|  |  |  |  |
| --- | --- | --- | --- |
| **Course Name:** | **Digital Image Processing** | **Semester:** | **VII** |
| **Date of Performance:** | **October 14, 2022** | **Batch No:** | **DIP 2** |
| **Faculty Name:** | **Prof. Gopal Gupta** | **Roll No:** | **1912052** |
| **Faculty Sign & Date:** |  | **Grade/Marks:** |  |

**Experiment No: 4**

**Title: To study process of image scaling.**

|  |
| --- |
| **Aim and Objective of the Experiment:** |
| To study the process of zoom and shrink with FPS determination |

|  |
| --- |
| **COs to be achieved:** |
| 1. **Develop and Design practical application** |

|  |
| --- |
| **Theory:** |
| https://miro.medium.com/max/1400/1*8SW0I1bh0S8s83sa5STAZQ.png  When working with digital images, there are plenty of manipulation techniques we can apply. A photographer can edit photos captured from an SD card during post and further retouch the images. One important detail is the size of the image, also called the resolution. The resolution is measured in pixels along the width (w) and height (h) of an image. You can change the size of an image by resizing and resampling. Resampling involves resizing, but resizing is not exactly resampling. The two techniques are not the same though the term itself is used interchangeably at times.  Altering its size thus involves a reduction in the number of pixels. The first technique would involve a simple method like in resizing, but the second would be more complex like in resampling. The process of resizing or resampling does not take place during image capture. When a photographer captures the image, their camera’s sensor uses the native resolution. A 32 MP camera is thus capable of using the full native resolution when capturing an image. It is in post or when the image has been transferred from the camera to a computer when its size can be modified.  A Guide to Using Frame Rates in Your Videos [ Artlist  Frame rate, which you will also hear referred to as ‘frames per second’ or ‘fps’ and maybe even ‘framerate’, means how many frames your camera records each second to create a motion picture. After all, a moving picture is just a series of still images played quickly in succession. When a video is displayed on a screen, the speed at which it is played is also known as the frame rate. For footage to appear ‘normal’, capture and playback frame rates should be the same.  In theory, you can shoot a motion picture anywhere from about 20 frames per second to as fast as your camera can record, write and refresh, but there are some rough standardizations when it comes to frame rate. If you were to shoot at under 12 frames per second, the human eye would be able to differentiate between the individual frames, so that’s not advisable. At about 16 frames per second, your videos will resemble old silent movies when the reels were hand-cranked. From 24 fps up, the frame rate that you choose will depend on where you expect your work to be seen, what you’re shooting and the impact that you want to have on your audience. Anything over 60 fps is referred to as high frame rate. |

|  |
| --- |
| **Stepwise-Procedure:** |
| 1. Take selfie 2. Transfer Image to local PC 3. Measure Resolution 4. Write a program to Local PC 5. Write a program to scale image 1.7\*& 0.3 6. Write a program to display image & blank Matrix with specific FPS 7. Reach 15 FPS & find your own t (duration of POV) 8. Repeat same on python |

|  |
| --- |
| **Output** |
| Upload picture screenshots for steps  clear all;  close all;  clc;  selfie = imread('manush.jpg');  imwrite(selfie,'myphoto.jpg');  [m,n,z] = size(selfie);  my\_scaled= imresize(selfie,[m\*2,n\*1]);  [r2,c2,z2] = size(my\_scaled);  figure;  imshow(my\_scaled);  title('My scaled selfie')  % black\_img = zeros(m,n,z);  % figure;  % imshow(black\_img);  % title('Black matrix ')  % I = mat2gray(black\_img); %converts matrix to image  % imwrite(I,'black.jpg');  % load the images  images = cell(2,1);  images{1} = selfie;  images{2} = background;  % create the video writer with 1 fps  writerObj = VideoWriter('mypov.avi');  writerObj.FrameRate = 1;  % set the seconds per image  secsPerImage = [2 2];  % open the video writer  open(writerObj);  % write the frames to the video  for u=1:length(images)  % convert the image to a frame  frame = im2frame(images{u});  for v=1:secsPerImage(u)  writeVideo(writerObj, frame);  end  end  % close the writer object  close(writerObj);  video = 'mypov.avi';  videoReader = VideoReader(video);  fps = get(videoReader, 'FrameRate');  disp(fps); % the fps is correct: it's the same declared in the video file properties  currAxes = axes;  while hasFrame(videoReader)  vidFrame = readFrame(videoReader);  image(vidFrame, 'Parent', currAxes);  currAxes.Visible = 'off';  pause(1/videoReader.FrameRate);  end  **SCALED IMAGE AND BLANK MATRIX**      **VIDEO FRAMES 1 & 2** |

|  |
| --- |
| **Conclusions:** |
| In this experiment we read a photo, my selfie and created a photo of same size all filled with zeroes. We created a video writer object with different fps set manually. Finally using video reader, we played it in a loop. |

|  |
| --- |
| **Post Lab Subjective/Objective type Questions:** |
| Answer the following questions:   1. Why FPS should not be very high or very low   **ANS: If** fps is too low, it will lead to bad user experience as the human eye would be able to differentiate between the individual frames, as if we are watching a slideshow. This is because only a few different frames appear on your screen per second.  **If** is fps is too high, too much processing power is wasted in rendering frames with unnoticeable improvement in visual performance. Larger buffer and more powerful processing units may or may not be available. Another downside of having high FPS is that, lighting requirement will increase significantly while shooting because less light enters the sensors as fps increases. Also, the on-screen images in high fps move so fast that they can become more distorted if GPU power is not enough to render frames so fast.   1. What is maximum scaling one can achieve before image gets pixelated.   **ANS**: If the image is raster format e.g. jpg, tiff, png, etc. there is a limit to how much the image can be scaled without losing quality, otherwise interpolations need to be done to fill in the scaled regions with pixels. However vector image formats like svm, vstm, fcm, etc. where images are stores as mathematical equations, so they can be scaled indefinitely without losing image quality. |

|  |
| --- |
| **Signature of faculty in-charge with Date:** |