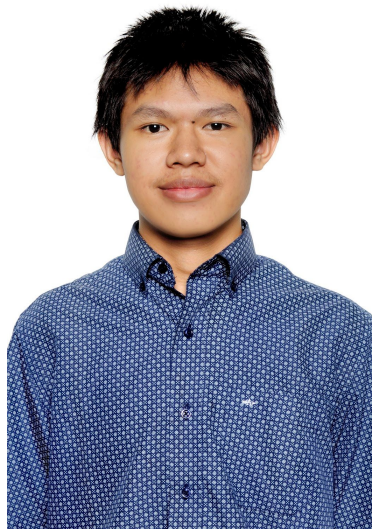


Laporan Tugas Besar IF4073 Interpretasi dan Pengolahan Citra

Automatic Plate Number Recognition

Disusun oleh:

Muhammad Alif Arifin 13516078



Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

2019

Daftar Isi

Daftar Isi	1
Pendahuluan	2
Dasar Teori	3
Citra Biner	3
Truecolor ke Grayscale	3
Pendeteksian Tepi pada Citra (Edge Detection)	3
Segmentasi Citra (Image Segmentation)	4
Resize Citra	4
Connected Component Labeling	5
Rancangan dan Implementasi	7
Deteksi Plat	7
Deteksi Karakter	8
Memahami Karakter	8
Pengujian	9
Hasil Pengujian	9
Akurasi	12
Kesimpulan	13

1. Pendahuluan

Pada tugas besar IF4073 Interpretasi dan Pengolahan Citra ini, persoalan yang diberikan adalah mengenali nomor plat kendaraan. Pengenalan nomor plat ini berguna untuk mencatat kendaraan yang melewati suatu area, seperti jalan tol ataupun tempat parkir. Di samping itu, pengenalan nomor plat kendaraan juga dapat digunakan untuk mendeteksi kendaraan yang melakukan pelanggaran lalu lintas seperti melebihi kecepatan yang seharusnya.



Gambar 1. Bentuk plat nomor kendaraan di Indonesia

Batasan masalah untuk tugas besar ini yaitu pengenalan nomor plat khusus untuk kendaraan roda empat (mobil). Masukan dari program yang dibuat berupa citra kendaraan tampak depan atau tampak belakang yang mengandung plat nomor kendaraan. Sedangkan keluaran dari program yaitu rangkaian karakter (string) dari nomor plat kendaraan.

2. Dasar Teori

2.1. Citra Biner

Citra biner adalah citra yang hanya memiliki dua nilai saja, yaitu warna hitam dan warna putih. Citra biner digunakan karena memiliki konsumsi memori yang sedikit (hanya 1 bit) dan digunakan untuk merepresentasikan hasil deteksi tepi pada citra. Citra biner juga digunakan untuk memisahkan (segmentasi) objek dari latar belakangnya.



Gambar 2. Contoh hasil deteksi tepi pada citra biner.

2.2. Truecolor ke Grayscale

Perubahan citra dari berwarna ke hitam putih dilakukan untuk memberikan abstraksi dan mempermudah melakukan deteksi tepi. Sehingga, sebelum dilakukan deteksi tepi dilakukan perubahan citra dari citra berwarna ke citra hitam putih. Perubahan dilakukan dengan menggunakan algoritma berikut.

$$0.2126 R + 0.7152 G + 0.0722 B$$

2.3. Pendeteksian Tepi pada Citra (*Edge Detection*)

Tepi adalah perubahan nilai intensitas derajat keabuan yang tinggi dalam jarak yang singkat. Tepi biasanya terdapat pada batas antara dua daerah yang berbeda intensitas dengan

perubahan yang sangat cepat di dalam citra. Empat macam tepi yaitu tepi curam (step edge), tepi landau (ramp edge), tepi garis (line edge), dan tepi atap (roof edge).

$$G_x = \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \quad G_y = \begin{bmatrix} -1 & 1 \end{bmatrix}$$

Gambar 3. Kernel yang digunakan untuk deteksi tepi dengan gradient

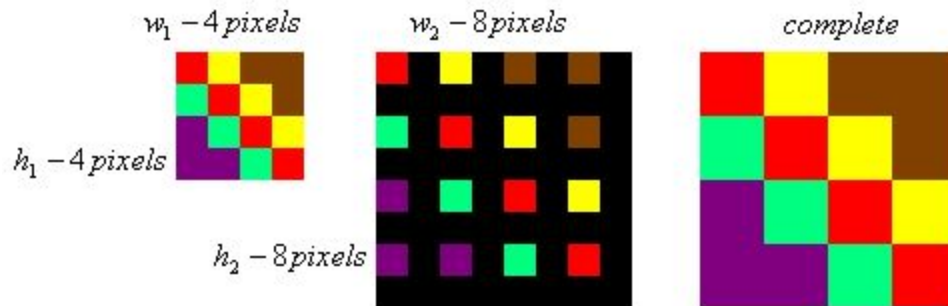
Pendeteksian tepi bertujuan untuk meningkatkan penampakan garis batas atau objek di dalam citra. Pendeteksian tepi mengekstraksi representasi gambar garis-garis di dalam citra. Banyak cara yang dapat dilakukan untuk mendeteksi tepi pada citra di antaranya adalah operator gradien, operator turunan kedua (Laplace), operator Laplace of Gaussian (LoG), operator Sobel, operator Roberts, operator Prewitt, dan operator Canny. Ide algoritma pendeteksian tepi adalah menapis citra dengan filter yang telah ditentukan oleh masing-masing operator.

2.4. Segmentasi Citra (*Image Segmentation*)

Segmentasi citra bertujuan untuk membagi citra menjadi region-region atau objek-objek. Segmentasi citra merupakan tahapan sebelum melakukan *image/object recognition*. Beberapa cara yang digunakan untuk melakukan segmentasi citra di antaranya adalah *threshloding*, *region growing*, *split and merge*, dan *clustering*. Segmentasi citra yang dilakukan hanya untuk citra *grayscale* menjadi citra biner.

2.5. Resize Citra

Resize citra diperlukan ketika ingin membandingkan citra karakter yang didapat dengan *template* yang ada. Resize dilakukan dengan menggunakan algoritma *nearest neighbor*. *Resize* ini akan memperhatikan rasio perubahan dari *image* sebelumnya dengan *image* yang baru. Apabila *scaling* dilakukan tidak genap maka akan dilakukan pembulatan untuk menentukan pixel mana yang akan diduplikasi.



Gambar 4. Contoh *resize*

2.6. Connected Component Labeling

Fungsi CCL diimplementasi dengan menggunakan CCL *two pass*. Berikut merupakan *pseudocode* dari CCL.

```

algorithm TwoPass(data)
    linked = []
    labels = structure with dimensions of data, initialized with
the value of Background

    First pass

    for row in data:
        for column in row:
            if data[row][column] is not Background

                neighbors = connected elements with the current
element's value

                if neighbors is empty
                    linked[NextLabel] = set containing NextLabel
                    labels[row][column] = NextLabel
                    NextLabel += 1

                else

                    Find the smallest label

                    L = neighbors labels
                    labels[row][column] = min(L)
                    for label in L

```

```
linked[label] = union(linked[label], L)

Second pass

for row in data
    for column in row
        if data[row][column] is not Background
            labels[row][column] = find(labels[row][column])

return labels
```

3. Rancangan dan Implementasi

Rancangan dan implementasi sistem dibangun sebagai berikut.

3.1. Deteksi Plat

Deteksi plat dilakukan awalnya dengan mengubah citra berwarna menjadi citra hitam putih. Lalu citra hitam putih akan diubah menjadi citra biner dengan menggunakan segmentasi biner dengan threshold sebesar 80. Lalu citra yang sudah menjadi biner akan dicari tepi dengan pendeteksi tepi *gradient*.

Setelah dicari tepi, akan dilakukan dilatasi sebanyak 1 kali. Dilatasi dilakukan dengan melakukan *convolusi* dengan kernel $3 \times 3 \{1, 1, 1, 1, 1, 1, 1, 1, 1\}$. Setelah itu, dicari pencarian kontur dengan menggunakan algoritma CCL (Connected Component Labelling). Dari seluruh kontur yang ada dibuat *boundary box* dan di filter dicari *boundary box* yang memiliki kemiripan dengan rasio plat nomor Indonesia.

Dari seluruh *boundary box*, dilakukan pendekatan *boundary box* yang sekiranya memiliki karakter di dalamnya. Pendekatan ini dilakukan dengan cara membuat *boundary box* untuk setiap kontur di dalam *boundary box* yang memiliki *boundary box* seperti huruf. Dari situ dicari *boundary box* yang memiliki karakter tertinggi dan dianggap menjadi plat. Berikut merupakan contoh hasil deteksi plat.



Gambar 5. Hasil deteksi plat

3.2. Deteksi Karakter

Deteksi karakter dilakukan dengan menggunakan pendekatan yang sama dengan plat. Dari plat ditarik garis lurus melintang horizontal dengan ketinggian di satu per tiga dari tinggi citra. Terdapat eliminasi *boundary box* apabila memiliki ukuran yang lebih besar atau lebih kecil dari rata-rata. Dan diambil seluruh *boundary box* yang beririsan dengan garis tersebut dan memiliki rasio yang sama dengan karakter. Berikut merupakan contoh hasil deteksi karakter.



Gambar 6. Hasil deteksi karakter

3.3. Memahami Karakter

Memahami karakter dilakukan dengan perbandingan dengan *template* yang sudah ada. Sebelum dibandingkan dilakukan *resize* pada kedua gambar (*template* dan gambar asli) agar dapat dibandingkan untuk setiap *pixel*-nya. Perbandingan dilakukan pada seluruh karakter dan dilakukan perhitungan *error* terkecil. *Error* yang paling kecil merupakan karakter yang dimaksud.




4. Pengujian




Untuk bagian pengujian, dilakukan dengan menggunakan 25 kasus uji. Kasus uji berupa citra kendaraan baik tampak depan ataupun tampak belakang.



4.1. Hasil Pengujian

Pengujian dilakukan dengan 3 kasus uji citra kendaraan roda empat (mobil) tampak depan ataupun tampak belakang.

Tabel 1. Hasil Pengujian

No.	Gambar	Hasil	Akurasi
1		B8OJWYN	6/7
2		1262RFS	5/8
3		78PTCL1NA	4/8

4		B1JKW	1
5		8WV1396N01	7/8
6		524	3/7

7		83241TUC	7/8
8		01T1PLR	5/7

9		IIN	0
10		8AI2QS	3/8

4.2. Akurasi

Dari 10 hasil percobaan didapatkan akurasi sebesar 62,5%

5. Kesimpulan

Sistem *automatic plate number recognition* yang dibuat telah mampu mengenali plat nomor kendaraan walaupun masih memiliki banyak kekurangan. Terlebih pada plat nomor yang tidak berwarna hitam dan mobil yang berwarna hitam (sulit untuk menemukan tepi-nya).

Daftar Pustaka

- Munir, Rinaldi. (2019). *Slide kuliah interpretasi dan pengolahan citra*. Diakses dari <http://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2019-2020/citra19-20.htm> (pada tanggal 20 November 2019)
- Tech Algorithm. (2007). Nearest Neighbor Image Scaling. <http://tech-algorithm.com/articles/nearest-neighbor-image-scaling/> (pada tanggal 20 November 2019)
- Wikipedia. Connected Component Labeling https://en.wikipedia.org/wiki/Connected-component_labeling (pada tanggal 20 November 2019)