**1. Blur (Box Blur)**

Blur biasa atau box blur bekerja dengan menghitung rata-rata dari piksel-piksel di sekitar setiap piksel dalam gambar, menghasilkan gambar yang lebih halus.

**Kode untuk Blur:**

python

Copy code

import cv2

# Membaca gambar

image = cv2.imread('image.jpg')

# Menerapkan blur biasa (Box Blur)

blurred\_image = cv2.blur(image, (5, 5)) # Ukuran kernel 5x5

# Menampilkan gambar asli dan hasil blur

cv2.imshow('Gambar Asli', image)

cv2.imshow('Blur Biasa', blurred\_image)

cv2.waitKey(0)

cv2.destroyAllWindows()

**2. Gaussian Blur**

Gaussian Blur lebih halus dibandingkan dengan box blur karena menggunakan distribusi Gaussian untuk menghitung rata-rata nilai piksel, memberikan bobot lebih pada piksel pusat dan mengurangi bobot piksel di sekitarnya.

**Kode untuk Gaussian Blur:**

python

Copy code

import cv2

# Membaca gambar

image = cv2.imread('image.jpg')

# Menerapkan Gaussian blur

gaussian\_blurred\_image = cv2.GaussianBlur(image, (5, 5), 0)

# Menampilkan gambar asli dan hasil Gaussian blur

cv2.imshow('Gambar Asli', image)

cv2.imshow('Gaussian Blur', gaussian\_blurred\_image)

cv2.waitKey(0)

cv2.destroyAllWindows()

**3. Sharpening**

Sharpening digunakan untuk meningkatkan kontras dan ketajaman gambar dengan menyoroti detail yang ada. Ini dilakukan dengan menggunakan filter yang menekankan perbedaan intensitas antara piksel dan tetangganya.

**Kode untuk Sharpening:**

python

Copy code

import cv2

import numpy as np

# Membaca gambar

image = cv2.imread('image.jpg')

# Matriks kernel untuk sharpening

kernel\_sharpening = np.array([[-1, -1, -1],

[-1, 9, -1],

[-1, -1, -1]])

# Menerapkan filter sharpening

sharpened\_image = cv2.filter2D(image, -1, kernel\_sharpening)

# Menampilkan gambar asli dan gambar yang sudah di-sharpening

cv2.imshow('Gambar Asli', image)

cv2.imshow('Sharpening', sharpened\_image)

cv2.waitKey(0)

cv2.destroyAllWindows()

**4. Edge Detection (Deteksi Tepi)**

Edge detection digunakan untuk menemukan batas atau kontur objek dalam gambar. Salah satu teknik paling populer adalah **Canny Edge Detection**, yang sangat efektif dalam menemukan tepi objek.

**Kode untuk Edge Detection (Canny):**

python

Copy code

import cv2

# Membaca gambar

image = cv2.imread('image.jpg')

# Mengubah gambar menjadi grayscale (Canny membutuhkan gambar grayscale)

gray\_image = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

# Menerapkan Canny edge detection

edges = cv2.Canny(gray\_image, 100, 200) # 100 dan 200 adalah nilai threshold

# Menampilkan gambar asli dan hasil deteksi tepi

cv2.imshow('Gambar Asli', image)

cv2.imshow('Edge Detection (Canny)', edges)

cv2.waitKey(0)

cv2.destroyAllWindows()

**Penjelasan:**

1. **Blur**: Menghaluskan gambar dengan menghitung rata-rata intensitas piksel.
2. **Gaussian Blur**: Menggunakan distribusi Gaussian untuk menghitung blur, memberikan efek yang lebih alami dan halus.
3. **Sharpening**: Menggunakan filter untuk meningkatkan ketajaman gambar dan menekankan detail objek.
4. **Edge Detection**: Menggunakan algoritma (seperti Canny) untuk mendeteksi tepi objek dalam gambar, sering digunakan untuk segmentasi objek.