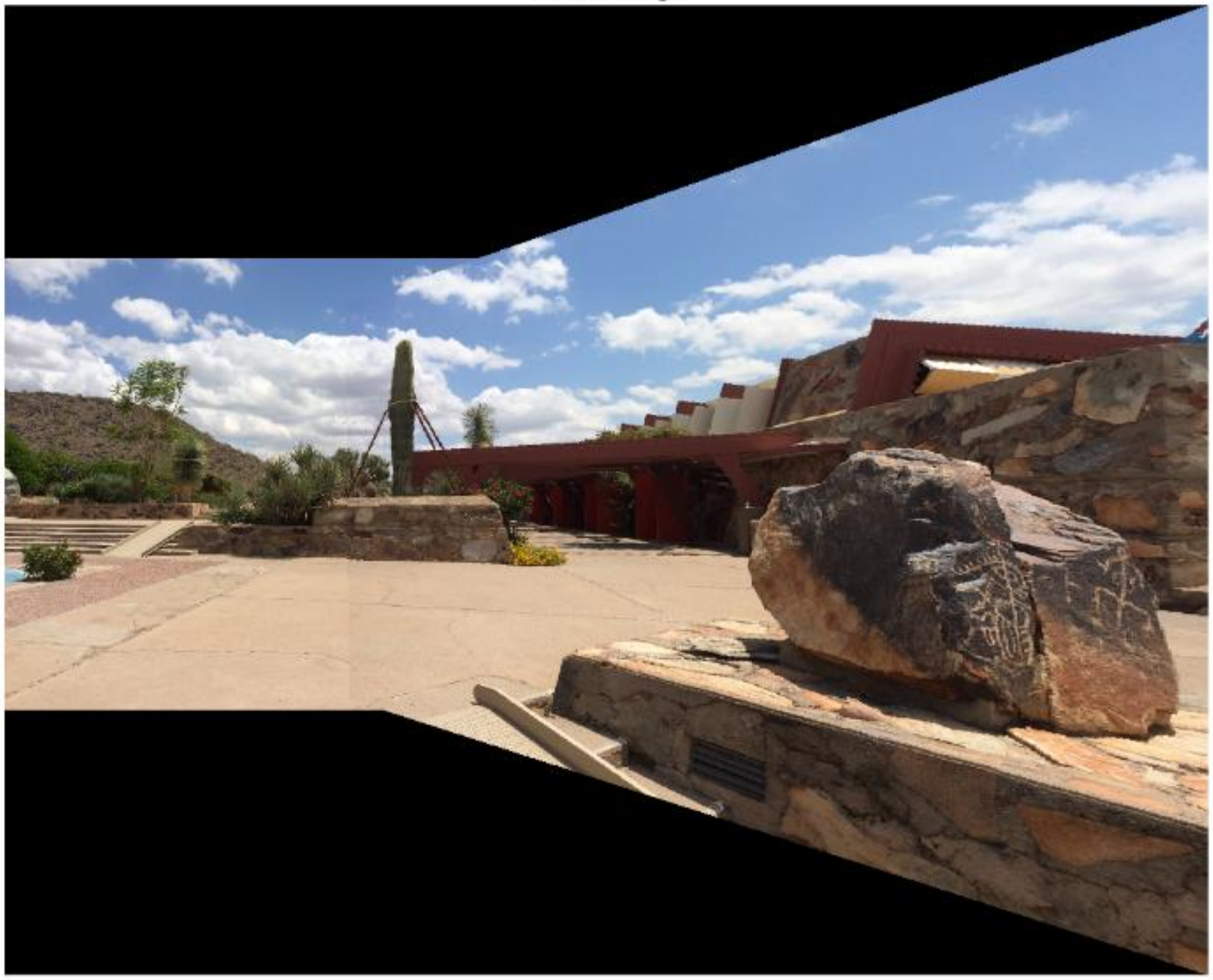


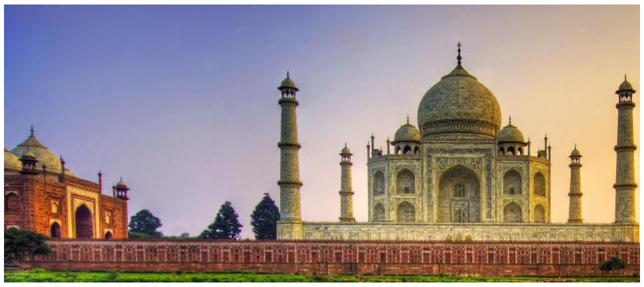


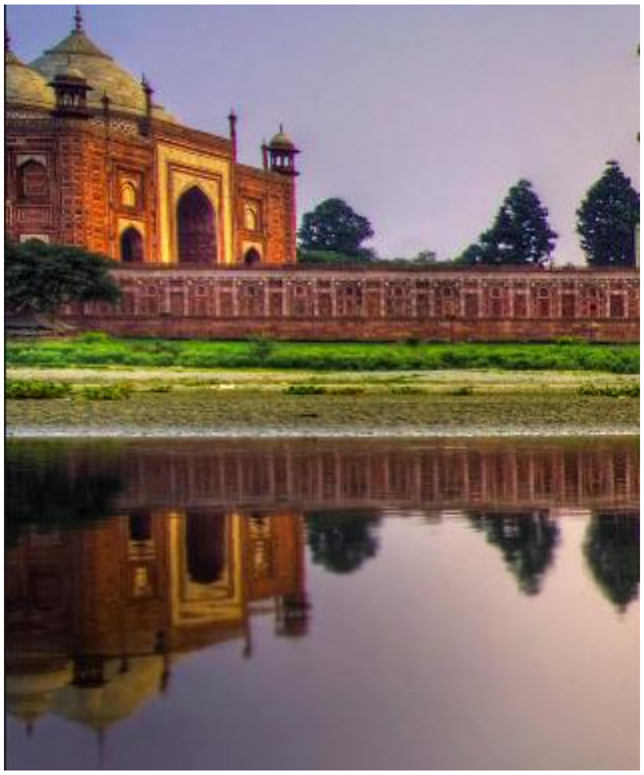




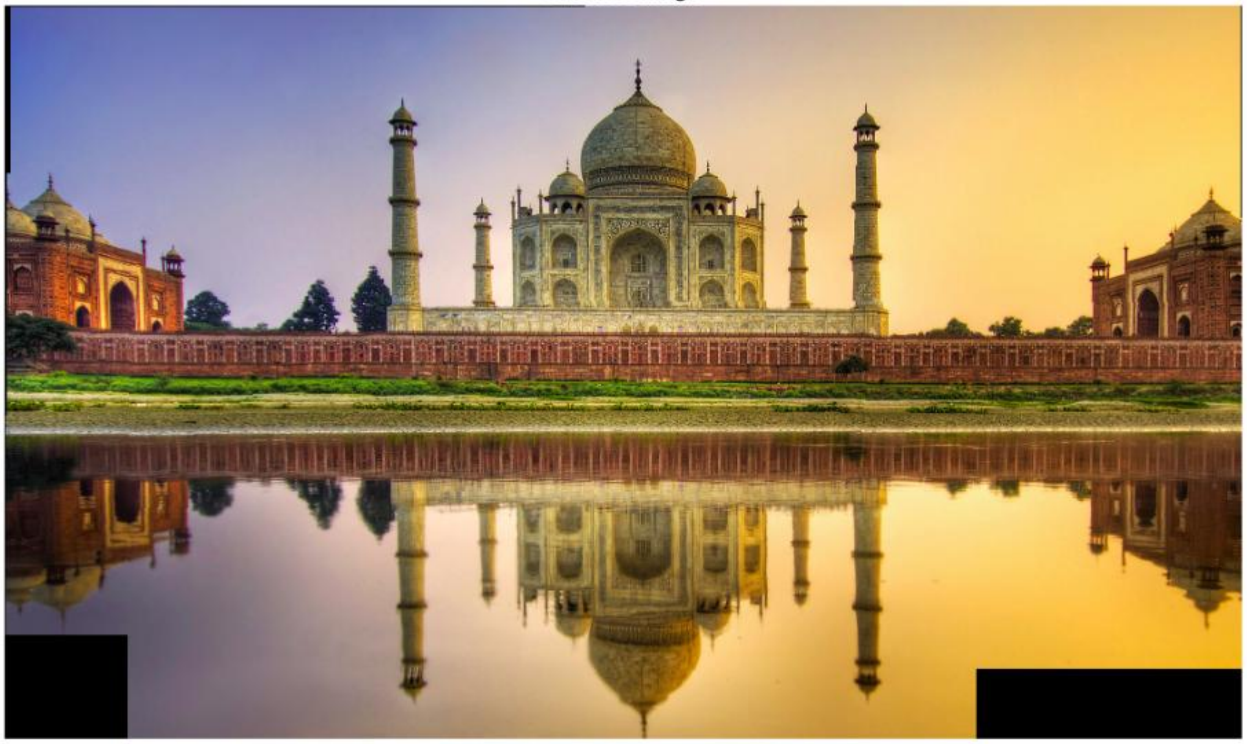
Stitched Image



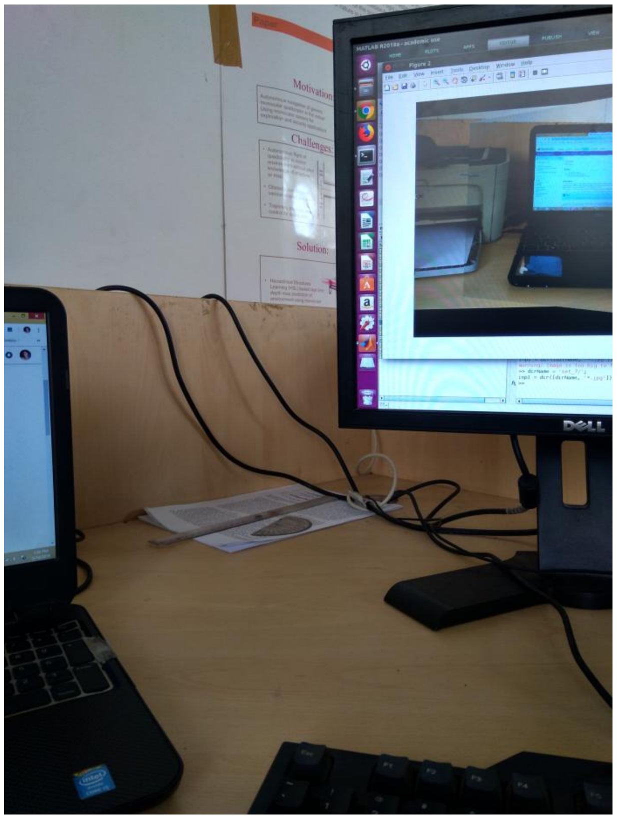


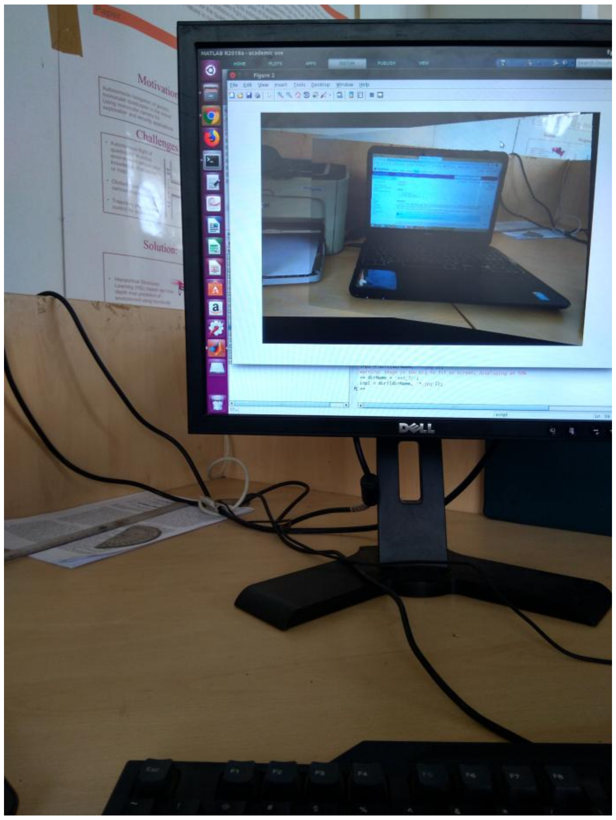


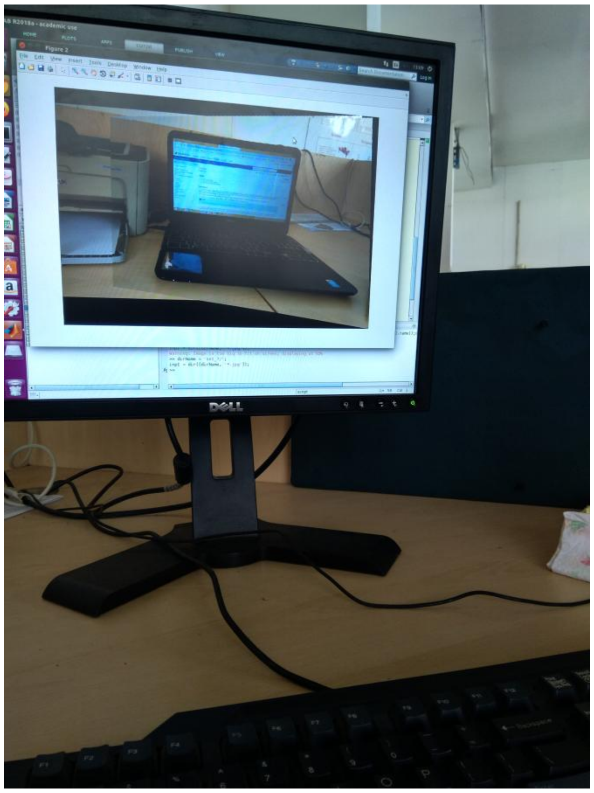
Stitched Image



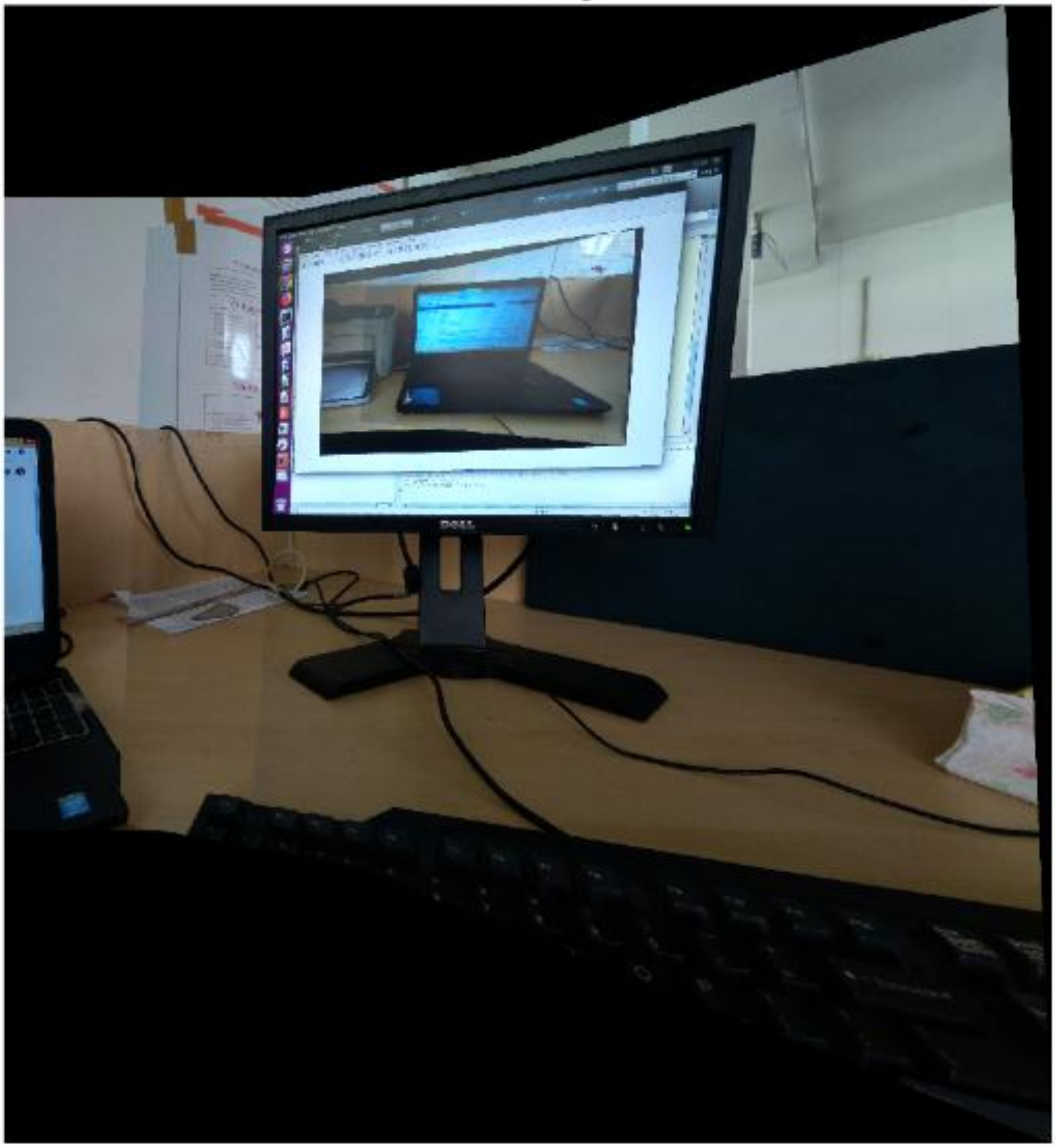
Results of the images taken by my camera:

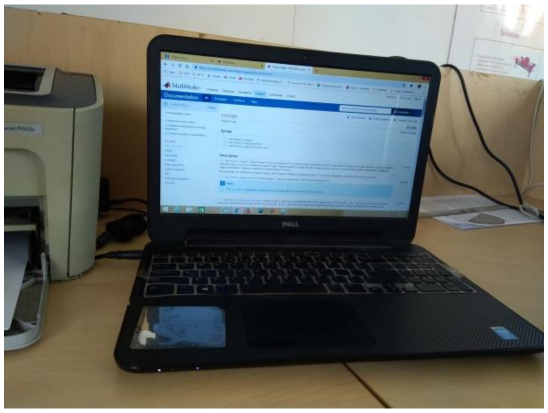


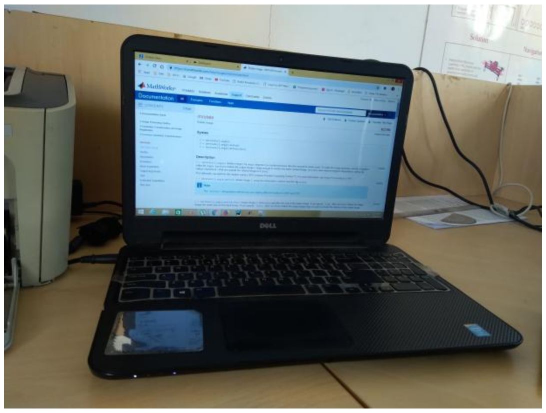




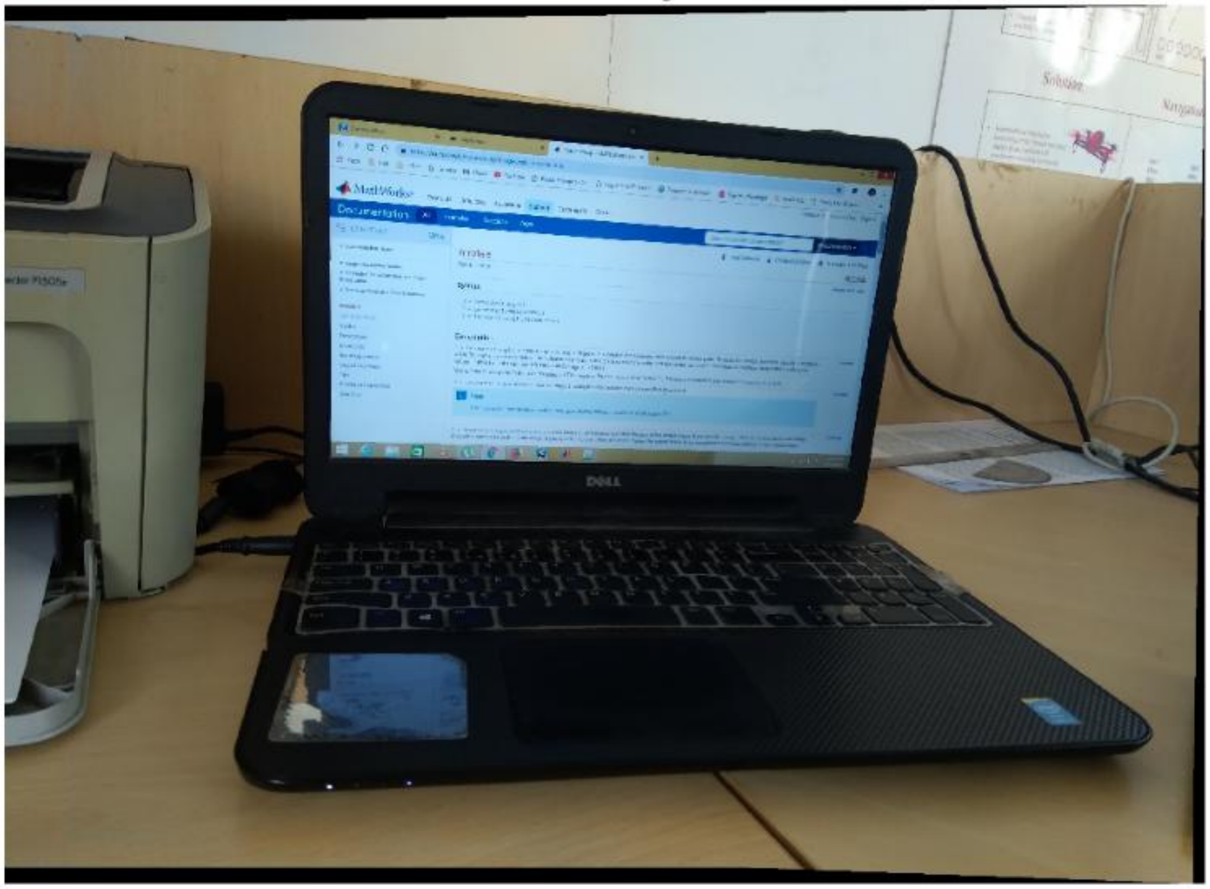
Stiched image



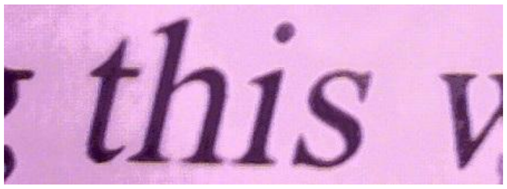


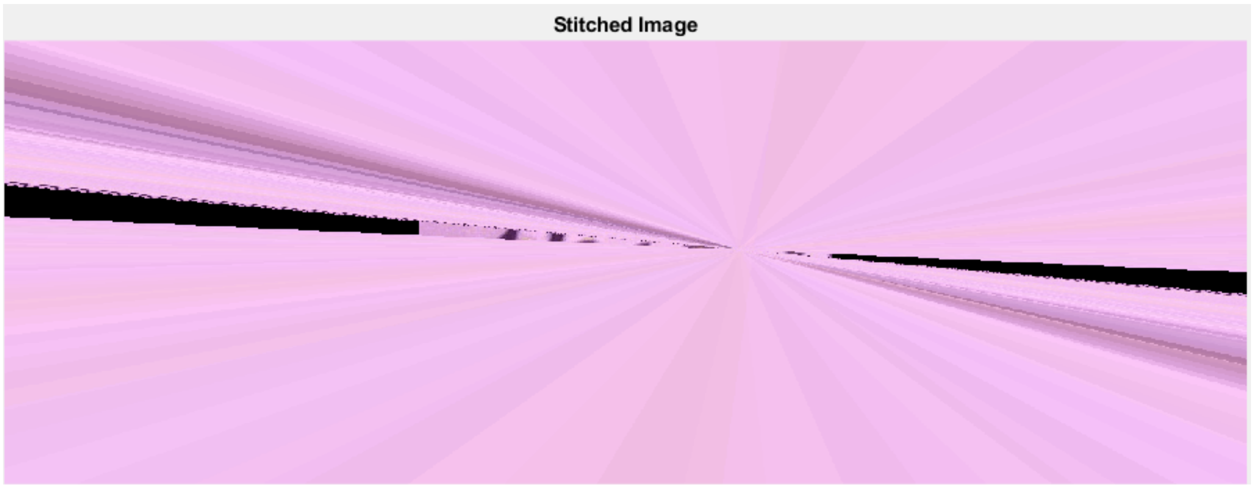


Stitched Image



RESULTS (FAILURE):





CODE:

1) For image stitching

```
clear all;
close all;
clc;

% Include vlfeat
run('C:/Users/TANU/Documents/Softwares added/vlfeat-0.9.21-bin/vlfeat-
0.9.21/toolbox/vl_setup');

% Name of the directory containing input images. Image types considered are
% either "*.png" or "*.jpg".
dirName = 'set_3/';
inpI = dir([dirName, '*.png']);

if isempty(inpI)
    inpI = dir([dirName, '*.jpg']);
end

I_2 = imread(inpI(1).name);

for i = 1:length(inpI)-1
    I_1 = imread(inpI(i+1).name);

    % Find matching points
    I1 = single(rgb2gray(I_1));
    I2 = single(rgb2gray(I_2));
    [f1, d1] = vl_sift(single(I1));
    [f2, d2] = vl_sift(single(I2));
    [matches, scores] = vl_ubcmatch(d1, d2);
    pts1 = f1(1:2,matches(1,:));
    pts2 = f2(1:2,matches(2,:));

    % Find Homography matrix
    H21 = estimateHomographyUsingRansac(pts1', pts2');

    % Stitch the two images
    Istitch = stichImages(I_1, I_2, H21);
    I_2 = Istitch;
end

% Show the stitched image
imshow(I_2); title('Stitched Image');


function H = estimateHomographyUsingRansac(pts1, pts2)
%ESTIMATEHOMOGRAPHYUSINGRANSAC Applies RANSAC to estimate best homography
%matrix between the two set of image points.
% Inputs:
%   pts1 = Points of image 1. Size (len X 2)
%   pts2 = Points of image 2. Size (len X 2)
```

```
    % RANSAC
    itrs = 100;
    n = 4;
    errThr = 0.05;
    h = ones(itrs, 3, 3);
    inliers = ones(itrs, 1);
    len = length(pts1);

    for itr = 1:itrs
        % Select points randomly
        idx = randperm(numel(1:size(pts1, 1)));
        pts = idx(1:n);

        % Calculate projection matrix
        h(itr, :, :) = calcHomographyMatrix(pts1(pts,:), pts2(pts,:));

        % Calculate reprojection error
        gen_pts1 = reshape(h(itr,:,:), 3, 3) * [pts2, ones(len,1)]';
        gen_pts1 = gen_pts1./gen_pts1(3,:);
        err = pts1 - gen_pts1(1:2,:)';
        err = err.^2;
        err = sqrt(sum(err(:,1:2), 2));

        inliers(itr) = size(find(err < errThr),1);
    end

    % Find best intrinsic and extrinsic parameters
    [~, max_index] = max(inliers);

    H = reshape(h(max_index, :, :), 3, 3);

end

function I = stichImages(I1, I2, H21)
%STICHIMAGES Summary of this function goes here
%   Detailed explanation goes here

    r = size(I1,1); c = size(I1,2);
    I = I2;
    I = padarray(I, [r c], 'both');

    for row = 1:size(I,1)
        for col = 1:size(I,2)
            x = H21 * [col-c; row-r; 1];
            x = floor(x/x(3));
            if (1 <= x(2) && x(2) <= r && 1 <= x(1) && x(1) <= c)
                    I(row,col,:) = I1(x(2), x(1), :);
            end
        end
    end


    % Post process stitched image to remove dark regions.
    gI = rgb2gray(I);
    [r,c] = find(gI~=0);
    t = [r,c];
    minR = min(t(:,1));    maxR = max(t(:,1));
    minC = min(t(:,2));    maxC = max(t(:,2));

    I = I(minR:maxR,minC:maxC,:);

end
```