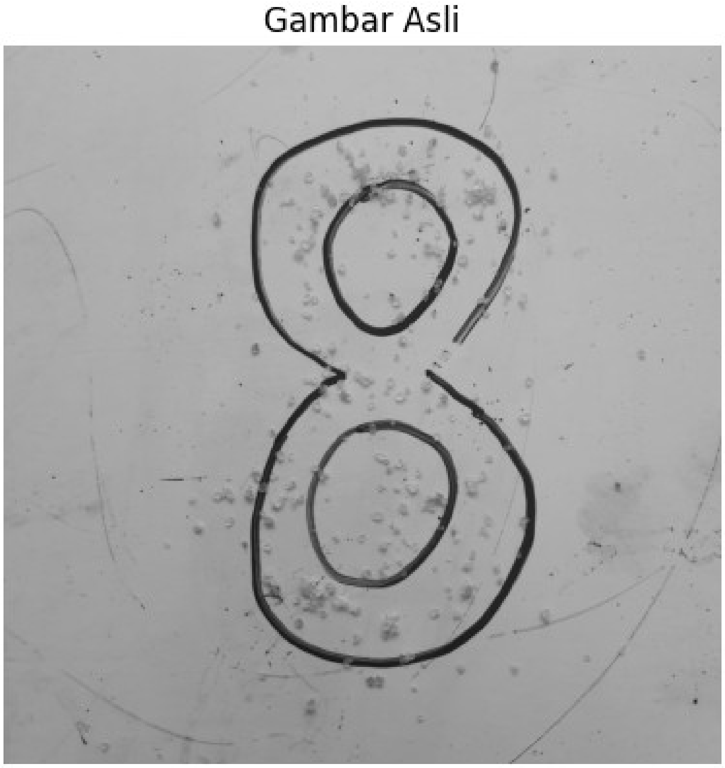
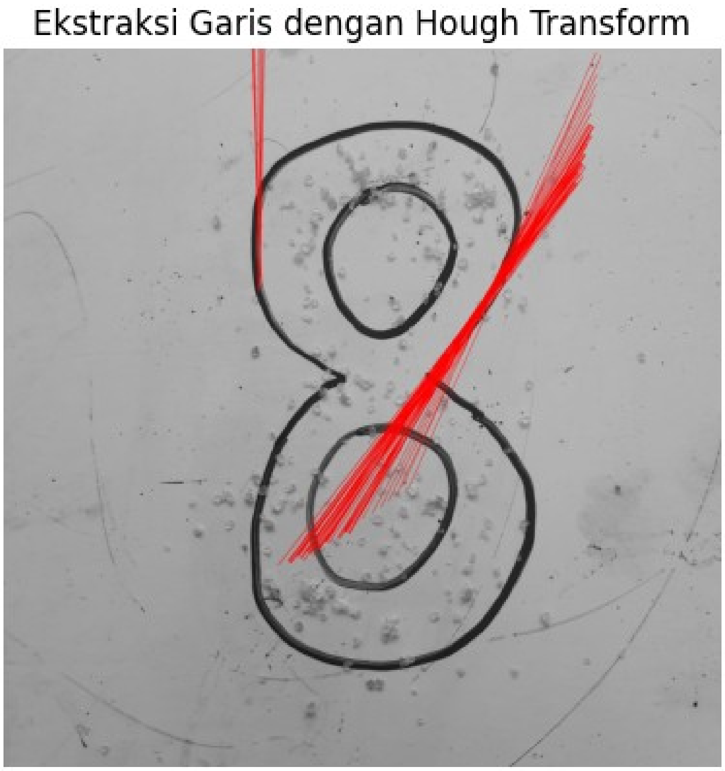
```python
import cv2
import numpy as np
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
from google.colab import files
import io
from PIL import Image

def upload_image():
    # Unggah file gambar menggunakan Google Colab
    uploaded = files.upload()
    for filename in uploaded.keys():
        print(f'File {filename} diunggah')
        return filename

def display_original_image(image_path):
    # Menampilkan gambar asli
    img = cv2.imread(image_path)
    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title('Gambar Asli')
    plt.axis('off')
    plt.show()

# 1. Ekstraksi Garis dengan Hough Transform
def hough_line_detection(image_path):
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    edges = cv2.Canny(gray, 50, 150, apertureSize=3)

    lines = cv2.HoughLines(edges, 1, np.pi / 180, 200)
    if lines is not None:
        for line in lines:
            rho, theta = line[0]
            a = np.cos(theta)
            b = np.sin(theta)
            x0 = a * rho
            y0 = b * rho
            x1 = int(x0 + 1000 * (-b))
            y1 = int(y0 + 1000 * (a))
            x2 = int(x0 - 1000 * (-b))
            y2 = int(y0 - 1000 * (a))
            cv2.line(img, (x1, y1), (x2, y2), (0, 0, 255), 2)

    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title('Ekstraksi Garis dengan Hough Transform')
    plt.axis('off')
    plt.show()

if __name__ == "__main__":
    print("Silakan Unggah Gambar:")
    image_path = upload_image()
    if image_path:
        display_original_image(image_path)
        hough_line_detection(image_path)
```

```
Silakan Unggah Gambar:

<IPython.core.display.HTML object>

Saving 81.jpg to 81.jpg
File 81.jpg diunggah
```




Ekstraksi Garis dengan Hough Transform

Garis merah yang terlihat di atas gambar menunjukkan hasil deteksi garis menggunakan Transformasi Hough berdasarkan tepi yang ditemukan oleh algoritma Canny Edge Detection.

```python
def upload_image():
    # Unggah file gambar menggunakan Google Colab
    uploaded = files.upload()
    for filename in uploaded.keys():
        print(f'File {filename} diunggah')
        return filename

def display_original_image(image_path):
    # Menampilkan gambar asli
    img = cv2.imread(image_path)
    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title('Gambar Asli')
    plt.axis('off')
    plt.show()

# 2. Template Matching untuk Deteksi Objek
def template_matching(image_path, template_path):
    img = cv2.imread(image_path)
    template = cv2.imread(template_path)
    h, w = template.shape[:2]

    res = cv2.matchTemplate(img, template, cv2.TM_CCOEFF_NORMED)
    threshold = 0.8
    loc = np.where(res >= threshold)
    for pt in zip(*loc[::-1]):
        cv2.rectangle(img, pt, (pt[0] + w, pt[1] + h), (0, 255, 0), 2)

    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title('Template Matching')
    plt.axis('off')
    plt.show()

if __name__ == "__main__":
    print("Silakan Unggah Gambar:")
    image_path = upload_image()
    if image_path:
        print("Unggah file template")
        template_path = upload_image()
        display_original_image(image_path)
        if template_path:
            template_matching(image_path, template_path)
```

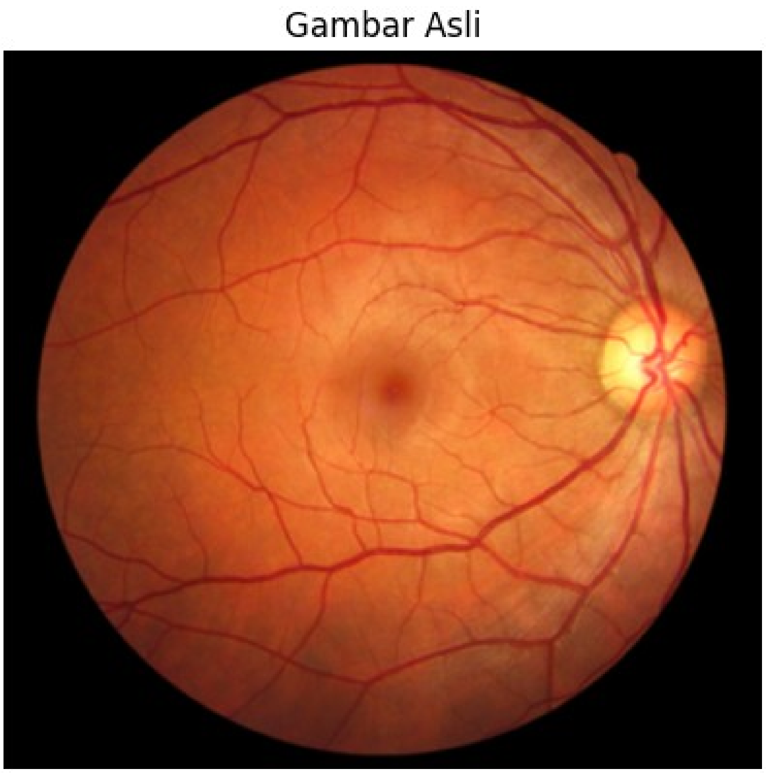
```
Silakan Unggah Gambar:
```

```
<IPython.core.display.HTML object>
```

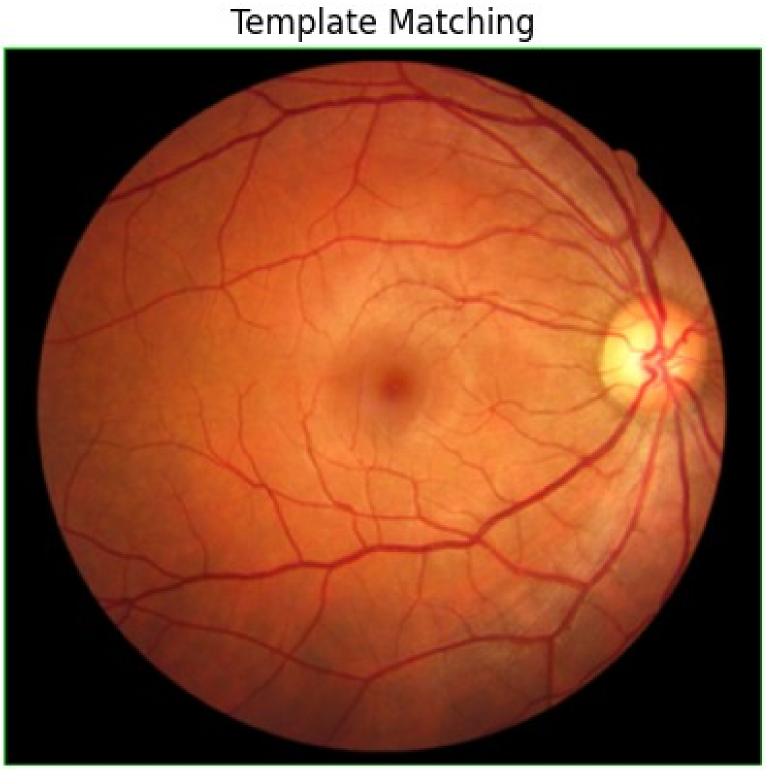
```
Saving 82.jpg to 82.jpg
File 82.jpg diunggah
Unggah file template
```

```
<IPython.core.display.HTML object>
```

```
Saving 82.jpg to 82 (1).jpg
File 82 (1).jpg diunggah
```



Gambar Asli



Template Matching

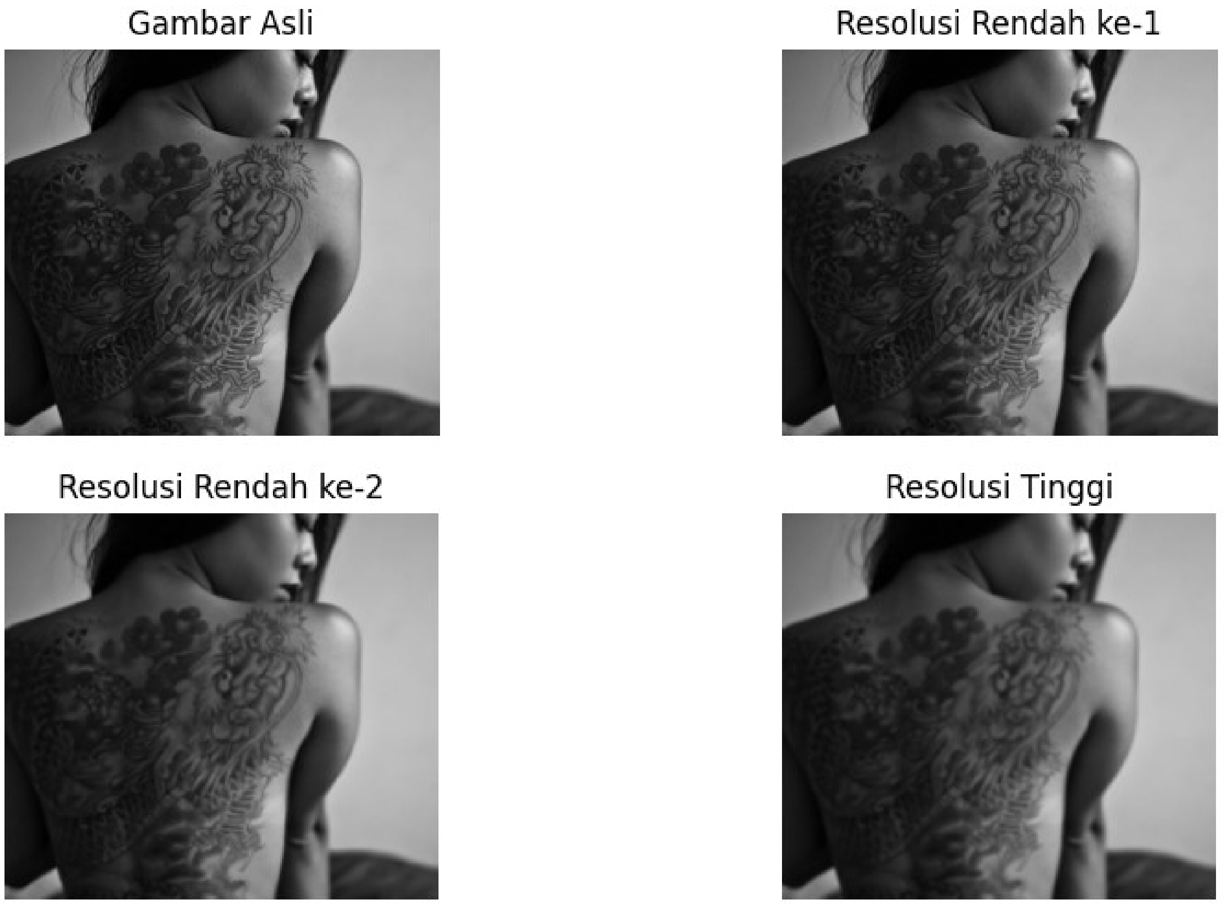
Persegi panjang hijau pada gambar hasil menunjukkan lokasi dalam gambar utama yang cocok dengan template berdasarkan nilai ambang batas (threshold = 0.8).

```
def upload_image():
    # Unggah file gambar menggunakan Google Colab
    uploaded = files.upload()
    for filename in uploaded.keys():
        print(f'File {filename} diunggah')
        return filename

# 3. Pembuatan Pyramid Gambar
def image_pyramid(image_path):
    img = cv2.imread(image_path)

    lower_reso1 = cv2.pyrDown(img)
    lower_reso2 = cv2.pyrDown(lower_reso1)
    higher_reso = cv2.pyrUp(lower_reso2)

    plt.figure(figsize=(10, 6))
    plt.subplot(2, 2, 1)
    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title('Gambar Asli')
    plt.axis('off')

    plt.subplot(2, 2, 2)
    plt.imshow(cv2.cvtColor(lower_reso1, cv2.COLOR_BGR2RGB))
    plt.title('Resolusi Rendah ke-1')
    plt.axis('off')

    plt.subplot(2, 2, 3)
    plt.imshow(cv2.cvtColor(lower_reso2, cv2.COLOR_BGR2RGB))
    plt.title('Resolusi Rendah ke-2')
    plt.axis('off')

    plt.subplot(2, 2, 4)
    plt.imshow(cv2.cvtColor(higher_reso, cv2.COLOR_BGR2RGB))
    plt.title('Resolusi Tinggi')
    plt.axis('off')
    plt.show()

if __name__ == "__main__":
    print("Silakan Unggah Gambar:")
    image_path = upload_image()
    if image_path:
        image_pyramid(image_path)
```

```
Silakan Unggah Gambar:
```

```
<IPython.core.display.HTML object>
```

```
Saving cek.jpg to cek.jpg
File cek.jpg diunggah
```



Gambar Asli

Resolusi Rendah ke-1

Resolusi Rendah ke-2

Resolusi Tinggi

pada hasil filter pada gambar ini menunjukan gambar yg di turunkan resolusinya menjadi lebih halus dan gammbar yang dinaikan ketajamannya menjadi kasar dan agak berbintik

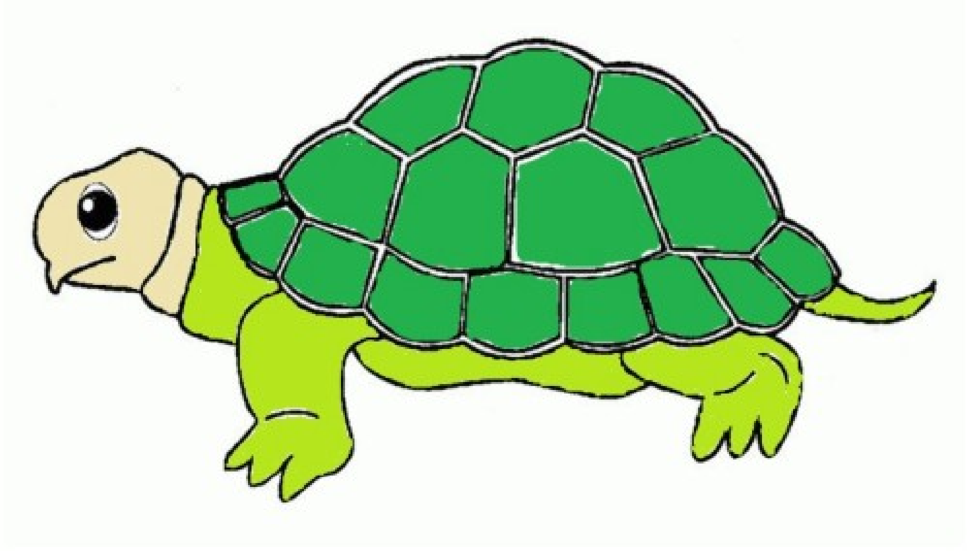
```python
def upload_image():
    # Unggah file gambar menggunakan Google Colab
    uploaded = files.upload()
    for filename in uploaded.keys():
        print(f'File {filename} diunggah')
        return filename

def display_original_image(image_path):
    # Menampilkan gambar asli
    img = cv2.imread(image_path)
    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title('Gambar Asli')
    plt.axis('off')
    plt.show()

# 4. Deteksi Lingkaran Menggunakan Hough Transform
def hough_circle_detection(image_path):
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    gray = cv2.medianBlur(gray, 5)

    circles = cv2.HoughCircles(gray, cv2.HOUGH_GRADIENT, 1, 20,
param1=50, param2=30, minRadius=0, maxRadius=0)
    if circles is not None:
        circles = np.uint16(np.around(circles))
        for i in circles[0, :]:
            cv2.circle(img, (i[0], i[1]), i[2], (0, 255, 0), 2)
            cv2.circle(img, (i[0], i[1]), 2, (0, 0, 255), 3)

    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title('Hough Circle Transform')
    plt.axis('off')
    plt.show()

if __name__ == "__main__":
    print("Silakan Unggah Gambar:")
    image_path = upload_image()
    if image_path:
        display_original_image(image_path)
        hough_circle_detection(image_path)
```

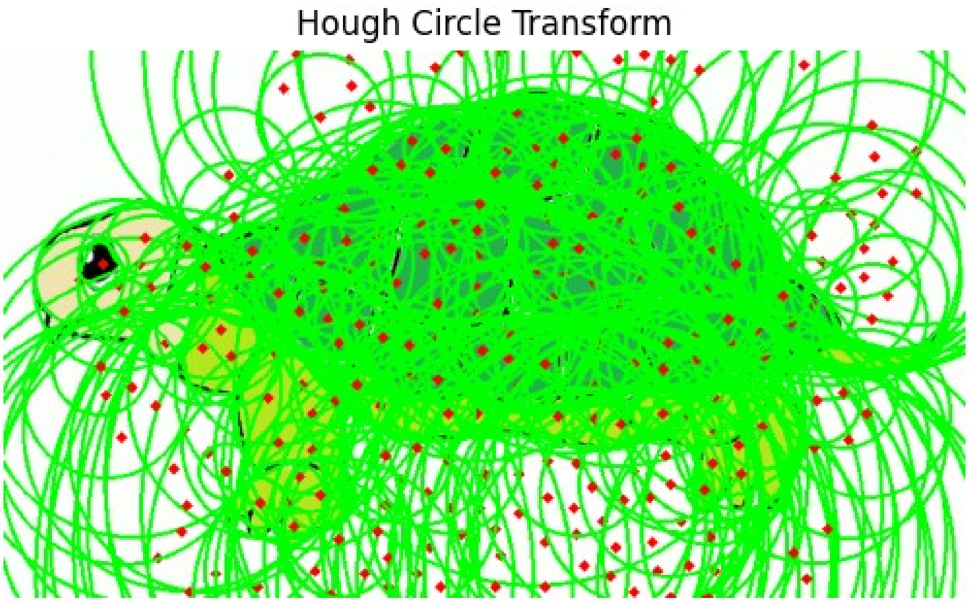
```
Silakan Unggah Gambar:
```

```
<IPython.core.display.HTML object>
```

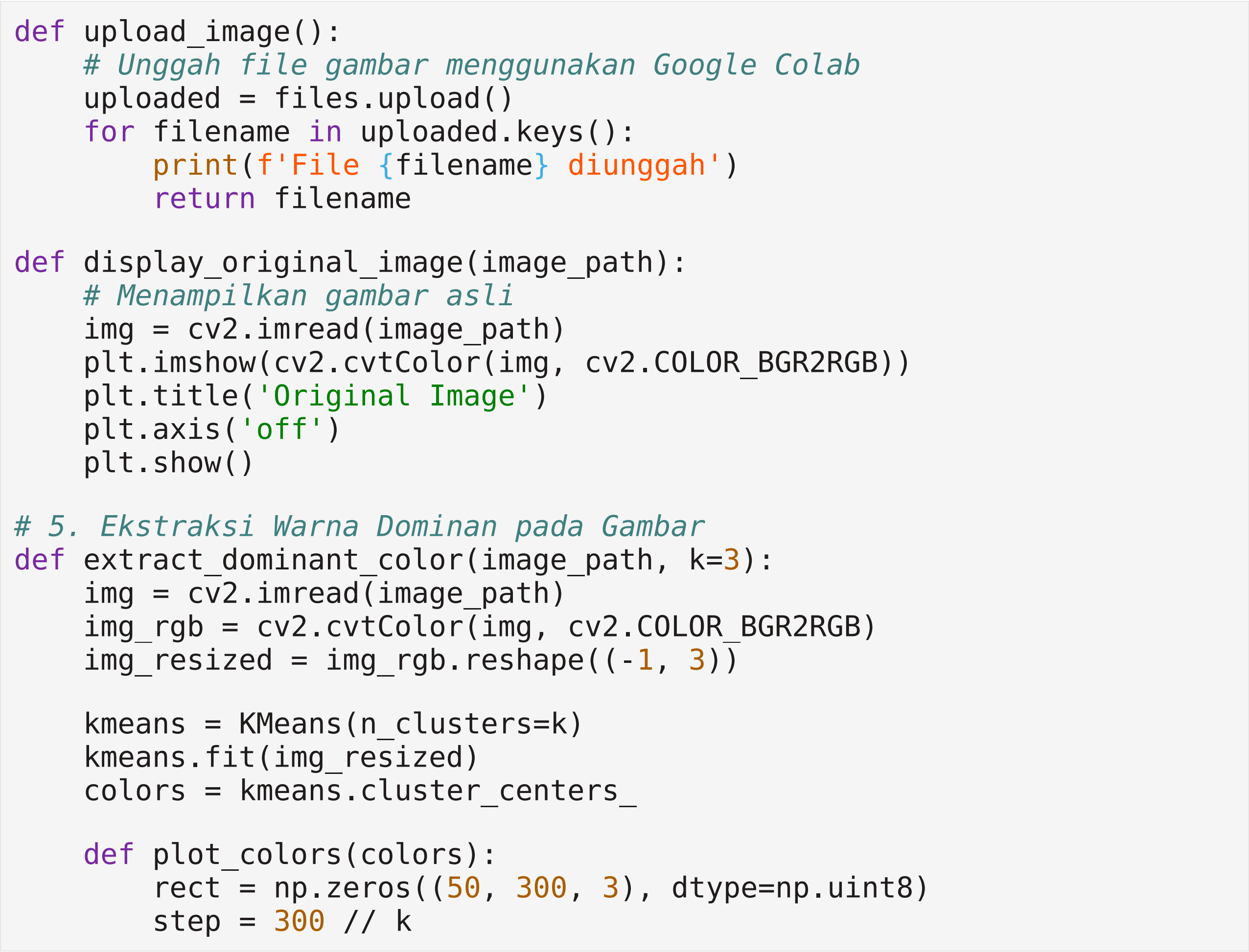
```
Saving kurawarna.jpg to kurawarna.jpg
File kurawarna.jpg diunggah
```

Gambar Asli



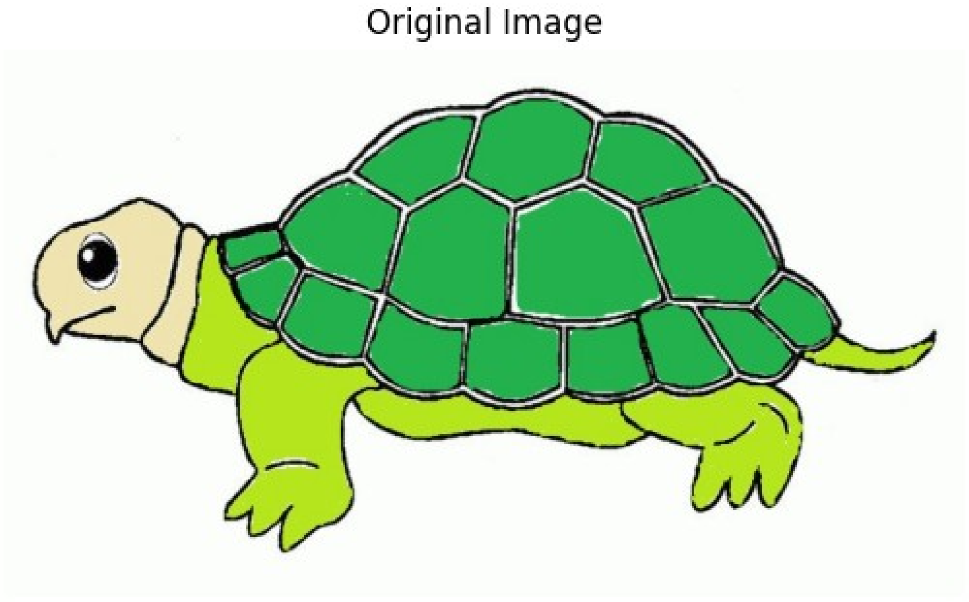


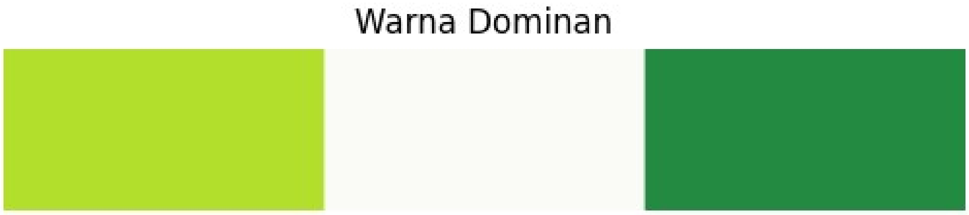
memproses gambar dalam format grayscale dan kemudian menerapkan filter median untuk mengurangi noise. gambar kura-kura yang terdeteksi ditandai dengan warna hijau



```
def upload_image():
    # Unggah file gambar menggunakan Google Colab
    uploaded = files.upload()
    for filename in uploaded.keys():
        print(f'File {filename} diunggah')
        return filename

def display_original_image(image_path):
    # Menampilkan gambar asli
    img = cv2.imread(image_path)
    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title('Original Image')
    plt.axis('off')
    plt.show()

# 5. Ekstraksi Warna Dominan pada Gambar
def extract_dominant_color(image_path, k=3):
    img = cv2.imread(image_path)
    img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img_resized = img_rgb.reshape((-1, 3))

    kmeans = KMeans(n_clusters=k)
    kmeans.fit(img_resized)
    colors = kmeans.cluster_centers_

    def plot_colors(colors):
        rect = np.zeros((50, 300, 3), dtype=np.uint8)
        step = 300 // k
        for i, color in enumerate(colors):
            rect[:, i*step:(i+1)*step] = color
        return rect

    color_rect = plot_colors(colors)
    plt.imshow(color_rect)
    plt.axis('off')
    plt.title('Warna Dominan')
    plt.show()

if __name__ == "__main__":
    print("Silakan Unggah Gambar:")
    image_path = upload_image()
    if image_path:
        display_original_image(image_path)
        extract_dominant_color(image_path)
```

```
Silakan Unggah Gambar:

<IPython.core.display.HTML object>

Saving kurawarna.jpg to kurawarna (1).jpg
File kurawarna (1).jpg diunggah
```





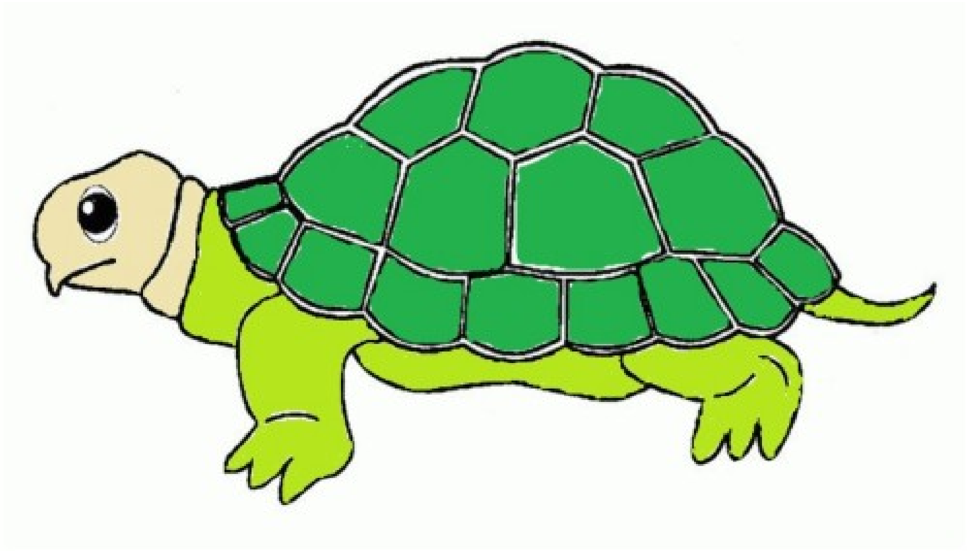
Program ini sangat berguna untuk aplikasi yang melibatkan analisis warna, seperti pengenalan objek berdasarkan warna. dan hanya menganalisis warna yang banyak keluar dari gambar

```
def upload_image():
    # Unggah file gambar menggunakan Google Colab
    uploaded = files.upload()
    for filename in uploaded.keys():
        print(f'File {filename} diunggah')
        return filename

def display_original_image(image_path):
    # Menampilkan gambar asli
    img = cv2.imread(image_path)
    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title('Gambar Asli')
    plt.axis('off')
    plt.show()

# 6. Deteksi Kontur pada Gambar
def contour_detection(image_path):
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    _, thresh = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)
    contours, _ = cv2.findContours(thresh, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)

    cv2.drawContours(img, contours, -1, (0, 255, 0), 2)

    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title('Deteksi Kontur')
    plt.axis('off')
    plt.show()

if __name__ == "__main__":
    print("Silakan Unggah Gambar:")
    image_path = upload_image()
    if image_path:
        display_original_image(image_path)
        contour_detection(image_path)
```

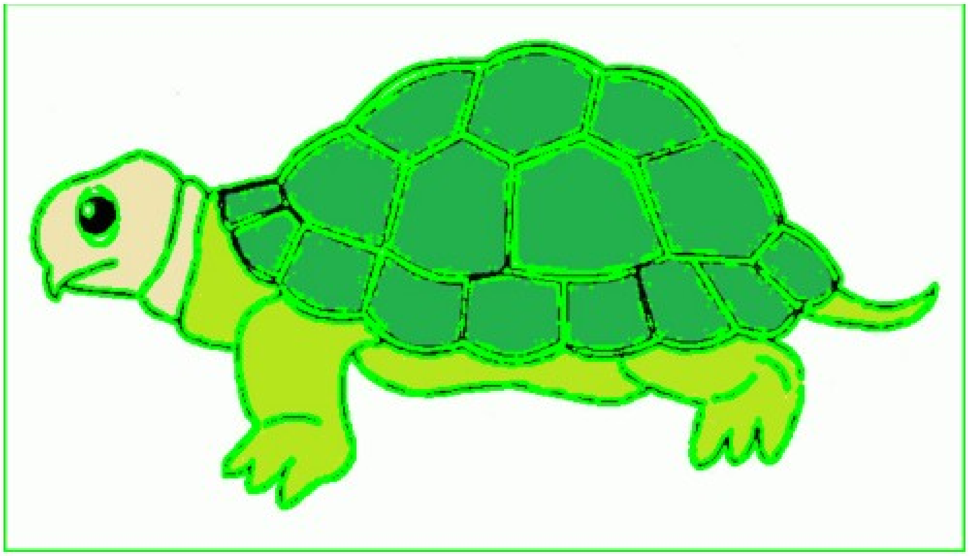
```
Silakan Unggah Gambar:
```

```
<IPython.core.display.HTML object>
```

```
Saving kurawarna.jpg to kurawarna (2).jpg
File kurawarna (2).jpg diunggah
```



Gambar Asli



Deteksi Kontur

kontur akan memeberi warna hijau agar dapat membantu dalam pengenalan bentuk, segmentasi objek, dan analisis citra.