

CS 7367

MACHINE VISION

ASSIGNMENT 1

#### INSTRUCTOR

Dr. Sanghoon Lee

**Your Name: Rashaad Washington**

**KSU ID: 001190767**

**1. ABSTRACT**

In this assignment, I coded a python program that uses an input picture with pixel-size 1024x1024 and down-sampled to different sizes. Such sizes are 512x512, 256x256, 128x128, 64x64, and 32x32. Afterwards, using the downsampled 32x32 image as the base image, I proceeded to upsample this image to the same sizes in reverse order. The last test that the program performed was changing the gray level in an 8-bit gray level image from 256 gray levels to 128, 64, 32, 16, 8, 4, and 2 gray levels.

**2. Test RESULTS**

|  |
| --- |
| **2.1 Test Results for Down-sampling**    Figure 1: Preprocessed (original) grayscale images (rose.jpg), downsampled images (rose.jpg) at 2x, 4x, 8x, 16x, 32x magnitude respectively.    Figure 2: Downsampled images (rose.jpg) at 8x, 16x, 32x magnitude respectively. |

|  |
| --- |
| **2.2 Test Results for Up-sampling**  A close-up of a flower  AI-generated content may be incorrect.  Figure 3: Preprocessed 32x32 grayscale images (rose.jpg) up-sampled to 64x64 and 128x128 size respectively.  A close-up of a flower  AI-generated content may be incorrect.  Figure 4: Up-sampled images at 256x256, 512x512, and 1024x1024 sizes respectively. |
| **2.3 Test Results for Gray level Change**    Figure 5: Rose.jpg image (256 gray levels) changed to 128, 64, 32, 16, 8, 4, & 2 gray levels respectively |

**3. Discussion**

My results were on par with my beliefs. That is, when an image is downsampled, it visually appears to be of ‘less quality’ when viewing on a high resolution device. It can become down-sampled to the extent where each pixel can be seen by the human eye. However, thats not the same idea with up-sampling an image. When given a 32x32 image, it is near impossible to attain the same quality of an oringinally 1024x1024 than an image that was up-sampled to 1024x1024. My assumption is that crucial image details and quality are ‘permanently’ lost when downsampled. For the 2nd test, the results were positive. When looking at the results in reverse, the image with 2 gray samples can be seen with the naked eye as just 2 colors. As the amount of gray levels increases, it becomes difficult to determine.   
  
If given time, a potential next step for this assignment would be to use an image that has colors other than black, gray, and white. In theory, this would provide a unique insight when decreasing the amount of gray levels in the image. However, the program is not coded to use reduce or increase specific RGB values, just gray levels.

**4. CODES**

**3.1 Code for Image Negative**

# Rashaad Washington

# Machine Vision

# Kennesaw State University

import numpy as np

import cv2

import matplotlib.pyplot as plt

# Read in image

orig\_img = cv2.imread('rose.jpg', 0)

[a, b] = orig\_img.shape

#Image Downsample Function

def downsamplethis(image, multiplier):

new\_img = np.zeros((a//multiplier, b//multiplier), dtype=int)

for i in range(0, a, multiplier):

for j in range(0, b, multiplier):

new\_img[i//multiplier][j//multiplier] = image[i][j]

return new\_img

#Image Upsample Function

def upsamplethis(image, multiplier):

height, width = image.shape

new\_height = height \* multiplier

new\_width = width \* multiplier

new\_img = np.zeros((height,width), dtype=int)

for i in range(height):

for j in range(width):

small\_i = i//multiplier

small\_j = j // multiplier

new\_img[i][j] = image[small\_i][small\_j]

return new\_img

#Downsample

fig, axes = plt.subplots(2, 6, figsize=(25, 10))

axes = axes.flatten()

#Preprocessed Image

axes[0].imshow(orig\_img, cmap="gray")

axes[0].set\_title('Preprocessed Image')

#Downsample image to 512x512

axes[1].imshow(downsamplethis(orig\_img, 2), cmap="gray")

axes[1].set\_title('Downsample image to 512x512')

#Downsample image to 256x256

axes[2].imshow(downsamplethis(orig\_img, 4), cmap="gray")

axes[2].set\_title('Downsample image to 256x256')

#Downsample image to 128x128

axes[3].imshow(downsamplethis(orig\_img, 8), cmap="gray")

axes[3].set\_title('Downsample image to 128x128')

#Downsample image to 64x64

axes[4].imshow(downsamplethis(orig\_img, 16), cmap="gray")

axes[4].set\_title('Downsample image to 64x64')

#Downsample image to 32x32

axes[5].imshow(downsamplethis(orig\_img, 32), cmap="gray")

axes[5].set\_title('Downsample image to 32x32')

print('\n')

#--------------------------------------------------------------------

#Upsample

#store 32x32 image

upimg = downsamplethis(orig\_img, 32)

#Preprocessed 32x32 Image

axes[6].imshow(upimg, cmap="gray")

axes[6].set\_title('Preprocessed 32x32 Image')

#Upsample image to 64x64

axes[7].imshow(upsamplethis(upimg, 2), cmap="gray")

axes[7].set\_title('Upsample image to 64x64')

#Upsample image to 128x128

axes[8].imshow(upsamplethis(upimg, 4), cmap="gray")

axes[8].set\_title('Upsample image to 128x128')

#Upsample image to 256x256

axes[9].imshow(upsamplethis(upimg, 8), cmap="gray")

axes[9].set\_title('Upsample image to 256x256')

#Upsample image to 512x512

axes[10].imshow(upsamplethis(upimg, 16), cmap="gray")

axes[10].set\_title('Upsample image to 512x512')

#Upsample image to 1024x1024

axes[11].imshow(upsamplethis(upimg, 32), cmap="gray")

axes[11].set\_title('Upsample image to 1024x1024')

plt.show()

#Reduce Gray Levels

def reducegray(image, level):

mod = 256 // level

return (image//mod) \* mod

fig, axes1 = plt.subplots(2, 4, figsize=(25, 10))

axes1 = axes1.flatten()

#Preprocessed Image

axes1[0].imshow(orig\_img, cmap="gray")

axes1[0].set\_title('Preprocessed Image')

#Change gray level to 128

axes1[1].imshow(reducegray(orig\_img, 128), cmap="gray")

axes1[1].set\_title('128 Gray Level')

#Change gray level to 64

axes1[2].imshow(reducegray(orig\_img, 64), cmap="gray")

axes1[2].set\_title('64 Gray Level')

#Change gray level to 32

axes1[3].imshow(reducegray(orig\_img, 32), cmap="gray")

axes1[3].set\_title('32 Gray Level')

#Change gray level to 16

axes1[4].imshow(reducegray(orig\_img, 16), cmap="gray")

axes1[4].set\_title('16 Gray Level')

#Change gray level to 8

axes1[5].imshow(reducegray(orig\_img, 8), cmap="gray")

axes1[5].set\_title('8 Gray Level')

#Change gray level to 4

axes1[6].imshow(reducegray(orig\_img, 4), cmap="gray")

axes1[6].set\_title('4 Gray Level')

#Change gray level to 2

axes1[7].imshow(reducegray(orig\_img, 2), cmap="gray")

axes1[7].set\_title('2 Gray Level')

plt.show()