

# **CS5500 – Digital Image Processing**

**Cal Poly Pomona**

**Homework 1**

**Fall 2023**

## **Description:**

Spatial Resolution

Gray Level Resolution

Name: Fidelis Prasetyo

Email: ([fprasetyo@cpp.edu](mailto:fprasetyo@cpp.edu))

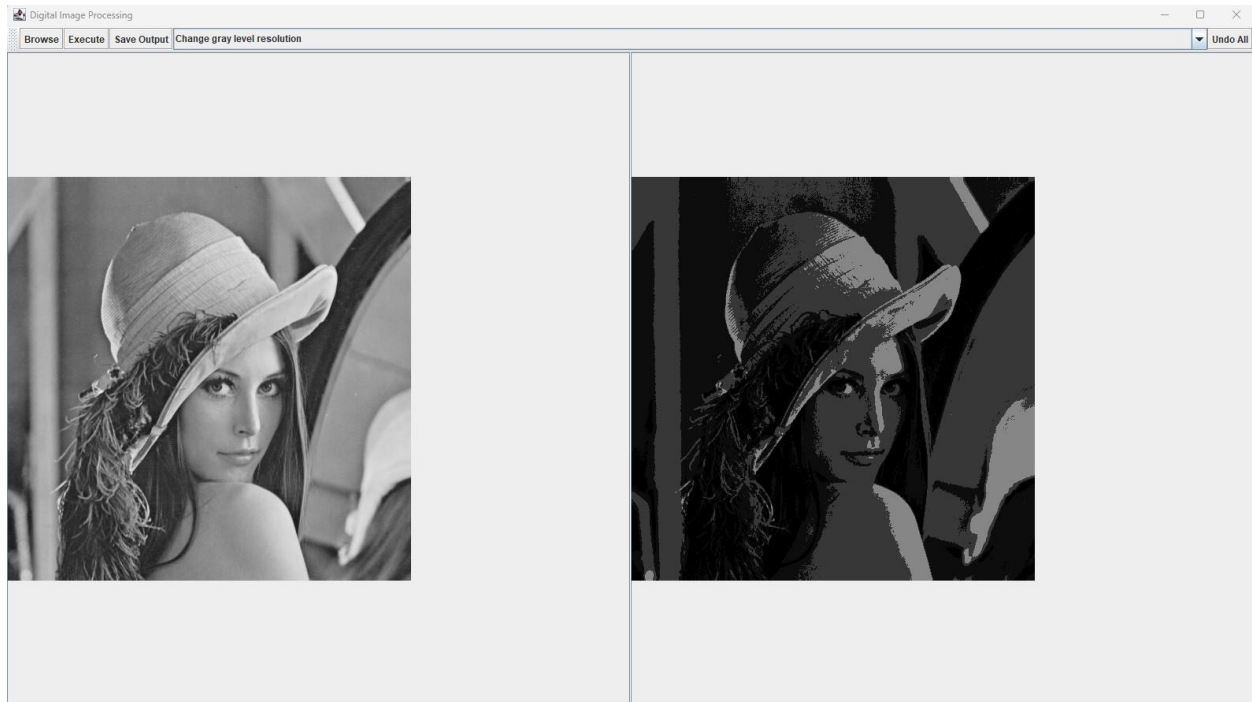
BroncoID: 015765555

Github & Source code:

<https://github.com/fidelisprasetyo/DigitalImageProcessing>

# Program Description

Preview of GUI of the program:



Description:

- Left image: the original image.
- Right image: the processed image.
- Browse button: to open the desired image file.
- Execute button: apply the chosen action.
- Save output: save the processed image (right image) to a file.
- Undo all: revert all changes to the original image.

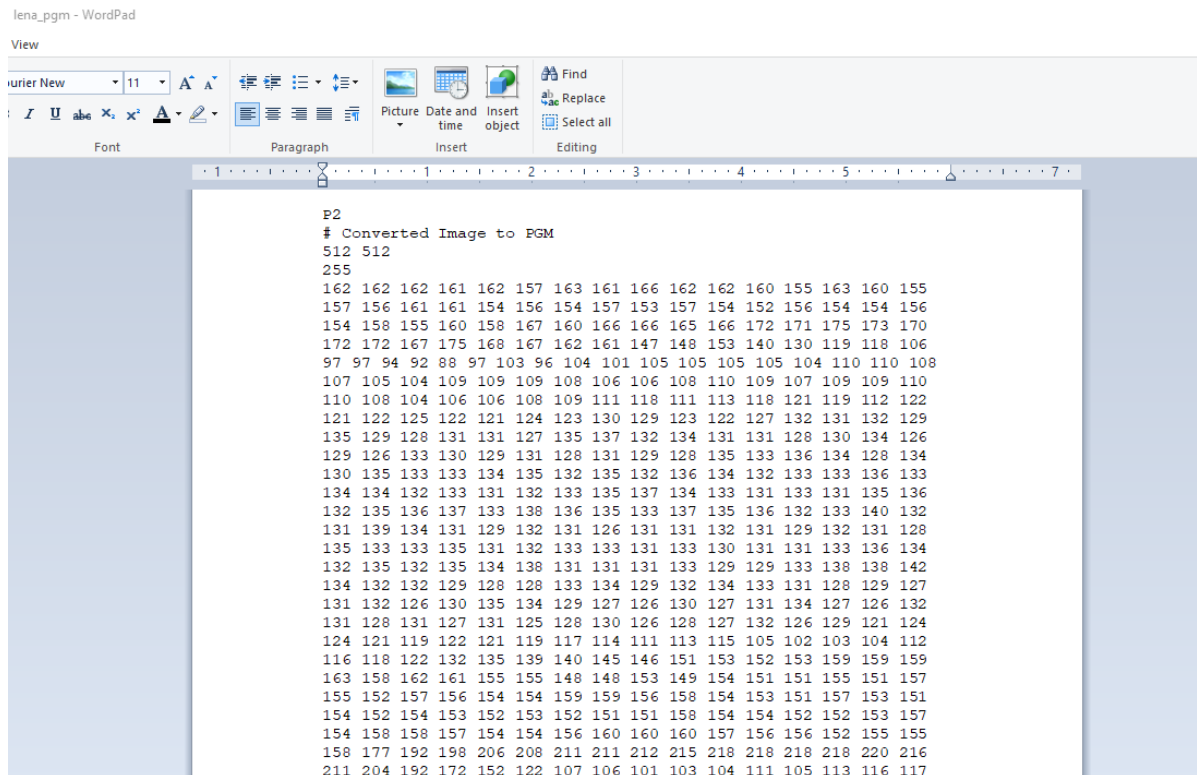
Implemented Features:

- Convert to PGM: convert the loaded image file into a PGM file.
- Change spatial resolution: resize the spatial resolution of the image using:
  - Nearest neighbor method
  - Linear interpolation
  - Bilinear interpolation
- Change gray level resolution: reduce the gray level of the image.

# Program Demonstration

## 1. Convert to PGM

The program is able to convert the loaded image into a readable PGM file. Here's the preview of the content inside the pgm file of lena.gif.

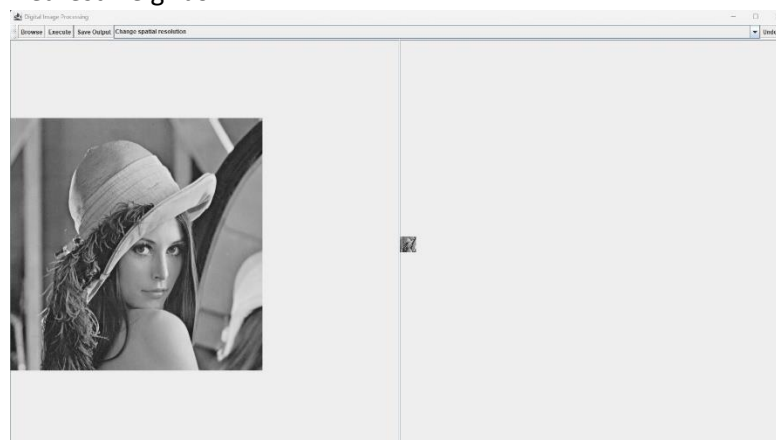


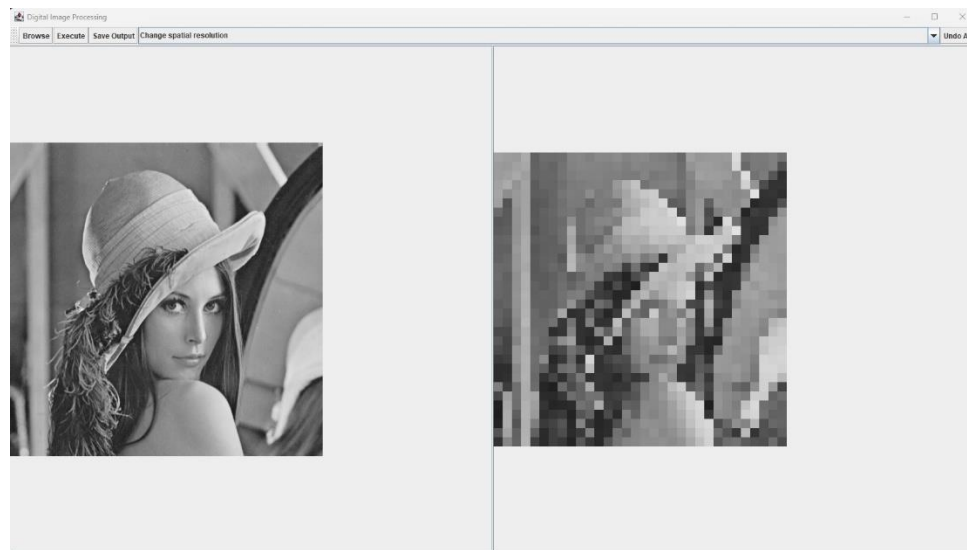
```
P2
# Converted Image to PGM
512 512
255
162 162 162 161 162 157 163 161 166 162 162 160 155 163 160 155
157 156 161 161 154 156 154 157 153 157 154 152 156 154 154 156
154 158 155 160 158 167 160 166 166 165 166 172 171 175 173 170
172 172 167 175 168 167 162 161 147 148 153 140 130 119 118 106
97 97 94 92 88 97 103 96 104 101 105 105 105 105 104 110 110 108
107 105 104 109 109 109 108 106 106 108 110 109 107 109 109 110
110 108 104 106 106 108 109 111 118 111 113 118 121 119 112 122
121 122 125 122 121 124 123 130 129 123 122 127 132 131 132 129
135 129 128 131 131 127 135 137 132 134 131 131 128 130 134 126
129 126 133 130 129 131 128 131 129 128 135 133 136 134 128 134
130 135 133 133 134 135 132 135 132 136 134 132 133 133 136 133
134 134 132 133 131 132 133 135 137 134 133 131 133 131 135 136
132 135 136 137 133 138 136 135 133 137 135 136 132 133 140 132
131 139 134 131 129 132 131 126 131 131 132 131 129 132 131 128
135 133 133 135 131 132 133 133 131 133 130 131 131 133 136 134
132 135 132 135 134 138 131 131 131 133 129 129 133 138 138 142
134 132 132 129 128 128 133 134 129 132 134 133 131 128 129 127
131 132 126 130 135 134 129 127 126 130 127 131 134 127 126 132
131 128 131 127 131 125 128 130 126 128 127 132 126 129 121 124
124 121 119 122 121 119 117 114 111 113 115 105 102 103 104 112
116 118 122 132 135 139 140 145 146 151 153 152 153 159 159 159
163 158 162 161 155 155 148 148 153 149 154 151 151 155 151 157
155 152 157 156 154 154 159 159 156 158 154 153 151 157 153 151
154 152 154 153 152 153 152 151 151 158 154 154 152 152 153 157
154 158 158 157 154 154 156 160 160 160 157 156 156 152 155 155
158 177 192 198 206 208 211 211 212 215 218 218 218 220 216
211 204 192 172 152 122 107 106 101 103 104 111 105 113 116 117
```

## 2. Spatial Resolution

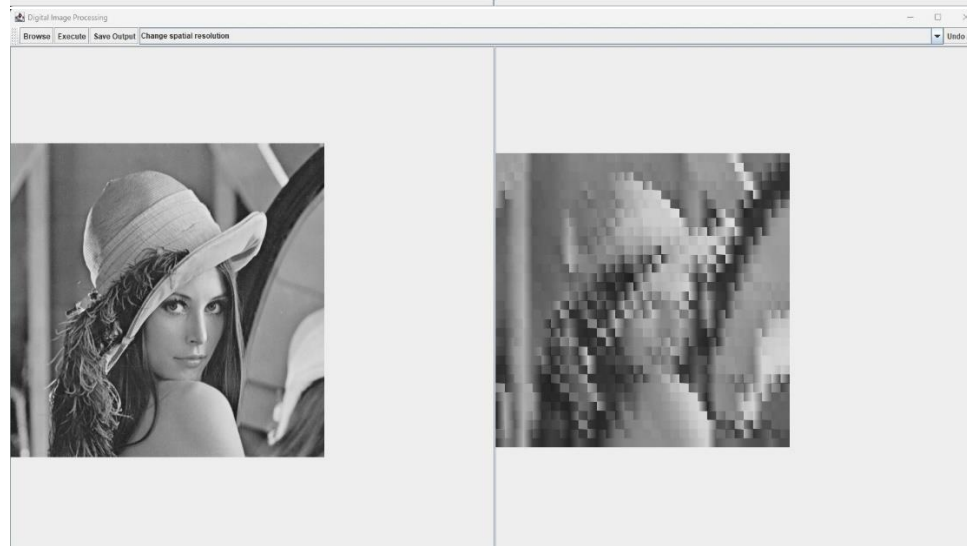
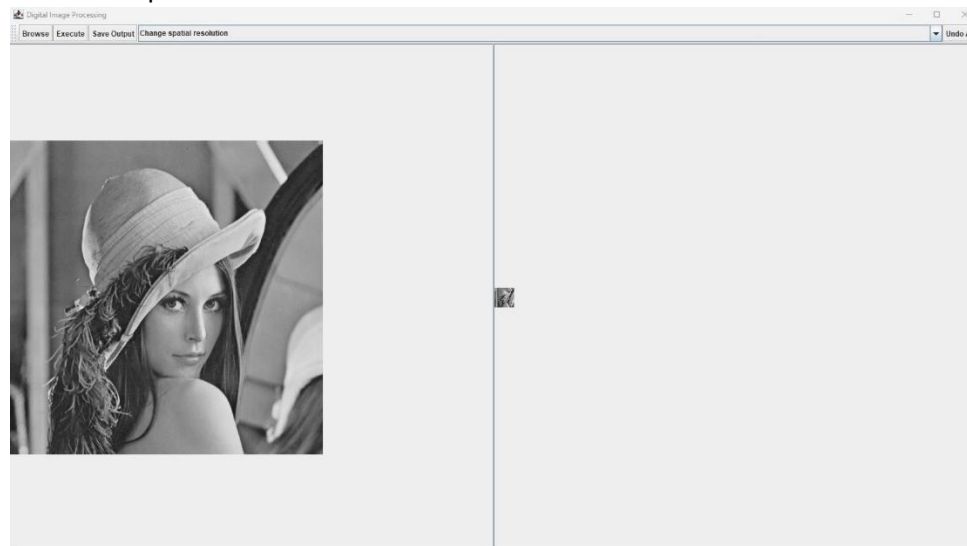
The original lena.gif image will be resized from 520x520 pixels into 32x32 in this demo. After that, the resized image will be zoomed back in into 480x480 pixels image to compare the difference between the scaling algorithms.

- Nearest Neighbor

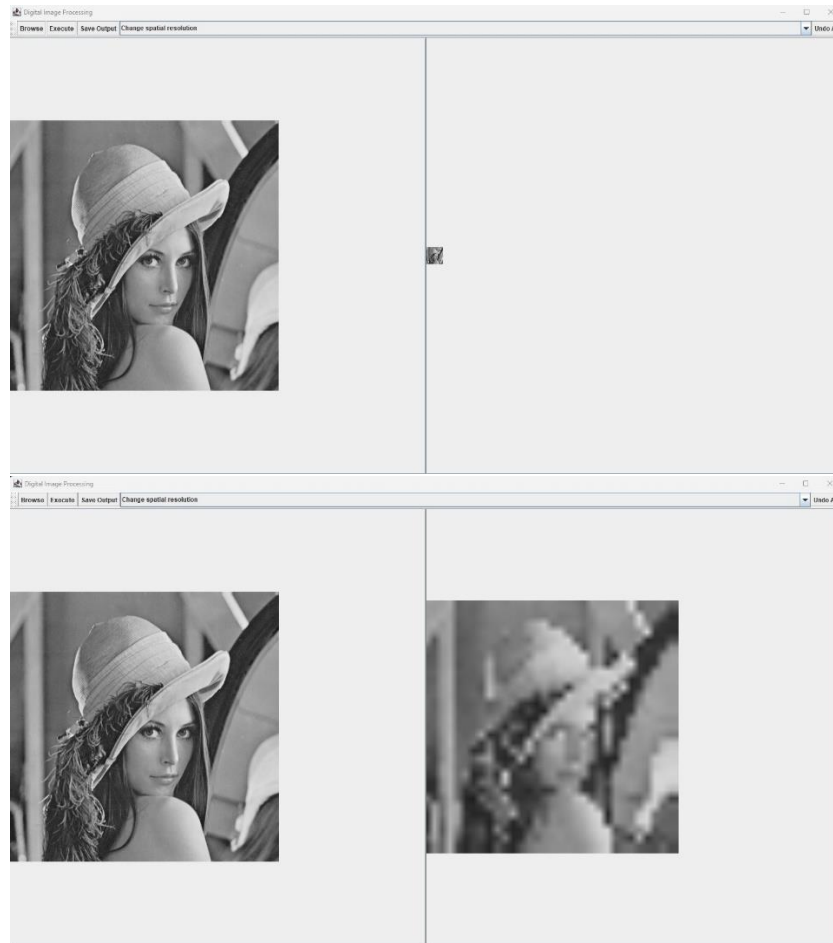




- Linear Interpolation



- Bilinear Interpolation



### 3. Gray Level Resolution

The original 8-bit image will be reduced to 7-bit and 2-bit images in this demo. The results, however, are not perfect as the output images are rather dark.



7-bit



2-bit

I suspect that the problem is in the thresholding. Theoretically, in a 1-bit image, we would want to categorize the 8-bit pixel values into binary values, which are 0 (black) and 255 (white). To decide the categorization of the pixel values, theoretically, we would pick the middle range of the original gray level resolution, which is 128. In other words, all pixels ranging from 0 to 127 would be converted to 0, while 128 to 255 would be converted to 255. However, the implemented feature doesn't behave that way. I'm still experimenting with other ways to implement this feature better, but I think this current implementation would suffice for now.

## Source Code & Supporting Files

The source code, this pdf file, and output images can be obtained from this GitHub repository:

<https://github.com/fidelisprasetyo/DigitalImageProcessing>