Assignment 1

1. **Camera Calibration:**

For camera calibration we have chosen a chess board of a square size 19.50 mm and placed it at a distance of 939.8 mm

Calibrating the camera using matlab.

% Auto-generated by cameraCalibrator app on 25-Sep-2022

%-------------------------------------------------------

% Define images to process

imageFileNames = {'E:\GSU\CV\new\_worked imqgew\image1.jpg',...

'E:\GSU\CV\new\_worked imqgew\image2.jpg',...

'E:\GSU\CV\new\_worked imqgew\image3.jpg',...

'E:\GSU\CV\new\_worked imqgew\image4.jpg',...

'E:\GSU\CV\new\_worked imqgew\image5.jpg',...

'E:\GSU\CV\new\_worked imqgew\image6.jpg',...

'E:\GSU\CV\new\_worked imqgew\image7.jpg',...

'E:\GSU\CV\new\_worked imqgew\image10.jpg',...

'E:\GSU\CV\new\_worked imqgew\image13.jpg',...

'E:\GSU\CV\new\_worked imqgew\image14.jpg',...

'E:\GSU\CV\new\_worked imqgew\image15.jpg',...

'E:\GSU\CV\new\_worked imqgew\image18.jpg',...

'E:\GSU\CV\new\_worked imqgew\image19.jpg',...

'E:\GSU\CV\new\_worked imqgew\image20.jpg',...

};

% Detect calibration pattern in images

detector = vision.calibration.monocular.CheckerboardDetector();

[imagePoints, imagesUsed] = detectPatternPoints(detector, imageFileNames);

imageFileNames = imageFileNames(imagesUsed);

% Read the first image to obtain image size

originalImage = imread(imageFileNames{1});

[mrows, ncols, ~] = size(originalImage);

% Generate world coordinates for the planar pattern keypoints

squareSize = 1.950000e+01; % in units of 'millimeters'

worldPoints = generateWorldPoints(detector, 'SquareSize', squareSize);

% Calibrate the camera

[cameraParams, imagesUsed, estimationErrors] = estimateCameraParameters(imagePoints, worldPoints, ...

'EstimateSkew', false, 'EstimateTangentialDistortion', false, ...

'NumRadialDistortionCoefficients', 2, 'WorldUnits', 'millimeters', ...

'InitialIntrinsicMatrix', [], 'InitialRadialDistortion', [], ...

'ImageSize', [mrows, ncols]);

% View reprojection errors

h1=figure; showReprojectionErrors(cameraParams);

% Visualize pattern locations

h2=figure; showExtrinsics(cameraParams, 'CameraCentric');

% Display parameter estimation errors

displayErrors(estimationErrors, cameraParams);

% For example, you can use the calibration data to remove effects of lens distortion.

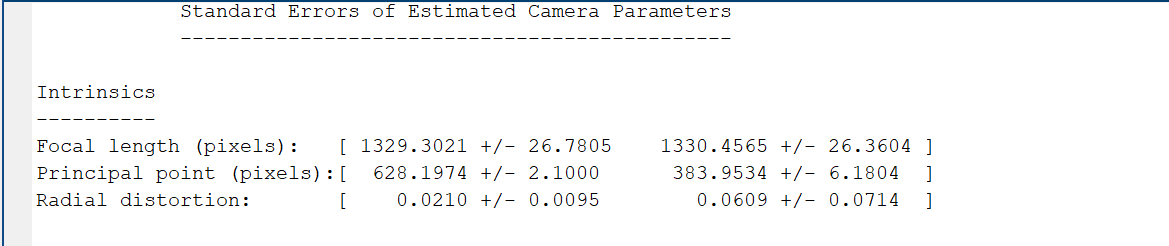
undistortedImage = undistortImage(originalImage, cameraParams);

% See additional examples of how to use the calibration data. At the prompt type:

% showdemo('MeasuringPlanarObjectsExample')

% showdemo('StructureFromMotionExample')

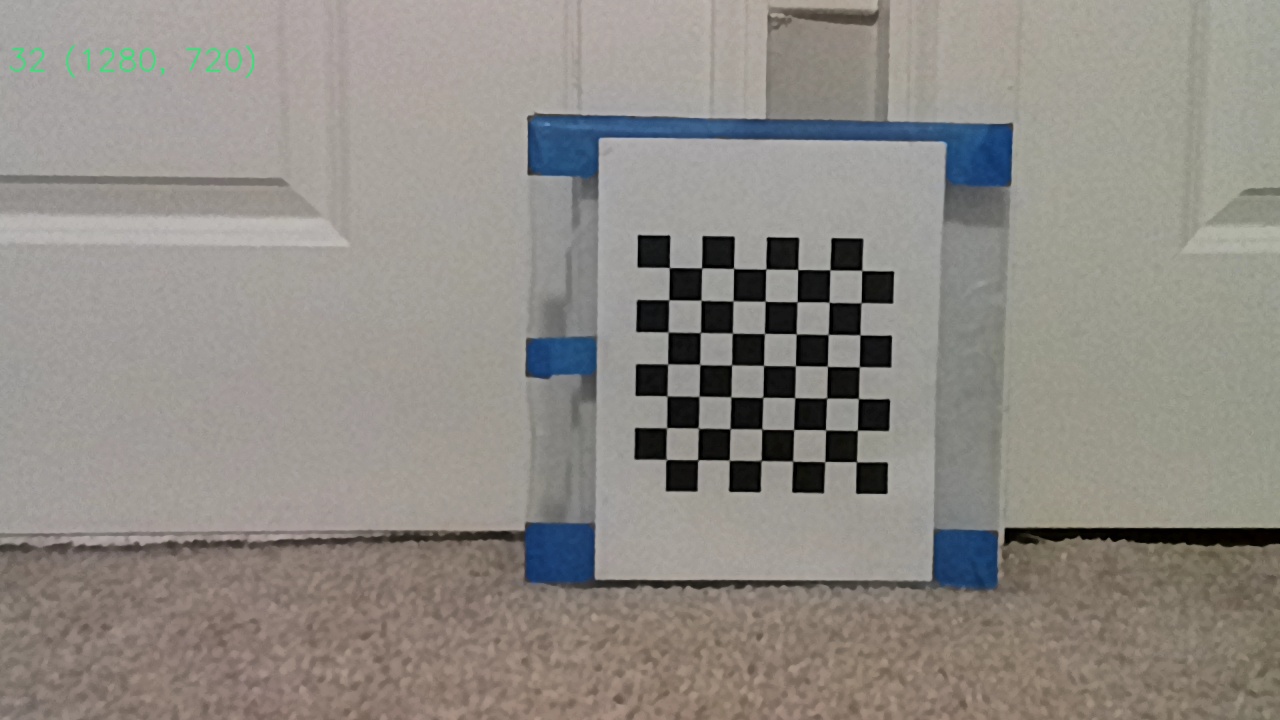
**Output:**



**Part B**

**Question 2**

**Code:**



I=imread("E:\GSU\CV\new\_worked imqgew\image1.jpg")

imshow(I)

[x,y]=ginput(2)

z\_dist=939.8

fx=1329.3020

fy=1330.4565

x1=z\*(x(1)/fx)

x2=z\*(x(2)/fx)

y1=z\*(y(1)/fy)

y2=z\*(y(2)/fy)

distance=sqrt((y2-y1)^2+(x2-x1)^2)

fprintf("Estimated distance B/W the 2 points",distance)

O/P

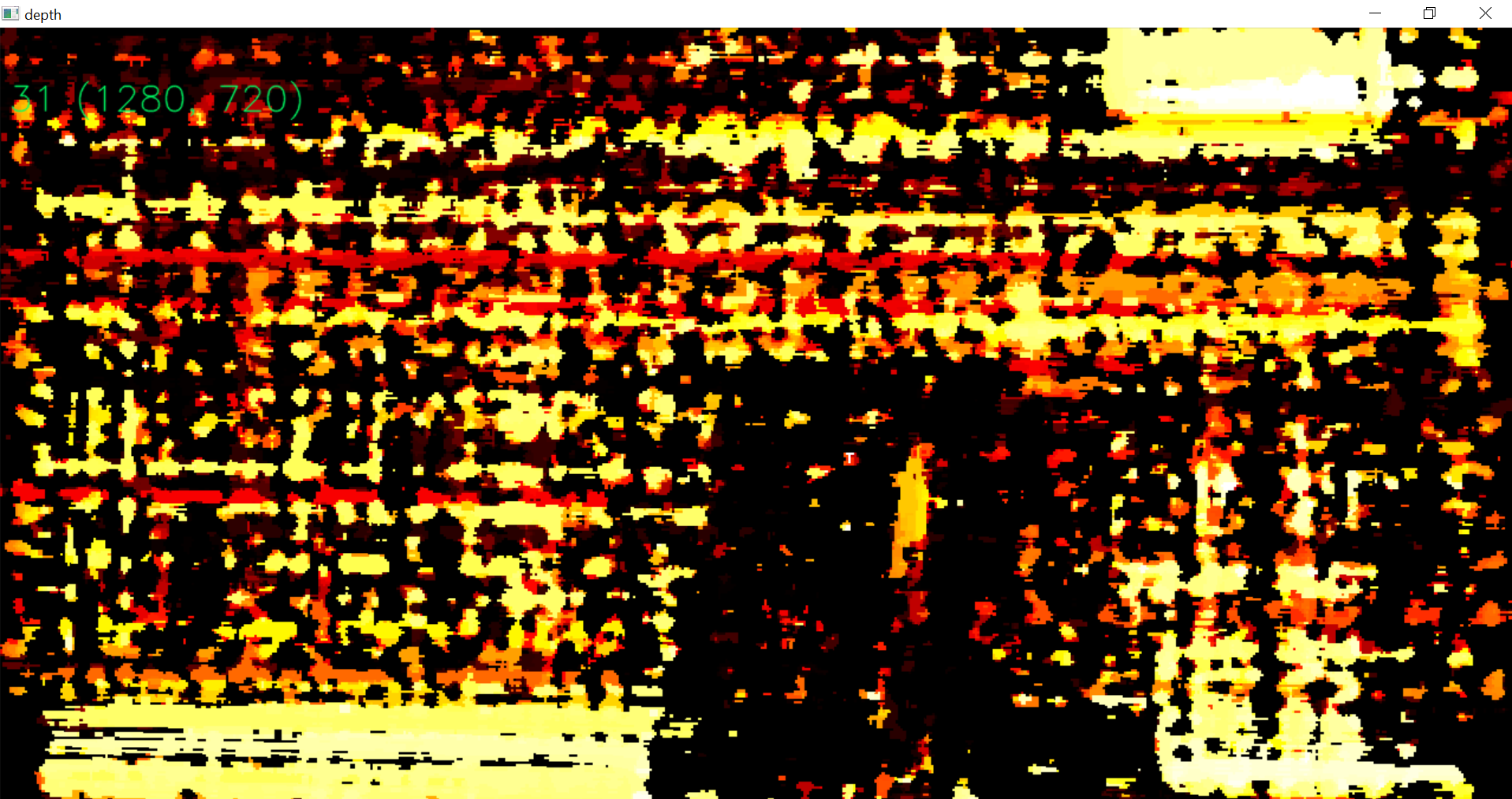
Estimated distance B/W the 2 points 32.9802 mm

Original distance is 31.75 mm

**Part 3:**

**Question 3**

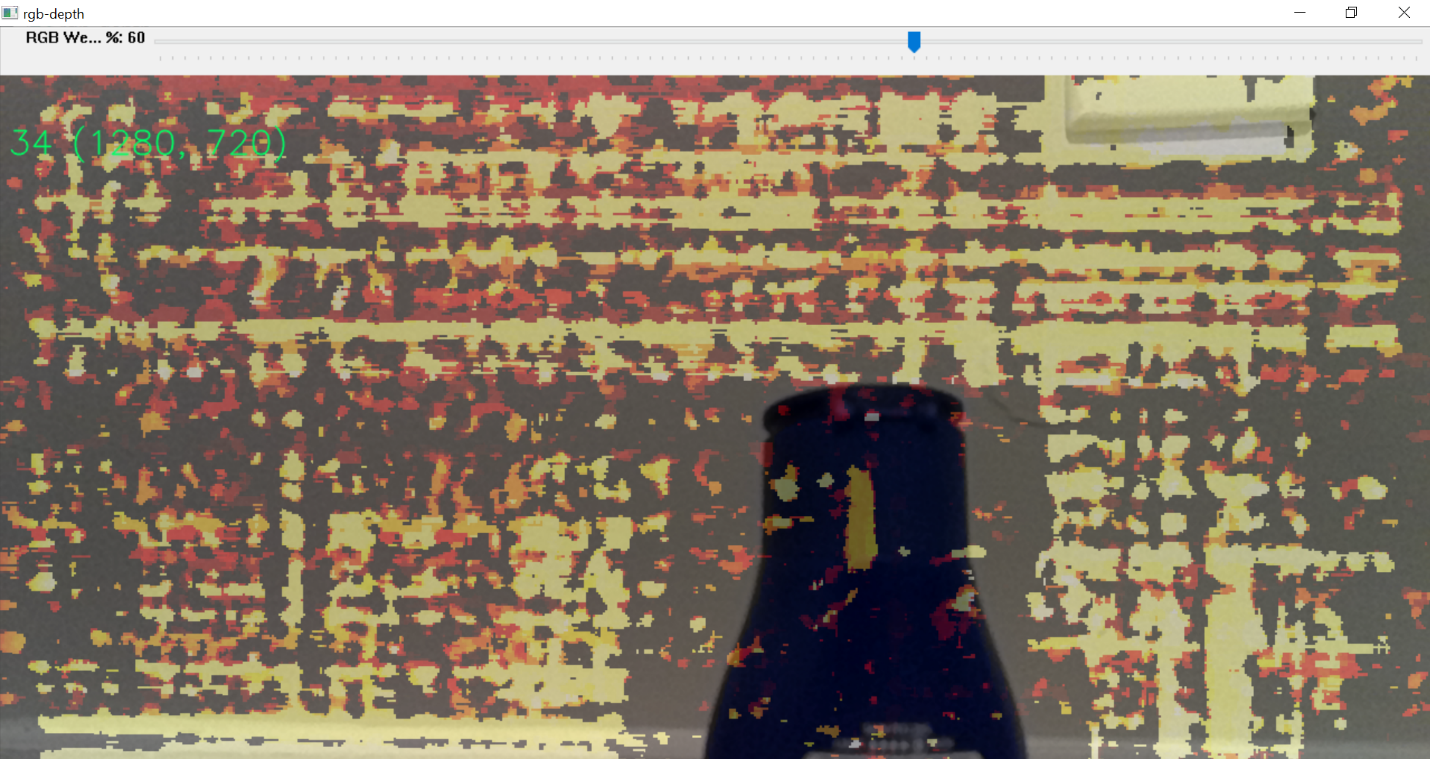
**Yes, it is feasible to run both stereo camera and mono camera simultaneously**



Depth Map



RBG image



Combination Of both rgb and depth Map

Maximum 34 fps

1280 720 resolution(720p)

Video link

<https://drive.google.com/file/d/1JujT9_UjpocDn0CMJP69wXNX5Fz9gXy-/view>

Github link

<https://github.com/harshithkamisetty/Compuer_Vision>