**PRAKTIKUM 11**

# Segmentasi Citra
# (*Image Segmentation*)

**Materi:**

- Thresholding dan deteksi tepi (*edge detection*)
- Contoh implementasi segmentasi citra pada OpenCV

**Tujuan Praktikum:**

- Mahasiswa dapat mengimplementasikan teknik thresholding dan edge detection menggunakan OpenCV

## A. PENYAJIAN

### Segmentasi Citra

Segmentasi citra merupakan proses mempartisi citra menjadi beberapa daerah atau objek. Umumnya segmentasi bertujuan untuk memisahkan suatu objek citra dengan objek lainnyadan atau memisahkan objek citra dengan latar belajang citra. Segmentasi citra didasarkan pada dua hal yaitu diskontinuitas dan kemiripan intensitas piksel. Pendekatan diskontinuitas disebut juga pendekatan berbasis edge (*edge-based*), sementara pendekatan kemiripan intensitas, seperti thresholding, clustering (contoh: k-means), klasifikasi, pendekatan teori graf (contoh: GrabCut), region growing, dan region splitting & merging.

### Thresholding

Thresholding merupakan teknik yang paling dasar untuk melakukan segmentasi citra. Teknik thresholding dapat digunakan untuk melakukan segmentasi citra berdasaran warna tertentu atau berdasarkan suatu nilai piksel tertentu.

### Deteksi Tepi (Edge Detection)

Deteksi tepi adalah operasi yang dijalankan untuk mendeteksi garis tepi yang membatasi dua wilayah citra homogen yang memiliki tingkat kecerahan yang berbeda. Edge adalah bagian dari citra dimana intensitas kecerahan berubah secara drastis. Tujuan dari deteksi tepi pada umumnya yaitu untuk mengurangi jumlah data pada citra pada langkah image processing berikutnya. Metode edge detection dibagi menjadi dua, yaitu *first order edge detection* dan *second order edge detection*.

### First Order Derivative Edge Detection

Metode deteksi tepi yang termasuk kedalam first order derivative adalah Roberts operators, Prewitt operators, Sobel operators, dan First-order of gaussian (FDOG). Pada modul ini akan dicoba untuk operator Sobel, Prewitt, dan Roberts. Setiap operator memiliki row gradient dan column gradient.

**Roberts Operator**

| | | |
|---|---|---|
| $z_1$ | $z_2$ | $z_3$ |
| $z_4$ | $z_5$ | $z_6$ |
| $z_7$ | $z_8$ | $z_9$ |

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | | 0 | 0 | 0 |
| 0 | -1 | 0 | | 0 | 0 | -1 |
| 0 | 0 | 0 | | 0 | 1 | 0 |

$G_x = (z_9 - z_5)$ $G_y = (z_8 - z_6)$

**Prewitt Operator**

| | | | | | | |
|---|---|---|---|---|---|---|
| -1 | -1 | -1 | | -1 | 0 | 1 |
| 0 | 0 | 0 | | -1 | 0 | 1 |
| 1 | 1 | 1 | | -1 | 0 | 1 |

$G_x = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$ $G_y = (z_1 + z_6 + z_9) - (z_1 + z_4 + z_7)$

**Robert Operator**

| | | | | | | |
|---|---|---|---|---|---|---|
| -1 | -2 | -1 | | -1 | 0 | 1 |
| 0 | 0 | 0 | | -2 | 0 | 2 |
| 1 | 2 | 1 | | -1 | 0 | 1 |

$G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$ $G_y = (z_1 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$

### Second Order Derivative Edge Detection

#### Canny detection

Materi lebih lanjut tentang Canny dapat dilihat pada file Canny Operator di LMS. Untuk melakukan canny, dilakukan 5 langkah berikut:

1. Smoothing, melakukan blurring untuk meghilangkan noise.
2. Menemukan gradien

   Algortime Canny pada dasarnya menemukan tepi dengan mencari area yang memiliki perubahan intensitas grayscale yang paling signifikan. Area ini ditemukan dengan mencari gradien. Jenis operator Sobel dapat digunakan untuk mendapatkan gradien.

3. Non-maximum suppresion

   Tujuan dari step ini adalah untuk meng-*convert* edge 'blur' dari step sebelumnya menjadi edge yang 'sharp'. Algoritme ini dilakukan untuk setiap pixel pada citra gradient. Hanya local maxima yang ditandai sebagai edge.

4. Double thresholding

   Hasil dari non-maximum suppresion memang sudah mencirikan edge asli, tetapi bisa saja edge tersebut dihasilkan oleh noise. Maka dari itu thresholding dilakukan sehingga edge yang digunakan hanya edge yang paling kuat. Canny menggunakan dua threshold. Edge yang lebih kuat dari threshold tertinggi ditandai sebagai 'strong', edge yang lebih lemah dari threshold terendah ditandai sebagai 'weak'.

5. Edge tracking menggunakan hysteresis

   Edge tracking dilakukan untuk melihat edge mana saja yang termasuk ke true edge. Tidak semua edge strong adalah true edge, begitupun tidak semua edge weak adalah noise. Edge tracking dapat diimplementasikan menggunakan analisis BLOB (Binary Large Object). Pixel edge dibagi kedalam BLOB connected neighborhood.

## B. LATIHAN

Latihan Canny Operator

```
#include <iostream>
#include <cv.h>
#include <highgui.h>

using namespace std;
using namespace cv;

int main()
```

```
{
    Mat src, src_gray;
    Mat dst;

    src = imread("Lenna.png");
    dst.create(src.size(),src.type());

    cvtColor(src,src_gray,CV_BGR2GRAY);


    Canny(src_gray,dst,50,100,3);
    imshow("test",dst);

    waitKey(0);
}
```

Latihan Sobel Operator

```
#include <iostream>
#include <cv.h>
#include <highgui.h>

using namespace std;
using namespace cv;

int main()
{
    Mat src, src_gray;
    Mat dst;
    Mat grad;


    src = imread("Lenna.png");
    dst.create(src.size(),src.type());

    GaussianBlur( src, src, Size(3,3), 0, 0, BORDER_DEFAULT );

    cvtColor(src,src_gray,CV_BGR2GRAY);

    Sobel( src_gray, grad, CV_16S, 1, 0, 3, 1, 0, BORDER_DEFAULT );
    convertScaleAbs(grad,dst);
    imshow("test",dst);

    waitKey(0);
}
```

## C. TUGAS

1. Lakukan sobel filter pada citra lena dengan nilai untuk parameter x_order=1, y_order=0, KernelSize=3,5 dan 7. Serta, lakukan sobel filter pada citra lena dengan nilai untuk parameter (x_order=0, y_order=1, KernelSize=3,5 dan 7). Bandingkan dan analisis hasil yang diberikan.