

LAPORAN
TUGAS AKHIR
MATA KULIAH PENGOLAHAN CITRA
Image Enhancement



Kelompok 2 :

- | | | |
|----|------------------------------|---------------|
| 1. | Bryan Faizal Bagaskara Putra | (19103020034) |
| 2. | Wahyu Anggara Putra | (2013020040) |
| 3. | Al Dian | (2013020149) |
| 4. | Shandy Sadewa Asmoro | (2013020206) |

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS NUSANTARA PGRI KEDIRI
TAHUN 2023

A. PROGRAM IMAGE ENHANCEMENT

Image enhancement, atau perbaikan kualitas citra, merupakan proses yang bertujuan untuk memperoleh citra yang lebih sesuai digunakan untuk aplikasi lebih lanjut, seperti pengenalan objek dalam citra. Teknik ini termasuk dalam tahap awal pengolahan citra, yang sering disebut sebagai *preprocessing*.

Image enhancement menjadi penting karena citra dapat mengandung berbagai masalah yang mempengaruhi kualitasnya. Beberapa masalah umum termasuk keberadaan derau (*noise*) dalam citra, kecerahan atau kegelapan yang tidak ideal, ketajaman yang kurang, dan citra yang kabur (*blur*). Selain itu, saat proses akuisisi citra, seperti pengambilan foto, juga dapat terjadi cacat yang mempengaruhi kualitas citra. Cacat ini bisa disebabkan oleh lensa yang menghasilkan objek atau latar belakang yang buram (*blurring*), atau karena objek yang difoto bergerak dan menyebabkan efek *motion blurring*. Terakhir, distorsi geometrik juga bisa terjadi akibat penggunaan lensa yang tidak sempurna atau sudut pengambilan yang tidak ideal.

Maka dari itu, *image enhancement* menjadi penting dalam memperbaiki kualitas citra dan mengatasi masalah-masalah yang telah disebutkan. Dengan menerapkan teknik-teknik enhancement yang sesuai, citra dapat ditingkatkan sehingga lebih jelas, tajam, dan bebas dari gangguan seperti derau, kegelapan, kecerahan yang berlebihan, dan distorsi geometrik. Hal ini akan meningkatkan kemampuan pengolahan citra lebih lanjut dan meningkatkan hasil dari aplikasi yang menggunakan citra tersebut.

Dalam program yang telah dibuat, terdapat beberapa teknik *image enhancement* yang diimplementasikan, yaitu pengubahan kecerahan gambar (*image brightening*), pelembutan citra (*image smoothing*), citra negatif (*image negative*), perataan histogram (*histogram equalization*), dan spesifikasi histogram (*histogram specification / histogram matching*).

Teknik pengubahan kecerahan gambar (*image brightening*) memungkinkan pengguna untuk meningkatkan atau mengurangi tingkat kecerahan gambar. Dalam program ini, pengguna dapat menggunakan tombol "Apply Brightness" untuk menerapkan teknik ini. Penerapan teknik *image brightening* dilakukan dengan memodifikasi nilai piksel gambar sesuai dengan faktor kecerahan yang telah ditentukan. Hasilnya adalah gambar yang lebih terang atau lebih gelap sesuai dengan preferensi pengguna.

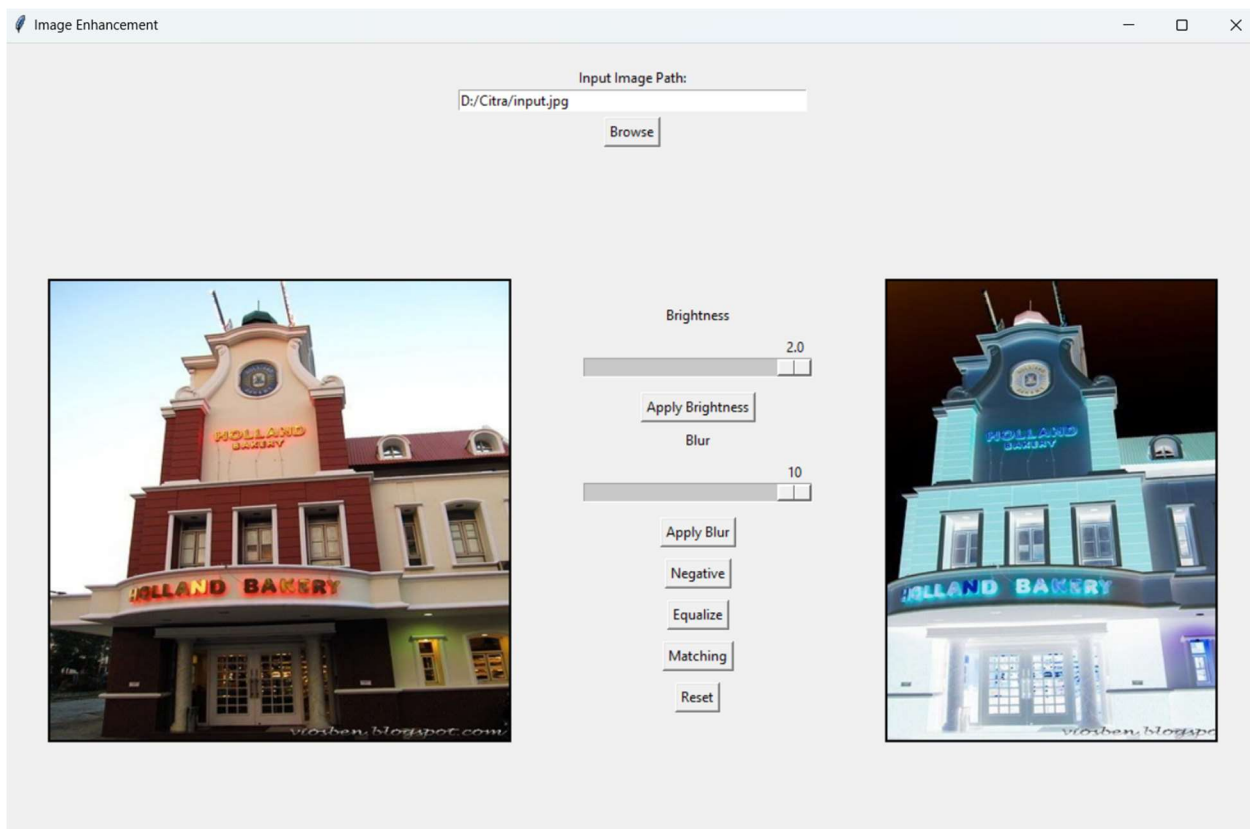
Teknik pelembutan citra (*image smoothing*) digunakan untuk menghasilkan efek blur pada gambar, yang menghaluskan atau melembutkan tampilan gambar. Pengguna dapat mengatur nilai keburaman dan menggunakan tombol "Apply Blur" untuk menerapkan teknik ini. Dalam program ini, efek *blur* diterapkan menggunakan *filter Gaussian*. Proses ini melibatkan konvolusi gambar dengan kernel *Gaussian*, yang menghasilkan gambar dengan detail yang lebih halus dan tampilan yang lebih lembut.

Selanjutnya, terdapat teknik citra negatif (*image negative*) yang menghasilkan citra yang merupakan kebalikan dari citra asli. Pengguna dapat menggunakan tombol "Negative" untuk menerapkan teknik ini. Pada penerapan teknik ini, setiap piksel pada gambar akan diubah menjadi nilai kebalikannya. Misalnya, piksel dengan nilai 0 akan menjadi 255, piksel dengan nilai 100 akan menjadi 155, dan seterusnya. Hal ini menciptakan efek kebalikan warna pada gambar.

Kemudian teknik perataan histogram (*histogram equalization*) bertujuan untuk memperoleh penyebaran histogram yang merata, sehingga setiap derajat keabuan memiliki jumlah piksel yang relatif sama. Teknik ini dapat meningkatkan kontras dan detail gambar. Pengguna dapat menggunakan tombol "Equalize" untuk menerapkan teknik ini. Pada penerapan teknik perataan histogram, histogram gambar akan diubah sehingga distribusi piksel menjadi lebih merata. Dengan ini, gambar akan memiliki rentang intensitas yang lebih luas dan kontras yang lebih baik.

Terakhir, terdapat teknik spesifikasi histogram (*histogram specification / histogram matching*) yang digunakan untuk memetakan histogram gambar ke histogram referensi yang ditentukan pengguna. Pengguna dapat menggunakan tombol "Histogram" untuk menerapkan teknik ini. Dalam penerapan teknik ini, histogram gambar akan disesuaikan dengan histogram referensi sehingga gambar memiliki distribusi piksel yang serupa dengan referensi. Hal ini berguna ketika pengguna ingin mencapai tampilan gambar yang serupa dengan referensi tertentu.

Program *image enhancement* ini memungkinkan pengguna untuk membuka gambar, memilih teknik enhancement yang diinginkan, dan melihat hasilnya secara *real-time*. Selain itu, program juga menyediakan fitur *reset* untuk mengembalikan gambar ke keadaan asli. Berikut program yang telah dibuat:



Gambar 1 GUI program Image Enhancement

B. SOURCE CODE PROGRAM


Program di atas adalah sebuah GUI (Graphical User Interface) untuk melakukan image enhancement atau perbaikan kualitas citra. Program ini dibangun menggunakan library tkinter untuk tampilan GUI, library PIL (Python Imaging Library) untuk manipulasi citra, numpy untuk manipulasi array, dan skimage untuk teknik ekualisasi histogram.

Pada awalnya, program akan membuat sebuah jendela GUI dengan judul "Image Enhancement" dan ukuran 1100x700 piksel. Program memiliki fitur untuk memilih file citra, menampilkan citra input dan citra output, serta memiliki beberapa tombol untuk melakukan operasi enhancement. Berikut alur program:

- Saat pengguna memilih file citra, program akan membuka dialog file dan meminta pengguna untuk memilih file citra. Setelah dipilih, citra input akan dimuat dan ditampilkan di dalam kotak citra input.
- Ada beberapa operasi enhancement yang dapat dilakukan:
 1. Brightness Enhancement: Terdapat sebuah slider (`brightness_scale`) yang digunakan untuk mengatur faktor kecerahan. Ketika tombol "Apply Brightness" ditekan, citra output akan diubah kecerahannya berdasarkan faktor yang dipilih.
 2. Negative Enhancement: Tombol "Negative" akan menghasilkan citra negatif dari citra input.
 3. Blur Enhancement: Terdapat sebuah slider (`blur_scale`) yang digunakan untuk mengatur tingkat blur. Ketika tombol "Apply Blur" ditekan, citra output akan diberikan efek blur dengan tingkat yang dipilih.
 4. Equalize Enhancement: Tombol "Equalize" akan menghasilkan citra dengan histogram yang seimbang, sehingga meningkatkan kontras dan detail.
 5. Histogram Matching: Tombol "Matching" akan membuka dialog file dan meminta pengguna untuk memilih file citra referensi. Citra output akan disesuaikan histogramnya dengan histogram citra referensi yang dipilih.
- Setiap kali sebuah operasi enhancement diterapkan, citra output akan diperbarui dan ditampilkan di dalam kotak citra output.
- Terdapat juga tombol "Reset" yang dapat digunakan untuk mengembalikan citra output ke citra input semula.


Dengan menggunakan GUI ini, pengguna dapat dengan mudah memilih file citra, melakukan operasi enhancement yang diinginkan, dan melihat hasilnya secara real-time melalui tampilan citra input dan output yang ditampilkan di dalam GUI. Berikut source codenya:

- Library-library yang digunakan diimpor ke dalam program.



```
1 import tkinter as tk
2 from tkinter import filedialog
3 from PIL import Image, ImageTk, ImageEnhance, ImageOps, ImageFilter
4 import numpy as np
5 from skimage import exposure
```

- Kelas ImageEnhancementGUI digunakan untuk mengatur GUI dan operasi enhancement. Pada konstruktor, dilakukan inisialisasi variabel-variabel yang akan digunakan, seperti citra input dan output, serta slider untuk brightness dan blur.



```
1 class ImageEnhancementGUI:
2     def __init__(self, root):
3         self.root = root
4         self.root.title("Image Enhancement")
5         self.root.geometry("1100x700")
6
7         self.input_image = None
8         self.output_image = None
9
10        self.brightness_scale = None
11        self.blur_scale = None
12
13        self.create_widgets()
```

- Metode `create_widgets` digunakan untuk membuat elemen-elemen GUI, seperti frame, label, entry, tombol, dan slider.

```
1 def create_widgets(self):
2     # Input frame
3     input_frame = tk.Frame(self.root)
4     input_frame.pack(side=tk.TOP, padx=20, pady=20)
5
6     input_label = tk.Label(input_frame, text="Input Image Path:")
7     input_label.pack()
8
9     self.input_entry = tk.Entry(input_frame, width=50)
10    self.input_entry.pack()
11
12    browse_button = tk.Button(input_frame, text="Browse", command=self.choose_file)
13    browse_button.pack(pady=5)
14
15    # Input image frame
16    input_image_frame = tk.Frame(self.root)
17    input_image_frame.pack(side=tk.LEFT, padx=20, pady=10)
18
19    self.input_box = tk.Label(input_image_frame, relief=tk.SOLID, width=400, height=400)
20    self.input_box.pack(side=tk.LEFT, padx=20)
21
22    # Enhancement buttons frame
23    buttons_frame = tk.Frame(self.root)
24    buttons_frame.pack(side=tk.LEFT, padx=20)
25
26    self.brightness_label = tk.Label(buttons_frame, text="Brightness")
27    self.brightness_label.pack()
28
29    self.brightness_scale = tk.Scale(buttons_frame, from_=0, to=2, resolution=0.1, orient=tk.HORIZONTAL, length=200)
30    self.brightness_scale.set(1.0)
31    self.brightness_scale.pack(pady=5)
32
33    self.brightness_button = tk.Button(buttons_frame, text="Apply Brightness", command=self.apply_brightness)
34    self.brightness_button.pack(pady=5)
35
36    self.blur_label = tk.Label(buttons_frame, text="Blur")
37    self.blur_label.pack()
38
39    self.blur_scale = tk.Scale(buttons_frame, from_=0, to=10, resolution=1, orient=tk.HORIZONTAL, length=200)
40    self.blur_scale.set(0)
41    self.blur_scale.pack(pady=5)
42
43    self.blur_button = tk.Button(buttons_frame, text="Apply Blur", command=self.apply_blur)
44    self.blur_button.pack(pady=5)
45
46    self.negative_button = tk.Button(buttons_frame, text="Negative", command=self.apply_negative)
47    self.negative_button.pack(pady=5)
48
49    self.equalize_button = tk.Button(buttons_frame, text="Equalize", command=self.apply_equalize)
50    self.equalize_button.pack(pady=5)
51
52    self.histogram_button = tk.Button(buttons_frame, text="Matching", command=self.apply_histogram)
53    self.histogram_button.pack(pady=5)
54
55    self.reset_button = tk.Button(buttons_frame, text="Reset", command=self.reset)
56    self.reset_button.pack(pady=5)
57
58    # Output frame
59    output_frame = tk.Frame(self.root)
60    output_frame.pack(side=tk.LEFT, padx=20, pady=0)
61
62    self.output_box = tk.Label(output_frame, relief=tk.SOLID, width=400, height=400)
63    self.output_box.pack(side=tk.LEFT, padx=20)
```

- Metode `choose_file` dipanggil saat tombol "Browse" ditekan. Metode ini membuka dialog file dan meminta pengguna untuk memilih file citra. Setelah file dipilih, path file tersebut akan dimasukkan ke dalam kotak `input_entry`, lalu citra input akan dimuat dan ditampilkan di kotak citra input.

```
1 def choose_file(self):
2     file_path = filedialog.askopenfilename(filetypes=[("Image Files", "*.jpg;*.jpeg;*.png")])
3     self.input_entry.delete(0, tk.END)
4     self.input_entry.insert(0, file_path)
5     self.load_input_image()
```

- Metode `load_input_image` digunakan untuk memuat citra input dari path yang diambil dari `input_entry`. Citra tersebut akan direscale menjadi ukuran 400x400 piksel untuk ditampilkan di kotak citra input.

```
1 def load_input_image(self):
2     input_image_path = self.input_entry.get()
3     if input_image_path:
4         self.input_image = Image.open(input_image_path)
5         self.output_image = self.input_image.copy()
6         self.display_input_image()
```

- Metode `display_input_image` digunakan untuk menampilkan citra input yang sudah direscale di kotak citra input menggunakan library PIL.

```
1 def display_input_image(self):
2     input_image_resized = self.input_image.resize((400, 400))
3     input_photo = ImageTk.PhotoImage(input_image_resized)
4     self.input_box.configure(image=input_photo)
5     self.input_box.image = input_photo
```

- Metode `display_output_image` digunakan untuk menampilkan citra output yang sudah direscale di kotak citra output menggunakan library PIL.

```
1 def display_output_image(self):
2     output_image_resized = self.output_image.resize((400, 400))
3     output_photo = ImageTk.PhotoImage(output_image_resized)
4     self.output_box.configure(image=output_photo)
5     self.output_box.image = output_photo
```

- Metode `apply_brightness` dipanggil saat tombol "Apply Brightness" ditekan. Metode ini mengambil faktor kecerahan dari slider `brightness_scale`, lalu citra output akan diubah kecerahannya berdasarkan faktor tersebut menggunakan library PIL. Hasil perubahan citra output akan ditampilkan di kotak citra output.

```
1 def apply_brightness(self):
2     if self.input_image is None:
3         return
4
5     if self.output_image != self.input_image:
6         self.output_image = self.input_image.copy()
7
8     brightness_factor = self.brightness_scale.get()
9     enhancer = ImageEnhance.Brightness(self.output_image)
10    self.output_image = enhancer.enhance(brightness_factor)
11    self.display_output_image()
```


- Metode `apply_blur` dipanggil saat tombol "Apply Blur" ditekan. Metode ini mengambil nilai tingkat blur dari slider `blur_scale`, lalu citra output akan diberikan efek blur dengan tingkat tersebut menggunakan library PIL. Hasil citra dengan efek blur akan ditampilkan di kotak citra output.

```
1 def apply_blur(self):
2     if self.input_image is None:
3         return
4
5     if self.output_image != self.input_image:
6         self.output_image = self.input_image.copy()
7
8     radius = self.blur_scale.get()
9     self.output_image = self.output_image.filter(ImageFilter.GaussianBlur(radius))
10    self.display_output_image()
```

- Metode `apply_negative` dipanggil saat tombol "Negative" ditekan. Metode ini menghasilkan citra negatif dari citra output menggunakan library PIL. Hasil citra negatif akan ditampilkan di kotak citra output.

```
1 def apply_negative(self):
2     if self.input_image is None:
3         return
4
5     if self.output_image != self.input_image:
6         self.output_image = self.input_image.copy()
7
8     self.output_image = ImageOps.invert(self.output_image)
9     self.display_output_image()
```

- Metode `apply_equalize` dipanggil saat tombol "Equalize" ditekan. Metode ini melakukan ekualisasi histogram pada citra output menggunakan library PIL. Hasil citra dengan histogram yang seimbang akan ditampilkan di kotak citra output.

```
1 def apply_equalize(self):
2     if self.input_image is None:
3         return
4
5     if self.output_image != self.input_image:
6         self.output_image = self.input_image.copy()
7
8     self.output_image = ImageOps.equalize(self.output_image)
9     self.display_output_image()
```

- Metode `apply_histogram` dipanggil saat tombol "Matching" ditekan. Metode ini membuka dialog file dan meminta pengguna untuk memilih file citra referensi. Citra referensi akan diubah menjadi array, lalu histogram citra output akan disesuaikan dengan histogram citra referensi menggunakan library skimage. Hasil citra dengan histogram yang disesuaikan akan ditampilkan di kotak citra output.

```
1 def apply_histogram(self):
2     if self.input_image is None:
3         return
4
5     if self.output_image != self.input_image:
6         self.output_image = self.input_image.copy()
7
8     reference_image_path = filedialog.askopenfilename(filetypes=[("Image Files", "*.jpg;*.jpeg;*.png")])
9     if not reference_image_path:
10        return
11
12    reference_image = Image.open(reference_image_path).convert("RGB")
13    reference_array = np.array(reference_image)
14    output_array = np.array(self.input_image.convert("RGB"))
15
16    matched_array = exposure.match_histograms(output_array, reference_array)
17
18    self.output_image = Image.fromarray(matched_array.astype('uint8'))
19    self.display_output_image()
```

- Metode reset dipanggil saat tombol "Reset" ditekan. Metode ini mengembalikan citra output ke citra input semula dan menampilkan citra input di kotak citra output.



```
1 def reset(self):
2     if self.input_image is None:
3         return
4
5     self.output_image = self.input_image.copy()
6     self.display_output_image()
```

- Bagian ini menjalankan program GUI dengan membuat objek ImageEnhancementGUI dan menjalankan mainloop untuk menampilkan GUI.



```
1 if __name__ == "__main__":
2     root = tk.Tk()
3     app = ImageEnhancementGUI(root)
4     root.mainloop()
```