BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1 Model Pengembangan

Dalam pengerjaan tugas akhir ini memiliki tujuan untuk mengektraksi fitur yang terdapat pada karakter citra digital menggunakan metode *diagonal distance feature* agar mempermudah proses pengenalan data karakter plat nomor dari data gambar digital ke data dalam bentuk teks.

Pada sistem pengenalan karakter ini menggunakan media input berupa gambar plat nomor yang sudah diambil bagian kotak plat nomornya saja. Citra plat nomor di proses menggunakan metode *diagonal distance feature*, yakni dengan cara mencari nilai koordinat ke 4 diagonal dari suatu citra karakter yang telah di segmentasi. Selanjutnya nilai koordinat dari ke 4 diagonal tersebut digunakan sebagai inputan untuk proses pengenalan karakter menggunakan metode *template matching* sehingga proses pengenalan karakter lebih sederhana dan lebih cepat.

3.2 Prosedur Penelitian

Prosedur penelitian yang dipakai dalam pengerjaan tugas akhir ini adalah:

1. Studi literatur

Pada penelitian perancangan yang dilakukan adalah perancangan perangkat lunak. Adapun metode penelitian yang dilakukan antara lain:

Pencarian data-data literatur untuk perangkat lunak dari masing-masing proses, informasi dari internet, jurnal dan konsep teoritis dari buku-buku

penunjang tugas akhir ini, serta materi-materi perkuliahan yang telah didapatkan dan perancangan perangkat lunak yaitu menggunakan Microsoft Visual Studio dan OpenCV melalui pencarian dari internet, dan konsep-konsep teoritis dari buku-buku penunjang tersebut. Dari kedua bagian tersebut akan dipadukan agar dapat bekerja sama untuk 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 *Block Diagram* pada proses sistem secara keseluruhan.

b. Melakukan perancangan perangkat lunak yang meliputi:

   i. Merancang program *preprocessing* citra yang meliputi proses *grayscale*, *thresholding, noise filtering, edge detection,* dan *morphology filter* untuk mempermudah proses segmentasi karakter sehingga lebih fokus ke citra karakter yang akan di ambil.

   ii. Merancang proses segmentasi karakter yang akan digunakan dengan membuang beberapa citra yang tidak diperlukan untuk proses ekstraksi ciri.

   iii. Membuat program ekstraksi ciri citra karakter dengan menggunakan metode *diagonal distance feature*.

   iv. Membuat program untuk menyimpan nilai dari proses ekstraksi ciri dan membuat data *learning* dari nilai nilai tiap karakter.

   v. Membuat program untuk pengenalan karakter menggunakan nilai inputan dari hasil proses *learning* ekstraksi ciri.

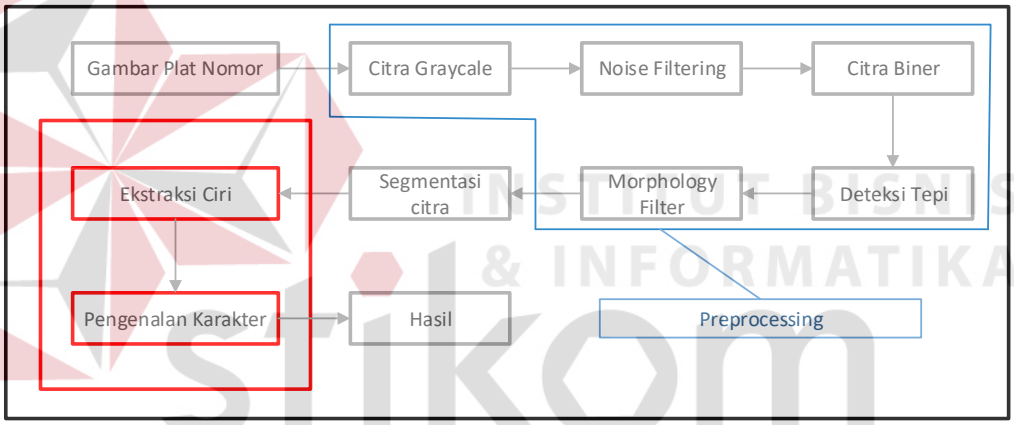
3. Melakukan uji coba sistem dan melakukan pencatatan data proses ekstraksi

ciri citra menggunakan metode *diagonal distance feature.*

4. Melaporkan hasil dari penelitian dan pembuatan jurnal untuk mempublikasikan tugas akhir.

3.3 Diagram Blok Sistem

Dalam perancangan tugas akhir kali ini terdapat beberapa proses yang dikerjakan mulai dari input berupa gambar plat nomor yang masih dalam bentuk citra RGB hingga proses pengenalan karakter yang nantinya mendapatkan hasil berupa data dalam bentuk teks. Berikut blok diagram yang pada sistem yang ditunjukkan oleh gambar 3.1 dibawah ini:



Gambar 3.1 Blok Diagram

Dalam mengerjakan Tugas Akhir ini, penulis hanya fokus pada 2 hal, yaitu yang pertama adalah proses ekstraksi fitur dengan menggunakan metode *diagonal distance feature* dan proses pengenalan karakter menggunakan metode *Template Matching.*

Gambar plat nomor yang telah diambil citra kotak plat nomornya saja sebagai inputan akan diproses menjadi citra grayscale. Selanjutnya dilakukan proses pengurangan derau atau *noise filtering* dengan menggunakan metode Gaussian Filter. Setelah proses *noise filtering* telah dilakukan, maka proses

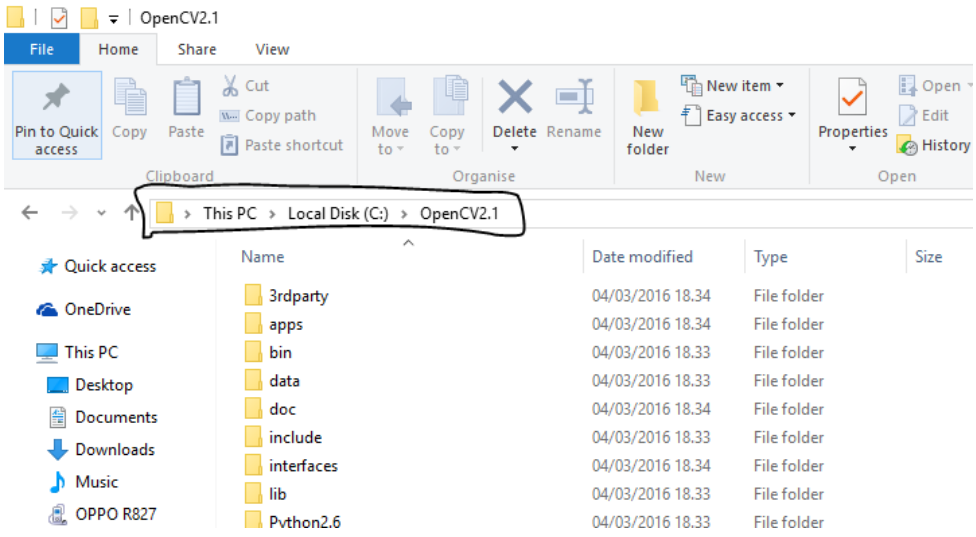
selanjutnya adalah proses *thresholding*. Selanjutnya dilakukan proses deteksi tepi menggunakan metode *Canny Edge Detection.* Hasil dari proses deteksi tepi digunakan sebagai inputan untuk proses *morphological operation* yang menggunakan metode erosi dan dilasi untuk mengurangi citra karakter yang tidak diperlukan. Kemudian dilakukan proses segmentasi untuk mendapatkan citra karakter yang dibutuhkan untuk proses ekstraksi ciri citra dan proses pengenalan karakter.

3.4 Rancangan Perangkat Lunak

Perangkat lunak yang digunakan untuk merancang sistem ini adalah Microsoft Visual Studio 2010 dengan menggunakan *library* OpenCV dan menggunakan bahasa pemrograman C++.

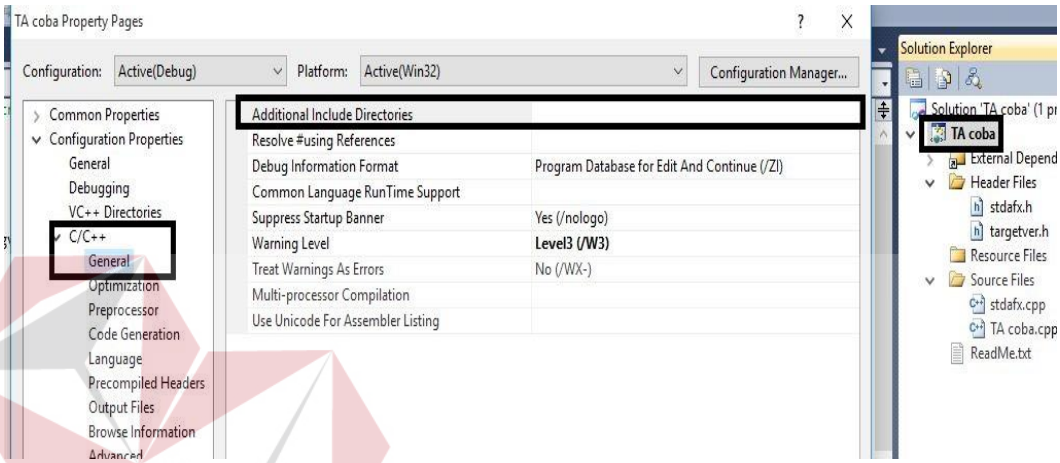
3.4.1 Microsoft Visual Studio dan OpenCV

Dalam tugas akhir ini penulis menggunakan openCV versi 2.1. setelah mengunduh *library* OpenCV, lakukan proses penginstalan, ikuti langkah-langkah instalasinya. Setelah selesai, biasanya openCV akan default di direktori C.



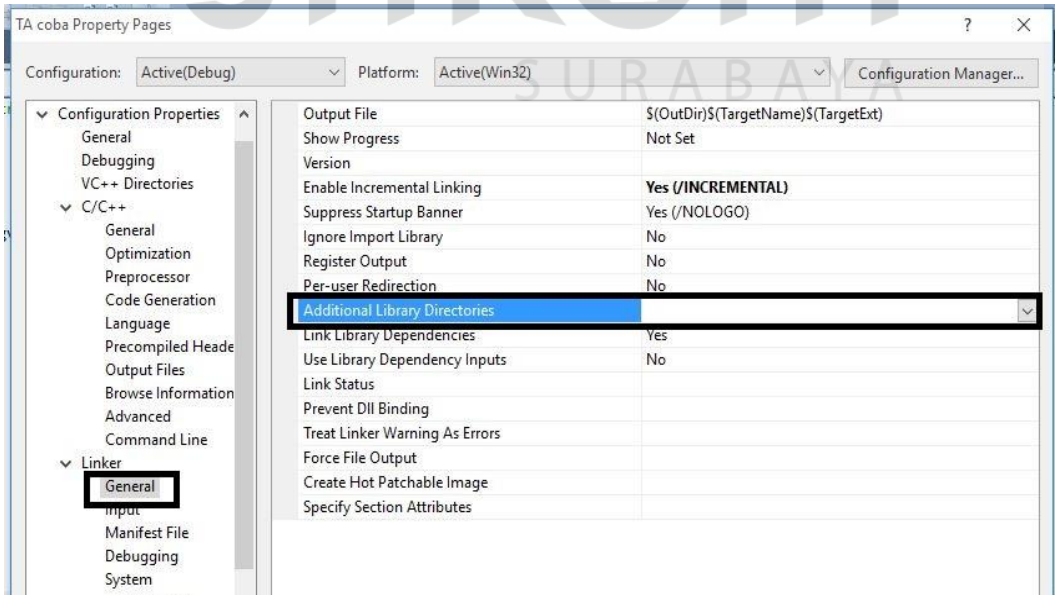
Gambar 3.2 Lokasi instalasi OpenCV

Langkah selanjutnya adalah buka project di Microsoft visual studio, klik kanan pada nama project kita lalu pilih properties. Kemudian akan muncul halaman properties dan pilih “C/C++”, klik “General” kemudian pilih Additional Include Directories seperti gambar 3.3. Setelah itu pilih direktori tempat include direktori berada, misalkan “C:\OpenCV2.1\include\opencv;%”



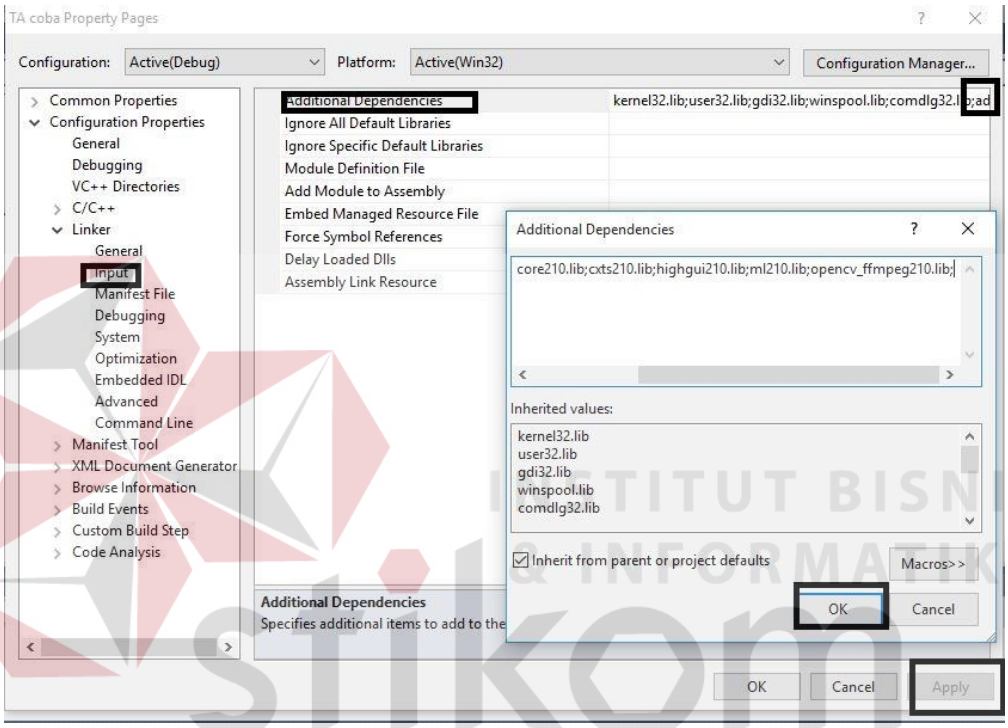
Gambar 3.3 Tampilan *Properties*

Setelah itu pilih “Linker”, pilih “Additional Libraby Directories” ari lokasi library direktori. Dalam hal ini pilih direktori “C:\OpenCV2.1\lib; “.



Gambar 3.4 Setting direktori *Library* OpenCV

Langkah terakhir adalah mengatur “input” pada bagian Linker, pilih “Additional Dependencies”, klik panah bawah dan edit semua tulisan menjadi “cv210.lib;cvaux210.lib;cxcore210.lib;cxts210.lib;highgui210.lib;ml210.lib;opencv_ffmpeg210.lib;”. setelah selesai klik “OK” kemudian klik “Apply” seperti pada gambar 3.5.



Gambar 3.5 Settingan Input

3.4.2 Perancangan *Preprocessing* Citra Plat Nomor

A. Gambar Plat Nomor

Gambar plat nomor yang digunakan adalah gambar plat nomor yang sudah di potong bagian kotak plat nomornya saja. Seperti gambar 3.6 dibawah ini.

Gambar 3.6 Contoh plat nomor yang akan di proses

B. *Preprocessing* Citra Plat Nomor

Proses *pre-processing* citra plat nomor terdiri dari proses *grayscale*, *noise filtering, thresholding, edge detection,* dan *morphology filter.* Gambar input yang berupa gambar plat nomor yang masih memiliki citra RGB akan di proses menjadi citra *grayscale* untuk mendapatkan derajat keabuan dari sebuah citra. Setelah itu dilakukan proses *noise filtering* dengan menggunakan metode *Gaussian Filter,* metode ini digunakan karena sangat cocok untuk menghilangkan *noise* yang dihasilkan oleh kamera. Berikut contoh citra *grayscale* pada gambar 3.7 dan citra *Gaussian filter* pada gambar 3.8.

Gambar 3.7 Citra Grayscale

Gambar 3.8 *Noise filtering* menggunakan *Gaussian filter*

Hasil dari *Gaussian filter* akan dijadikan inputan untuk proses *thresholding*. Cara kerja proses *thresholding* itu sendiri yakni dengan mengubah citra *grayscale* yang telah dilakukan *noise filtering* menjadi citra biner yang hanya memiliki nilai piksel 1 dan 0. Nilai piksel yang mempunyai derajat keabuan yang

nilainya lebih kecil dari batas akan bernilai 0 dan yang lebih besar akan di beri nilai 1. Metode *thresholding* yang digunakan adalah *Otsu Thresholding* tujuannya adalah untuk mendapatkan citra biner yang lebih baik.

Gambar 3.9 Metode *Otsu Thresholding*

Berikut kode program yang ditulis menggunakan bahasa pemrograman C++ dan menggunakan *library* OpenCV tentang proses *Grayscale, Gaussian filter* dan *Otsu thresholding* seperti dalam gambar 3.10 di bawah ini.

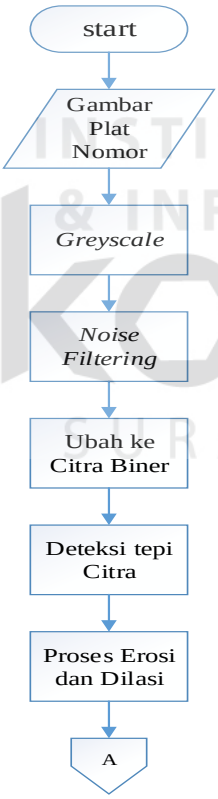
```
//grayscale
grayku=cvCreateImage(cvSize(src_img->width,src_img->height),8,1);
cvCvtColor(src_img,grayku, CV_BGR2GRAY);

//Gaussian filter
gauss=cvCreateImage(cvSize(grayku->width,grayku->height),8,1);
cv::GaussianBlur((cv::Mat)grayku, (cv::Mat)gauss, cv::Size(0, 0), 3);


//threshold
thres=cvCreateImage(cvSize(src_img->width,src_img->height),8,1);
cvThreshold( gauss, thres, 50, 255, CV_THRESH_OTSU );
```

Gambar 3.10 Program *Grayscale, Gaussian filter, dan Otsu threshold.*

Proses Deteksi tepi citra dilakukan untuk mempermudah menemukan tepi atau kontur dari sebuah citra. Pada proses deteksi tepi ini, penulis menggunakan metode *Canny Edge detection*. Hasil dari deteksi tepi di gunakan untuk proses *morphological operation.* Proses ini menggunakan metode *erotion* dan *dilation* yang digunakan untuk menyeleksi gambar yang tidak diperlukan.

Gambar 3.11 Metode *erotion*

Gambar 3.12 Metode *dilation*

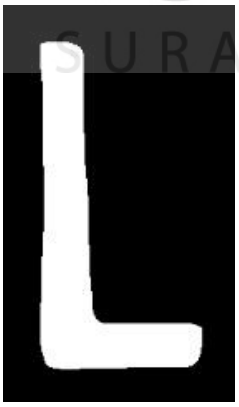


Gambar 3.13 *Flowchart Pre-processing* Citra

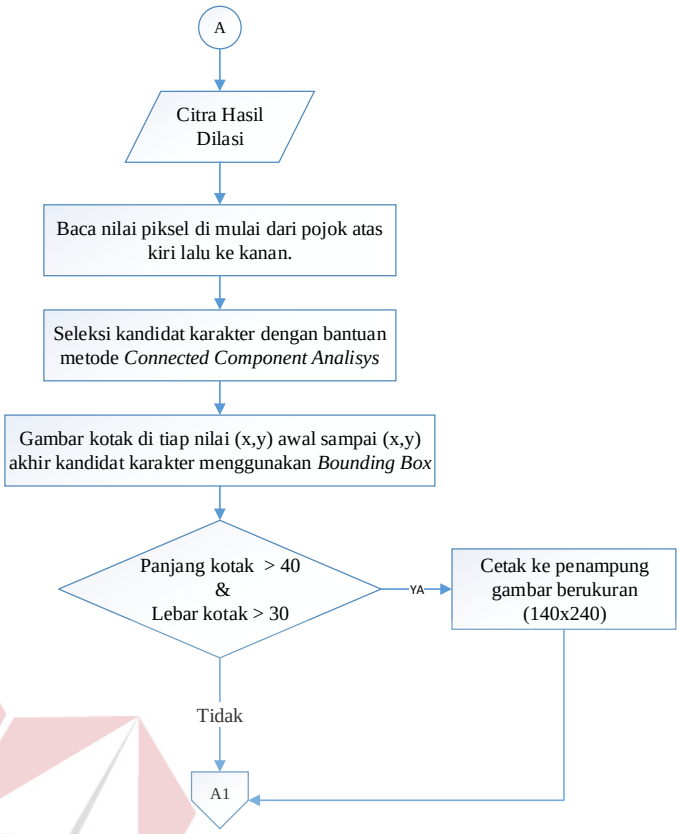
3.4.3 Segmentasi Citra

Pada proses segmentasi karakter dari citra plat nomor, proses yang digunakan yaitu dengan membaca nilai piksel gambar secara terus menerus dari kiri ke kanan, kemudian turun ke baris bawahnya hingga menemukan kandidat karakter yang dibutuhkan. Dalam hal ini citra masukan dari proses segmentasi hanya mempunyai nilai 1 atau 0, karena citra masukan diambil dari proses dilasi. Apabila dalam proses pencarian kandidat karakter bertemu dengan piksel yang bernilai 1, maka akan di simpan sebagai kandidat karakter. Dengan menggunakan bantuan metode *Connected Component Analisys* dan *Bounding Box,* proses segmentasi bisa dijalankan. Contoh dari proses segmentasi yang berhasil mengambil karakter L dapat dilihat di gambar 3.15 dibawah ini.

Gambar 3.14 Proses segmentasi karakter



Gambar 3.15 Karakter L hasil segmentasi



Gambar 3.16 *Flowchart* Segmentasi Citra

3.4.4 Ekstraksi ciri menggunakan *Diagonal Distance Feature*

Pada proses ekstraksi ciri ini menggunakan metode *diagonal distance feature* dengan mencari nilai 4 diagonal yang dihitung dari tiap pojok suatu citra hingga bertemu nilai piksel putih. Pada tahun 2015, pernah dilakukan penelitian mengenai *Ekstraksi FiturAngka Jawa Menggunakan Metode Diagonal Distance dan Longest Run Feature* (Umam, Muhammad Misbahul : 2015).

Citra yang sebelumnya sudah melalui proses segmentasi dan menghasilkan suatu karakter tertentu yang telah di *resize* menjadi ukuran (140,240) piksel di ektraksi cirinya dengan menghitung jarak pojok kiri atas dari gambar dan akan di proses secara diagonal hingga nantinya bertemu nilai piksel putih.

Pencarian diagonal dilakukan dengan cara *scanning* atau melakukan perulangan secara terus menerus sesuai dengan diagonal dalam program hingga bertemu dengan *array* yang memiliki nilai lebih besar dari 0, dan kemudian dicatat koordinat diagonalnya. Untuk diagonal pertama dicari dengan cara memulai perulangan pada kondisi x = 0 dan y = 0. Buat program untuk menaikkan kedua nilai (x,y) sesuai dengan diagonal kotak yang berukuran (140,240) ketika perulangan menemukan nilai lebih dari 0, maka hentikan perulangan, cetak lokasi koordinat saat ini. Pencarian jarak dilakukan dengan menerapkan rumus:

$$d1 = \sqrt{(x_{akhir} - x_{awal})^2 + (y_{akhir} - y_{awal})^2} \quad \text{..... (1)}$$

x_{akhir} merupakan posisi koordinat x saat di temukan *array* yang memiliki nilai lebih besar dari 0, x_{awal} adalah koordinat awal saat memulai proses dimana $x_{awal} = 0$. Begitu pula dengan y_{akhir} dan y_{awal}.

Diagonal kedua dicari dengan cara memulai perulangan pada kondisi x = 139 dan y = 0. Buat program untuk menurunkan nilai (x) dan menaikkan nilai (y) sesuai dengan diagonal kotak yang berukuran (140x240) ketika perulangan menemukan nilai lebih dari 0, maka hentikan perulangan, cetak lokasi koordinat saat ini. Pencarian jarak dilakukan dengan menerapkan rumus:

$$d2 = \sqrt{(x_{awal} - x_{akhir})^2 + (y_{akhir} - y_{awal})^2} \quad \text{..... (2)}$$

x_{akhir} merupakan posisi koordinat x saat di temukan *array* yang memiliki nilai lebih besar dari 0, x_{awal} adalah koordinat awal saat memulai proses dimana $x_{awal} = 139$. Begitu pula dengan y_{akhir} dan $y_{awal} = 0$.

Diagonal ketiga dicari dengan cara memulai perulangan pada kondisi x = 0 dan y = 239. Buat program untuk menaikkan nilai (x) dan menurunkan nilai (y) sesuai dengan diagonal kotak yang berukuran (140x240) ketika perulangan

menemukan nilai lebih dari 0, maka hentikan perulangan, cetak lokasi koordinat saat ini. Pencarian jarak dilakukan dengan menerapkan rumus:

$$d3 = \sqrt{(x_{akhir} - x_{awal})^2 + (y_{akhir} - y_{awal})^2} \quad \text{..... (3)}$$

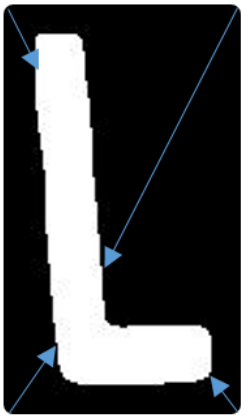
x_{akhir} merupakan posisi koordinat x saat di temukan *array* yang memiliki nilai lebih besar dari 0, x_{awal} adalah koordinat awal saat memulai proses dimana $x_{awal} = 0$. Begitu pula dengan y_{akhir} dan $y_{awal} = 239$.

Diagonal keempat dicari dengan cara memulai perulangan pada kondisi x = 139 dan y = 239. Buat program untuk menurunkan nilai (x,y) sesuai dengan diagonal kotak yang berukuran (140 , 240) ketika perulangan menemukan nilai lebih dari 0, maka hentikan perulangan, cetak lokasi koordinat saat ini. Pencarian jarak dilakukan dengan menerapkan rumus:

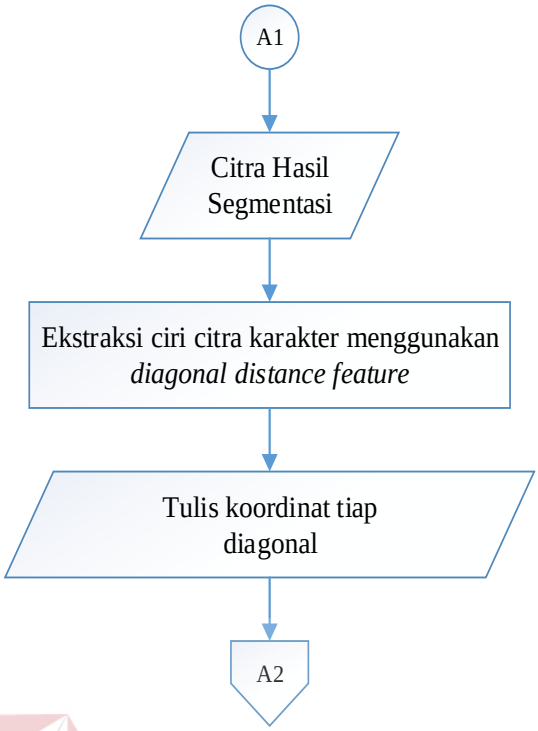
$$d4 = \sqrt{(x_{akhir} - x_{awal})^2 + (y_{akhir} - y_{awal})^2} \quad \text{..... (4)}$$

x_{akhir} merupakan posisi koordinat x saat di temukan *array* yang memiliki nilai lebih besar dari 0, x_{awal} adalah koordinat awal saat memulai proses dimana $x_{awal} = 139$. Begitu pula dengan y_{akhir} dan $y_{awal} = 239$.

Seperti ilustrasi dari gambar 3.17, tiap diagonal memiliki pola perulangan yang berbeda untuk mendapatkan koordinat diagonal yang akan digunakan.



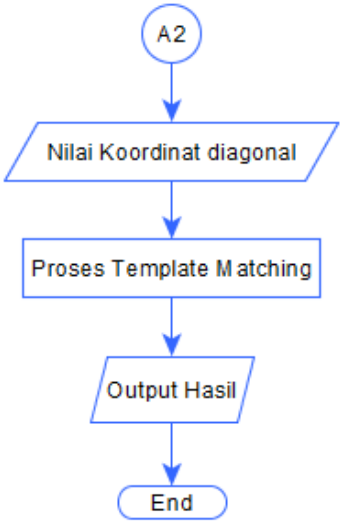
Gambar 3.17 Ekstraksi ciri menggunakan *Diagonal Distance Feature*



Gambar 3.18 *Flowchart Diagonal Distance Feature*

3.4.5 Pengenalan karakter menggunakan *Template Matching*

Proses pengenalan karakter menggunakan metode *template matching* menggunakan masukan berupa nilai hasil dari metode *diagonal distance feature*. nilai dari 4 diagonal yang telah didapatkan dari tiap karakter yang telah di normalisasi, menjadi citra berukuran (140,240) piksel disimpan dalam suatu data *learning*. Dalam proses *learning*, nilai koordinat ke 4 diagonal tiap karakter di uji sebanyak 5 kali. Dari hasil 5 kali percobaan ini didapatkan nilai koordinat 4 diagonal masing-masing karakter yang dijadikan *input-an* untuk pengenalan karakter. Hasil dari data *learning* dibandingkan dengan data gambar plat nomor yang di proses. Nilai yang memiliki selisih terkecil merupakan citra karakter yang paling sesuai. Hal ini dilakukan secara terus menerus hingga seluruh citra karakter pada plat nomor selesai di proses.



Gambar 3.19 *Flowchart Template Matching*

3.5 Teknik Pengumpulan dan Analisis Data

Pada pengerjaan proyek tugas akhir mengenai ekstraksi ciri dan pengenalan plat nomor ini, setelah melakukan pembuatan perangkat lunak, yang dilakukan selanjutnya adalah menganalisa kinerja sistem apakah sistem yang dibuat dapat bekerja sesuai rencana yang telah ditentukan.

3.5.1 Pengumpulan Data

Data sampel yang digunakan dalam tugas akhir ini diperoleh dengan mencari karakter plat nomor yang berbeda. Untuk 1 set (26 karakter huruf dan 10 karakter angka) dilakukan uji coba sebanyak 5 kali untuk menghasilkan sampel *Learning.*

3.5.2 Analisa Data

Data nilai ke 4 diagonal tiap karakter yang sudah melalui proses *learning*, dicari *range* tiap data dari hasil 5 kali percobaan. Dilakukan analisa nilai koordinat tiap karakter dan didapatkan bahwa karakter 3,6,8,9,0,B,O dan karakter X,H,N,E,M,K,Z memiliki nilai koordinat diagonal yang sebagian besar datanya sama. Hal ini tentu menjadi suatu masalah untuk proses pengenalan karakter yang

hanya menggunakan nilai koordinat ke 4 diagonal saja sebagai *input*-an untuk pengenalan karakter.

3.6 Uji Coba

Pada proses uji coba ini, program di uji coba dengan cara melakukan proses *diagonal distance feature* pada tiap karakter. Untuk mengetahui nilai dari ke 4 diagonal yang di dapatkan program akan mencetak nilainya pada *Command Window* kemudian di bandingkan dengan ekstraksi secara manual tiap piksel citra karakter. Jika nilai ke 4 diagonal sama, maka program ekstraksi *diagonal distance feature* berjalan dengan baik.

Untuk proses pengenlana karakter, akan di uji coba dengan membandingkan data teks yang di cetak di *Command Window* dengan gambar karakter yang telah di segmentasi. Apabila data teks yang di cetak merupakan karakter yang di maksud dalam gambar, maka program pengenalan karakter berjalan dengan baik.