**Department of Electrical Engineering**

|  |  |
| --- | --- |
| **Faculty Member:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Dated: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
|  |  |
| **Course/Section:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Semester: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
|  |  |

**CS-477 Computer Vision**

**Lab#7: Image interpolation**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **PLO4-CLO4** | **PLO5-CLO5** | **PLO8-CLO6** | **PLO9-CLO7** |
| **Name** | **Reg. No** | **Investigation**  **(5 marks)** | **Modern Tool Usage**  **(5 marks)** | **Ethics**  **(5 marks)** | **Individual and Team Work**  **(5 marks)** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Lab#7: image interpolation**

**Objectives**

The objective in this lab is to introduce digital images as a second useful type of signal. We will show how the A-to-D sampling and the D-to-A reconstruction processes are carried out for digital images. We will show a commonly used method of image zooming (reconstruction) that gives “poor” results.

* Familiarization with digital images.
* Working with images in Matlab.
* Sampling of images
* Familiarization with reconstruction of images

**Lab Instructions**

* This lab activity comprises of following parts: Lab Exercises, and Post-Lab Viva/Quiz session.
* The lab report shall be uploaded on LMS.
* Only those tasks that are completed during the allocated lab time will be credited to the students. Students are however encouraged to practice on their own in spare time for enhancing their skills.

**Lab Report Instructions**

All questions should be answered precisely to get maximum credit. Lab report must ensure following items:

* Lab objectives
* Python codes
* Results (graphs/tables) duly commented and discussed
* Conclusion

### Sampling of Images

Images that are stored in digital form on a computer have to be sampled images because they are stored in an MxN array (i.e., a matrix). The sampling rate in the two spatial dimensions was chosen at the time the image was digitized (in units of samples per inch if the original was a photograph). For example, the image might have been “sampled” by a scanner where the resolution was chosen to be 300 dpi (dots per inch).7 If we want a different sampling rate, we can simulate a *lower* sampling rate by simply throwing away samples in a periodic way. For example, if every other sample is removed, the sampling rate will be halved (in our example, the 300 dpi image would become a 150 dpi image). Usually this is called *sub-sampling* or *down-sampling*.8

*Down-sampling* throws away samples, so it will shrink the size of the image. This is what is done by the following scheme *wp = ww(1:p:end,1:p:end);* when we are down sampling by a factor of p.

### interpolation of Images

When an image has been sampled, we can ﬁll in the missing samples by doing interpolation. For images, this would be analogous to the sine-wave interpolation which is part of the reconstruction process in a D-to-A converter. We could use a “square pulse” or a “triangular pulse” or other pulse shapes for the reconstruction.



Figure 1: 2-D Interpolation broken down into row and column operations: the gray dots indicate repeated data values created by a zero-order hold; or, in the case of bilinear interpolation, they are the interpolated values.

.

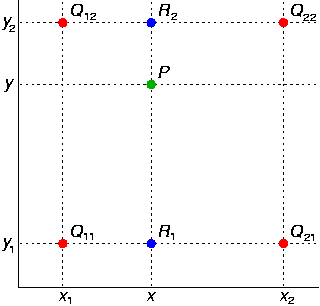
**Bi-linear Interpolation**

Linear interpolation in the x-direction followed by linear interpolation in y-direction.

**Key Points:**

**Q11 = (x1, y1), Q12 = (x1, y2),**

**Q21 = (x2, y1), Q22 = (x2, y2).**







### Lab tasks

1. Image Resolution.

1. Reduce the resolution of **7\_1.asc** by a factor of 4 in both horizontal and vertical dimensions (e.g., if the original image is 400 by 400, then result shall be 100 by 100) to create a decimated image using two different methods:

***HINT:*** To read in an “.asc”, : X=np.loadtxt(‘3\_1.asc’).

* + 1. Keep one pixel out of every 4x4 pixel area. Display the resulting image Y1.
    2. Replace every 4x4 pixel area in **7\_1.asc** by the average value of the pixel values in that region. Display the resulting image Y2.

1. Enlarge Image Y1 by a factor of 4 in both horizontal and vertical dimensions (e.g., from 100 by 100 to 400 by 400) using:
   1. Pixel repeating (zero order hold).
   2. Bilinear interpolation (do not use built-in interpolation function*, use your own code*).

Keep the result images from (b.i) and (b.ii) the same size as **7\_1.asc,** compare the images. Compare the quality of the linear interpolation result to the zero-order hold result. Point out regions where they differ and try to justify this difference by estimating the local frequency content. In other words, look for regions of “low-frequency” content and “high-frequency” content and see how the interpolation quality is dependent on this factor. A couple of questions to think about: Are edges low frequency or high frequency features? Is the background a low frequency or high frequency feature?