PROGRAM PENDETEKSI DAN PENGHITUNG JUMLAH PENGUNJUNG DENGAN TEKNIK PENCITRAAN DIGITAL MENGGUNAKAN *WEBCAM*

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan Untuk Memenuhi Persyaratan
Memperoleh Gelar Sarjana Teknik

Disusun Oleh:

WIRAWAN HIDAYAT

NIM. 0710630083-63

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN

UNIVERSITAS BRAWIJAYA

FAKULTAS TEKNIK

MALANG

2012

LEMBAR PERSETUJUAN

PROGRAM PENDETEKSI DAN PENGHITUNG JUMLAH PENGUNJUNG DENGAN TEKNIK PENCITRAAN DIGITAL MENGGUNAKAN *WEBCAM*

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan untuk memenuhi sebagian persyaratan
Memperoleh gelar Sarjana Teknik

Disusun oleh:

WIRAWAN HIDAYAT

NIM. 0710630083 - 63

Telah diperiksa dan disetujui oleh

Dosen Pembimbing:

Pembimbing I	**Pembimbing II**
Ir. Muhammad Aswin, MT.	**Adharul Muttaqin, ST., MT.**
NIP. 19640626 199002 1 001	**NIP. 19760121 200501 1 001**

LEMBAR PENGESAHAN

PROGRAM PENDETEKSI DAN PENGHITUNG JUMLAH PENGUNJUNG DENGAN TEKNIK PENCITRAAN DIGITAL MENGGUNAKAN *WEBCAM*

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan untuk memenuhi persyaratan

memperoleh gelar Sarjana Teknik

Disusun oleh:

WIRAWAN HIDAYAT
NIM. 0710630083 - 63

Skripsi ini telah diuji dan dinyatakan lulus pada

tanggal 09 Februari 2012

DOSEN PENGUJI

Waru Djuriatno, ST., MT.	**Sigit Kusmaryanto, Ir., M.Eng.**
NIP. 19690725 1997202 1 001	**NIP. 19700310 199412 1 001**

Ali Mustofa, ST., MT.
NIP. 19710601 200003 1 001

Mengetahui

Ketua Jurusan Teknik Elektro

Dr. Ir. Sholeh Hadi Pramono, MS.
NIP. 19580728 198701 1 001

PENGANTAR

Puji dan syukur penulis panjatkan kepada Allah SWT, karena dengan rahmat, taufik dan hidayah-Nya lah skripsi ini dapat diselesaikan.

Skripsi berjudul "Program Pendeteksi dan Penghitung Jumlah Pengunjung Dengan Teknik Pencitraan Digital Menggunakan *Webcam*" ini disusun untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Teknik di Jurusan Teknik Elektro Universitas Brawijaya.

Penulis menyadari bahwa penyusunan skripsi ini tidak terlepas dari bantuan berbagai pihak. Oleh karena itu, dengan ketulusan dan kerendahan hati penulis menyampaikan terima kasih kepada:

- Ibunda Endang Rusmini, Ayahanda Djoko Suminto, kakak Erwan Vembrianto dan Erbi Krismuharja, serta adik tercinta Digna Puspita Sari atas segala nasehat, kasih sayang, perhatian dan kesabarannya serta telah banyak mendoakan kelancaran penulis hingga terselesaikannya skripsi ini,
- Bapak Dr. Ir. Sholeh Hadi Pramono, MS. selaku Ketua Jurusan dan bapak Muhammad Aziz Muslim, ST., MT., PhD. selaku Sekretaris Jurusan Teknik Elektro Universitas Brawijaya,
- Bapak Waru Djuriatno, ST., MT.. selaku Ketua Kelompok Dosen Keahlian Rekayasa Komputer Jurusan Teknik Elektro Universitas Brawijaya,
- Bapak Ir. Muhammad Aswin., MT., dan Bapak Adharul Muttaqin, ST., MT., selaku Dosen Pembimbing atas segala bimbingan, nasehat, pengarahan, motivasi, saran dan masukan yang telah diberikan dalam pengerjaan skripsi,
- Karunia Tri Wahyuni, yang selalu memberi doa dan dukungan kepada penulis dalam pengerjaan skripsi ini,
- Aulia Wildan, ST., Fanni Kharisma, ST., Boby Septian S., Ichwan Maulana, Wildan Khoirurrizqi, Gallant Hendrayana, Dwyan Zakaria, dan Ida Bagus Willy,ST., Tito Cahyo Prasetyo, ST., Abdurrahman yang selalu memberi doa serta menemani saya dalam suka dan duka selama menjalani masa perkuliahan di Teknik Elektro Universitas Brawijaya,

- Teman-teman Core-nerz ’07, teman-teman di kelembagaan, senior serta semua pihak yang tidak mungkin bagi penulis untuk mencantumkan namanya satu-persatu, terima kasih banyak atas bantuan dan dukungannya.

Pada akhirnya, penulis menyadari bahwa skripsi ini masih belum sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun. Penulis berharap semoga skripsi ini dapat bermanfaat bagi pengembangan ilmu pengetahuan dan teknologi.

Malang, Januari 2012

Penulis

ABSTRAK

Wirawan Hidayat, Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya, Program Pendeteksi dan Penghitung Jumlah Pengunjung dengan Teknik Pencitraan Digital Menggunakan *Webcam*, Dosen Pembimbing: Ir. Muhammad Aswin, MT., dan Adharul Muttaqin, ST., MT.

Untuk mendapatkan data jumlah pengunjung yang masuk pada suatu gedung umumnya masih dilakukan secara manual dengan memanfaatkan tenaga manusia sebagai penghitung jumlah pengunjung. Semakin banyak pintu masuk yang digunakan, maka semakin banyak pula tenaga manusia yang dibutuhkan. Dengan berkembangnya teknologi, tugas tersebut dapat dibantu dengan melakukan penerapan salah satu ilmu komputer yaitu pencitraan digital yang mengolah gambar atau video menjadi informasi yang bermanfaat dan tercatat.

Pada skripsi ini ditunjukkan bagaimana membuat sebuah program pendeteksi dan penghitung jumlah pengunjung dengan pencitraan digital menggunakan *webcam*. Berdasarkan *image* setiap satu *frame* yang berhasil ditangkap *webcam,* dilakukan penghitungan untuk mencari selisih piksel antara *background image* dan *frame image*. Selisih piksel tersebut dikuadratkan dan dibagi dengan luas *detection window.* Nilai hasil perhitungan ini kemudian dibandingkan dengan nilai *threshold* untuk diketahui adanya pegunjung yang melewati *detection window.* Tingkat keberhasilan dari program ini sebesar 97,5% dengan pengujian menggunakan sumber cahaya yang berbeda-beda.

Kata kunci: *webcam,* penghitung pengunjung, metode penghitungan,*detection window*.

DAFTAR ISI

PENGANTAR i
ABSTRAK iii
DAFTAR ISI iv
DAFTAR GAMBAR vii
DAFTAR TABEL viii
BAB I 9
PENDAHULUAN 9
1.1 Latar Belakang Masalah 9
1.2 Rumusan Masalah 9
1.3 Batasan Masalah 10
1.4 Tujuan 10
1.5 Sistematika Penulisan 10
BAB II 12
Tinjauan Pustaka 12
2.1. Pengolahan Citra 12
2.1.1 Pengolahan Warna RGB 13
2.1.2 Grayscale 14
2.1.3 Pengambangan (*Thresholding*) 15
2.1.4. Low Pass Filter 17
2.2 Deteksi Objek 17
2.2.1 Deteksi Objek Tunggal 18
2.3 Moving Detector 18
2.4 Webcam 19
BAB III 20
METODE PENELITIAN 20
3.1 Studi Literatur 20
3.2 Diagram Sistem 20
3.3 Cara Kerja Sistem 21
3.4 Pengujian sistem 23
3.4.1. Prosentase *error* 24

3.5. Kesimpulan dan Saran 24
BAB IV 25
PERANCANGAN DAN IMPLEMENTASI 25
4.1 Perancangan Secara Umum 25
4.1.1 Blok Diagram Sistem 25
4.1.2 Cara Kerja Aplikasi 26
4.1.3 Konfigurasi Perangkat Keras 27
4.2 Perancangan Perangkat Lunak 27
4.2.1 Perancangan Antarmuka (*Interface*) 28
4.2.2 Perancangan Inisialisasi Webcam 29
4.2.3 Perancangan Program Pengolahan Digital 30
4.2.4 Perancangan Program Keseluruhan 37
4.3 Implementasi Sistem 38
4.3.1 Lingkungan Implementasi 38
4.3.2 Implementasi Algoritma Bagian Inisialisasi *Webcam* 39
4.3.3 Implementasi Algoritma Bagian Pengolahan Digital 40
4.4 Implementasi Antarmuka (Interface) 46
4.4.1 Implementasi Antarmuka (*Interface*) Saat Akan Dijalankan 48
4.4.2 Implementasi Antarmuka (interface) saat Pengambilan background 49
BAB V 51
PENGUJIAN 51
5.1 Pengujian webcam 51
5.1.1 Pengujian Koneksi *device* atau *Webcam* 51
5.2 Pengujian Program Pendeteksi dan Penghitung Jumlah Pengunjung 52
5.2.1 Pengujian Program Pendeteksi dan Penghitung Jumlah Pengunjung Dengan Kondisi Cahaya yang Berbeda-beda 52
5.2.2 Pengujian Program Pendeteksi dan Penghitung Jumlah Pengunjung Dengan Kondisi Kecepatan Jalan Normal 54
5.3 Pengujian Keseluruhan 58
5.4 Analisis Kegagalan 59
5.2.1 Analisis Kegagalan Program Pendeteksi dan Penghitung Jumlah Pengunjung 59
BAB VI 61
PENUTUP 61
6.1 Kesimpulan 61
6.2 Saran 61

DAFTAR PUSTAKA 62

DAFTAR GAMBAR

Gambar 2.1 fungsi citra dalam bentuk matriks 13
Gambar 2.2 Ilustrasi citra digital 13
Gambar 2.3 Nilai Warna RGB dalam Hexadesimal 13
Gambar 2.5 citra sebenarnya 15
Gambar 2.6 citra *grayscale* 15
Gambar 3.1 Diagram Sistem 21
Gambar 3.3 Proses Pengambilan Video dengan Menggunakan *webcam* tampak atas 22
Gambar 4.1 Blok diagram sistem 25
Gambar 4.2 Gambar Perancangan Sistem 27
Gambar 4.3 Flowchart perancangan aplikasi 28
Gambar 4.4 Perancangan Tampilan awal program 29
Gambar 4.5 Video capture 29
Gambar 4.6 Blok Diagram sistem *buffer* Windows API 30
Gambar 4.7 Flowchart bagian pengolahan digital 31
Gambar 4.8 Ilustrasi pengambilan *background* 32
Gambar 4.9 Proses penentuan *detection window* 33
Gambar 4.10 Proses *grayscaling* 34
Gambar 4.11 Proses Perhitungan nilai *threshold* 35
Gambar 4.12 Proses Perhitungan piksel 37
Gambar 4.13 Flowchart Proses keseluruhan 38
Gambar 4.14 Flowchart Inisialisasi *webcam* 39
Gambar 4.15 Detection windows pada image 42
Gambar 4.16 Tampilan Awal Program 47
Gambar 4.17 Tampilan program ketika akan dijalankan 49
Gambar 4.18Tampilan saat Pengambilan *background* 49
Gambar 5.1 Pengujian koneksi *webcam* 52
Gambar 5.2 Pengujian nilai *threshold* 1 56
Gambar 5.3 Pengujian nilai *threshold* 2 56
Gambar 5.4 Pengujian nilai *threshold* 3 57
Gambar 5.5 Pengujian nilai *threshold* 4 57
Gambar 5.6 Pengujian nilai *threshold* 5 58

DAFTAR TABEL

Tabel 2.1 Contoh thresholding 16
Tabel 2.2 Contoh Hasil perhitungan *thresholding* 17
Table 4.1. Sampel nilai pengali 45
Table 4.2. Tools yang digunakan pada form 47
Tabel 5.1 Data Hasil Pengujian 51
Tabel 5.2 Hasil pengujian dalam kondisi cahaya agak gelap (Lampu Ekonomat 12 W) 53
Table 5.3 Hasil pengujian kondisi cahaya terang (Lampu Philips ESSENTIAL 14 W) 53
Table 5.4 Hasil pengujian kondisi cahaya terang (Lampu Philips Tornado 24 W) 54
Tabel 5.5 Hasil pengujian dengan kecepatan normal (1 m/s) 54
Tabel 5.6 Hasil pengujian nilai *threshold* 55
Tabel 5.7 Hasil pengujian keseluruhan 58

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Data jumlah pengunjung suatu tempat perbelanjaan sangat penting untuk mengetahui seberapa besar minat pengunjung untuk mengunjungi tempat tersebut. Data-data tersebut selalu *update* setiap hari untuk dijadikan trend grafik jumlah pengunjung. Selain itu, data tersebut dapat menjadi prediksi akan perkembangan suatu tempat perbelanjaan mulai dari renovasi hingga reinovasi. Begitu pula data jumlah orang yang masih ada di dalam tempat tersebut, ketika ingin menutup tempat tersebut perlu diketahui terlebih dahulu apakah di dalam ruangan masih ada orang atau tidak.

Sampai saat ini untuk mendapatkan data jumlah pengunjung umumnya masih menggunakan cara manual, yaitu dengan cara menugaskan orang untuk menghitung jumlah pengunjung menggunakan *hand counter*. Cara manual ini jelas membutuhkan bantuan tenaga manusia untuk melaksanakannya. Belum lagi jika suatu tempat perbelanjaan memiliki banyak pintu masuk,maka akan lebih membutuhkan tenaga manusia untuk melakukan perhitungan jumlah pengunjung.

Salah satu solusi yang dapat digunakan untuk mengatasi masalah tersebut adalah memanfaatkan kecanggihan teknologi. Apabila pengambilan data dilakukan dengan otomatis yaitu menggunakan alat bantu berupa kamera, lalu diproses dengan PC (*Personal Computer*), maka jalannya proses perhitungan akan jadi lebih efisein, karena tidak memerlukan banyak tenaga manusia untuk pengambilan data.

Dalam skripsi ini, penulis mengajukan judul program pendeteksi dan penghitung jumlah pengunjung dengan teknik pencitraan digital menggunakan *webcam*. Pengambilan video dilakukan dengan kamera yang dipasang tepat di atas pintu masuk dan kemudian data video diproses oleh PC.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas maka rumusan masalah dari skripsi ini adalah:

1. Bagaimana cara mengambil data agar diperoleh jumlah pengunjung dengan menggunakan teknik pengolahan citra.
2. Bagaimana cara mendeteksi adanya suatu objek (pengunjung).
3. Bagaimana cara menghitung jumlah pengunjung.

1.3 Batasan Masalah

Agar penulisan ini lebih terarah dan sesuai dengan tujuan yang diinginkan maka permasalahan perlu dibatasi sebagai berikut:

1. Objek (pengunjung) yang melewati pintu masuk adalah manusia.
2. Objek (pengunjung) yang dihitung hanya yang tertangkap kamera dan melewati area deteksi (*detection window*) yang telah ditentukan posisi dan ukurannya.
3. Posisi kamera tetap (*static)* dan berada diatas pintu masuk tegak lurus 90^o terhadap bidang latar.
4. Hasil ditentukan saat perekaman berlangsung(*real time*).
5. Objek (pengunjung) yang dideteksi dan dihitung adalah objek tunggal.
6. Objek (pengunjung) melewati pintu masuk yang didesain untuk masuk satu per satu.

1.4 Tujuan

Adapun Tujuan skripsi ini adalah:

Dapat membuat *software* yang bisa mendeteksi dan menghitung jumlah objek (pengunjung) yang melewati pintu masuk.

1.5 Sistematika Penulisan

Sistematika penulisan laporan skripsi ini adalah sebagai berikut :

BAB I Pendahuluan

Memuat latar belakang, rumusan masalah, tujuan, batasan masalah, manfaat dan sistematika pembahasan.

BAB II Tinjauan Pustaka

Membahas teori-teori yang mendukung dalam perancangan dan pembuatan aplikasi mendeteksi objek, menghitung jumlah objek yang melewati pintu masuk secara *real time*.

BAB III Metode Penelitian

Berisi tentang metode dalam penelitian yang digunakan dalam perancangan dan pengujian program.

BAB IV Perancangan dan Implementasi

Membahas tentang langkah-langkah perancangan dan implementasi aplikasi serta menjelaskan algoritma aplikasi.

BAB V Pengujian dan Analisis

Memuat analisis dan hasil pengujian terhadap sistem yang telah diimplementasikan.

BAB VI Penutup

Memuat kesimpulan dan saran-saran.

BAB II
Tinjauan Pustaka

2.1. Pengolahan Citra

Pengolahan citra merupakan proses pengolahan suatu citra yang menghasilkan keluaran citra dengan kualitas yang lebih baik. Proses ini mempunyai ciri data masukan dan informasi keluaran yang berbentuk citra. Istilah pengolahan citra digital secara umum didefinisikan sebagai pemrosesan citra dua dimensi dengan komputer. Dalam definisi yang lebih luas, pengolahan citra digital juga mencakup semua data dua dimensi.

Umumnya citra digital digambarkan dalam bentuk persegi panjang atau bujur sangkar yang memiliki lebar dan tinggi tertentu. Ukuran ini biasanya dinyatakan dalam banyaknya titik *(point)* atau piksel sehingga ukuran citra selalu bernilai bulat. Setiap titik atau piksel memiliki koordinat sesuai posisinya masing-masing dalam citra. Koordinat ini biasanya dinyatakan dalam bilangan bulat positif, yang dapat dimulai dari indeks berapapun tergantung pada sistem yang digunakan. Setiap titik atau piksel juga memiliki nilai berupa angka digital yang merepresentasikan informasi yang diwakili oleh titik tersebut.

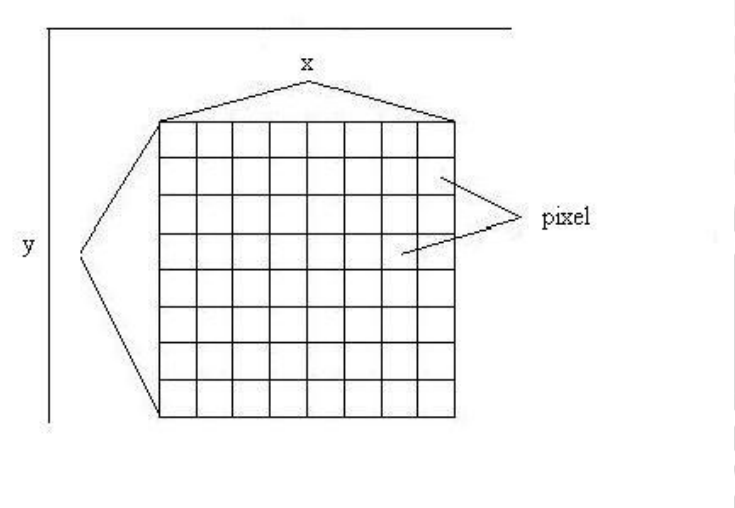
Format data citra digital sangat berhubungan erat dengan warna. Pada kebanyakan kasus, terutama untuk keperluan penampilan secara visual, nilai data digital merepresentasikan warna dari citra yang diolah. Format citra digital yang banyak dipakai adalah Citra Biner (*monokrom*), Citra Skala Keabuan (*grayscale*),dan Citra Warna Asli (*true color*).

Beberapa faktor menyebabkan perkembangan sistem pengolahan citra menjadi lebih berkembang pesat pada saat ini. Salah satu yang utama adalah dibutuhkannya suatu teknologi yang dapat memproses data-data yang diterima dan pada akhirnya teknologi tersebut dapat mengambil keputusan sendiri dari hasil pengolahan data tentunya dengan tingkat akurasi hasil yang lebih baik.

Citra digital adalah representasi objek berupa barisan bilangan nyata yang diwakili oleh bit-bit tertentu yang terhimpun dalam bentuk *array*. Citra digital dapat digambarkan dalam bentuk matrik yang didefinisikan sebagai fungsi *f(x,y) berukuran R baris dan C kolom*, dimana *x* dan *y* adalah koordinat dan nilai *f(x,y)* adalah nilai citra pada koordinat tersebut.

$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,C-1) \\ f(1,0) & f(1,1) & \cdots & f(1,C-1) \\ \vdots & \vdots & & \vdots \\ f(R-1,0) & f(R-1,1) & \cdots & f(R-1,C-1) \end{bmatrix}$$

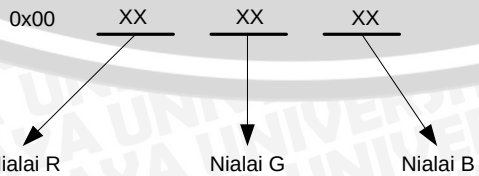
Gambar 2.1 fungsi citra dalam bentuk matriks
(**Sumber :** http://kuliahinformatika.wordpress.com)



Gambar 2.2 Ilustrasi citra digital
(Sumber: http://dendieisme.blogspot.com)

2.1.1 Pengolahan Warna RGB

Dasar pengolahan citra adalah pengolahan warna RGB (*Red Green Blue*) pada koordinat-kordinat tertentu. Pengolahan citra dipresentasikan dengan nilai heksadesimal dari 0x00000000 sampai 0x00ffffffff. Warna hitam adalah 0x00000000 dan warna putih adalah 0x00ffffff. Gambar 2.3 menunjukkan definisi nilai warna dan variable 0x00 menyatakan angka dibelakangnya adalah heksadesimal.



Gambar 2.3 Nilai Warna RGB dalam Hexadesimal
(**Sumber :** http://aqsha-eepis.blogspot.com)

Setiap warna mempunyai *range* nilai 00 (angka desimalnya adalah 0) dan ff (angka desimalnya 255) atau mempunyai nilai derajat keabuan 256 $=2^8$. Dengan demikaian, range warna yang digunakan adalah (2^8) (2^8) (2^8) = 2^{24} (atau dikenal dengan istilah *true colour* pada *windows*). Nilai warna yang digunakan merupakan gabungan warna cahaya merah(R), hijau(G), dan biru(B).

2.1.2 Grayscale

Proses awal yang banyak dilakukan dalam image processing adalah mengubah citra berwarna menjadi citra *grayscale*, hal ini digunakan untuk menyederhanakan model citra yang akan diamati. *Grayscale* banyak digunakan jika dalam suatu citra image terdapat perbedaan piksel dengan intensitas yang kecil antara objek yang diamati terhadap objek lainnya. Seperti telah dijelaskan di depan, citra berwarna terdiri dari 3 layer yaitu R-layer, G-layer dan B-layer. Sehingga untuk melakukan proses-proses selanjutnya tetap diperhatikan tiga layer di atas. Bila setiap proses perhitungan dilakukan menggunakan tiga layer, berarti dilakukan tiga perhitungan yang sama. Sehingga konsep itu diubah dengan mengubah 3 layer di atas menjadi 1 layer *grayscale* dan hasilnya adalah citra *grayscale*. Dalam citra ini tidak ada lagi warna, melainkan yang ada adalah derajat keabuan.

Untuk mengubah citra berwarna yang mempunyai nilai matrik masing-masing r, g dan b menjadi citra *grayscale* dengan nilai s, maka konversi dapat dilakukan dengan mengambil rata-rata dari nilai r, g dan b sehingga dapat dituliskan menjadi:

$$s = \frac{R + G + B}{3}$$

Hasil dari perhitungan nilai *grayscale* akan memberikan keluaran berupa nilai dari tiap piksel yang berbeda dari sebelumnya. Operasi konversi menggunakan proses *grayscale* umumnya digunakan untuk mempermudah dalam proses perhitungan selanjutnya. Berikut contoh gambar yang menunjukkan hasil dari *grayscaling* dari citra RGB menjadi *grayscale*:

(a) (b)

Gambar 2.5 citra sebenarnya
(**Sumber:** http://digilib.mdp.ac.id)

Gambar 2.6 citra *grayscale*
(**Sumber:** http://digilib.mdp.ac.id)

2.1.3 Pengambangan (*Thresholding*)

Operasi pengambangan (*thresholding*) digunakan untuk mengubah citra dengan format skala keabuan ke citra biner, yang hanya memiliki nilai (0 atau 1). Dalam hal ini, titik dengan nilai rentang nilai keabuan tertentu diubah menjadi berwarna hitam dan sisanya menjadi putih, atau sebaliknya. Penentuan nilai *threshold* dapat dilakukan dengan manual yaitu secara langsung menntukan nilai yang dikehendaki ,dan dapat dilakukan secara otomatis yakni dengan mendapatkan nilai *threshold* berdasarkan hasil perhitungan tertentu yang nantinya akan dijadikan suatu nilai parameter *threshold.* Untuk mendapatkan nilai *threshold* dengan menggunakan perhitungan biasanya memerlukan beberapa proses tambahan seperti contoh dalam proses perhitungan selisih piksel yang nantinya akan menjalankan sistem *counter* , maka untuk mendapatkan nilai *threshold* diperlukan proses perhitungan rata-rata yang nantinya akan didapatkan suatu nilai ambang tertentu yang menjadi nilai *threshold*.

Salah satu dari dua operasi pengambangan yang banyak digunakan adalah pengambangan tunggal. Pengambangan tunggal memiliki sebuah nilai batas ambang. Untuk menetukan nilai ambang dengan persamaan:

$$T = \frac{f_{maks} + f_{\min}}{2}$$

Dengan ketentuan :

T = nilai ambang

f_{maks} = nilai intensitas maksimum pada citra

$f_{\min}$ = nilai intensitas minimum pada citra

Apabila nilai ambang sudah ditemukan maka warna *pixel* akan digantikan dengan warna hitam atau putih tergantung kondisi berikut :

$$f(x,y) = 255, \text{ jika } f(x,y) \geq T \qquad (2.3)$$

$$f(x,y) = 0, \text{ jika } f(x,y) \leq T$$

dengan :

$f(x,y)$ = nilai intensitas *pixel* pada posisi (x,y)

T = nilai ambang

Sebagai contoh, diketahui citra grayscale 4x4 *pixel* dengan kedalaman 8 bit seperti berikut.

200	230	150	75
240	50	170	90
210	100	120	80
100	90	200	230

Tabel 2.1 Contoh thresholding

(**Sumber:** perancangan)

Dengan metode ini, nilai ambang dari citra diatas adalah

$$T = \frac{f_{maks} + f_{\min}}{2} = \frac{240+50}{2} = 145$$

Bila nilai T =145 diterapkan untuk citra tersebut diperoleh citra berikut:

255	255	255	0
255	0	255	0
255	255	0	0
0	0	255	255

Tabel 2.2 Contoh Hasil perhitungan *thresholding*
(**Sumber:** perancangan)

2.1.4. Low Pass Filter

Low pass filter dilakukan untuk menghilangkan ruang derau berfrekuensi tinggi dari sebuah gambar digital. Istilah derau atau "*Noise*" digunakan sebagai efek samping dari proses .*Noise* merupakan variasi yang tidak diinginkan terjadi dalam sebuah pixel. *Low pass filter* juga digunakan untuk mengurangi detail dari gambar atau justru membuat gambar menjadi lebih kabur dari sebelumnya namun memiliki *noise* yang lebih kecil dari sebelumnya. Salah satu bentuk dari penghilangan noise adalah dengan rerata. Rerata ini menggunakan cara merata-rata setiap data perhitungan yang diperoleh. Untuk rata-rata 5 data perhitungan, maka nilai data ke-(n) dirata-rata dengan jumlah data (n=1, 2, …, 5).

Fil(5) = (nilai(1) + nilai(2) + nilai(3) + nilai(4)+ nilai(5)) / 5

2.2 Deteksi Objek

Deteksi objek adalah suatu cara untuk mendapatkan atau mengetahui keberadaan dari suatu objek. Dalam pendeteksian atau mengetahui keberadaan suatu objek, dalam prosesnya biasanya melakukan pencarian objek dengan melakukan penelusuran (*scanning*) sehingga akan didapatkan nilai posisi dari objek yang dideteksi. Cara penelusuran (*scanning*) untuk mendeteksi objek dapat dilakukan pada daerah tertentu yang dijadikan fokus penelusuran, artinya proses penelusuran hanya dititik beratkan pada daerah tertentu yang diinginkan sehingga diluar daerah tersebut dihiraukan atau diabaikan saja. Cara ini biasanya dilakukan dengan tujuan agar suatu proses pendeteksian objek berjalan lebih cepat.

2.2.1 Deteksi Objek Tunggal

Objek tunggal disini merupakan objek yang dideteksi dipastikan hanya berjumlah satu,artinya dalam pengangkapan *image* nantinya kemungkinan hanya ada satu objek yang akan muncul. Jika daerah deteksi (*detection window*) sudah ditentukan posisinya dan kamera diposisikan *static* (diam), maka proses deteksi dilakukan cukup dengan mengamati daerah tersebut, tanpa menghiraukan daerah yang lainnya. Dengan memastikan bahwa objek akan melintasi daerah deteksi tersebut, maka proses deteksi dipusatkan pada daerah tersebut.

2.3 *Moving Detector*

Jika dalam sistem pendeteksian objek berkaitan dengan pendeteksian objek yang bergerak , maka salah satu metode yang dapat digunakan adalah *moving detector*. Dalam metode ini, proses pendeteksian tentang keberadaan dari suatu objek didapatkan dari perbedaan *pixel-pixel* sautu *image*. Jika tidak ada objek yang bergerak sedikitpun, maka tidak akan ada pula perbedaan *pixel* dari suatu perhitungan *image* tersebut. Namun sebaliknya, jika terdapat objek yang bergerak maka akan muncul perbedaan nilai *pixel* dari perhitungan suatu image baik berupa warna maupun intensitas. Berikut adalah perhitungan untuk mendeteksi adanya suatu objek yang bergerak:

$$NilaiBeda = \frac{\sum_{i=a}^{c}\sum_{j=b}^{d}(F_{i,j} - B_{i,j})^2}{N}$$

Keterangan:

F = nilai *pixel* pada *frame*

B = nilai *pixel* pada *background*

N = jumlah *pixel* pada area *detection window*

i = koordinat x pada *detection window*

j = koordinat y pada *detection window*

Perhitungan persamaan diatas merupakan salah satu metode untuk mendapatkan nilai perbedaan dari *pixel* yang disebut dengan metode NSSD (*Normalized Sum Squared Different*). Dengan munculnya suatu nilai perbedaan pada perhitungan diatas, mengindikasikan bahwa terdapat objek yang terdeteksi dalam area deteksi yang telah ditentukan posisinya. Penentuan adanya suatu objek yang bergerak disini setelah terdapat perbedaan nilai yang melebihi nilai *threshold* yang didapat dari sutu perhitungan teretentu.

2.4 *Webcam*

Webcam adalah kamera digital yang dikoneksikan ke komputer, digunakan untuk telekonferensi video atau tujuan lain. *Webcam* dapat menangkap gambar video gerak-penuh, dan beberapa model termasuk mikrofon dan kemampuan *zoom*. *Webcam* yang digunakan dalam fungsi *video* melakukan penangkapan *image* yang dilakukan dalam jumlah tertentu dalam waktu satu detik atau disebut dengan *frame per second* (fps).

BAB III
METODE PENELITIAN

Pada tahap ini dijelaskan mengenai langkah-langkah yang akan dilakukan unutk merancang dan mengimplementasikan perangkat lunak yang akan dibuat.Adapun langkah-langkah yang digunakan dalam penyusunan skripsi ini adalah sebagai berikut:

3.1 Studi Literatur

Dalam penyusunan skripsi ini, pencarian dan pengumpulan data dilakukan dengan melakukan studi literatur dengan sasaran tinjauan antara lain :

1) Buku tinjauan referensi.
2) Referensi Internet

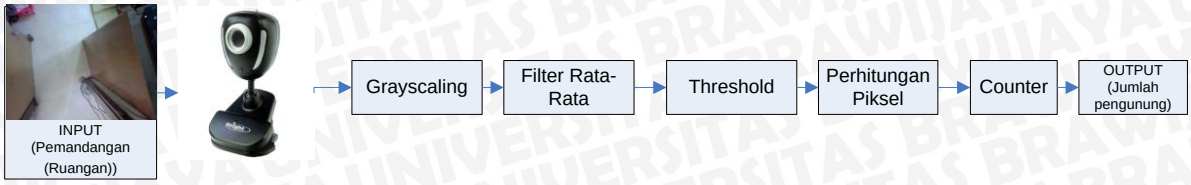
Studi literatur ini dilakukan dengan tujuan untuk mengkaji hal-hal yang berhubungan dengan teori-teori yang mendukung dalam perancangan dan pengaplikasian aplikasi. Adapun teori-teori yang dikaji tersebut adalah sebagai berikut:

1) Teori mengenai *moving detection* pada pencitraan digital.
2) Aplikasi *software* Visual Basic 6.0 yang digunakan untuk membuat aplikasi.

3.2 Diagram Sistem

Diagram sistem merupakan tahap awal dari perancangan suatu sistem agar perancangan sistem berjalan secara sistematis. Perancangan yang sistematis menghasilkan aplikasi yang sistematis pula. Dalam sistem ini, citra masukan adalah *image* dari video yang ditangkap oleh webcam.

Komputer akan memproses *image* yang tertangkap oleh webcam menggunakan teknik pengolahan citra digital. Hasil dari pengolahan citra digital ini kemudian digunakan untuk mengidentifikasi bahwa objek berhasil ditemukan atau tidak dan menampilkannya. Berikut ini merupakan diagram sistem dari aplikasi:

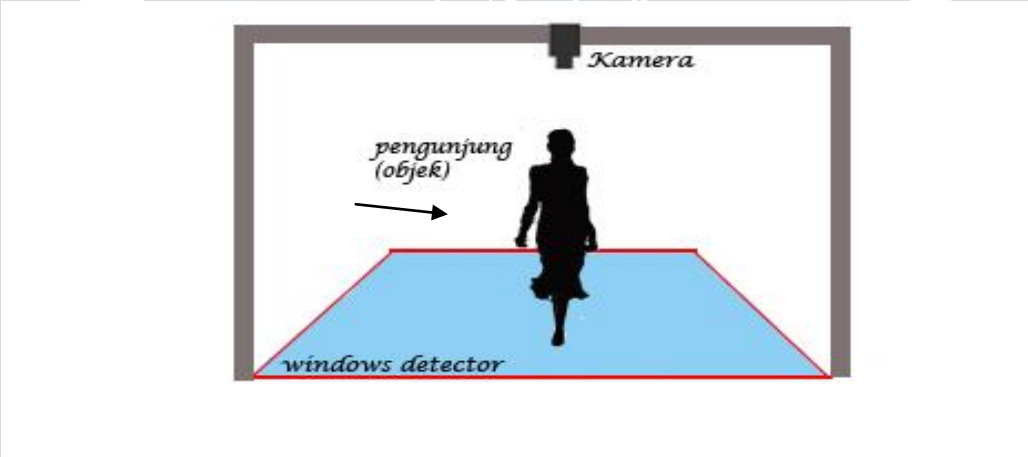


Gambar 3.1 Diagram Sistem

(**Sumber:** Perancangan)

3.3 Cara Kerja Sistem

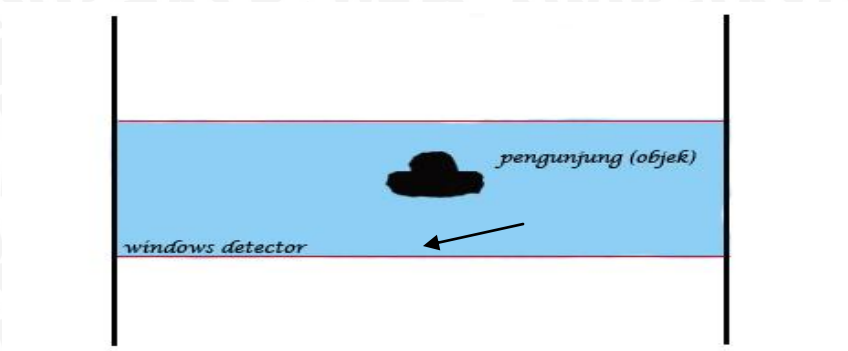
Berikut ini adalah ilustrasi dari pengambilan *image* menggunakan *webcam* tampak dari depan. Posisi *webcam* berada tepat tegak lurus terhadap bidang latar dan posisi *detection window* berada pada bidang latar.



Gambar 3.2 Proses Pengambilan Video dengan Menggunakan webcam tampak depan.

(Sumber: Perancangan**)**

Sedangkan ilustrasi pengambilan gambar video menggunakan webcam tampak dari atas adalah sebagai berikut:



Gambar 3.3 Proses pengambilan *image* dengan menggunakan *webcam* tampak atas

(**Sumber:** perancangan)

Berdasarkan pada **Gambar 3.1**, input pada sistem ini merupakan *image* dari video yang berhasil tertangkap oleh *webcam*. Langkah pertama yang harus dilakukan adalah mengambil *image background* yang akan dipakai sebagai acuan pemisahan objek. Pengambilan *image background* dilakukan dengan cara menangkap *image* dari *webcam* sebelum dilakukan proses selanjutnya. Dengan demikian diharapkan *background* yang didapat mewakili *frame* kosong yang paling sesuai dengan kondisi terakhir sebelum dilakukan perekaman *webcam*. Setelah mendapatkan *image background* yang diinginkan, langkah berikutnya adalah menentukan lingkup atau batasan pendeteksian pada *image background* dan *image frame* atau dapat disebut dengan istilah *detection window*. *Detection window* pada *image background* dan *image frame* diletakkan pada posisi yang sama. Hal ini dilakukan dengan tujuan untuk membatasi daerah yang akan diproses. Setelah menentukan *detection window*, langkah berikutnya adalah mengubah *image frame* dan *image background* yang semula dalam format RGB menjadi *grayscale*. Proses *grayscale* disini merupakan penyederhanaan nilai dari tiap piksel yang nantinya akan menjadi tahap awal dalam menentukan nilai *threshold*. Proses *threshold* (*thresholding*) dilakukan agar setiap *pixel* pada $f(x_n,y_n)$ mempunyai nilai parameter tertentu yang menjadi suatu batasan. Nilai *threshold* ini yang nantinya akan menjadi acuan dalam proses selanjutnya. Dengan menentukan nilai *threshold* yang tepat, dapat dilihat apakah ada objek yang melewati *detection window* atau tidak. Berikutnya adalah proses perhitungan piksel yang nantinya digunakan sebagai pendeteksi adanya suatu objek yang bergerak.

Dari setiap nilai *pixel grayscale* yang didapatkan dari *image background* dan *image frame* melalui proses sebelumnya, hasilnya dikurangkan sehingga terdapat perbedaan atau selisih nilai *pixel*, inilah yang dikenali sebagai keberadaan suatu objek bergerak. Jika saat proses perhitungan nilai selisih tersebut tidak terdapat suatu perbedaan nilai, maka hal

tersebut menandakan bahwa tidak ada objek bergerak yang terdeteksi oleh *webcam*. Untuk menghindari nilai negatif dari operasi tersebut, hasil yang diperoleh dikuadratkan agar semua nilai yang muncul dari proses bernilai positif. Kemudian hasilnya dibagi dengan luasan *detection window*. Perhitungan tersebut dapat dituliskan sebagai berikut :

$$NilaiBeda = \frac{\sum_{i=a}^{c}\sum_{j=b}^{d}(F_{i,j} - B_{i,j})^2}{N}$$

Keterangan:

F = nilai *pixel* pada *frame*

B = nilai *pixel* pada *background*

N = jumlah *pixel* pada area *detection window*

i = koordinat x pada *detection window*

j = koordinat y pada *detection window*

Dalam perhitungan ini, acuan yang digunakan adalah perbedaan atau selisih dari jumlah nilai *pixel* yang terdapat pada *image background* dan pada *image frame*. Hasil perhitungan yang berupa selisih jumlah nilai piksel memungkinkan untuk menghasilkan nilai negatif sehingga hasil selisih tersebut dikuadratkan agar selalu didapatkan nilai yang positif. Nilai negatif dapat terjadi bilamana warna dari obyek pada *image frame* lebih gelap daripada warna *image background*. Sebab warna gelap memiliki nilai yang lebih kecil dari warna terang.

Pada program penghitungan orang yang lewat ini diberi batasan hanya dilakukan di pintu masuk bagi pengunjung manusia, pintu masuk didesain agar pengunjung dapat masuk satu per satu. Demikian pula untuk sumber cahaya pada saat penghitungan dilakukan. Sumber cahaya akan mempengaruhi warna dari objek yang akan dihitung. Untuk itu pengujian penghitungan orang dilakukan pada ruang tertutup dimana sumber cahaya yang digunakan berbeda-beda. Proses penghitungan dilakukan apabila ada objek yang melewati *detection window* yang telah ditentukan.

3.4 Pengujian sistem

Pengujian sistem dilakukan dalam ruangan tertutup , kemudian menggunakan sumber cahaya yang berbeda-beda pada tiap pengujiannya. Kemudian melewatkan suatu

objek (pengunjung) melewati daerah pendeteksi (*detection window*) berulang- ulang sebanyak 4 kali. Selanjutnya menganalisis % *error* yang dihasilkan oleh sistem ini.

3.4.1. Prosentase *error*

Presontase *error* dari sistem ini dapat diketahui dari perbedaan jumlah objek sebenarnya yang melewati pintu dengan jumlah hasil objek yang terdeteksi oleh *webcam*. Nilai prosesntase *error* ini dapat dihitung berdasarkan persamaan berikut:

$$\%\, error = 100\% - \%keberhasilan$$

3.5. Kesimpulan dan Saran

Pada tahap ini, diambil dari hasil pengujian dan analisa terhadap sistem pendeteksi dan penghitung jumlah pengunjung dengan teknik pencitraan digital menggunakan *webcam*. Tahap berikutnya adalah membuat saran untuk perbaikan terhadap penelitian selanjutnya sehinga dapat menyempurnakan kekurangan-kekurangan yang ada serta pengembangan agar diperoleh suatu sistem yang lebih komplek.

BAB IV
PERANCANGAN DAN IMPLEMENTASI

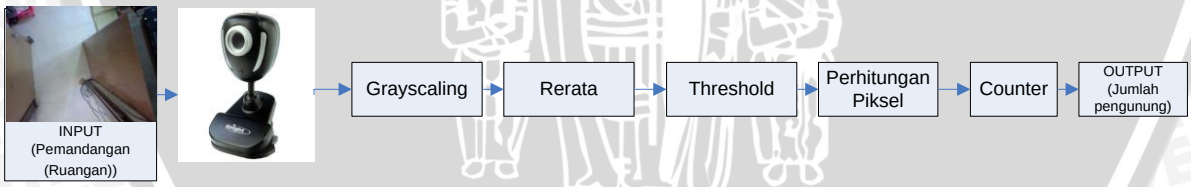
Bab ini membahas peranacangan dan implementasi program pendeteksi dan penghitung jumlah pengunjung menggunakan teknik pencitraan digital menggunakan *webcam.* Pada bab ini juga akan dibahas mengenai proses inisialisasi *webcam*, pengolahan citra , dan penghitungan objek dari hasil pengolahan citra yang bertujuan agar mudah dalam menganalisa program secara keseluruhan.

4.1 Perancangan Secara Umum

Perancangan aplikasi secara umum merupakan tahap awal sebagai acuan dalam perancangan aplikasi yang akan dibuat. Perancangan ini didahului dengan pendefinisian kegiatan pelaku atau *user* dalam menggunakan program pendeteksi dan penghitung jumlah pengunjung dengan pengolahan citra serta perangkat yang digunakan meliputi blok sistem diagram dan cara kerja aplikasi.

4.1.1 Blok Diagram Sistem

Secara garis besar desain yang akan dibuat untuk aplikasi pendeteksi dan penghitung jumlah pengunjung terdiri dari beberapa tahap. Berikut ini adalah blok diagram dari program pendeteksi dan penghitung jumlah pengunjung dengan teknik pencitraan digital menggunakan *webcam*,ditunjukkan pada Gambar 4.1.



Gambar 4.1 Blok diagram sistem

(**Sumber :** Peranacangan)

Fungsi masing-masing bagian dalam blok diagram ini adalah sebagai berikut:

1 Pemandangan (Ruangan) yang akan digunakan sebagai *input* sistem.

2 *Webcam* digunakan untuk menangkap *image* pada video atau disebut *image capture*

3 Dilakukan proses *grayscale* pada *image* yang tertangkap oleh *webcam*

4 Hasil *grayscaling* kemudian direrata dan hasilnya digunakan dalam proses

perhitungan piksel untuk mengetahui apakah ada selisih piksel yang menjadi acuan dalam penentuan adanya pengunjung atau tidak.

5 Jika ada pengunjung maka akan mengaktifkan *counter*.

4.1.2 Cara Kerja Aplikasi

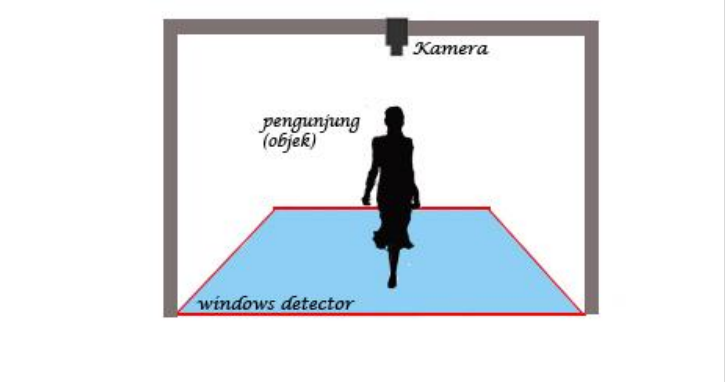
Cara kerja dari program pendeteksi dan penghitung jumlah pengunjung dengan teknik pengolahan citra digita menggunakan *webcam* yaitu dimulai dengan langkah pertama yang harus dilakukan adalah mengambil *image background* yang akan dipakai sebagai acuan pemisahan objek. Pengambilan *image background* dilakukan dengan cara menangkap *image* dari *video* input sebelum dilakukan perekaman. Perlu diperhatikan bahwa posisi *webcam* berada diatas pintu masuk dan tegak lurus 90^o terhadap bidang latar. Dengan demikian diharapkan *image background* yang didapat mewakili *frame* kosong yang paling sesuai dengan kondisi terakhir sebelum dilakukan perekamann *webcam*. Setelah mendapatkan *image background* yang diinginkan, langkah berikutnya adalah menentukan lingkup atau batasan pendeteksian pada *image background* dan *image frame* atau dapat disebut dengan istilah *detection window*. *Detection window* pada *image background* dan *image frame* diletakkan pada posisi yang sama. Hal ini dilakukan dengan tujuan untuk membatasi daerah yang akan diproses. Proses pendeteksian dan perhitungan dilakukan dengan cara mencari perbedaan nilai piksel yang terdapat pada *detection window*. Setelah mendapatkan perbedaan nilai piksel yang terdapat pada *detection window* maka selanjutnya adalah membandingkan antara nilai perbedaan piksel tersebut dengan nilai *threshold* yang didapatkan dari perhitungan nilai rerata dengan nilai pengali. Nilai *threshold* ini selalu berubah ubah sesuai dengan nilai masukan dari jumlah hasil nilai perbedaan piksel dengan jumlah *frame* saat perhitungan. Jika nilai perbedaan piksel pada *detection window* lebih besar dari nilai *threshold* maka menandakan adanya suatu objek yang bergerak melewati *detection window*. *Counter* akan bertambah (aktif) jika pada saat objek(pengunjung) yang melewati *detection window* secara penuh melintasi daerah tersebut, artinya terdapat 2 titik koordinat pada garis *detection window* yaitu titik A(Xa,Ya) pada awal garis dan titik B(Xb,Yb) garis akhir yang nantinya akan menjadi suatu penanda bahwa objek(pengunjung) tersebut benar-benar melewati secara penuh daerh *detection window* tersebut atau tidak sehingga jika melewati secara penuh akan mengaktifkan *counter* .

Dalam program ini, penginisialisasian *webcam* menggunakan library

"avicap32.dll" yang merupakan teknologi *video capture* dalam bentuk API (*Application Programming Interface*) yang dikenal dengan VFW (*Video for Windows*). Teknik pengolahan citra digital ini dilakukan dengan pengurangan nilai piksel pada *detection window* yang telah ditentukan. Keluaran dari program adalah mendeteksi adanya pengunjung yang masuk yang kemudian hasilnya akan menjalankan perhitungan.

4.1.3 Konfigurasi Perangkat Keras

Pengolahan citra ini dapat bekerja dengan baik jika memenuhi parameter-parameter yang sesuai dengan program yang telah ditentukan.Tujuannya adalah agar citra yang telah diolah dapat terdeteksi bahwa ada atau tidak pengunjung yang melewati pntu masuk. Untuk mendapatkan citra yang sesuai dengan keinginan, dibutuhkan konfigurasi *webcam* yang tepat pada tempat sistem diimplementasikan. Gambaran untuk konfigurasi tempat seperti pada gambar 4.2.



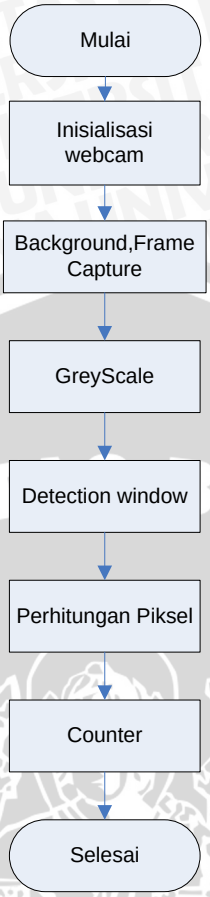
Gambar 4.2 Gambar Perancangan Sistem
(**Sumber :** Perancangan)

4.2 Perancangan Perangkat Lunak

Perancangan perangkat lunak dibangun dengan menggunakan bahasa pemrograman Microsoft Visual Basic 6.0. Sistem ini dirancang dengan dengan spesifikasi sebagai berikut:

1 Mengakses perangkat keras *webcam*.
2 Melakukan *Capture Background* dan *Capture Frame*
3 Menentukan *detection window*
4 Melakukan Proses *grayscale*
5 Melakukan Perhitungan piksel
6 *Counting* jika ada gerakan yang terdeteksi melewati *detection window*

Desain perancangan program secara umum ditunjukkan pada gambar 4.3.

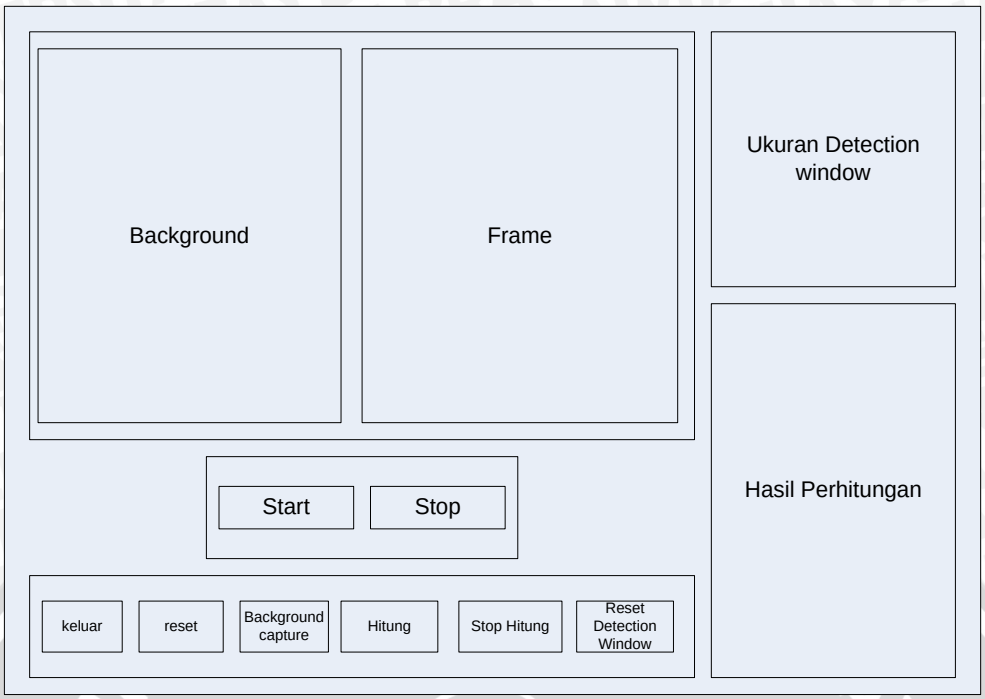


Gambar 4.3 Flowchart perancangan aplikasi

(**Sumber :** Perancangan)

4.2.1 Perancangan Antarmuka (*Interface*)

Program pendeteksi dan penghitung jumlah pengunjung ini dirancang sesederhana mungkin dengan tujuan agar dapat memudahkan saat penggunaaan program sehingga pengguna mudah beradaptasi dan mudah memahami penggunaan program ini. Rancangan tampilan awal *form* program pendeteksi dan penghitung jumlah pengunjung seperti pada gambar 4.4.



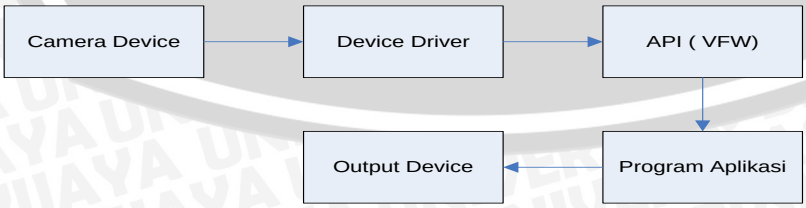
Gambar 4.4 Perancangan Tampilan awal program

(**Sumber :** Perancangan)

4.2.2 Perancangan Inisialisasi Webcam

Perancangan inisialisasi *webcam* dilakukan dengan menmanfaatkan fitur komponen VFW (*Video for Windows*) sebagai antar muka perangkat keras. VFW merupakan teknologi pertama kali yang dikeluarkan Windows untuk teknologi *video capture* dalam bentuk API (*Application Programming Interface*), yang tersimpan dalam library "avicap32.dll". Library ini meginisialisasikan pemanggilan fungsi pengambilan *frame* dan *copy image frame*

Pada perancangan inisialisasi *webcam* ini , VFW menghubungkan rangkaian antara *camera device* hingga ke program aplikasi. Seperti pada gambar 4.5.

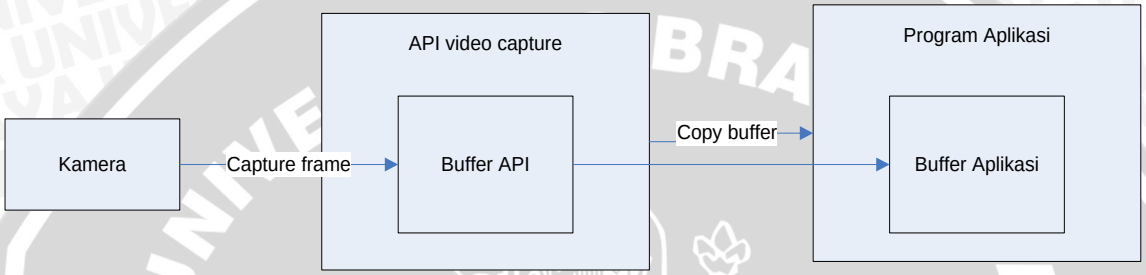


Gambar 4.5 Video capture

(Sumber : Perancangan)

Pemanggilan Fungsi API tersebut melalui beberapa tahapan yaitu tahap pertama

adalah membaca semua driver kamera yang terpasang kemudian memberikan penomoran pada setiap driver yang telah tebaca. Dari tiap driver kamera yang terbaca tersebut kemudian dibaca jenis dari kameranya dan diberi penjelasan bahwa ada driver kamera yang terpasang. Sebelum dilakukan pembacaan *image* hasil tangkapan dari *webcam* , terebih dahulu disediakan *buffer* untuk penyimpanan data *image* sementara. Setiap *video capture* berhasil menangkap satu gambar (satu *frame*), maka data *frame* tersebut akan disimpan dalam suatu *buffer* yang terdapat pada API *video capture*. Setelah itu API akan mengirimkan sinyal *trigger* yang akan mengaktifkan suatu *event* selanjutnya. Berikut adalah blok diagram sistem *buffer* Windows API pada gambar 4.6.

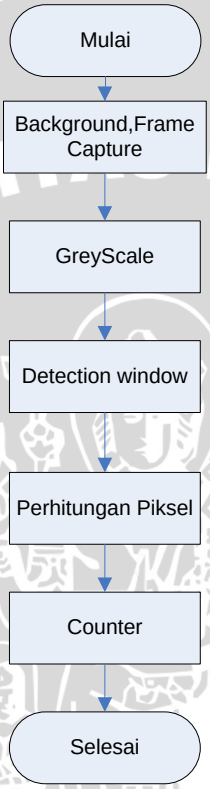


Gambar 4.6 Blok Diagram sistem *buffer* Windows API
(Sumber : Perancangan)

4.2.3 Perancangan Program Pengolahan Digital

Pada perancangan program pengolahan digital terdiri dari beberapa bagian yaitu melakukan *background capture*, menentukan *detection window,* melakukan proses *grayscale* ,melakukan perhitungan piksel, dan menghitung jika terdeteksi bahwa ada gerakan yang melewati *detection window* tersebut. Proses *background capture* ini digunakan untuk menyimpan *image* dari *frame* ketika kamera sedang aktif sebelum proses perekaman dimulai. Selanjutnya menentukan daerah deteksi (*detection window*) yang nantinya akan menjadi daerah untuk perhitungan piksel *frame background* dengan piksel *frame* yang baru ditangkap oleh *webcam*. Jika terdapat perubahan nilai piksel dari perhitungan pengurangan piksel dan nilai dari perubahan piksel tersebut melebihi dari nilai *threshold* maka akan diindikasikan bahwa terdapat pergerakan yang terjadi pada area deteksi yang kemudian akan mengaktifkan *counter*. Sebaliknya jika tidak terdapat perubahan selisih piksel atau nilai dari perbedaan piksel tersebut tidak melebihi (lebih kecil) dari nilai *threshold* maka mengindikasikan bahwa tidak terdapat pergerakan dalam *detection window* dan *counter* tidak aktif. *Counter* akan bertambah (aktif) jika pada saat

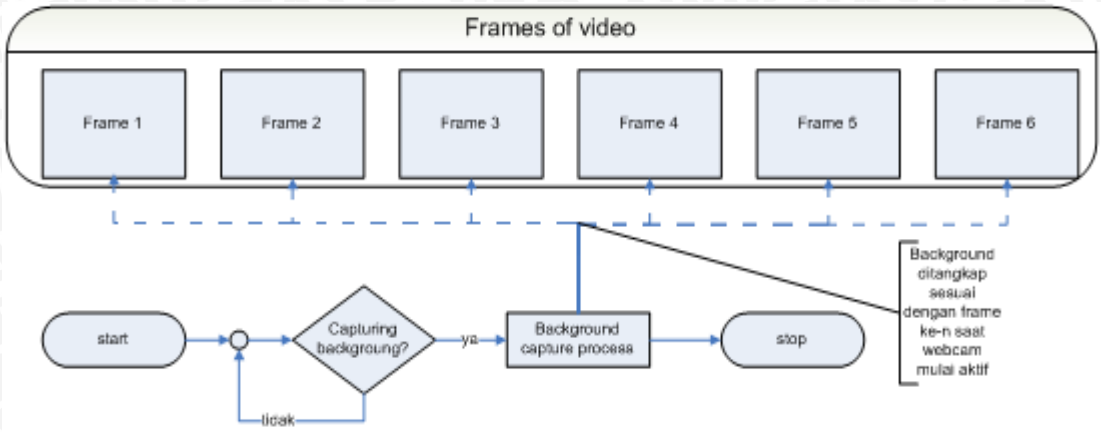
objek(pengunjung) yang melewati *detection window* secara penuh melintasi daerah tersebut, artinya terdapat 2 titik koordinat pada garis *detection window* yaitu titik A(Xa,Ya) pada awal garis dan titik B(Xb,Yb) garis akhir yang nantinya akan menjadi suatu penanda bahwa objek(pengunjung) tersebut benar-benar melewati secara penuh daerah *detection window* tersebut atau tidak sehingga jika melewati secara penuh akan mengaktifkan *counter*. Flowchart untuk perancangan program pengolahan digital ditunjukkan pada gambar 4.7.



Gambar 4.7 Flowchart bagian pengolahan digital
(**Sumber :** Perancangan)

4.2.3.1 Perancangan sub Pengolahan Citra Bagian Background Capture dan Frame Capture

Perancangan bagian *background capture* ini dilakukan saat program berjalan. Ketika *webcam* berhasil dijalankan, maka langkah selanjutnya adalah menangkap sebuah *frame* yang kemudian disimpan dalam sistem dan dijadikan acuan sebagai *background*. Pengambilan *background* ini dapat diilustrasikan seperti pada gambar 4.8.



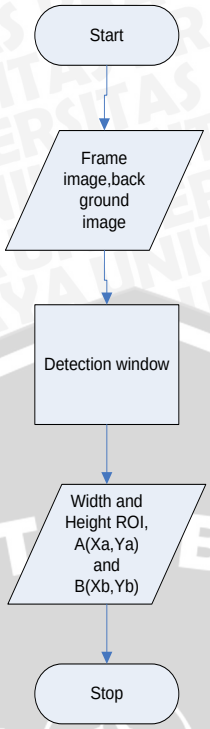
Gambar 4.8 Ilustrasi pengambilan *background*

(**Sumber :** Perancangan)

Pada saat proses pengambilan *background* , jika menjalankan proses tersebut maka *frame* ke-n pada saat kamera aktif itulah yang akan ditangkap dan disimpan yang kemudian akan dijadikan *background* untuk proses selanjutnya. Namun pada saat proses *frame capture* terdapat sedikit perbedaan dengan proses *background capture*, saat proses *frame capture* pengambilan *image* dari *webcam* dilakukan saat program perhitungan dimulai.

4.2.3.2 Perancangan sub Pengolahan Citra Bagian Detection Window

Pada sub pengolahan citra bagian *detection window* , *input* merupakan hasil dari proses *background capture* dan proses *frame capture* yang berupa *image*. *Image* tersebut kemudian ditentukan *detection window* pada daerah tertentu yang ditentukan sendiri oleh user. Penentuan letak atatu posisi dari *detection window* pada *background image* dan *frame image* harus sama agar memudahkan dalam proses perhitungan piksel. Berikut proses penentuan *detection window* pada gambar 4.9.



Gambar 4.9 Proses penentuan *detection window*
(**Sumber :** Perancangan)

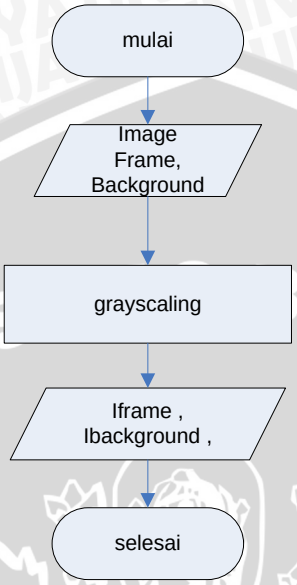
Penjelasan flowchart proses penentuan *detection window*

1. Input adalah *background image* dan *frame image*.
2. Menentukan daerah *detection window* pada *background image* dan *frame image*.
3. Setelah didapatkan letak *detection window* pada kedua *image* tersebut, didapatkan lebar dan tinggi ROI ,nilai dari tiap koordinat dari *detection window* dan juga jumlah piksel.
4. Didapatkan pula 2 titik koordinat yang akan menjadi suatu penanda pada proses perhitungan jumlah pengunjung. dimana titik A dengan koordinat A(Xa,Ya) merupakan koordinat tertentu pada garis awal *detection window*,sedangkan titik B dengan koordinat B(Xb,Yb) merupakan koordinat tertentu pada garis akhir *detection window*.

4.2.3.3 Perancangan sub Pengolahan Digital Bagian Grayscale

Proses *grayscale* dilakukan setelah hasil dari proses *detection window* yang menghasilkan sebuah data / variable berupa nilai dari piksel pada setiap koordinat dalam *detection window* baik pada *background image* maupun pada *frame image*. Dalam proses

ini dilakukan perubahan *image* dari format RGB menjadi *grayscale*. Sehingga hasil dari proses ini adalah *image* dalam format *grayscale* dan nilai piksel pada setiap koordinat didalam *detection window* yang terdapat pada *background image* dan *frame image*. Berikut adalah proses *grayscaling* seperti pada gambar 4.10.



Gambar 4.10 Proses *grayscaling*

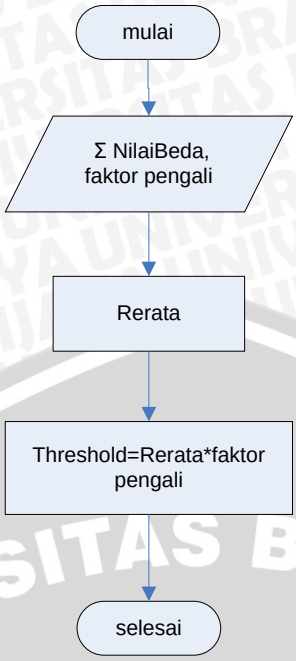
(**Sumber :** Perancangan)

Penjelasan flowchart proses *grayscale*

1. Input dari proses *Image* kemudian dilakukan *grayscaling*.
2. Proses *greyscale* ini kemudian didapatkan nilai piksel dari tiap koordinat juga di dalam *detection window*..

4.2.3.4 Perancangan Sub Pengolahan Digital Bagian Perhitungan Nilai *Threshold*

Pada proses perhitungan pengunjung, sebelum melakukan proses perhitungan hasil perhitungan pengunjung, dilakukan terlebih dahulu perhitungan nilai dari *threshold* yang nantinya akan menjadi batasan dimana akan terjadi perhitungan jumlah obyek pengunjung. Nilai *threshold* disini bersifat adaptif artinya nilainya selalu berubah ubah sesuai dengan niilai masukan pada proses sebelumnya. Untuk mendapatkan nilai *threshold* disini maka diperlukan suatu konstanta yang bernilai tertentu yang disebut dengan faktor pengali. Faktor pengali disini berfungsi sebagai penentu nilai batas ambang nantinya, yang didapatkan dari beberapa pengujian pada kondisi tertentu sehingga didapatkan nilai faktor pengali yang tepat untuk mendapatkan nilai *threshold*. Berikut merupakan perancangan *flowchart* seperti pada gambar 4.11



Gambar 4.11 Proses Perhitungan nilai *threshold*

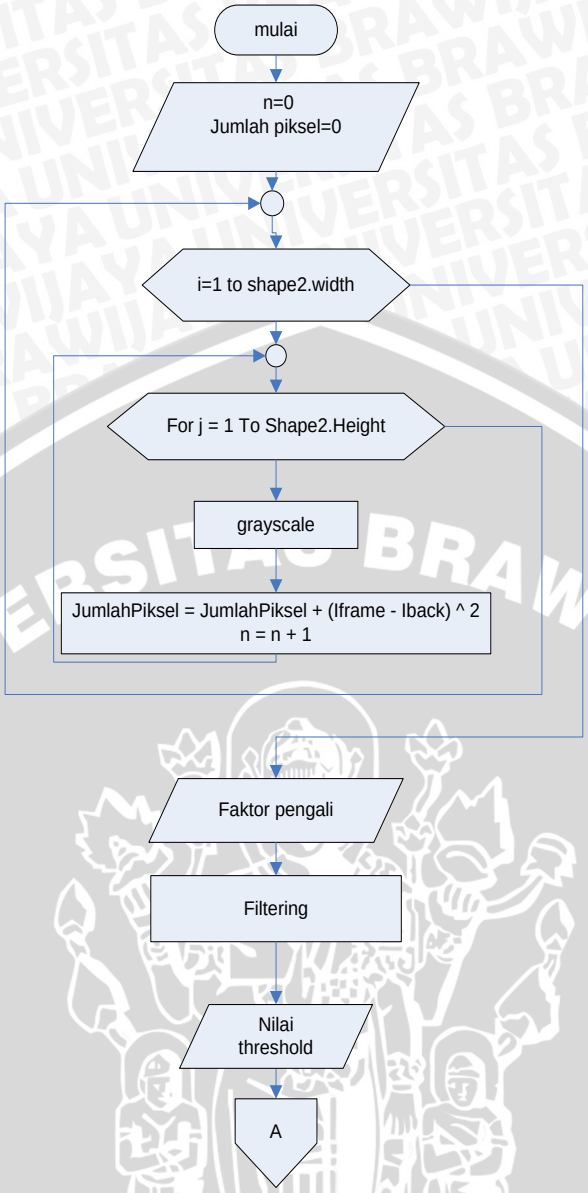
(**Sumber :** Perancangan)

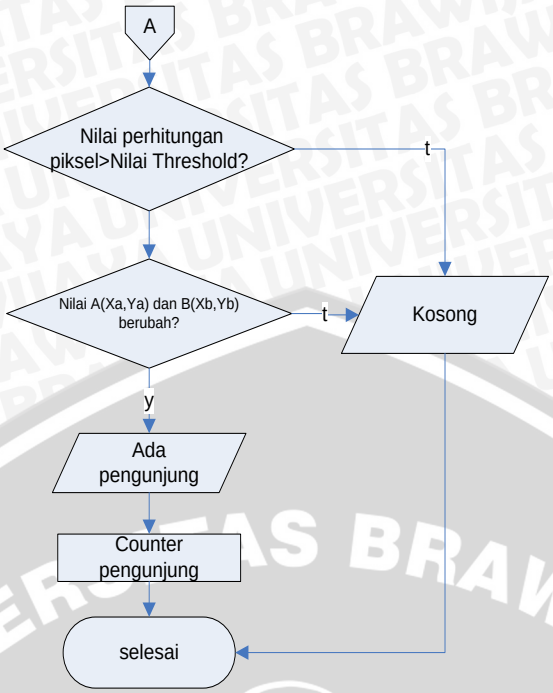
Penjelasan *flowchart* proses penetuan nilai *threshold*

1. Nilai rerata didapatkan dari jumlah perhitungan piksel (Σ NilaiBeda) dibagi jumlah frame yang diproses.
2. Nilai rerata kemudian dikalikan dengan nilai faktor pengali yang kemudian didapatkan nilai *threshold.* Nilai *threshold* ini nantinya digunakan sebagai batas ambang dimana terjadi perhitungan objek atau tidak.

4.2.3.5 Perancangan sub Pengolahan Digital Bagian Perhitungan Piksel

Pada proses penghitungan obyek (pengunjung) memanfaatkan hasil akhir pada proses *grayscale* dan penentuan nilai *threshold*. Hasil dari proses *detection window* dan *grayscaling* kemudian dilakukan porses perhitungan piksel. Untuk mendapatkan hasil perhitungan, dilakukan menggunakan metode yang telah ditentukan sebelumnya. Selengkapnya seperti pada gambar 4.12.


mulai
n=0
Jumlah piksel=0
i=1 to shape2.width
For j = 1 To Shape2.Height
grayscale
JumlahPiksel = JumlahPiksel + (Iframe - Iback) ^ 2
n = n + 1
Faktor pengali
Filtering
Nilai
threshold
A



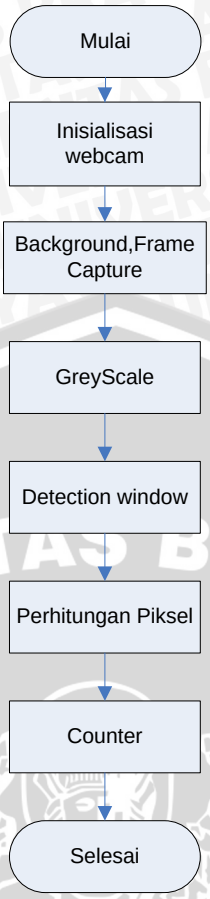
Gambar 4.12 Proses Perhitungan piksel

(**Sumber :** Perancangan)

Setelah mendapatkan nilai perhitungan piksel, kemudian dibandingkan dengan nilai *threshold* apakah lebih besar atau tidak. Jika nilai perhitungan piksel lebih besar daripada nilai *threshold* maka menandakan ada pengunjung, sebaliknya jika nilai perhitungan piksel lebih kecil dari nilai *threshold* maka menandakan tidak ada pengunjung. *Counter* akan bertambah (aktif) jika pada saat objek (pengunjung) yang melewati *detection window* secara penuh melintasi daerah tersebut, artinya terdapat 2 titik koordinat pada garis *detection window* yaitu titik A(Xa,Ya) pada awal garis dan titik B(Xb,Yb) garis akhir yang nantinya akan menjadi suatu penanda bahwa objek (pengunjung) tersebut benar-benar melewati secara penuh daerah *detection window* tersebut atau tidak sehingga jika melewati secara penuh akan mengaktifkan *counter*.

4.2.4 Perancangan Program Keseluruhan

Setelah membuat perancangan pada tiap-tiap bagian program yang meliputi inisialisasi *webcam* dan pengolahan citra digital pada perancangan secara umum, selanjutnya melakukan perancangan secara utuh agar menjadi satu kesatuan program yang dapat dijalankan. Secara keseluruhan ,rangakaian program ditunjukkan seperti pada gambar 4.13.



Gambar 4.13 Flowchart Proses keseluruhan

(**Sumber :** Perancangan)

4.3 Implementasi Sistem

Setelah tahap perancangaan, selanjutnya adalah tahap implementasi. Implementasi ini merupakan proses penulisan hasil perancangan perangkat lunak ke dalam sebuah bahasa pemrograman.

4.3.1 Lingkungan Implementasi

Program ini dibuat dengan menggunakan bahasa pemrograman Microsoft Visual Basic 6.0 . Program diimplementasikan dengan menggunakan spesifikasi sebagai berikut:

1. Perangkat keras

   A. Komputer

   Spesifikasi

   - Prosesor : Intel® Core™ 2 Duo @ 2.80 GHz (2 CPU's)
   - Memory : 3072 MB RAM
   - Graphic Card : NVidia GeForce 9400 GT
   - OS : Microsoft Windows XP Professional

B. *Webcam*

Spesifikasi

1. OKAYA WEBCAM

   a. Resolusi : 320 x 240 piksel

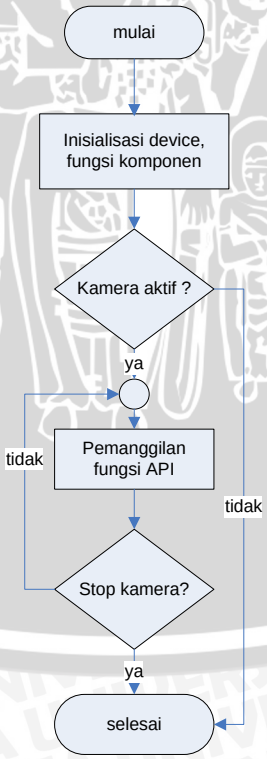
2. Perangkat Lunak

   A. Sistem Operasi : Microsoft Windows XP Professional

   B. Bahasa Pemrograman : Microsoft Visual Basic 6.0

4.3.2 Implementasi Algoritma Bagian Inisialisasi *Webcam*

Inisialisasi *webcam* yang merupakan tahap awal dari proses keseluruhan merupakan suatu proses dimana terjadi pemanggilan fungsi *video capture* menggunakan komponen VFW (*Video for Windows*) yang terdapat pada satu paket *library* "avicap32.dll". Hasil tangkapan dari *video capture* ini merupakan sutau *image* yang berjalan secara *real-time* yang tersimpan sementara dalam suatu *buffer* API yang nantinya pada saat proses *capture*, *image* dari tiap *frame* tersebut disimpan pada *hard disk* perangkat computer dalam format JPG. *Flowchart* proses inisialisasi *webcam* seperti ditunjukkan pada gambar 4.14.



Gambar 4.14 Flowchart Inisialisasi *webcam*

(**Sumber :** Perancangan)

Pada proses inisialisasi *webcam* ini, *event* akan di jalanan setelah sinyal *trigger* dikirimkan oleh fungsi API. Pemanggilan fungsi API ini seperti yang telah dijelaskan pada sub bab 4.2.2 sebelumnya. Dalam bahasa pemrograman Bisual Basic 6.0 dijelaskan sebagai berikut:

```
'inisialisasi fungsi pengiriman sinyal trigger
Public Declare Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal hwnd As Long, ByVal wMsg As Long, ByVal wParam As Long, lParam As Any) As Long

'inisialisasi fungsi pada library "avicap32.dll"
Public Declare Function capCreateCaptureWindow Lib "avicap32.dll" Alias "capCreateCaptureWindowA" (ByVal lpszWindowName As String, ByVal dwStyle As Long, ByVal x As Long, ByVal y As Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hwndParent As Long, ByVal nID As Long) As Long

'inisialisasi variable  dan konstanta
Public mCapHwnd As Long
Public Const CONNECT As Long = 1034
Public Const DISCONNECT As Long = 1035
Public Const GET_FRAME As Long = 1084
Public Const COPY As Long = 1054
Public Const WM_CAP_SET_VIDEOFORMAT = &H400 + 45
Public Const WM_USER = &H400
Public Const WM_CAP_START = WM_USER
Public Const WM_CAP_SET_PREVIEW = WM_CAP_START + 50
```

4.3.3 Implementasi Algoritma Bagian Pengolahan Digital

Program pendeteksi dan penghitung jumlah pengunjung menggukaan teknik pencitraan digital melalui tahap *background capture*, menentukan *detection window*, melakukan proses *grayscale* ,melakukan perhitungan piksel, dan menghitung jika ada gerakan yang melewati *detection window* tersebut. Proses implementasi algoritma bagian pengolahan digital didasarkan pada flowchart gambar 4.7.

5.3.3.1 *Background Capture* dan *Frame Capture*

Proses *background capture* dan *frame capture* merupakan proses awal sebelum proses pengolahan citra digital dilakukan. Proses *background capture* dan *frame capture* sebenarnya menggunakan proses yang sama namun yang membedakan adalah jumlah *image* yang ditangkap oleh *webcam*. Pada proses *background capture*, *image* yang ditangkap hanya satu saja karena digunakan untuk patokan atau referensi pada proses selanjutnya. Sedangkan pada proses *frame capture*, *image* yang ditangkap adalah *image* yang ”berjalan” seiring dengan *frame* yang berhasil ditangkap dengan skala waktu tertentu. Ilustrasi proses *background capture* seperti pada gambar 4.8. Implementasi *background capture* dalam bahasa pemrogramana Visual Basic 6.0 dijelaskan sebagai berikut:

```
' pengiriman sinyal trigger
SendMessage mCapHwnd, GET_FRAME, 0, 0 // mengambil image pada frame yang tertangkap
SendMessage mCapHwnd, COPY, 0, 0  // menyalin image sementara
PictBack.Picture = Clipboard.GetData: Clipboard.Clear

'menghapus semua file berekstensi .jpg
 HapusFileJpg (App.Path & "\*.jpg")
 menyimpan image dengan nama "pic_bg.jpg"
SavePicture PictBack.Picture, App.Path & "\pic_bg.jpg"
'Messagebox informasi
MsgBox "Tentukan Detection Window!", vbOKOnly, "Information"
```

Penjelasan dari *script* diatas adalah:

1. Ketika proses *background capture* aktif, maka akan mengirimkan sinyal *trigger* untuk mengaktifkan fungsi pengambilan *image frame*, kemudian disimpan dalam *buffer API*, kemudian dilakukan pemindahan *image frame* dari *Buffer API* ke *hard disk* .
2. Jika terdapat file berekstensi .jpg pada direktori maka akan dihapus.
3. *Image frame* yang berhasil disimpan pada direkktori (*hard disk*) dengan nama "pic_bg.jpg" lalu menampilkan *messagebox* informasi.

Proses *frame capture* dilakukan hampir sama dengan proses *background capture* namun yang membedakan dalam proses *frame capture* ini adalah penangkapan *image* tiap *frame* yang berkala secara terus menerus sesuai dengan waktu pengambilan *frame* yang telah ditentukan.

```
' mengirimkan sinyal trigger
SendMessage mCapHwnd, GET_FRAME, 0, 0
SendMessage mCapHwnd, COPY, 0, 0
'menyimpan image tiap frame dengan format .jpg
SavePicture PictFrame.Picture, App.Path & "\pic_" & IndexFrame & ".jpg"
IndexFrame = IndexFrame + 1
'menghapus semua file berekstensi .jpg
HapusFileJpg (App.Path & "\*.jpg")
```

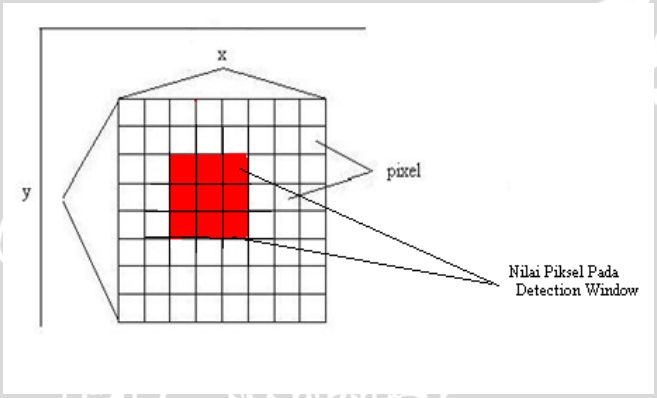
Penjelasan dari *script* diatas sebagai berikut:

1. Ketika proses *frame capture* aktif,maka akan mengirimkan sinya *trigger* untuk mengaktifkan fungsi pengambilan *image frame*,kemudian disimpan dalam *buffer API* kemudian dilakukan pemindahan *image frame* ke *hard disk*.

2. *Image frame* tersebut kemudian disimpan dengan format .jpg dan diberi nomor sesuai dengan index *frame* ke-n yang berhasil disimpan. Index *frame* dimlai dari 0 hingga ke-n
3. Setelah proses tersebut selesai, maka semua file yang berkestensi .jpg dihapus.

4.3.3.2 Proses Detection Window

Pada program bagian *detection window*, hasil pemrosesan pada tahap sebelumnya yaitu proses *background capture* dan *frame capture* yang kemudian didapatkan suatu *image* yang tersimpan pada *hard disk.* Kemudian ditentukan *detection window* dari *image* tersebut. *Detection window* ini merupakan daerah yang digunakan untuk proses perhitungan, yang memiliki koordinat *f(x,y)* tertentu yang telah ditentukan sebelumnya. Sebagai contoh digambarkan pada ilustrasi berikut:



Gambar 4.15 Detection windows pada image
(**Sumber:** perancangan)

Proses *detection window* ini bertujuan untuk mendapatkan koordinat dari *detection window* (ROI) beserta nilai piksel dari tiap koordinat tersebut yang terdappat pada *image.* Implementasi proses *detection window* dalam bahasa pemrograman Visual Basic 6.0 adalah sebagai berikut:

```
Private Sub PictBack_MouseDown(Button As Integer, Shift As Integer, x As Single, y As Single)

    'set nilai awal pada detection window
    Shape1.Height = 0
    Shape1.Width = 0
    Shape1.Top = 0
    Shape1.Left = 0
    Shape2.Height = 0
    Shape2.Width = 0
    Shape2.Top = 0
    Shape2.Left = 0
```

```
'mengambil koordinat  x  dan y pada detection window 1 dan 2
  Shape1.Top = y
  Shape1.Left = x
  Shape2.Top = y
  Shape2.Left = x
'menampilkan nilai x dan y koordinat awal
  TxtPanjang = "(" & x & "," & y & ")"
  'mendapatkan koordinat titik A(Xa,Ya)
  b = y + (y / 2)
  txtx = "(" & x & "," & b & ")"
  End Sub

  Private Sub PictBack_MouseUp(Button As Integer, Shift As Integer, x As Single, y As Single)

  Shape1.Height = y - Shape1.Top
  Shape1.Width = x - Shape1.Left
  Shape2.Height = y - Shape2.Top
  Shape2.Width = x - Shape2.Left
  'menampilkan koordinat x dan y pada koordinat akhir
  TxtLebar = "(" & x & "," & y & ")"
  'mendapatkan kootdinat titik B(Xb,Yb)
  a = y - (y / 2)
  txty = "(" & x & "," & a & ")"
  End Sub
```

Output dari proses ini yang berupa nilai dari tiap koordinat dalam *detection window* yang nantinya digunakan dalam proses selanjutnya. Selain itu dari proses ini didapatkan pula koordinat 2 titik yaitu titik A(Xa,Ya) yang terletak di tengah-tengah sumbu y pada lebar garis awal *detection window* dan titik B(Xb,Yb) yang terletak pada lebar garis akhir *detection window.* Kedua titik ini akan menjadi parameter yang menentukan bahwa *counter* nantinya akan aktif menghitung jumlah pengunjung atau tidak.

4.3.3.3 Proses Grayscale

Jika pada proses *detection window* telah ditentukan *detection window* (ROI) nya, maka langkah selanjutnya adalah proses *grayscale*. Pada proses ini dilakukan pengubahan pada *image*, namun dalam hal ini proses *grayscaling* dilakukan hanya pada nilai piksel yang berada di daerah *detection window* karena proses perhitungan yang akan dilakukan hanya pada daerah yang ditentukan sebelumnya (*detection window*). Proses *grayscale* dalam bahasa pemrograman Visual Basic 6.0 adalah sebagai berikut:

```
JumlahPiksel = 0
n = 0
For i = 1 To Shape2.Width Step 15
  For j = 1 To Shape2.Height Step 15
    warna = PictBack.Point(i + Shape1.Left, j + Shape1.Top)
    r = warna And RGB(255, 0, 0)
```

```
g = Int((warna And RGB(0, 255, 0)) / 256)
b = Int(Int((warna And RGB(0, 0, 255)) / 256) / 256)
'proses konversi  image background  menggunakan grayscaling
Iback = (r + g + b) / 3

warna = PictFrame.Point(i + Shape1.Left, j + Shape1.Top)
r = warna And RGB(255, 0, 0)
g = Int((warna And RGB(0, 255, 0)) / 256)
b = Int(Int((warna And RGB(0, 0, 255)) / 256) / 256)
'proses konversi image frame menggunakan grayscaling
Iframe = (r + g + b) / 3
'PictFrame.PSet (i, j), RGB(Iframe, Iframe, Iframe)

warna = PictBack.Point(x, a)
r = warna And RGB(255, 0, 0)
g = Int((warna And RGB(0, 255, 0)) / 256)
b = Int(Int((warna And RGB(0, 0, 255)) / 256) / 256)
Ititik = (r + g + b) / 3

JumlahPiksel = JumlahPiksel + (Iframe - Iback) ^ 2
' jumlah piksel dalam detection window (ROI)
n = n + 1
Next j
Next i
```

Image dari hasil proses *detection window* kemudian di ubah menjadi format *grayscale*. Setelah dilakukan proses *grayscale* akan menghasilkan nilai piksel dalam format *grayscale* yang kemudian didapatkan jumlah piksel dalam *window detection* yang terdapat pada *image*. Nilai ini yang nantinya akan digunakan dalam proses perhitungan piksel.

4.3.3.4 Penentuan Nilai *Threshold*

Nilai *threshold* merupakan suatu nilai batas ambang dimana akan ditentukan terjadinya suatu perhitungan atau tidak. Pada program ini, nilai *threshold* didapatkan dari proses rerata yang kemudian dikalikan dengan faktor pengali.

Nilai pengali digunakan untuk menetukan nilai *threshold.* Dalam perhitungan nilai pengali sangat berpengaruh karena nilai pengali disini digunakan sebagai penentuan besar nilai batas ambang yang akan menentukan kapan terjadinya perhitungan jumlah pengunjung. Nilai pengali dapat ditentukan secara manual oleh user sehingga nilai ambang bernilai berapa kali dari nilai perhitungan untuk dijadikan batasan. Pada program ini digunakan besar nilai pengali 0.6 berdasarkan sample yang telah dilakukan sebelumnya untuk kondisi tertentu. Pengambilan sample dilakukan sebanyak 2 kali falam tiap pengambilan sample dilakukan 20 kali pengunjung lewat.Hasil dari pengambilan sample terlihat seperti pada table 4.1.

Table 4.1. Sampel nilai pengali

No	Faktor pengali	Jumlah pengunjung sesungguhnya	Jumlah Pengunjung Terhitung	% keberhasilan
1	0,2	40	24	60%
2	0,4	40	35	87,5%
3	0,6	40	38	95%
4	0,9	40	32	80%

Besarnya nilai *threshold* dalam program yang secara *real time* ini tidak selalu konstan, karena dipengaruhi oleh besar nilai masukan yang dihasilkan oleh nilai rerata yang selalu berubah-ubah pula.

```
‘ perhitungan perbedaan piksel
Rumus = JumlahPiksel / n
JumlahFrame = JumlahFrame + 1
JumlahRumus = JumlahRumus + Rumus
Filter = JumlahRumus / JumlahFrame
‘perhitungan nilai threshold
Threshold = Filter * 0.6
```

4.3.3.5 Proses Perhitungan Piksel (*Moving Detection)*

Dalam proses perhitungan piksel ini dilakukan perhitungan antara *background image* dengan *frame image*. Terdapat beberapa sub proses dalam proses perhitungan ini yaitu *filtering* dan *thresholding*. Sub proses *filtering* ini bertujuan untuk mengurangi *noise* pada saat perhitungan dan sub proses *thresholding* menghasilkan nilai *threshold* digunakan sebagai acuan perubahan atau selisih nilai piksel. Sesuai dengan persamaan perhitungan filter rata rata yang kita sebut dengan perhitugan rerata, jumlah perhitungan dibandingkan dengan jumlah *frame* seperti berikut :

Filter(n) = (Jumlah perhitungan piksel(1) + Jumlah perhitungan piksel (2) + Jumlah perhitungan piksel (3) + Jumlah perhitungan piksel (4) + Jumlah perhitungan piksel (n)) / n

Dimana n adalah jumlah *frame* yang tertangkap *webcam* dan yang terlibat dalam proses perhitungan nilai *thresholding* didapatkan dari perkalian nilai rerata dengan factor pengali.

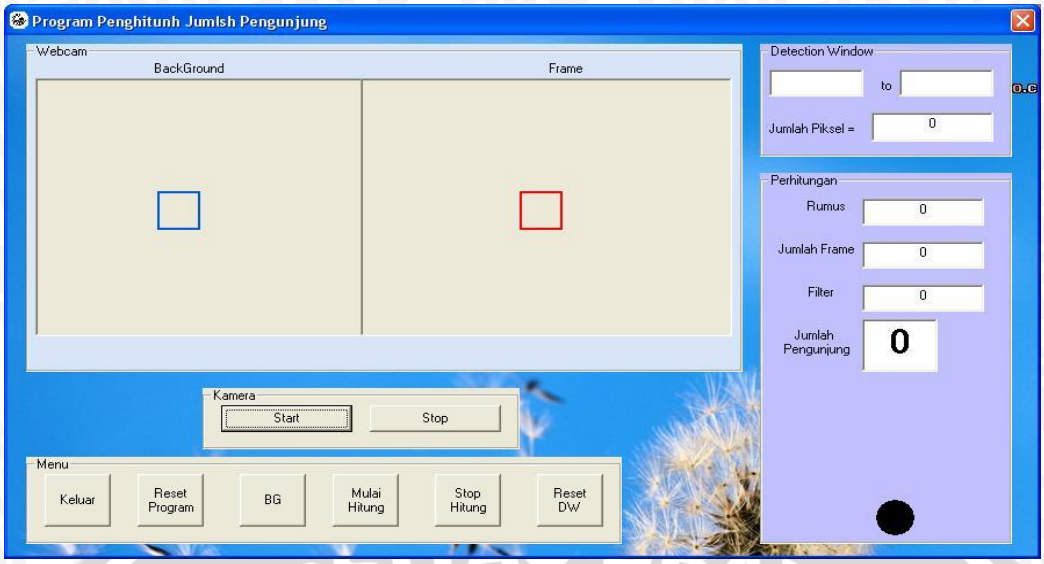
Besarnya factor pengali ini sangat berpengaruh dalam proses perhitungan piksel,oleh karena itu digunakan nilai pengali yang sesuai dengan karakteristik yang akan digunakan. Penulisan dalam bahasa pemrograman Visual Basic 6.0 seperti berikut:

```
‘ perhitungan perbedaan piksel
Rumus = JumlahPiksel / n

JumlahFrame = JumlahFrame + 1
JumlahRumus = JumlahRumus + Rumus
Filter = JumlahRumus / JumlahFrame
Threshold = Filter * 0.6

‘mendeteksi adanya pengunjung
If Rumus > Threshold And (Iframe <> Ititik) Then
 Status = "Ada Pengunjung"
 Shape3.FillColor = RGB(255, 0, 0)

 If Temp <> Status Then
  Beep
   JumlahLewat = JumlahLewat + 1
    Temp = Status
  End If
Else

 Status = "Kosong"
 Shape3.FillColor = RGB(0, 0, 0)
 Temp = Status
End If
```

Proses penghitungan ini berdasarkan pada metode perhitungan selisih piksel. Jika nilai perhitungan hasilnya lebih besar dari nilai *threshold* maka diindikasikan ada pengunjung bergerak melewati *detection window* yang kemudian mengaktifkan *counter*.

4.4 Implementasi Antarmuka (Interface)

Tampilan *form* utama program pendeteksi dan penghitung jumah pengunjung ketika pertama kali dijalankan ditunjukkan pada Gambar 4.15.



Gambar 4.16 Tampilan Awal Program

(**Sumber :** Pengujian)

Berikut akan dijelaskan mengenai komponen komponen yang digunakan dalam program pendeteksi dan penghitung jumlah pengunjung sesuai implementasi antarmuka yang ditunjukkan pada Gambar 4.16. Komponen yang digunakan meliputi picture box, command button, frame, label , shape, combo box, serta text box., untuk lebih jelasnya ditunjukkan pada table 4.2

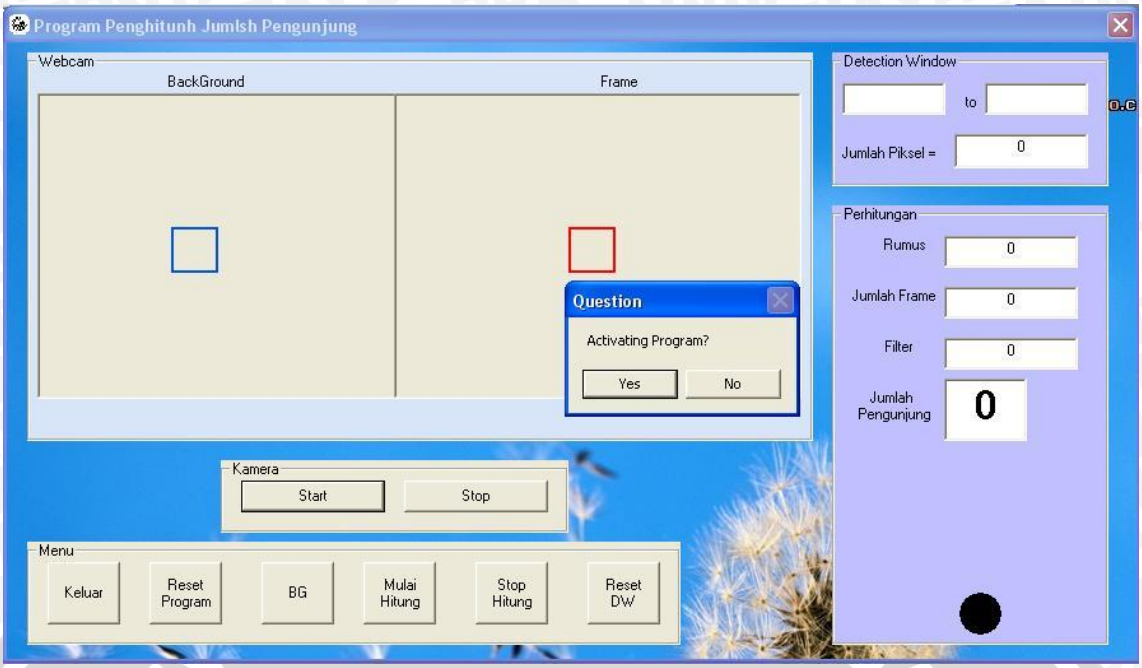
Table 4.2. Tools yang digunakan pada form

Menu	Jenis Komponen	Keterangan
Start	Command Button	Digunakan untuk mengaktifkan program sekaligus mengaktifkan *webcam*.
Stop	Command Button	Dugunakanan untuk mengakhiri program sekaligus mematikan fungsi *webcam*
Keluar	Command Button	Keluar dari program
Reset Program	Command Button	Mengembalikan pada kondisi awal program dijalankan
BG	Command Button	Digunakan untuk menyimpan *background image*

Mulai Hitung	Command Button	Digunakan untuk memulai proses perhitungan
Stop Hitung	Command Button	Digunakan untuk menghentikan perhitungan
Reset DW	Command Button	Mengatur ulang pengaturan *detection window*
Detection Window	Text box	Menampilkan koordinat awal dan akhir untuk *detection window*
Rumus	Text box	Menampilkan hasil perhitungan
Jumlah Piksel	Text box	Menampilkan jumlah piksel pada daerah *detection window*
Jumlah Frame	Text box	Menampilkan frame yang berhasil ditangkap oleh *webcam*
Filter	Text box	Menampilkan nilai Filter
Jumlah Pengunjung	Text box	Menampilkan banyaknya pengunjung yang melewati *detection window*
Status	Label	Menampilkan status sebagai Indikator pendeteksi
Shape	Shape	Indicator pendeteksi

4.4.1 Implementasi Antarmuka (*Interface*) Saat Akan Dijalankan

Saat program pertama kali akan dijalankan, maka akan memunculkan suatu pilihan untuk memulai program atau tidak. Jika ya maka akan mengaktifkan program dan siap untuk dignakan selanjutnya Tampilan awal program yang akan dijalankan adalah seperti terlihat pada gambar 4.17.

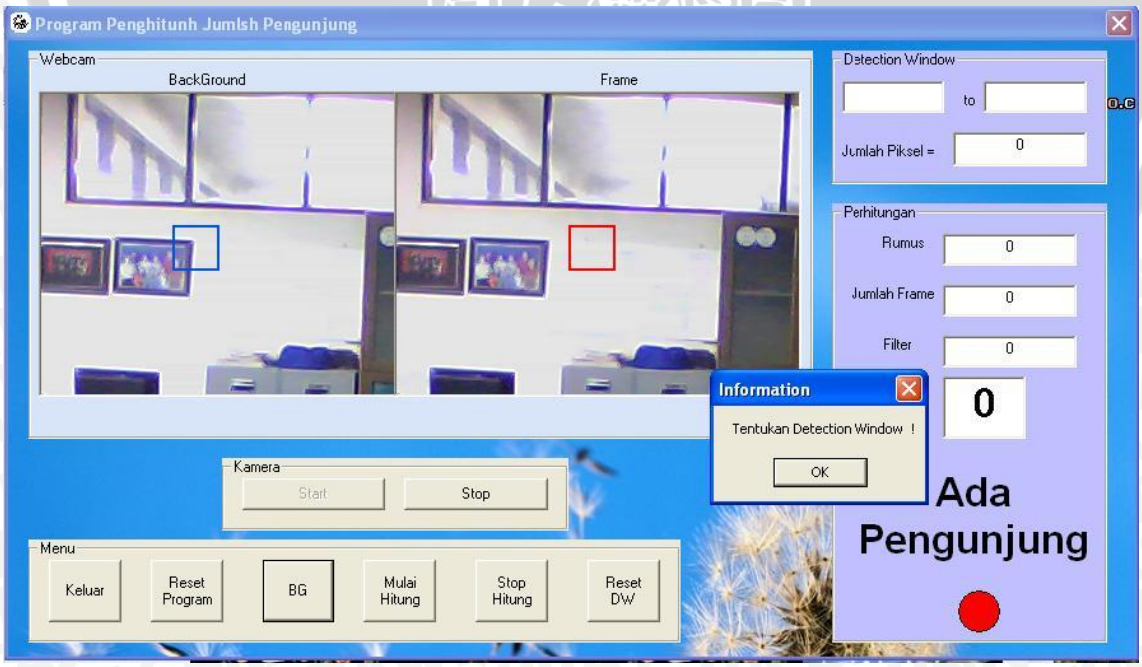


Gambar 4.17 Tampilan program ketika akan dijalankan

(**Sumber :** Pengujian)

4.4.2 Implementasi Antarmuka (interface) saat Pengambilan background

Proses pengambilan *background image* menyimpan *frame* yang berhasil ditangkap *webcam* pada saat tombol Background Capture di aktifkan atau ditekan. Tampilan program seperti ditunjukkan gambar 4.18.



Gambar 4.18 Tampilan saat Pengambilan *background*

(**Sumber :** Pengujian)

BAB V
PENGUJIAN

Untuk mengetahui program sesuai dengan perangacangan atau tidak dan dapat bekerja denga baik, maka perlu dilakukan pengujian pada program. Pengujian dalam bab ini adalah sebagai berikut:

1. Pengujian *device* atau *webcam*
2. Pengujian program pendeteksi dan penghitung jumlah pengunjung.
3. Analisis kegagalan
4. Kesimpulan hasil pengujian

5.1 Pengujian webcam

Pengujian *webcam* disini bertujuan untuk mengetahui apakah *webcam* yang tersambung pada perangkat computer tersebut telah berfungsi atau tidak dan dapat bekerja sesuai yang diharapkan. Dalam pengujian disini menggunakan *webcam* OKAYA.

5.1.1 Pengujian Koneksi *device* atau *Webcam*

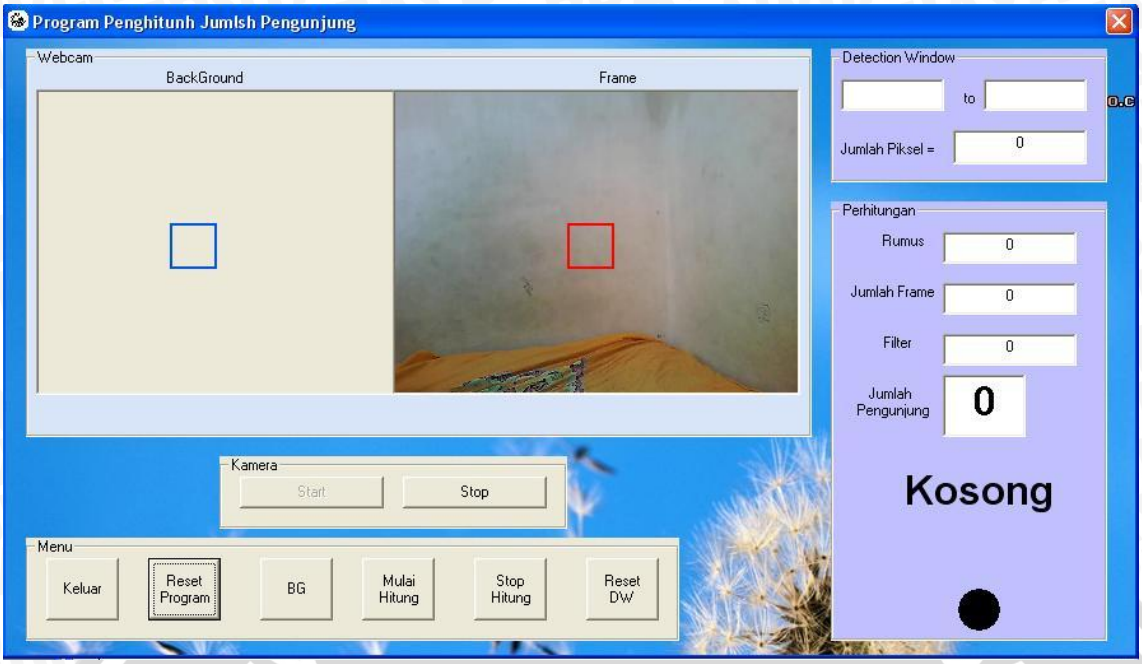
Pengujian ini bertujuan untuk mengetahui apakah *webcam* dapat terkoneksi dengan laptop yang menandakan bahwa komponen VFW (*Video for Window*) dapat berkerja dengan baik. Pengujian koneksi *webcam* dilakukan secara langsung ketika program dijalankan. Hasil pengujian ditunjukkan pada table 5.1

Tabel 5.1 Data Hasil Pengujian

No	*Device*	Hasil
1	Okaya *webcam*	Berhasil Terhubung

(Sumber : Pengujian Aplikasi)

Table diatas adalah hasil pengujian koneksi *webcam* terhadap laptop. Dari hasil pengujian didapatkan hasil bahwa komponen VFW (*Viedeo for Window*) bekerja dengan baik. Sehingga *webcam* dapat terhubung.



Gambar 5.1 Pengujian koneksi *webcam*

(**Sumber :** Pengujian)

5.2 Pengujian Program Pendeteksi dan Penghitung Jumlah Pengunjung

Pengujian ini bertujuan untuk mengetahui kenerja dari program terhadap beberapa bentuk kondisi pengujian yang berbeda-beda sehingga didapatkan program aplikasi yang sesuai dengan kebutuhan.

5.2.1 Pengujian Program Pendeteksi dan Penghitung Jumlah Pengunjung Dengan Kondisi Cahaya yang Berbeda-beda

Program pendeteksi dan penghitung jumlah pengunjung ini dilakukan pengujian dengan sumber cahaya yang berbeda-beda yaitu dalam kondisi cahaya yang agak gelap, normal, dan terang dengan tujuan agar didapatkan porsi cahaya yang sesuai dengan kemampuan kinerja program.

5.2.1.1 Pengujian Program Pendeteksi dan Penghitung Jumlah Pengunjung Dengan Kondisi Cahaya Agak Gelap

Pengujian program pendeteksi dan penghitung jumlah pengunjung dilakukan pada kondisi cahaya yang agak gelap dengan tujuan agar mengetahui respon program terhadap kondisi tersebut. Pengujian ini dilakukan dengan menggunakan sumber cahaya Lampu Ekonomat 12 W .Hasil pengujian program seperti pada table 5.2

Tabel 5.2 Hasil pengujian dalam kondisi cahaya agak gelap (Lampu Ekonomat 12 W)

No	Jumlah sebenarnya	Jumlah terhitung	% keberhasilan
1	20	20	100%
2	20	17	85%
3	20	20	100%
4	20	20	100%
Rata-rata			96%

(**Sumber :** Pengujian)

5.2.1.2 Pengujian Program Pendeteksi dan Penghitung Jumlah Pengunjung Dengan Kondisi Cahaya Normal

Pengujian program pendeteksi dan penghitung jumlah pengunjugn ini dilakukan pada kondisi cahaya yang normal dengan tujuan agar mengetahui respon program terhadap kondisi tersebut. Pengujian ini dilakukan dengan menggunakan sumber cahaya Lampu Philips ESSENTIAL 14 W. Hasil pengunjian program seperti pada table 5.3

Table 5.3 Hasil pengujian kondisi cahaya terang (Lampu Philips ESSENTIAL 14 W)

No	Jumlah sebenarnya	Jumlah terhitung	% keberhasilan
1	20	20	100%
2	20	20	100%
3	20	20	100%
4	20	20	100%
Rata-rata			100%

(**Sumber :** Pengujian)

5.2.1.3 Pengujian Program Pendeteksi dan Penghitung Jumlah Pengunjung Dengan Kondisi Cahaya Terang

Pengujian program pendeteksi dan penghitung jumlah pengunjung dilakukan pada kondisi cahaya yang terang dengan tujuan agar megetahui respon program terhadap kondisi

tersebut. Pengujian ini dilakukan dengan menggunakan sumber cahaya Lampu Philips Tornado 24 W. Hasil pengujian program seperti pada table 5.4

Table 5.4 Hasil pengujian kondisi cahaya terang (Lampu Philips Tornado 24 W)

No	Jumlah sebenarnya	Jumlah terhitung	% keberhasilan
1	20	21	95%
2	20	20	100%
3	20	22	90%
4	20	20	100%
Rata-rata			96,25%

(**Sumber :** Pengujian)

5.2.2 Pengujian Program Pendeteksi dan Penghitung Jumlah Pengunjung Dengan Kondisi Kecepatan Jalan Normal

Pengujian program pendeteksi dan penghitung jumlah pengunjung dilakukan dengan kondisi dimana kecepatan pengunjung normal (1 m/s). Tujuan dari pengujian dalam kondisi ini adalah didapatkan respon dari program terhadap kondisi pengunjung yang melewati memeiliki kecepatan 1 m/s.

5.2.2.1 Pengujian Program Pendeteksi dan Penghitung Jumlah Pengunjung Dengan Kecepatan Normal (1 m/s)

Pengujian program pendeteksi dan penghitung jumlah pengunjung dilakukan dengan kecepatan normal (1 m/s) bertujuan agar dapat mengetahui respon program terhadap kondisi tersebut. Hasil pengujian program seperti pada table 5.5

Tabel 5.5 Hasil pengujian dengan kecepatan normal (1 m/s)

No	Jumlah sebenarnya	Jumlah terhitung	% keberhasilan

1	20	20	100%
2	20	19	95%
3	20	20	100%
4	20	20	100%
Rata-rata			98%

(**Sumber :** Pengujian)

5.2.2.2 Pengujian Nilai *Threshold* dan perhitungan piksel

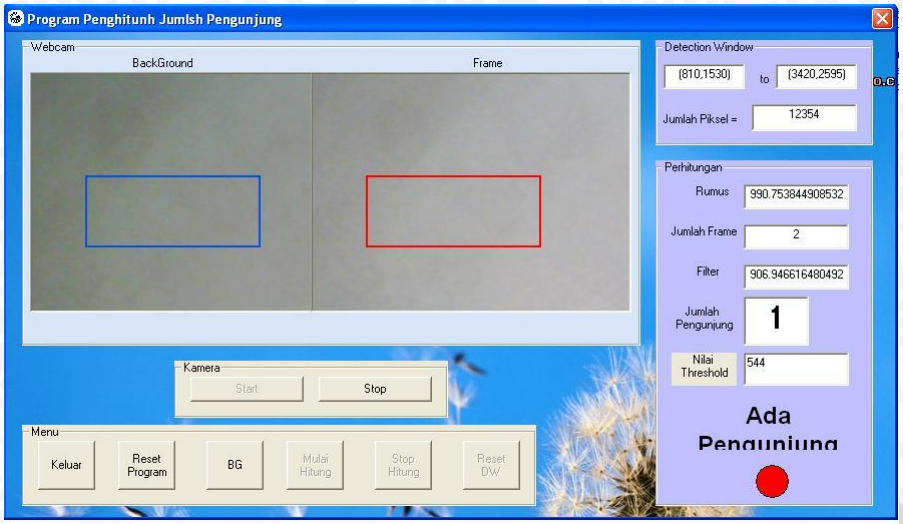
Pengujian nilai *threshold* dilakukan dengan tujuan mengetahui nilai ambang yang dihasilkan dari proses perhitungan. Hasil pengujian program seperti pada table 5.6

Tabel 5.6 Hasil pengujian nilai *threshold*

No	Nilai Filter rata-rata	Faktor pengali	Nilai Threshold	Nilai perhitungan piksel (rumus)
1	906	0.6	544	990
2	796	0.6	478	629
3	26	0.6	16	27
4	2074	0.6	1245	3072
5	603	0.6	362	80

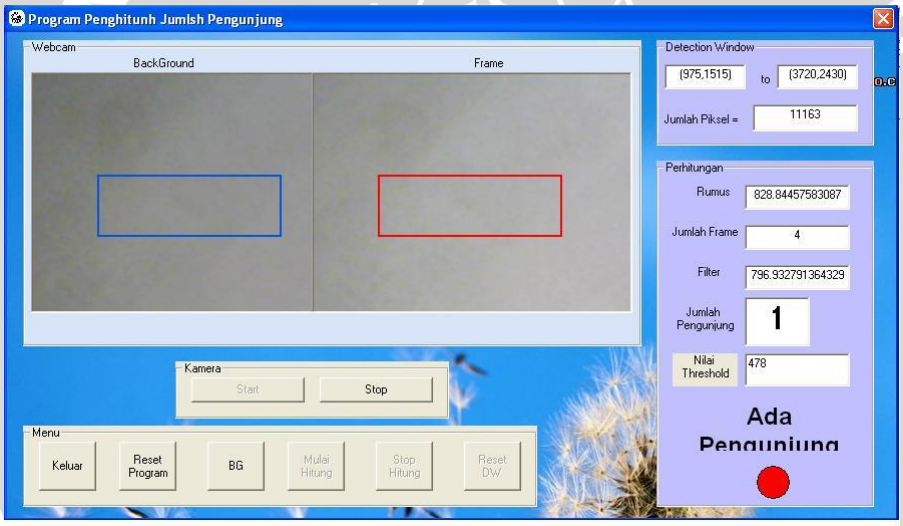
(**Sumber :** Pengujian)

Berikut merupakan hasil pengujian dalam bentuk *screen shoot*. Pada gambar berikut menunjukkan bahwa nilai *threshold* yang didapat selau berubah ubah dikarenakan nilai rerata yang berubah-ubah pula.



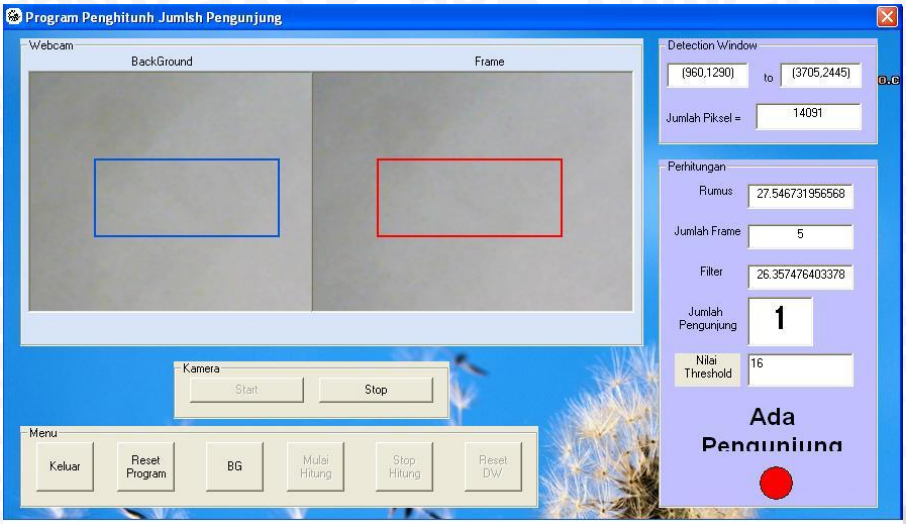
Gambar 5.2 Pengujian nilai *threshold* 1

(**Sumber :** Pengujian)



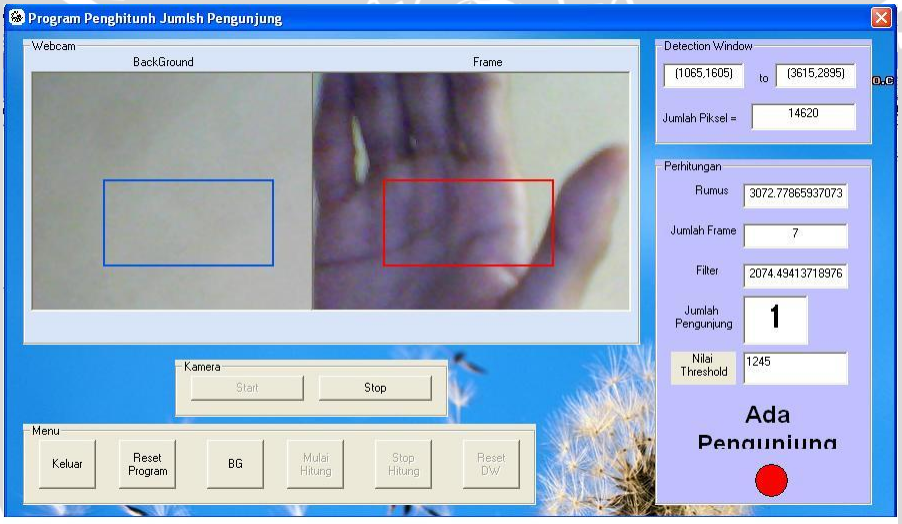
Gambar 5.3 Pengujian nilai *threshold* 2

(**Sumber :** Pengujian)



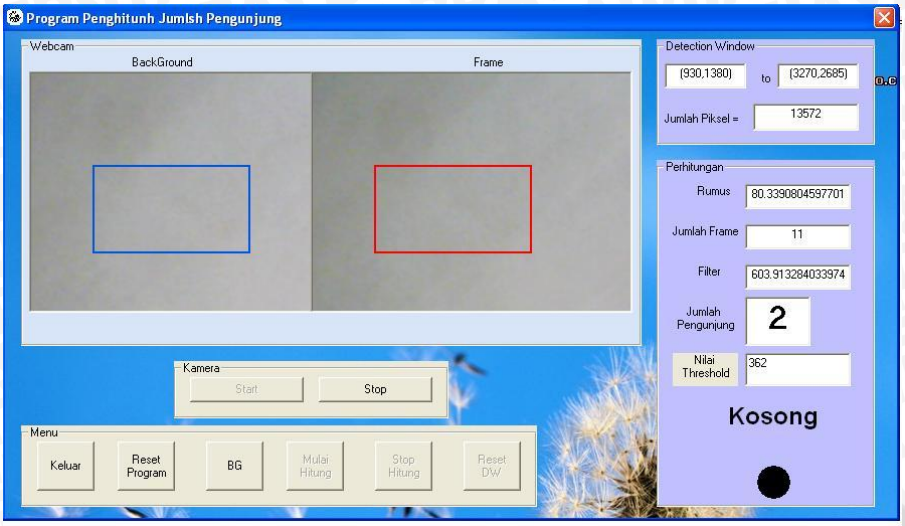
Gambar 5.4 Pengujian nilai *threshold* 3

(**Sumber :** Pengujian)



Gambar 5.5 Pengujian nilai *threshold* 4

(**Sumber :** Pengujian)



Gambar 5.6 Pengujian nilai *threshold* 5

(**Sumber :** Pengujian)

5.3 Pengujian Keseluruhan

Pengujian dilakukan secara keseluruhan bertujuan untuk mengetahui respon program jika dilakukan dalam kondisi cahay agak gelap, normal, terang, kecepatan pengujung normal (1m/s) berikut hasil dari pengujian keseluruhan pada tabel 5.7

Tabel 5.7 Hasil pengujian keseluruhan

Pengujian	Jumlah sebenarnya	Jumlah terhitung	% keberhasilan
Cahaya agak gelap (lampu ekonomat 12W)	80	73	96%
Cahaya normal (lampu Philips Essential 14W)	80	80	100%
Cahaya terang (lampu Philips Tornado 24W)	80	83	96,25%
Kecepatan pengunjung normal (1m/s)	80	79	98%
Rata-rata			97,5%

(**Sumber :** Pengujian)

Berdasarkan tabel 5.7, hasil pengujian keseluruhan program pendeteksi dan penghitung jumlah pengunjung menunjukkan prosentasi keberhasilan penhitungan sebesar 97, 5%. Prosentasi error yang dihasilkan dari program ini sebesar:

% error = 100% - prosentase keberhasilan

= 100% - 97, 5%

= 2, 5%

Dari gambar pengujian nilai *treshold* 1 sampai gambar pengujian nilai threshold 5 dapat dibuktikan bahwa *counter* jumlah pengunjung dilakukan atau terjadi saat nilai perhitungan piksel (nilaibeda) lebih besar daripada nilai *threshold*. Sedangkan pada gambar pengujian nilai *threshold* 6 dapat diketahui bahwa nilai perhitungan piksel (nilaibeda) lebih kecil daripada nilai *threshold* yang dihasilkan ,maka *counter* tidak terjadi.

5.4 Analisis Kegagalan

Saat pengujian dilaksanakan ,program pendeteksi dan penghitung jumlah pengunjung tidak menghasilkan data keluaran yang sama dengan data masukan . Hal ini terjadi dikarenakan faktor-faktor diluar sistem maupun didalam sistem (lingkungan maupun perangkat).

5.2.1 Analisis Kegagalan Program Pendeteksi dan Penghitung Jumlah Pengunjung

Pada saat program dijalankan terdapat beberapa hal yang menyebabkan data hasil keluaran tidak sama dengan data masukan program. Hal ini disebabkan oleh beberapa faktor diantaranya:

1. Intensitas cahaya yang berubah-ubah ketika terdapat pengunjung melewati pintu masuk menyebabkan nilai hasil perhitungan berubah-ubah pula. Seperti contoh, nilai *threshold* telah didapatkan berdasarkan hasil perhitungan saat intensitas cahaya *background* lebih kecil dari intensitas cahaya *frame.* Namun saat program dijalankan, intensitas cahaya pada *image background* lebih besar daripada intensitas cahaya pada *image frame*, menyebabkan nilai yang dihasilkan dari perhitungan yang berbeda. Sehingga nilai perhitungan melebihi nilai *threshold* yang telah ditentukan tadi,menyebabkan counter aktif walaupun masih belum ada objek begerak yang melewati *detection window.*
2. Kecepatan pengunjung ketika melewati *detection window* yang lebih dari 1 m/s menyebabkan *webcam* tidak berhasil melakukan penangkapan *image.* Sehingga saat pengunjung melewati *detection window* kemungkinan sagat

kecil untuk berhasil ditangkap oleh *webcam.* Hal ini menyebabkan program tidak mendeteksi adanya perubahan piksel yang terjadi pada *detection window* walaupun pada kenyataannya ada pengunjung yang melewati daerah tersebut.

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, pengujian ,dan analisis sistem maka dapat diambil kesimpulan sebagai berikut:

1. Program ini berhasil mendeteksi dan mendapatkan data jumlah pengunjung yang melewati *detection window* dengan tingkat kesalahan (*error*) sebesar 2,5%.
2. Tingkat keberhasilan penghitungan dan pendeteksian pada program ini dipengaruhi oleh kecepatan pengunjung melewati *detection window*. Jika kecepatan pengunjung yang melewati *detection window* lebih dari 1 m/s maka besar kemungkinan pergerakan tidak terdeteksi oleh *webcam*.

6.2 Saran

Saran yang dapat diberikan untuk pengembangan tugas akhir ini adalah:

1. Pendeteksian dan penghitungan dilakukan terhadap semua objek yang melewati *detection window* tanpa membedakan jenis objeknya sehingga perlu dikembangkan metode untuk menentukan apakah obyek yang masuk adalah manusia atau bukan.
2. Penghitungan jumlah pengunjung hanya dilakukan pada pintu arah masuk saja dan masih memerlukan suatu metode untuk mendeteksi dua arah yaitu arah masuk dan arah keluar dalam satu proses.

DAFTAR PUSTAKA

Pardosi,Mico,2005,"Visual Basic 6.0",Selaras,Surabaya.

Sigit, Riyanto., Basuki, Achmad., Ramadijanti, Nana., Pramadihanto, Dadet.,. 2005. Step by Step Pengolahan Citra Digital. Yogyakarta: Andi

Kuncara,Chandra,2004,"Perancangan dan Pembuatan Program Penghitung Jumlah Orang Menggunakan Webcam.",URL: http://dewey.petra.ac.id/jiunkpe_dg_4414.html (diakses 14 Desember 2011)

Jae-Won, Kim, Choi,Kang-Sun, Choi, Byeong-Doo, Ko,Sung-Jea. 2010. *Real-time Vision-based People Counting System for the Security Door***.** PDF.

Kurnia,Dede,2011," Pengukuran Kepadatan Arus Lalu Lintas Menggunakan Sensor Kamera ", URL: http://dhetiastapurwatna.wordpress.com/2009/05/19/sistem-lalu-lintas/. (diakses tanggal 16 Desember 2011)

Suntoyo,Andi,2011," Pengambilan Gambar dari *Web Camera* untuk Membangun Sistem Informasi",URL : http://www.andisun.com/jurnal/pengambilan-gambar-dari-web-camera-untuk-membangun-sistem-informasi-2. (diakses tanggal 18 Desember 2011)

Emysil,John.,Paler,L. "Webcam Using VB". http://planetcodes.blogspot.com/2010/04/webcam-using-vb60.html (diakses 18 Desember 2011)

RobDog888 . Screen Resolution . http://www.vbforums.com/showthread.php?t=360570 (diakses tanggal 23 Januari 2012)

Anonimous. "Visual Basic 6.0: Pengenalan Form" http://wartawarga.gunadarma.ac.id/2011/04/visual-basic-6-0-pengenalan-form/

(Diakses pada tanggal 23 Januari 2012)

Idhawati.2011," Pengolahan Citra.PDF"