

BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1 Model Pengembangan

Tujuan dari tugas akhir ini adalah untuk dapat mendeteksi letak posisi plat nomor mobil dengan menggunakan metode morfologi. Operasi morfologi yang akan digunakan adalah dilasi. Operasi dilasi digunakan untuk memperbesar ukuran segmen objek dengan menambah lapisan di sekeliling objek. Biasanya operasi dilasi ini diterapkan dalam objek-objek yang terputus disebabkan hasil pada saat pengambilan citra yang terganggu oleh noise, kerusakan objek fisik yang dijadikan citra digital, atau dapat juga dikarenakan resolusi yang jelek.

3.2 Prosedur Penelitian

Prosedur penelitian yang dipakai dalam pengerjaan tugas akhir ini adalah:

1. Studi literatur

Pada penelitian ini terdapat perancangan yang akan dilakukan yaitu, perancangan perangkat lunak. Adapun metode penelitian yang dilakukan antara lain:

Pencarian data-data literatur untuk perangkat lunak dari masing-masing informasi dari internet dan konsep teoritis dari buku-buku penunjang tugas akhir ini, serta materi-materi perkuliahan yang telah didapatkan dan perancangan perangkat lunak yaitu menggunakan OpenCV dan Microsoft Visual Studio melalui pencarian dari internet, dan beberapa konsep teoritis dari beberapa buku penunjang

tersebut, dari kedua bagian ini akan dipadukan agar dapat bekerja sama untuk dapat menjalankan sistem dengan baik.

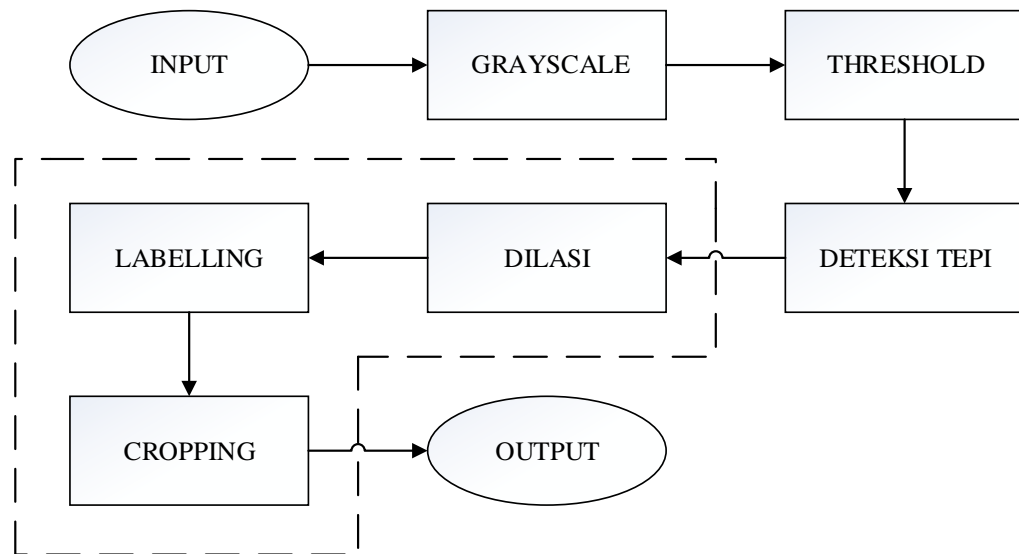
2. Tahap perancangan dan pengembangan sistem

Dalam membuat pengembangan sistem, terdapat beberapa langkah rancangan sistem yang diambil antara lain:

- a. Membuat Blok Diagram pada proses sistem secara keseluruhan
- b. Melakukan perancangan perangkat lunak yang meliputi:
 - Merancang program *pre-processing* citra yang meliputi proses *grayscale*, *thresholding*, *dilasi*, *labelling* dan *cropping* untuk mempermudah proses pendeteksian plat nomor kendaraan roda 4 yang akan di ambil.
 - Membuat program pendeteksian plat nomor mobil menggunakan metode morfologi.
3. Melakukan uji coba sistem dan melakukan pencatatan data pada proses pendeteksian plat nomor mobil menggunakan metode morfologi.
4. Melaporkan hasil penelitian dan pembuatan jurnal untuk mempublikasikan tugas akhir.

3.3 Diagram Blok Sistem

Pengerjaan tugas akhir ditunjukkan dalam bentuk blok diagram pada gambar 3.1. Blok diagram ini menggambarkan proses dari sampel citra hingga output berupa pendeteksian plat nomor mobil.



Gambar 3.1 Blok Diagram

Pengerjaan tugas akhir ini hanya fokus pada bagian dalam dari area garis putus-putus diatas. Input berupa gambar digital mobil. Selanjutnya, citra digital mobil yang semula RGB akan diubah menjadi citra grayscale dengan proses grayscale. Setelah gambar sudah dalam bentuk citra grayscale, maka proses selanjutnya adalah merubah gambar menjadi citra biner dengan cara melakukan proses *thresholding* terhadap objek citra *grayscale*. Tujuannya agar mendapatkan citra yang bernilai hitam dan putih sehingga lebih mudah untuk membedakan masing-masing objek yang diinginkan.

Setelah proses threshold, maka proses berikutnya adalah mendeteksi tepi yang sudah menjadi citra threshold. Tujuannya agar meningkatkan penampakan garis batas atau daerah pada suatu citra. Setelah sudah dideteksi tepi maka proses selanjutnya adalah dilasi. Tujuannya untuk penumbuhan atau penebalan pada objek citra, agar lebih mudah dalam mendeteksi posisi plat nomor.

Setelah melalui proses dilasi maka proses selanjutnya adalah *labelling*. Tujuannya untuk menandai suatu area sebagai objek yang dikenali untuk

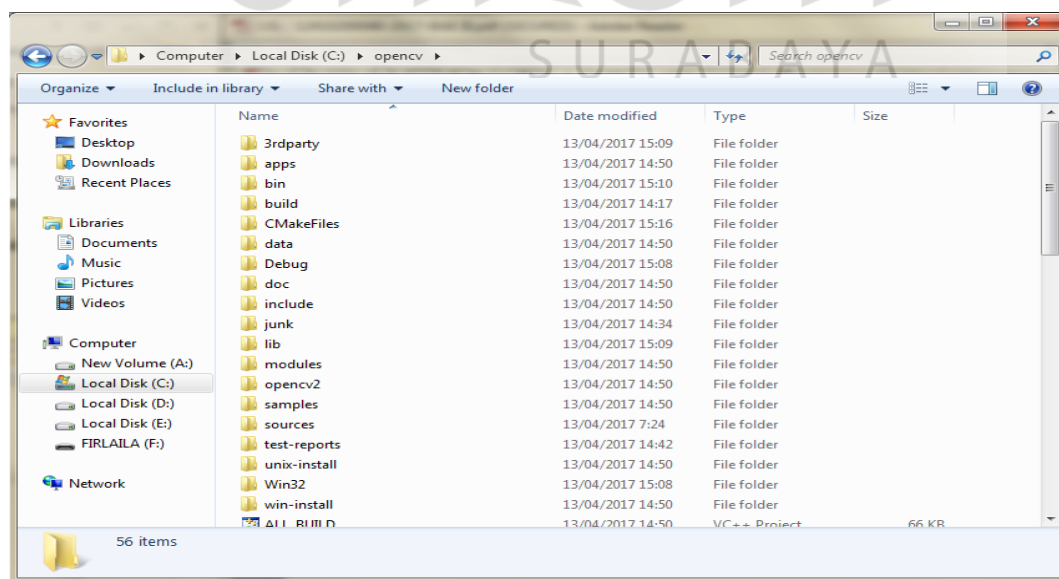
mempermudah pendeteksian lokasi, sehingga akan didapatkan citra akhir berupa lokasi dari plat nomor kendaraan dari gambar input. Posisi citra yang sudah ditandai/dikenali pada proses sebelumnya, maka proses selanjutnya *cropping*, tujuannya untuk memotong citra.

3.4 Rancangan Perangkat Lunak

Microsoft Visual Studio 2013 dengan menggunakan Library OpenCV merupakan perangkat lunak yang digunakan untuk merancang sistem ini, bahasa yang akan digunakan adalah bahasa pemrograman C++.

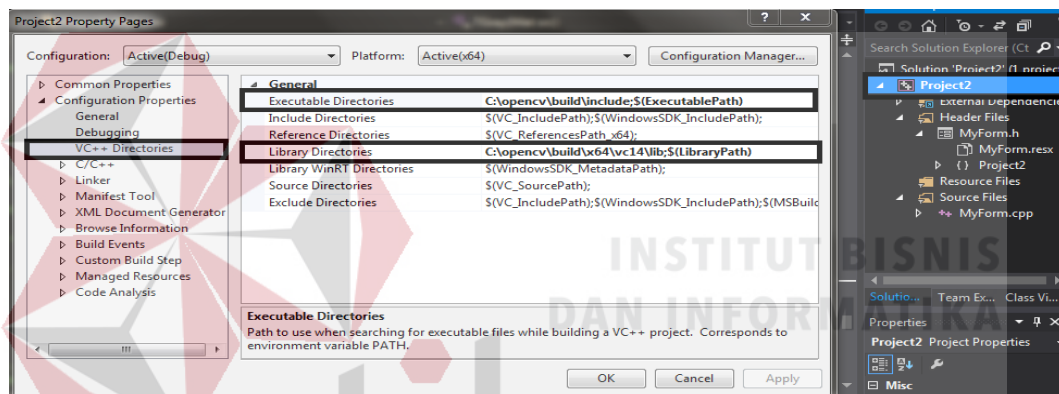
3.4.1 Microsoft Visual Studio dan OpenCV

OpenCV yang digunakan dalam tugas akhir ini penulis menggunakan versi 3.2. setelah mengunduh library OpenCV, maka lakukan proses penginstalan, kemudian ikuti langkah – langkah dari instalasinya. Setelah selesai, copy folder “opencv” pindahkan ke direktori C, seperti pada gambar 3.2.



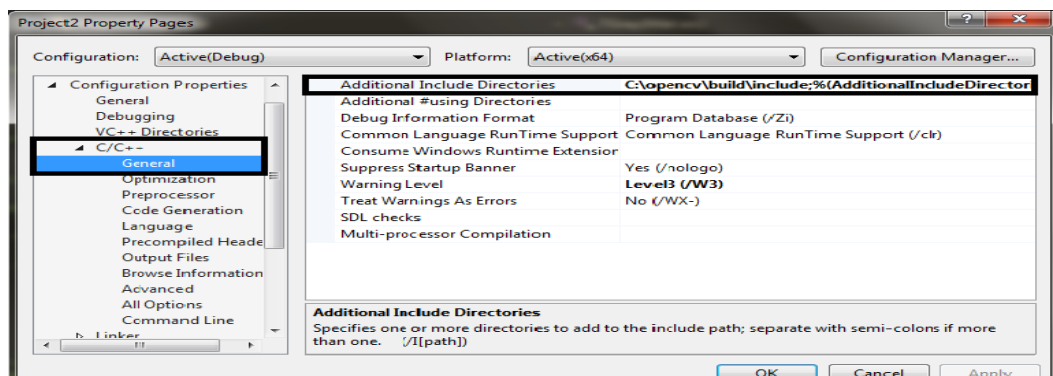
Gambar 3.2 Setelah Dipindahkan Library OpenCV

Langkah selanjutnya adalah membuka project di Microsoft visual studio, setelah terbuka klik kanan pada nama project kita lalu pilih properties. Halaman properties akan muncul setelah memilihnya kemudian pilih “VC++ Directories”, kemudian pilih “Executable Directories” dan “Library Directories” seperti pada gambar 3.3. Setelah itu pilih direktori tempat include direktori berada untuk Executable Directories dan direktori tempat library direktori berada untuk Library Directories, misalkan untuk include “C:\opencv\build\include” dan untuk library “C:\opencv\build\x64\vc14\lib”.



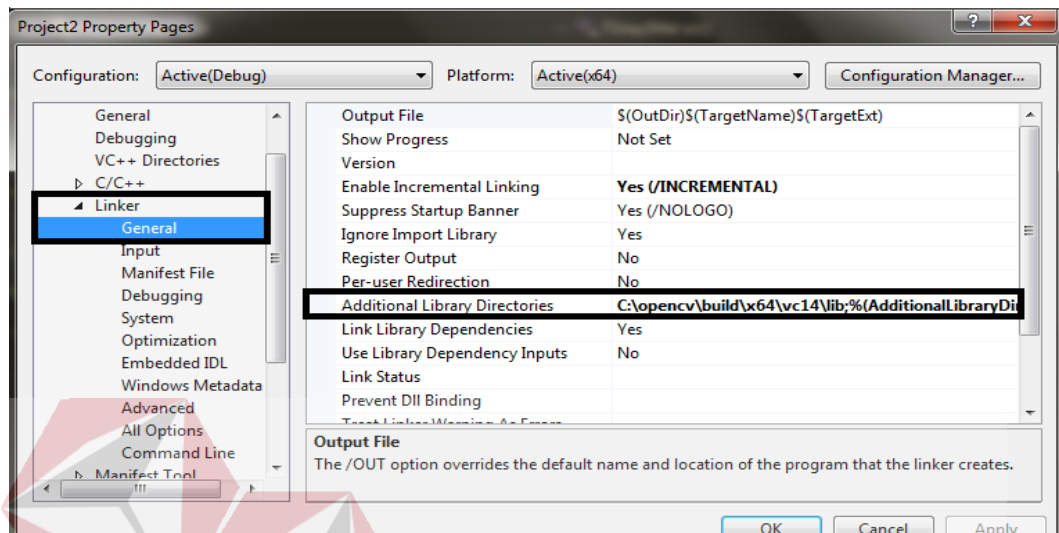
Gambar 3.3 Tampilan Properties VC++ Directories

Setelah itu pilih “C/C++”, klik “General” kemudian pilih “Additional Include Directories” seperti gambar 3.4. Setelah itu pilih direktori tempat direktori include berada, misalkan “C:\opencv\build\include”



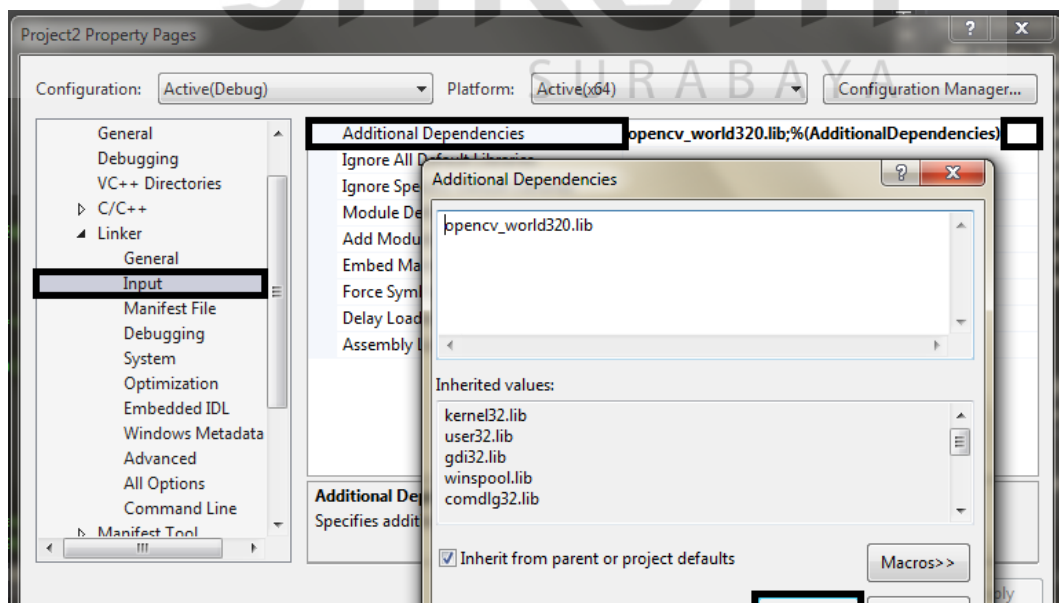
Gambar 3.4 Tampilan Properties C/C++

Setelah itu pilih “Linker”, klik “General” kemudian pilih “Additional Include Directories” cari lokasi library direktori yang berada di direktori “C:\opencv\build\x64\vc14\lib”, seperti pada gambar 3.5.



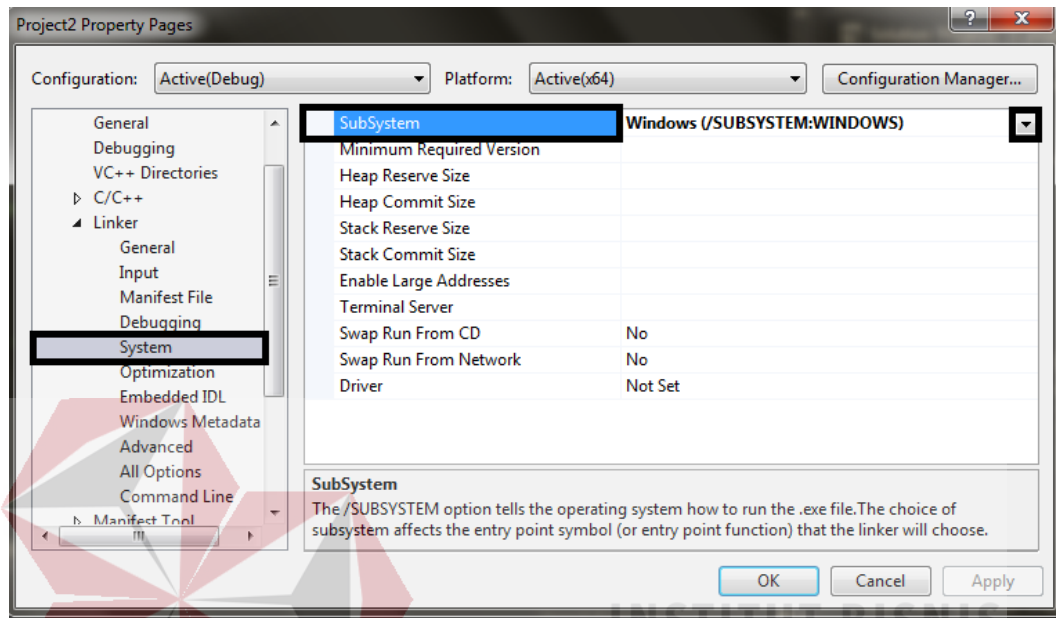
Gambar 3.5 Tampilan Properties Linker General

Setelah itu pilih “Input” pada bagian Linker, pilih “Additional Dependencies”, klik panah bawah dan edit semua tulisan menjadi “opencv_world320.lib”, setelah itu klik “OK”, seperti pada gambar 3.6.



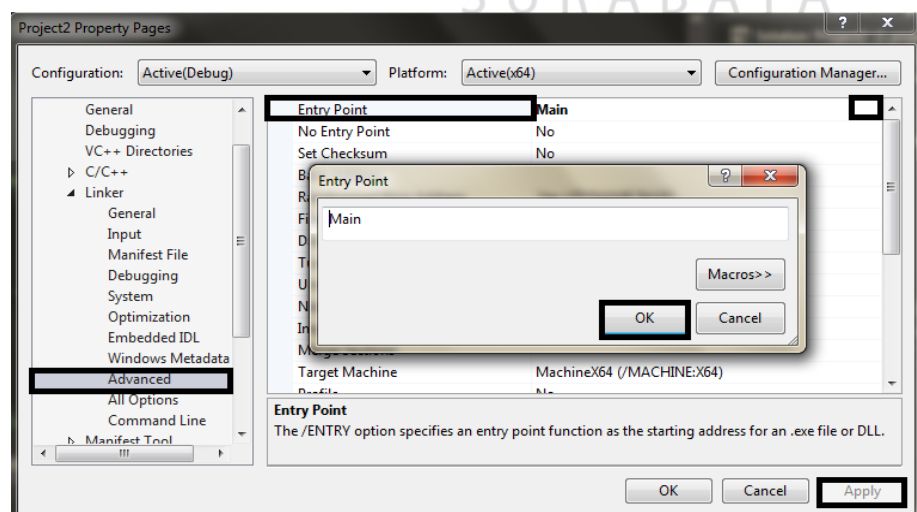
Gambar 3.6 Tampilan Properties Linker Input

Kemudian pilih “System” pada bagian Linker, pilih “SubSystem”, klik panah bawah dan pilih “Windows (/SUBSYSTEM:WINDOWS)”, seperti pada gambar 3.7.



Gambar 3.7 Tampilan Properties Linker System

Langkah terakhir adalah memilih “Advanced” pada bagian Linker, pilih “Entry Point”, klik panah bawah dan edit semua tulisan menjadi “Main”, setelah itu klik “OK” kemudian klik “Apply” seperti pada gambar 3.8.



Gambar 3.8 Tampilan Properties Linker Advanced

3.5 Perancangan *Preprocessing* Citra

3.5.1 Gambar Citra Input

Gambar citra input yang digunakan adalah gambar mobil yang akan memasuki parkir seperti pada gambar 3.9. citra input ini berasal dari kamera yang ada di pintu masuk parkir stikom, dengan format file bertipe .JPG atau .JPEG dan memiliki ukuran file tidak lebih dari 1 MB.

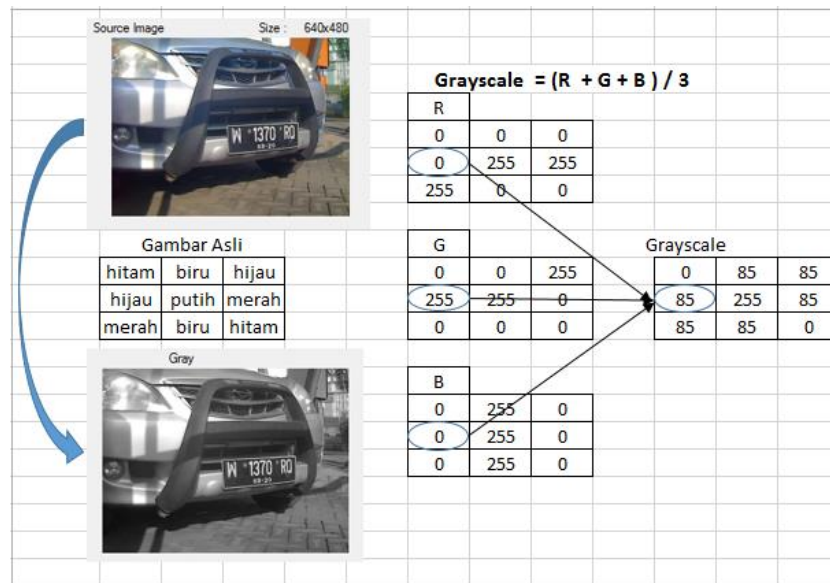


Gambar 3.9 Citra Input

3.5.2 *Preprocessing*

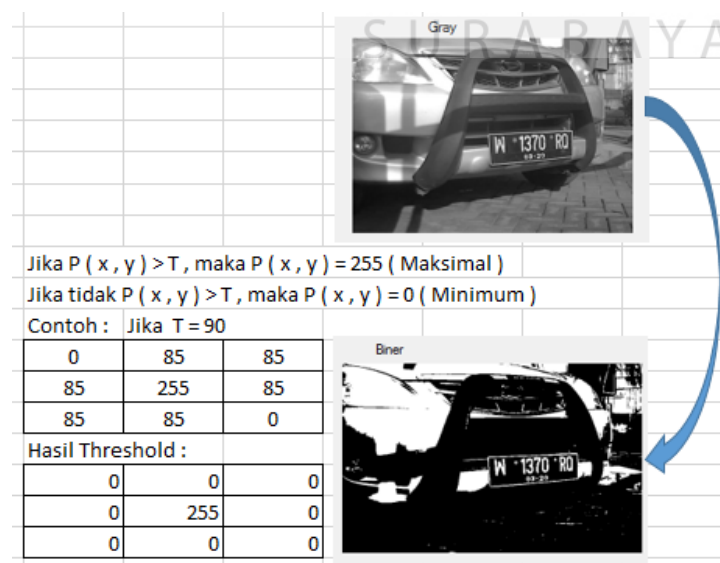
Gambar yang semulanya memiliki citra RGB (Red,Green,Blue) diproses ke citra grayscale sehingga yang didapatkan derajat keabuan pada citra, rumus yang digunakan pada proses grayscale ini adalah $Grayscale = \frac{(Red+Green+Blue)}{3}$, Dapat dilihat pada gambar 3.10, nilai piksel Red = 0, Green = 255, dan Blue = 0 kemudian dihitung menggunakan rumus yang digunakan pada proses grayscale, perhitungannya sebagai berikut :

$$Grayscale = \frac{(Red + Green + Blue)}{3} = \frac{(0 + 255 + 0)}{3} = \frac{255}{3} = 85$$





Gambar 3.10 Citra Grayscale

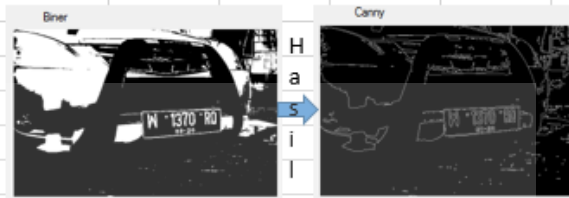
Hasil proses dari grayscale selanjutnya akan dijadikan inputan pada proses thresholding, cara kerja dari thresholding yang tadinya citra mempunyai nilai piksel derajat keabuan bernilai kecil dari batas maka akan bernilai 0 sedangkan yang lebih besar bernilai 1. Berikut contoh citra grayscale yang sudah diproses menggunakan thresholding, seperti pada gambar 3.11.



Gambar 3.11 Citra Thresholding

Setelah proses threshold, maka proses berikutnya adalah deteksi tepi yang sudah menjadi citra threshold. Tujuannya agar meningkatkan penampakan garis batas atau daerah pada suatu citra. Seperti pada gambar 3.12.

Rumus :	hasil horisontal = $ P(x, y) - p(x, y+1) $							
	hasil vertikal = $ P(x, y) - p(x+1, y) $							
Contoh :								
2	2	10	10	Hasil	0	8	0	0
2	2	10	10		0	8	0	0
10	10	2	2		0	8	0	0
10	10	2	2		0	8	0	0
				Vertikal	0	8	0	0
hasil								
Horizontal								
0	0	0	0	Biner		H a s i l	Canny	
8	8	8	8					
0	0	0	0					
0	0	0	0					



Gambar 3.12 Citra Deteksi Tepi

3.6 Dilasi

Gambar inputan yang akan dilakukan pada proses dilasi adalah hasil dari proses deteksi tepi yang menggunakan metode canny, pada saat melakukan proses deteksi tepi menghasilkan garis – garis tepi pada objek, beberapa garis – garis yang sudah terdeteksi ada sebagian garisnya yang mulai hilang atau sudah mulai terputus, yang dimaksud terputus disini adalah garisnya sudah menjadi objek lain padahal yang aslinya objeknya hanya satu. Agar gambar tidak terputus maka dilakukanlah proses dilasi, berikut contoh hasil dari proses deteksi tepi yang akan digunakan sebagai inputan pada proses dilasi ini, seperti pada gambar 3.13.



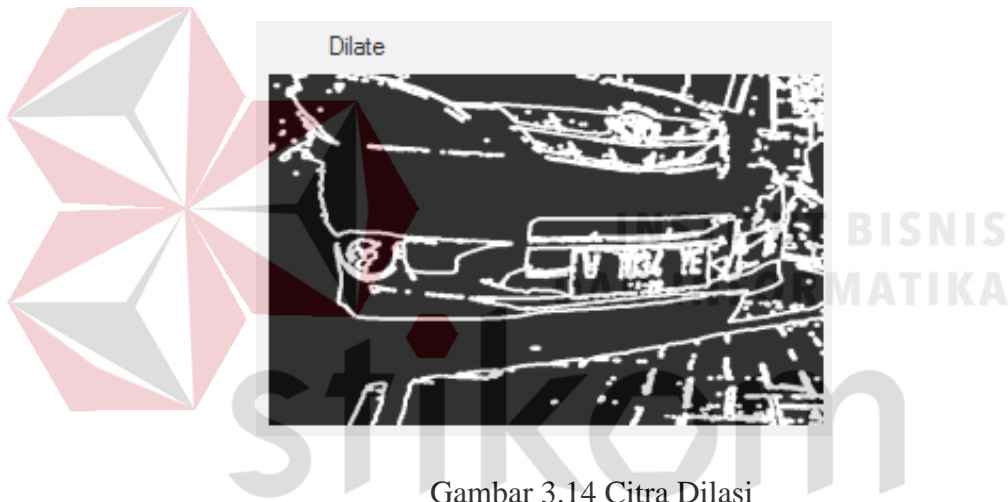
Gambar 3.13 Proses Deteksi Tepi

Tujuan dari proses dilasi adalah untuk menyambungkan kembali garis – garis yang mulai terputus, agar tidak menjadikan objek lain pada citra. Proses dilasi ini merupakan operasi morfologi filter, pada operasi morfologi filter yang menjadi kunci penting adalah pemilihan structuring element, structuring element memiliki dua komponen penting yaitu bentuk dan ukuran, yang mempengaruhi hasil pengujian. Pemilihan bentuk structuring element juga mempengaruhi citra hasil operasi morfologi. Pada pemrosesan citra biasa (bukan secara morfologi) structuring element dapat diibaratkan dengan mask. Structuring element juga memiliki titik poros, yang disebut juga dengan titik origin/titik asal/ titik acuan.

Pada saat melakukan proses dilasi ini, structuring elements yang digunakan berbentuk *rectangle* (persegi panjang) dan menggunakan ukuran (5,5) yang dimana 5 adalah lebarnya sedangkan 5 merupakan tingginya. Penulis sebelumnya mencoba menggunakan ukuran (3,3) dan berbentuk *ellipse* tetapi hasil yang didapatkan tidak sesuai dengan tujuan pendeteksian plat nomor ini, dan juga pernah menggunakan ukuran yang lebih besar yaitu (15,15) tetapi hasil yang didapatkan juga tidak sesuai pada saat dilakukannya proses berikutnya. Berikut kode program yang ditulis

menggunakan bahasa pemrograman C++ dan menggunakan *library* OpenCV yang digunakan pada proses dilasi menggunakan citra hasil deteksi tepi dan hasil dari kode program pada proses dilasi dapat dilihat pada gambar 3.14.

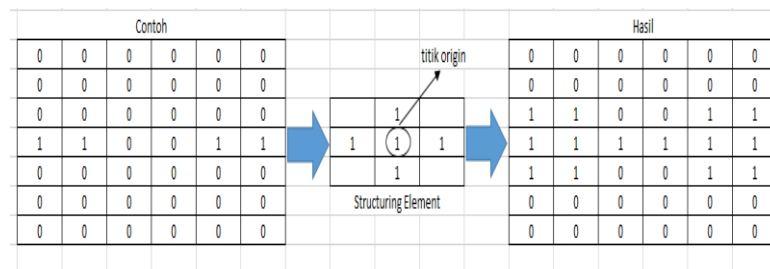
```
private: Mat TDilate (Mat src) {
    Mat dst;
    Mat Element;
    Element = getStructuringElement(MORPH_RECT, cv::Size(5, 5));
    dilate(src, dst, Element);
    return dst;}
```



Gambar 3.14 Citra Dilasi

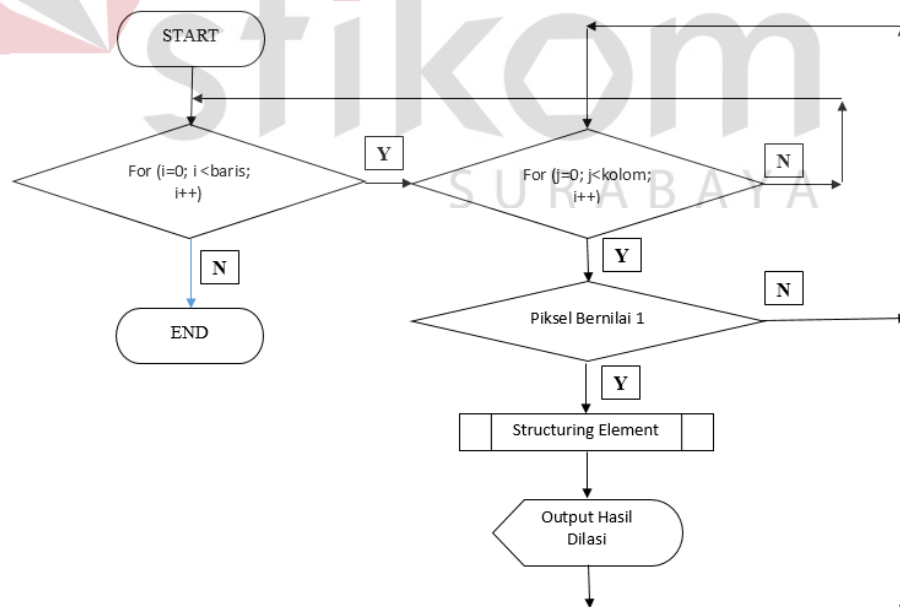
Cara kerja dari proses dilasi ini yaitu setiap pixel pada citra akan di bandingkan dengan titik origin / titik asal / titik acuan pada structuring element, jika terdapat nilai yang sama di citra dengan yang ada di titik origin / titik asal / titik acuan pada structuring element maka kanan kiri atas bawahnya pada pixel yang sedang dibandingkan akan diberi nilai 1 pada pixel, jika berbeda maka tidak akan memberi nilai 1 pada pixel, maksudnya jika bernilai 0 pada pixel yang sedang dibandingkan maka tidak akan diberi nilai 1 jadi nilai pada pixel akan tetap 0. Proses ini berulang – ulang pada setiap pixel yang terdapat pada citra mulai dari

piksel kiri atas hingga pada piksel yang berada di kanan bawah, seperti pada gambar 3.15.



Gambar 3.15 Contoh Proses Dilasi

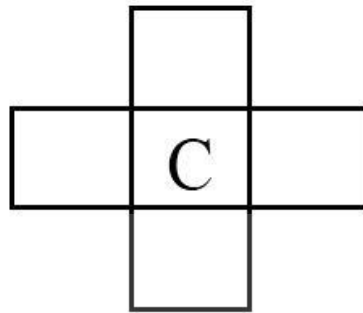
Pada contoh proses dilasi diatas hasil yang didapatkan akan seperti pada gambar 3.15 , karena proses perbandingannya tidak menggunakan hasil yang sudah dibandingkan antara citra awal dengan structuring element, tetapi tetap menggunakan citra awal. Terlihat pada gambar bahwa hasil yang didapatkan sesuai dengan tujuan dari proses dilasi. Alur flowchart dari proses dilasi dapat dilihat pada gambar 3.16.



Gambar 3.16 Alur Flowchart Dilasi

3.7 Connected Component Labelling

Pada proses labelling ini metode yang digunakan adalah connected component labelling. Connected component labelling yang digunakan pada proses ini adalah 4 connected neighbors, yaitu suatu pola yang memiliki 4 lengan dan memiliki 1 counter ditengahnya, seperti pada gambar 3.17.



Gambar 3.17 4 Connected Neighbors

Citra yang dapat diolah pada proses labelling yang menggunakan metode connected component labelling ini yaitu citra biner atau citra yang memiliki warna hitam dan putih saja. Tujuan dari proses labelling ini adalah untuk menandai objek yang dicari pada citra biner, namun pada proses labelling ini inputan yang digunakan adalah hasil dari proses dilasi, seperti pada gambar 3.18.



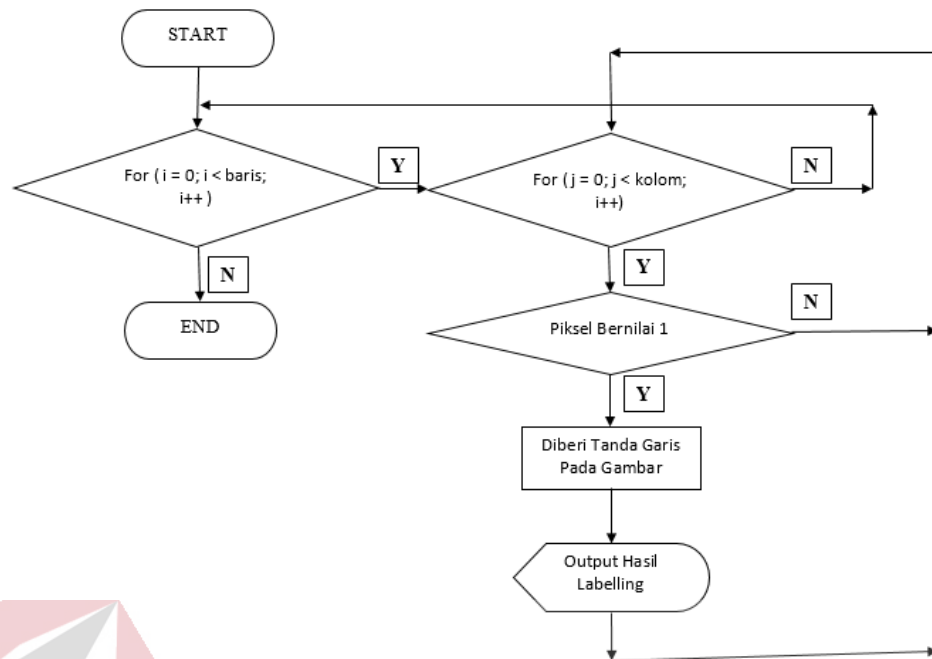
Gambar 3.18 Citra Inputan Labelling

Cara kerja dari proses connected component labelling yaitu pertama nilai counter sama dengan 0, kemudian yang kedua yaitu melakukan scanning piksel pada setiap piksel - piksel yang ada pada citra inputan, scanning piksel ini dimulai dari kiri atas hingga kekanan bawah, kiri atas merupakan piksel pertama pada citra sedangkan kanan bawah merupakan piksel terakhir pada citra. Ketiga cari yang bernilai 1, kemudian yang keempat jika terdapat nilai 1 pada piksel maka counter di increment, Kelima nilai keempat lengannya dibuat sama seperti pada counter, dan yang keenam jika pada piksel terdapat nilai 0 maka kembali lagi ke cara kerja yang kedua yaitu melakukan scanning piksel. Seperti yang dapat dilihat pada gambar 3.19.

Contoh						Hasil					
1	1	1	0	0	0	1	1	1	0	0	0
1	1	1	0	0	0	1	1	1	0	0	0
0	0	0	1	0	0	0	0	0	2	0	0
0	0	0	1	0	0	0	0	0	2	0	0
0	1	1	0	1	1	0	3	3	0	4	4
0	1	1	0	1	1	0	3	3	0	4	4
0	1	1	0	0	0	0	3	3	0	0	0
0	0	0	0	1	1	0	0	0	0	5	5

Gambar 3.19 Contoh Cara Kerja Labeling

Alur *flowchart* yang digunakan pada proses labelling yang menggunakan metode connected component labelling dapat dilihat pada gambar 3.20.



Gambar 3.20 Alur Flowchart Labeling

Setelah melakukan tahapan sesuai dengan yang ada di alur flowchart labelling maka didapatkan hasilnya seperti yang dapat dilihat pada gambar 3.21.



Gambar 3.21 Citra Labelling

3.8 Cropping

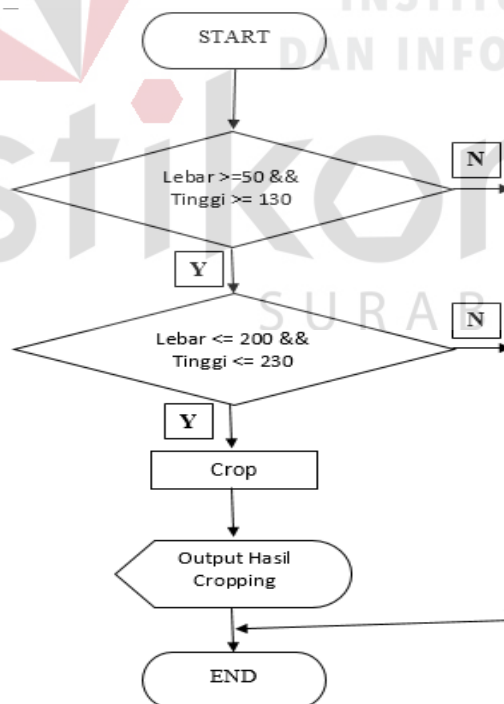
Cropping adalah memotong satu bagian dari citra sehingga diperoleh citra yang berukuran lebih kecil. Proses ini bertujuan untuk memotong objek yang sudah ditandai pada proses labelling yang menggunakan metode connected component

labelling dengan 4 connected neighbors, sehingga nantinya keluaran dari proses ini berupa potongan objek yang dicari. Berikut adalah contoh inputan pada proses cropping ini, yang dapat dilihat pada gambar 3.22.



Gambar 3.22 Contoh Citra Inputan Cropping

Cara kerja dari proses cropping dapat dilihat pada alur *flowchart* seperti pada gambar 3.23.



Gambar 3.23 Alur Flowchart Cropping

Pertama dicari lebar dengan ukuran lebih dari sama dengan 50 dan tinggi lebih dari sama dengan 130, jika tidak ada yang sesuai dengan ukuran maka proses langsung selesai dan tidak mendapatkan hasil plat nomor mobil, jika iya maka dicari ukurannya kembali. Kedua dicari ukuran lebar yang kurang dari sama dengan 200 dan tinggi kurang dari sama dengan 230, jika iya maka akan di crop kemudian ditampilkan hasil dari proses cropping tersebut, namun jika tidak maka proses cropping selesai, proses cropping selesai dikarenakan tidak ada ukuran yang sesuai dengan yang dicari pada citra. Contoh hasil dari proses cropping yang berhasil diambil dapat dilihat pada gambar 3.24.



Gambar 3.24 Citra Cropping