Name : Muhamad Fahraz Firdaus

NIM : 20220040131

Class : TI22 I

Course : Pengolahan Citra Digital

Remedial UTS

Project Case Study:

Edge Detection Displaying the contours of objects in the image using an edge detection algorithm.

Step by Step :

1. Installation and Library Import

Steps:

Ensure the required Python libraries (OpenCV, NumPy, and Matplotlib) are installed.

OpenCV: Used for image manipulation and analysis.

NumPy: Used for numerical data manipulation.

Matplotlib: Used for result visualization.

code :

```
!pip install opencv-python-headless matplotlib numpy
```

2. Reading and Converting the Image

Objective:

To read an image from the file path and convert it into grayscale format. Grayscale format simplifies analysis by reducing the RGB color dimensions into a single channel (intensity of gray shades).

Process:

- cv2.imread(): Reads the image.
- cv2.cvtColor(): Converts the image to grayscale.

code :

```
import cv2
```

```
import numpy as np

import matplotlib.pyplot as plt


# Path ke gambar

image_path = '/content/WIN_20240627_13_59_47_Pro.jpg'   # Ganti dengan
path gambar


# Fungsi untuk membaca gambar dan mengubah ke format grayscale

def read_and_convert_image(image_path):

    # Membaca gambar

    img = cv2.imread(image_path)

    # Konversi ke skala abu-abu

    gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    return img, gray_img
```

3. Edge Detection with the Canny Algorithm

Objective:

To detect edges in the image based on intensity changes using the Canny algorithm.

Process:

- cv2.Canny() takes the grayscale image and two threshold parameters (lower_threshold and upper_threshold) to detect significant pixel intensity changes.
- These parameters can be adjusted for optimal results based on the image's level of detail.

code :

```
# Fungsi untuk mendeteksi tepi menggunakan Canny

def detect_edges(gray_img, lower_threshold=50, upper_threshold=150):

    edges = cv2.Canny(gray_img, lower_threshold, upper_threshold)

    return edges
```

4. Finding and Analyzing Contours

Objective:

To identify contours in the image based on detected edges and analyze the properties of each contour.

Process:

1. cv2.findContours(): Finds all contours in the edge-detected image.
2. The analysis includes calculations for:
    - Area: Using cv2.contourArea().
    - Bounding Box: Using cv2.boundingRect().
    - Centroid: The center point of the bounding box, calculated as (x + w // 2, y + h // 2).

code :

```
# Fungsi untuk menemukan dan menganalisis kontur

def find_and_analyze_contours(edges):

    # Menemukan kontur

          contours,   _   =   cv2.findContours(edges,   cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

    contour_data = []

    for contour in contours:

        # Menghitung luas dan koordinat bounding box

        area = cv2.contourArea(contour)

        x, y, w, h = cv2.boundingRect(contour)

        contour_data.append({

            "Area": area,

            "Bounding Box": (x, y, w, h),

            "Centroid": (x + w // 2, y + h // 2)

        })

    return contours, contour_data
```

5. Drawing Contours on the Original Image

Objective:

To visualize the detected contours by overlaying them on the original image.

Process:

- cv2.drawContours(): Draws all contours in green with a specified line thickness.

code :

```
# Fungsi untuk menggambar kontur pada gambar asli
def draw_contours(img, contours):
    result_img = img.copy()
    cv2.drawContours(result_img, contours, -1, (0, 255, 0), 2)  # Hijau untuk kontur
    return result_img
```

6. Creating and Running a process Function

Objective:

To encapsulate the entire program into a single function named process for easier execution.

code :

```
# Fungsi utama untuk deteksi dan analisis
def process_image(image_path):
    img, gray_img = read_and_convert_image(image_path)
    edges = detect_edges(gray_img)
    contours, contour_data = find_and_analyze_contours(edges)
    result_img = draw_contours(img, contours)
    return result_img, edges, contour_data, contours # Return contours here

# Proses gambar
```

```
original_img, gray_img = read_and_convert_image(image_path)

result_img, edges, contour_data, contours = process_image(image_path) # Assign contours here
```

7. Visualizing Results

Objective:

To display the original image, the edge-detected image, and the image with contours side by side for better understanding of the analysis.

Process:

- plt.subplot(): Splits the view into three sections for parallel visualization of images.

code :

```
# fungsi Visualisasi

def visualize_results(img, edges, result_img):

    plt.figure(figsize=(15, 5))

    plt.subplot(1, 3, 1)

    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))

    plt.title("Gambar Asli")

    plt.axis("off")


    plt.subplot(1, 3, 2)

    plt.imshow(edges, cmap="gray")

    plt.title("Deteksi Tepi")

    plt.axis("off")


    plt.subplot(1, 3, 3)

    plt.imshow(cv2.cvtColor(result_img, cv2.COLOR_BGR2RGB))

    plt.title("Kontur pada Gambar")

    plt.axis("off")
```

```
    plt.show()


# Visualisasi hasil

visualize_results(original_img, edges, result_img)
```

The results obtained

8. Analysis Information

Objective:

To display the properties of the detected contours, such as:

- Number of objects in the image.
- Area: The size of each object.
- Bounding Box: Coordinates (x, y) and dimensions (w, h).
- Centroid: The center point of each object.

code :

```
# Menampilkan informasi objek

print(f"Jumlah objek terdeteksi: {len(contours)}")

print("Informasi tiap objek:")

for i, data in enumerate(contour_data):

    print(f"Objek {i + 1}:")

    print(f"  - Area: {data['Area']:.2f}")

    print(f"  - Bounding Box (x, y, w, h): {data['Bounding Box']}")

    print(f"  - Centroid: {data['Centroid']}")
```

The analysis information that has been gathered

```
Informasi tiap objek:
Objek 1:
  - Area: 107.00
  - Bounding Box (x, y, w, h): (304, 714, 58, 6)
  - Centroid: (333, 717)
Objek 2:
  - Area: 0.00
  - Bounding Box (x, y, w, h): (0, 712, 7, 2)
  - Centroid: (3, 713)
Objek 3:
  - Area: 0.00
  - Bounding Box (x, y, w, h): (473, 705, 3, 15)
  - Centroid: (474, 712)
Objek 4:
  - Area: 0.50
  - Bounding Box (x, y, w, h): (472, 705, 2, 2)
  - Centroid: (473, 706)
```

Complete information is available at this link:

https://github.com/muhamadazz/PengolahanCitraDigital/blob/main/Remedial%20UTS/Analysis_data.csv