BAB III

METODE PENELITIAN

Dalam tugas akhir ini penguji melakukan pengujian dari judul tugas akhir sebelumnya, yang dilakukan oleh Isana Mahardika. dalam tugas akhir tersebut membahas pendeteksian tempat parkir kosong. Dalam pengujian yang dilakukan, penguji melakukan pengembangan untuk aplikasi streaming dan membuat peregangan kontras.

Untuk melakukan pengumpulan data yang diperlukan dalam melaksanakan tugas akhir, ada beberapa cara yang telah dilakukan, antara lain:

1. Studi kepustakaan

Studi kepustakaan berupa pencarian data-data literatur dari fungsi pada *library* OpenCV, melalui pencarian dari internet, dan konsep-konsep teoritis dari buku-buku penunjang serta metode yang akan digunakan untuk melakukan pengolahan citra.

2. Penelitian

Penelitian dilakukan dengan menggunakan aplikasi perangkat lunak, implementasi perangkat lunak, dan pengambilan data pengujian aplikasi, kemudian melakukan evaluasi dari data hasil pengujian yang telah dilakukan.

3.1. Perancangan Sistem dan Blok Diagram Sistem

Model penelitian yang akan dilakukan adalah model penelitian pengembangan. Untuk mempermudah dalam memahami sistem yang akan dibuat dapat dijelaskan melalui blok diagram pada Gambar 3.1.

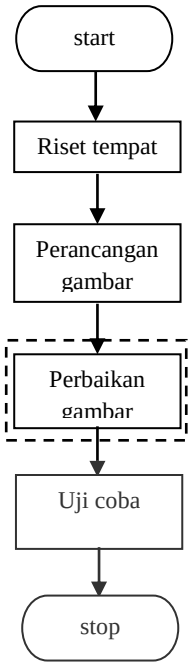
Gambar 3.1 Blok Diagram Sistem Secara Umum

Sebagai input, citra didapatkan dari kamera yang terpasang pada miniatur. Kemudian diproses menggunakan *console application* Visual C++ 2008 dengan memanfaatkan *library* OpenCV. Citra yang diperoleh adalah citra keadaan blok parkir mobil. Setelah mendapatkan citra tersebut, maka selanjutnya citra akan diproses untuk kemudian disubtraksi/dikurangkan (*subtraction*) dengan citra sampel (citra blok tanpa mobil). Dari hasil subtraksi tersebut dilakukan klasifikasi benda yang teridentifikasi merupakan mobil atau tidak.

Input citra yang didapatkan dari tugas akhir sebelumnya akan diolah untuk mendapatkan citra yang sempurna. Sehingga input citra nanti dapat bekerja secara maksimal untuk mendeteksi tempat parkir yang kosong berdasarkan nilai histogram yang didapatkan. Semua proses dilakukan menggunakan aplikasi visual C++ 2008, dengan memanfatkan *library* OpenCV. Pada akhirnya akan ditampilkan informasi pada PC berupa *output* nomor tempat yang kosong (diasumsikan bahwa setiap tempat parkir terdapat nomor urut), selain itu juga citra yang diambil secara *streaming* akan ditampilkan pada layar PC.

3.2. Perancangan Perangkat Keras

Flowchart perancangan dan pembuatan miniatur sebagai berikut :



Gambar 3.2 *Flowchart* Pembuatan Miniatur

Dalam Tugas Akhir ini perangkat keras yang digunakan adalah miniatur tempat parkir, miniatur telah dilakukan riset desain dan ukuran tempat parkir dalam tugas akhir sebelumnya. Selanjutnya diimplementasikan di aera terbuka agar sesuai dengan konsep yang dikerjakan terhadap tugas akhir kali ini.

Hasil yang didapat dari area terbuka adalah pencahayaan yang selalu berubah - ubah setiap waktu, sehingga kamera diprogram agar dapat mengatur pencahayaan yang masuk melalui konversi histogram.

3.3. Pengambilan Citra Sampel

Citra sampel diambil dalam ruangan terbuka atau dalam area parkir *outdoor*, dan dalam situasi dan kondisi yang selalu berubah – ubah pencahayaanya.

Citra sampel tersebut digunakan untuk data yang akan disubtraksi dengan citra *update*. Pengambilan citra sampel dilakukan secara manual dan akan disimpan pada direktori D:**file TA\\motiondetection1.jpg**. Ketika program berjalan program akan memuat citra sampel menggunakan fungsi *cvLoadImage* dan disimpan dalam variabel img, format variabel img adalah *Iplimage.* Berikut potongan program untuk memuat citra sampel .

```
IplImage*img=cvLoadImage ("D:\\file TA\\motiondetection1.jpg" );
```

3.4. Penerimaan Data Citra

Setiap data citra yang dikirimkan dari kamera diakses dengan *pointer CvCapture* dan *videocapture* menggunakan fungsi *cvCaptureFromCAM(1), VideoCapture cap(1);* Angka 1 pada fungsi *cvCaptureFromCAM(1)* merupakan indeks dari kamera yang digunakan. Berikut adalah potongan program untuk proses penerimaan data citra dari Kamera menggunakan.

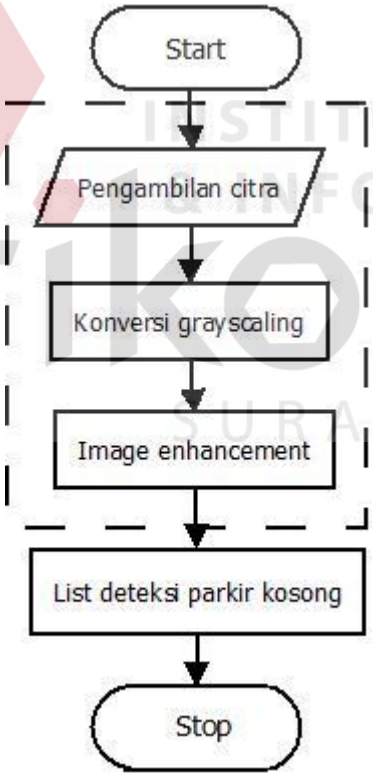
```
CvCapture* capture = cvCaptureFromCAM(1);
```

Data citra yang ditangkap adalah data citra dengan ruang warna RGB dan disimpan langsung pada variabel Ipl*image* (*Intel Image Processing Library*) yaitu stuktur data untuk penyimpanan data citra pada OpenCV. Urutan *channel* data dalam *Iplimage* adalah BGR sehingga untuk menampilkan warna sesungguhnya. Kemudian data citra yang didapat diolah menjadi *histogram* untuk mendapatkan hasil yang diinginkan.

Data citra yang ditangkap akan diolah menjadi ke data histogram, untuk mendapatkan hasil yang maksimal sebelum dinormalisasi kembali dari hasil histogram yang didapatkan.

3.5. Pengolahan Citra

Proses pengolahan citra adalah proses yang paling utama dalam pengerjaan program untuk melakukan *image processing* pada Tugas Akhir ini karena menggunakan Kamera *webcam* sebagai sensor untuk mendeteksi cahaya yang masuk dan mengkonversi citra yang ditangkap dan kemudian diolah. Berikut adalah *Flowchart* pengolahan citra secara garis besar :



Gambar3.3 *Flowchart* Pengolahan Citra

Metode yang digunakan untuk proses pengolahan citra adalah metode *contrast streching*. Untuk mendukung metode *contrast streching* juga dilakukan

konversi warna kedalam *grayscaling,* kemudian dikonversi menjadi histogram untuk mengkonversi cahaya yang masuk.

Proses pengolahan citra disini yang dimaksud adalah bagaimana proses pengolahan citra dari awal hingga mendapatkan hasil yang diinginkan. Untuk mendukung proses tersebut dilakukan proses konversi warna dari RGB (*Red green blue*) menjadi citra berwarna *grayscale* sebelum dirubah menjadi histogram, sehingga mendapatkan hasil yang diinginkan.

Ketika cahaya berubah – ubah maka kamera tidak bisa mendapatkan citra yang sempurna, oleh karna itu dilakukan proses histogram supaya mendapatkan citra yang sempurna. Setelah melakukan proses histogram, langkah selanjutnya yang dilakukan adalah normalisasi citra, normalisasi citra bertujuan untuk mendapatkan hasil dari histogram tersebut.

Setelah dilakukan proses pengolahan citra dengan menggunakan metode *contrast streching* nantinya akan diproses kedalam deteksi tempat parkir kosong, yang sudah pernah dibahas dalam Tugas akhir sebelumnya.

3.6. Grayscaling

Grayscale adalah suatu format citra atau gambar yang tiap-tiap *pixel* gambar hanya terdiri dari 1 *channel* warna. Proses perubahan warna dari RGB menjadi *Grayscale* bertujuan untuk mempermudah proses selanjutnya yaitu proses perubahan *Grayscale* menjadi biner. Sehingga setelah proses subtraksi berhasil dilakukan maka langkah selanjutnya ialah melakukan konversi format gambar dari RGB menjadi *Grayscale*. Untuk mengubah RGB menjadi *Grayscale* dapat digunakan rumus.

Grayscale = 0.299*R* + 0.587*G* + 0.114*B*

atau dapat menggunakan algoritma dengan merata-rata nilai ketiga buah *channel* RGB.

Grayscale = (R + G + B) / 3

Perubahan gambar RGB menjadi *Grayscale* menggunakan library openCV pada visual C++ menggunakan perintah sebagai berikut.

```
cvCvtColor(region1,gimask1,CV_RGB2GRAY);
```

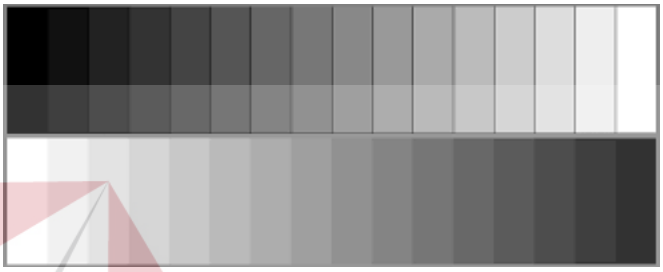
Pada perintah tersebut sudah terdapat dua *frame* yaitu region1 dan gimask1. region1 adalah *frame* RGB hasil subtraksi sedangkan gimask1 adalah *frame* yang disediakan untuk *Grayscale* yang akan dibuat. Sehingga maksud dari potongan perintah tersebut adalah mengubah gambar region1 menjadi *Grayscale* dengan fungsi CV_RGB2GRAY lalu disimpan pada *frame* bernama gimask1.Tetapi terlebih dahulu dibuat deklarasi pointer untuk *image* graycale, yaitu dengan cara seperti berikut.

```
IplImage*gimask1=cvCreateImage(cvGetSize(region1),IPL_DEPTH_8U,1);
```

Pada potongan program tersebut terdapat IPL_DEPTH_8U yang artinya adalah tiap-tiap *pixel* bernilai 8 bit. Sedangkan angka 1 setelah koma dibelakang IPL_DEPTH_8U bermakna tiap-tiap *pixel* hanya terdiri dari sebuah *channel*.

Proses awal yang sering dilakukan dalam melakukan *image processing* adalah mengubah citra berwarna menjadi citra *grayscale*. Hal ini digunakan untuk menyedarhanakan model citra. (Dharmawan, 2013). Citra *grayscale* hanya menggunakan warna satu tingkatan warna abu – abu. (Sutoyo, 2012). Warna abu – abu adalah satu – satunya warna pada ruang RGB (Red, Green, Blue) yang mempunyai itensitas warna yang sama.

Pada citra berasas keabu–abuan hanya mememlukan nilai intensitas untuk tiap piksel sebagai nilai tunggal, sedangkan pada citra yang berwarna memerlukan sebanyak tiga nilai intensitas untuk setiap pikselnya. Intensitas dari citra berwarna grayscale akan disimpan ke dalam 8 bit integer yang memberikan 256 kemungkinan yang mana akan dimulai dari level 0 hingga 255 (0 untuk warna hitam dan 255 untuk warna putih dan diantara 0 – 255 adalah derajat warna keabuan). (sunu jatmika, 2014)



Gambar 3.4 Grayscale level (sunu jatmika, 2014)

3.7. *Image Enhancement*

Teknik *image enhancement* digunakan untuk memperbaiki citra yang diambil, seperti *brightness, contrast,* kemudian dirubah menjadi citra *image grayscale, noise,* dan deteksi tepi. (basuki, 2007)

Image enhancement (perbaikan citra) bertujuan untuk meningkatkan kualitas citra untuk pengekihatan manusia atau untuk mengkonversi suatu citra agar lebih baik citra yang sebelumnya.

Gambar 3.5. Citra dalam bentuk *Image enhancement*

3.8. Metode Pengujian dan Evaluasi Sistem

Untuk dapat mengetahui apakah aplikasi sudah bisa berjalan sesuai dengan apa yang diharapkan, maka akan dilakukan pengujian dan evaluasi terhadap sistem, dan dilakukan evaluasi untuk setiap tahapan dalam pembuatan sistem aplikasi. Dimulai dari *streaming*, *update* terhadap citra dalam berbagai kondisi cuaca, *grayscaling*, dan histogram untuk dapat mengetahui perhitungan dalam mendeteksi parkir.

3.8.1. Pengujian *Streaming* Citra

Untuk mengetahui apakah data citra sudah dapat diakses langsung melalui Kamera, maka dilakukan pengujian dengan cara menjalankan (*running*) program pemanggilan kamera dari Visual C++ 2008, yaitu untuk mengakses *console*

Kamera secara langsung dari program. Kemudian citra yang tampil akan diuji apakah dapat menampilkan data citra secara *streaming*.

3.8.2. Pengujian *Update* Citra dan *streaming*

Pengujian program dilakukan apakah data berupa citra kondisi parkir dapat melakukan *update* secara otomatis, dan data *update* sengaja diberi nama *file* yang sama dari sebelumnya, agar dapat diketahui apakah pengujian bisa sempurna (D:\\File TA\\motiondetection1.jpg), citra yang diambil akan *update* secara otomatis.

Sedangkan *streaming* untuk mengetahui apakah cahaya yang masuk terlalu berlebihan atau terlalu rendah, sehingga pengujian yang dilakukan dapat diketahui melalui proses histogram yang ada pada *streaming* tersebut. dan pada saat streaming *update* citra akan tetap dilakukan dan disimpan kedalam direktori yang sudah tersedia.

3.8.3. Pengujian *Image enhancement*

Pengujian *image enhancement* dilakukan dengan cara melakukan perbaikan terhadap citra yang didapat pada saat melakukan pengujian *streaming*, citra yang didapat kemudian dikonversi kedalam *image grayscale* kemudian dilakukan perbaikan terhadap citra.

Setelah citra dilakukan perbaikan, maka hasil yang didapatkan akan berbeda dengan citra *grayscale* yang telah dilakukan konversi. Pengujian dilakukan secara straming dengan kondisi citra awal dan *grayscale.*

Gambar 3.6 Citra berwarna *grayscale*

Gambar 3.7. Citra yang sudah enhancement (perbaiki)

3.8.4. Evaluasi Sistem Keseluruhan

Setelah melalui seluruh proses pengujian di atas maka perlu dilakukan pengujian sistem secara keseluruhan. Dimulai dari melihat data citra yang ditangkap oleh Kamera, dan melihat tampilan data citra yang ditampilkan *window image.* Setelah itu, melalui tahap *update* citra, yaitu ketika waktu sistem menunjukkan detik ke-5 atau kelipatannya maka gambar yang tersimpan pada direktori **D://File TA//motiondetection1.jpg** akan berubah sesuai kondisi citra pada saat menit ke-5 atau kelipatannya dan apabila waktu belum mencapai 5 detik atau kelipatannya maka gambar yang tersimpan tidak akan berubah. Kemudian dilanjutkan dengan melihat hasil tahap pengolahan citra, yaitu ketika tiap 5 detik sekali, maka program akan memperbarui *list* nomor parkir. Kemudian sebagai tambahan, kamera pada miniatur juga mengirimkan citra yang disorot untuk ditampilkan pada PC secara *streaming*..

Secara keseluruhan sistem mampu berjalan sesuai dengan apa yang diharapkan, citra berwarna RGB yang diubah menjadi citra *grayscale* dan kemudian dikonversi kedalam *image enhancement* untuk mengetahui setiap detail citra yang ditangkap. Jika keseluruhan sistem telah berjalan sesuai dengan langkah-langkah tersebut, maka secara keseluruhan sistem ini sudah dikatakan baik