BAB III

METODE DAN PERANCANGAN SISTEM

3.1. Metode Pengembangan Sistem

Agar pengembangan sistem dapat dilakukan dalam waktu yang cukup singkat, model RAD (*Rapid Application Development*) digunakan sebagai basis pengembangan. RAD merupakan sebuah *life cycle* pengembangan *software* yang bertujuan untuk waktu pengembangan yang lebih cepat dan hasil yang lebih berkualitas dibandingkan dengan *life cycle* tradisional (Roth, 2003).

Penggunaan RAD dapat mengurangi waktu dan biaya pembangunan dan meningkatkan keberhasilan (Rosenblatt, 2013). Selain itu, beliau juga menyatakan bahwa model pengembangan RAD terdiri dari empat buah fase, antara lain:

1. Perencanaan kebutuhan, tahap berguna untuk menemukan isu-isu kunci yang perlu diselesaikan.
2. *User Design*, pada tahap ini desain dilakukan secara meningkat hingga pengguna dapat mengerti, memodifikasi dan menyetujui kinerja sistem. Dengan kata lain, desain dilakukan hingga kebutuhan terpenuhi.
3. Pembangunan, fokus dari tahap ini adalah membangun aplikasi layaknya pada model SDLC, hanya saja pengguna dapat terus memberikan masukan dan perubahan aplikasi tetap dimungkinkan.
4. *Cutover*, fase ini merupakan fase akhir dalam pengembangan, termasuk *testing*, konversi data dan penyesuaian lingkungan sistem yang dibuat.

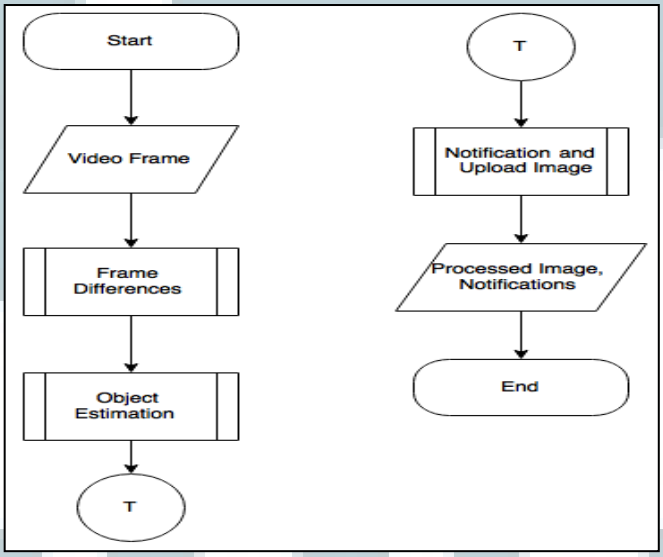
3.2. Perancangan Sistem

Pada bagian ini akan ditampilkan perancangan sistem, yang direpresentasikan dalam *flowchart* dan desain antarmuka.

3.2.1. *Flowchart* Aplikasi

Pada bagian ini ditampilkan desain *flowcharts* dari aplikasi. *Flowcharts* dibuat untuk tiap *subroutine* yang berkaitan dengan algoritma *frame differences* di dalam aplikasi. *Subroutine* adalah sebuah bagian kecil dari program komputer yang menjalankan tugas yang spesifik (Mattson, 2010:281).

A. *Flowchart Subroutine Security Camera*

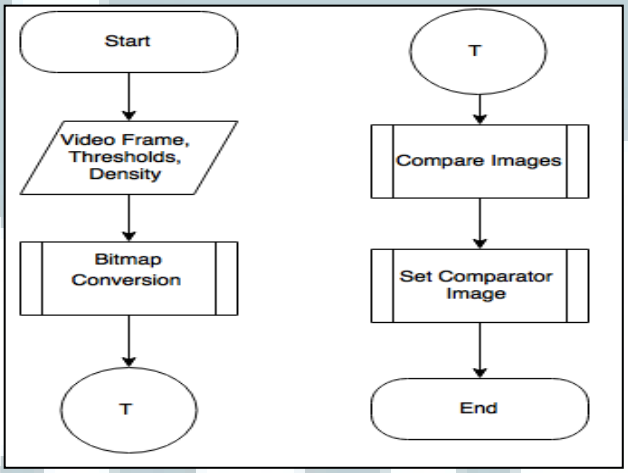


Gambar 3.1. *Flowchart* untuk *Subroutine Security Camera*

Gambar 3.1 menampilkan alur proses-proses umum dalam *subroutine security camera*. Algoritma *frame differences* akan bekerja setiap kali terdapat *frame input* baru, algoritma ini berfungsi untuk melakukan deteksi gerakan atau objek pada *frame*. Setelah itu, proses *object estimation* lalu akan berjalan, proses ini akan menandai objek yang berada di dalam gambar berdasatkan estimasi.

Setelah selesai, aplikasi kemudian akan mengirimkan notifikasi kepada pengguna apabila terdeteksi adanya objek atau gerakan. Sehingga, pada akhirnya *output* yang didapatkan oleh user adalah notifikasi SMS, alarm dan gambar atau *capture* dari gerakan yang terdeteksi melalui e-mail. Video *frames* yang menjadi *input* bagi *algoritma frame differences* memiliki jeda sebesar 1 milisekon dari kumpulan *frames web camera* yang sesungguhnya (*real time)*. Nilai 1 milisekon diambil karena merupakan nilai terendah yang dapat digunakan pada *timer* VB.NET dan paling mendekati *real time*.

B. *Flowchart Subroutine Frame Differences*

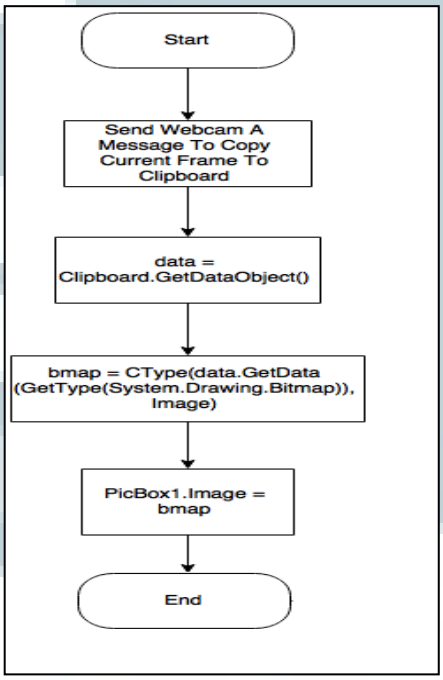


Gambar 3.2. *Flowchart* untuk *Subroutine Frame Differences*

Gambar 3.2 menampilkan alur proses di dalam *subroutine frame differences*. Selain menerima *video frame*, Algoritma *frame differences* juga menerima nilai *threshold* dan *density*. Pada awalnya, nilai-nilai ini bernilai *default* dapat diubah oleh user melalui aplikasi. Setelah itu, *frame* yang diterima kemudian juga akan diubah menjadi *bitmap* (gambar). Setelah itu, proses

pembandingan gambar dapat dimulai, gambar *input* akan dibandingkan dengan gambar pembanding untuk menentukan apakah terjadi gerakan atau tidak. Apabila terjadi gerakan, maka gambar pembanding akan diperbarui sesuai dengan metode *dynamic adaptive template matching*.

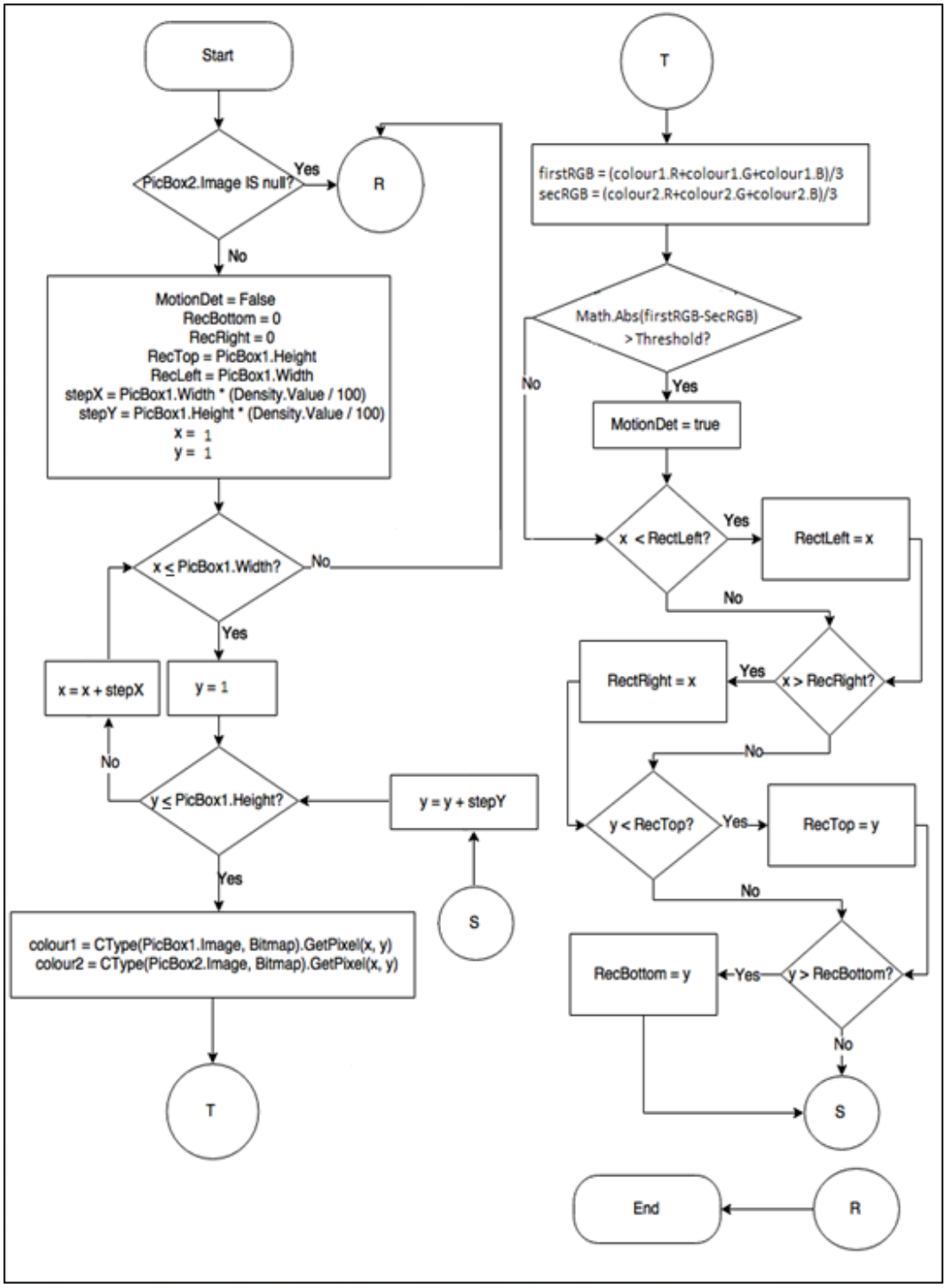
C. *Flowchart Subroutine Bitmap Conversion*



Gambar 3.3. *Flowchart* untuk *Subroutine Bitmap Conversion*

Gambar 3.3 menampilkan alur proses *subroutine bitmap conversion*. Awalnya, *subroutine* ini akan mengirimkan pesan ke *webcam device* untuk menyalin *frame* kedalam *clipboard*, setelah itu *clipboard* akan disalin kedalam variabel data. Selanjutnya, data ini akan dikonversikan menjadi bentuk *bitmap* sehingga dapat diproses lebih lanjut. *Picture box* PicBox1, yakni yang digunakan untuk menampilkan *detection result* lalu akan disi dengan gambar bitmap yang baru saja didapat.

D. *Flowchart Subroutine Compare Images*



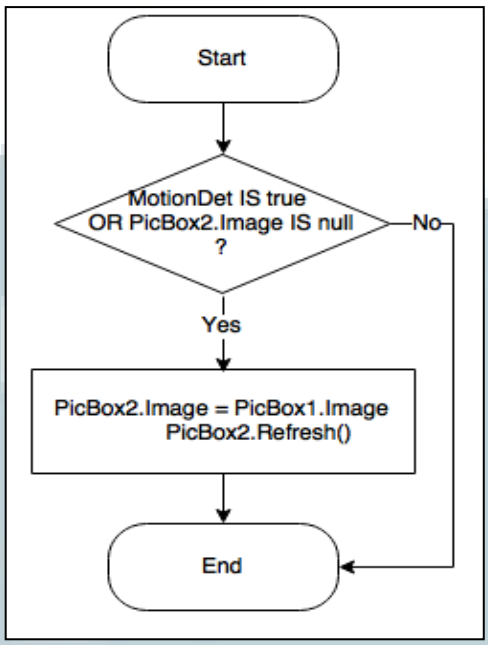
Gambar 3.4. *Flowchart* untuk *Subroutine Compare Images*

Gambar 3.4 menampilkan alur proses di dalam *subroutine compare images*. Proses utama dari algoritma *frame differences* berada didalam *subroutine* ini, awalnya *picture box* kedua, tempat menampung *frame* atau gambar pembanding dicek, apabila masih bernilai *null* maka dari itu proses pengecekan gambar tidak akan dilakukan. Sebaliknya, apabila telah terdapat gambar pembanding maka akan gambar akan dibandingkan.

Berbeda dengan algoritma *frame differences* pada umumnya yang melakukan pengecekan kepada seluruh piksel, algoritma *frame differences* pada penelitian ini dikombinasikan dengan konsep *density*. Sehingga, algoritma tidak perlu mengecek seluruh piksel. *Density* merupakan tingkat kerapatan piksel, penjelasan mengenai nilai *density* dan dampaknya dalam pengecekan piksel pada algoritma *frame differences* terdapat pada subbab 2.7.5.

Setiap piksel yang perlu dicek lalu akan dilihat nilai RGBnya, serta dicari rata-ratanya. Setelah nilai rata-rata RGB pada piksel di posisi x, y gambar saat ini dan gambar pembanding didapatkan, maka keduanya akan diselisihkan. Nilai selisih (*difference*) ini kemudian diubah kedalam bentuk absolut untuk mempermudah perbandingan. Selanjutnya, nilai *difference* RGB akan dibandingkan dengan *threshold*. Jika melebihi *threshold* maka *flag* MotionDet akan diubah menjadi *true*. Seiring pengecekan, dilakukan juga penandaan kepada piksel-piksel yang terdeteksi untuk membentuk kotak, sebagai estimasi dari objek yang terdeteksi. Proses komparasi gambar selesai disini. Selanjutnya, *subroutine set comparator image* akan berjalan.

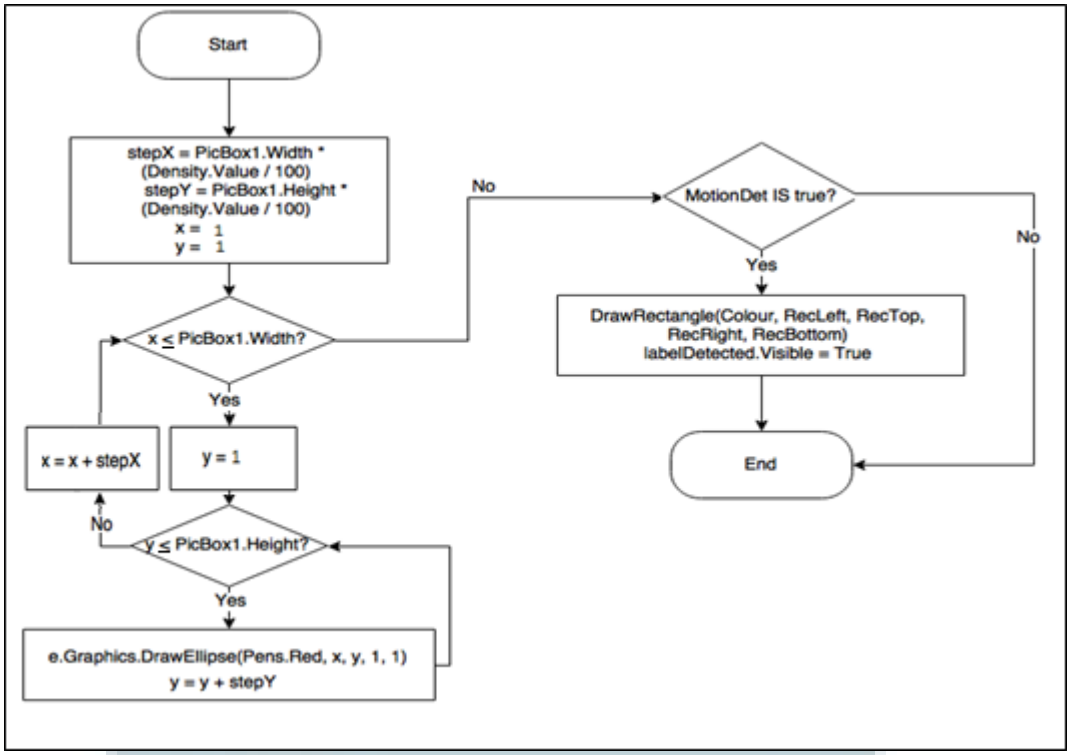
E. *Flowchart Subroutine Set Comparator Image*



Gambar 3.5. *Flowchart* untuk *Subroutine Set Comparator Image*

Gambar 3.5 menampilkan alur proses di dalam *subroutine set comparator image*. *Subroutine* ini akan mengecek hasil dari perbandingan gambar menunjukan dengan melihat ke dalam variabel MotionDet dan isi dari *picture box* gambar pembanding. Jika isi dari *picture box* gambar pembanding masih bernilai *null* yang berarti aplikasi baru saja berjalan dan belum ada gambar pembanding sama sekali, maka gambar yang baru saja ditangkap akan disalin ke dalam *picture box* gambar pembanding. Selain itu, jika nilai variabel MotionDet bernilai *true picture box*, gambar pembanding juga akan diperbarui. Hal ini merupakan implementasi dari teori metode *dynamic adaptive template matching*, dimana apabila terdapat perubahan di dalam lingkungan maka *frame* pembanding diperbarui (Widyawan *et al*., 2012).

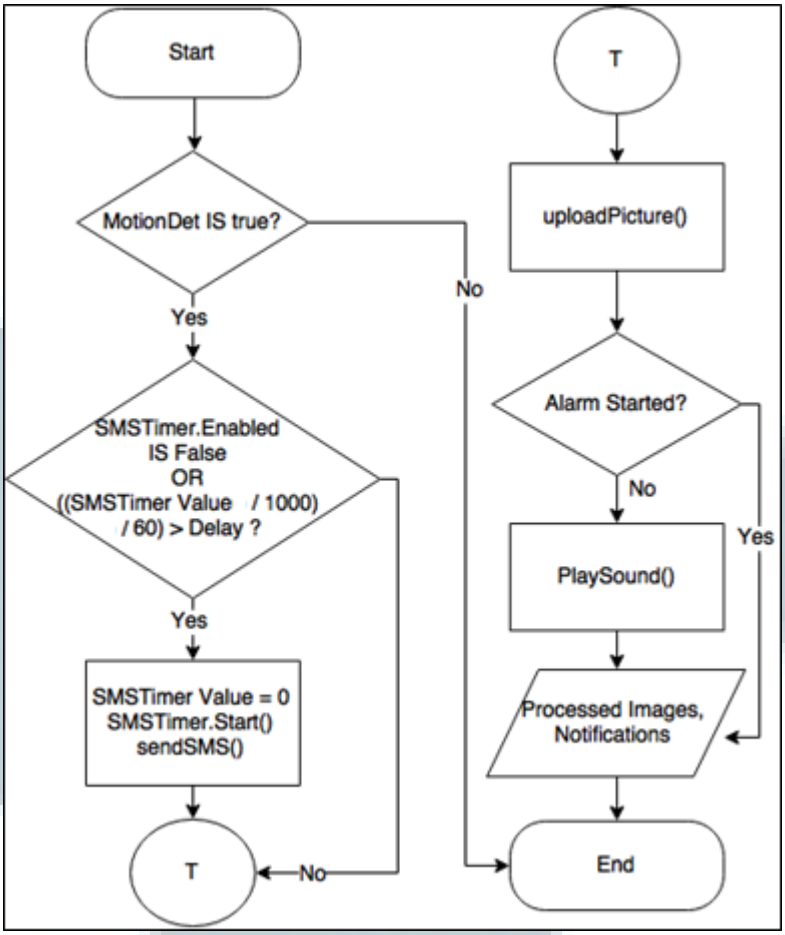
F. *Flowchart Subroutine Object Estimation And Grid Drawing*



Gambar 3.6. *Flowchart* untuk *Subroutine Object Estimation And Grid Drawing*

Gambar 3.6 menampilkan alur proses di dalam *subroutine object estimation and grid drawing. Subroutine* ini akan menggambarkan *grid* pada *picturebox* PicBox1, *grid* menunjukan piksel mana saja yang dicek pada saat algoritma berjalan sesuai dengan *density*. Setelah itu, isi variabel MotionDet akan dicek. Jika MotionDet bernilai ya, maka bentuk kotak akan dibuat pada *picture box* yang menggambarkan gambar *detection result* yakni PicBox1. Titik-titik yang digunakan untuk membuat kotak berdasarkan nilai variabel RecLeft, RecTop, RecRight dan RecBottom, seluruh nilai ini telah diisi pada *subroutine compare image*. Kotak yang digambarkan pada *picture box* berguna sebagai estimasi area dari objek yang bergerak dan juga untuk menandakan adanya terdeteksi.

G. *Flowchart Subroutine Notification And Upload Image*



Gambar 3.7. *Flowchart* untuk *Notification and Upload Image*

Gambar 3.7 menampilkan alur proses di dalam *subroutine notification and upload image* melalui *e-mail.* Setelah gambar *input* dan pembanding selesai dikomparasi, *subroutine* ini kemudian berjalan. Pertama-tama akan dilakukan pengecekan kepada nilai MotionDet, jika bernilai *false* maka akan dilakukan *upload* gambar yang baru saja terdeteksi melalui e-mail, setelah itu *alarm* akan dibunyikan jika tidak sedang aktif.

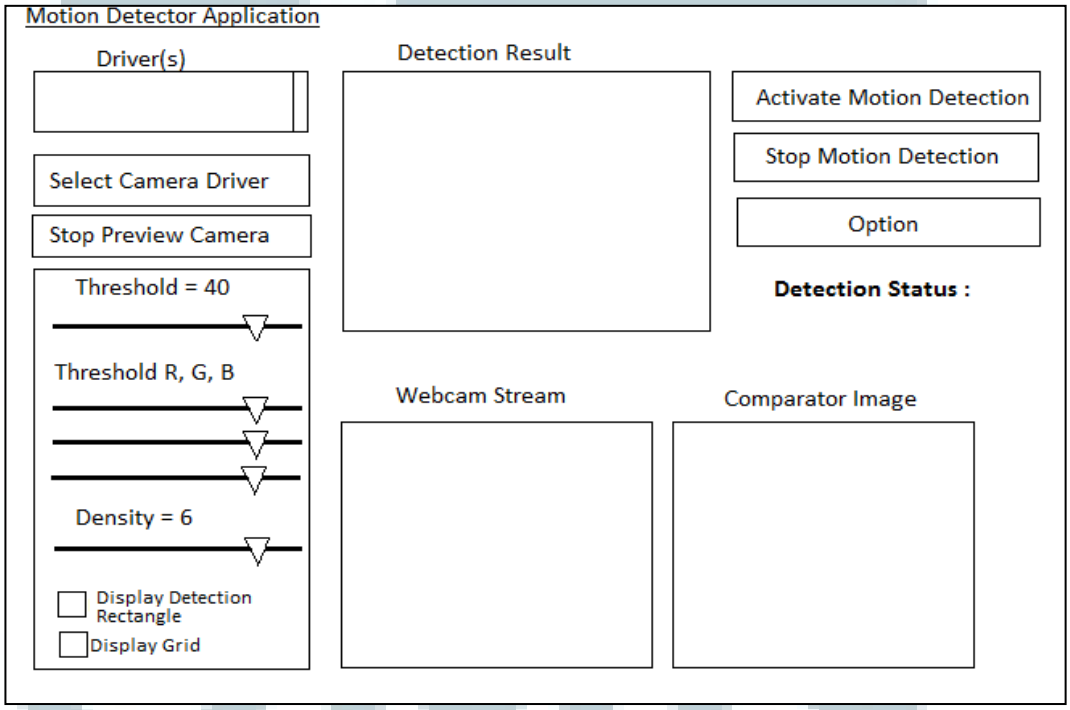
Akan tetapi, jika nilai MotionDet adalah *true* maka, akan dilakukan pengecekan terhadap *timer* SMSTimer, jika belum aktif maka timer akan

diaktifkan. Lalu SMS akan dikirim kepada *user* untuk memberitahu adanya deteksi setelah itu, proses *upload* gambar akan dilakukan melalui *attachment* melalui e-mail, kemudian e-mail akan dikirimkan dan *alarm* akan dicek. Namun, jika *timer* telah aktif maka akan di cek nilainya, jika *timer* belum berjalan lebih dari batas menit yang ditentukan oleh *user* pada aplikasi maka, SMS tidak akan dikirmkan. Hal ini menghindari terjadinya *spam* SMS.

3.2.2 Desain Antarmuka

Pada bagian ini akan ditampilkan desain antarmuka dari aplikasi.

A. Desain Antarmuka *Form Main*

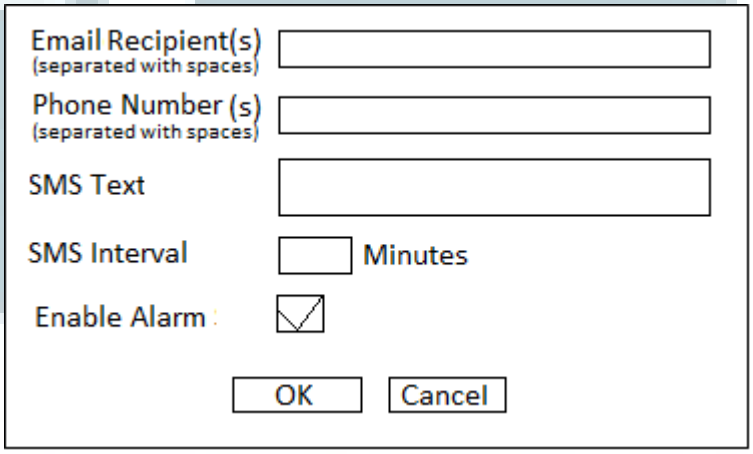


Gambar 3.8. Desain Antarmuka untuk *Form Main*

Aplikasi ini hanya memiliki satu buah *form* yang langsung ditampilkan ketika *user* membuka aplikasi, *form* ini berisi fitur-fitur utama dari aplikasi. Pada *form main*, *user* dapat memilih *driver webcam* untuk melakukan koneksi

atau diskoneksi dengan menekan tombol *select camera device* atau *stop preview camera*. Setelah itu *user* dapat memulai atau menghentikan deteksi dengan menekan tombol *activate motion detection* atau *stop motion detection*. Terdapat juga kolom untuk mengatur sensitivitas RGB dan *density* untuk digunakan pada algoritma *frame difference*, opsi untuk menampilkan *grid*, opsi untuk menampilkan *rectangle detection* serta tombol *option* untuk mengatur notifikasi.

B. Desain Antarmuka *Form Option*



Gambar 3.9. Desain Antarmuka untuk *Form Option*

Form option berisi opsi-opsi untuk mengatur notifikasi dari aplikasi, seperi e-mail yang akan mendapatkan gambar-gambar deteksi, nomor tujuan untuk menerima notifikasi SMS dan *delay* antar pengiriman SMS yang harus dipenuhi agar tidak terjadi *spamming*. E-mail yang digunakan haruslah berada pada server *gmail*, terdapat validasi untuk hal ini pada saat *user* menekan tombol OK. Selain itu, pada *form* ini notifikasi alarm (suara) juga dapat diaktifkan atau dinonaktifkan.