BAB IV

ANALISIS DAN PERANCANGAN SISTEM

Pada bab ini akan dijelaskan mengenai analisis kebutuhan sistem yang ingin dikembangkan, kemudian dilanjutkan dengan perancangan sistem yang memenuhi hal – hal yang telah dianalisis sebelumnya untuk diwujudkan menjadi sebuah sistem yang dapat digunakan dengan baik.

4.1 ANALISIS KEBUTUHAN

Bagian analisis ini tersusun atas berbagai permasalahan yang melatarbelakangi pengembangan sistem ini beserta dengan kebutuhan yang harus dapat diatasi oleh sistem ini. Setelah itu akan dijelaskan juga mengenai gambaran umum dari sistem yang akan dikembangkan.

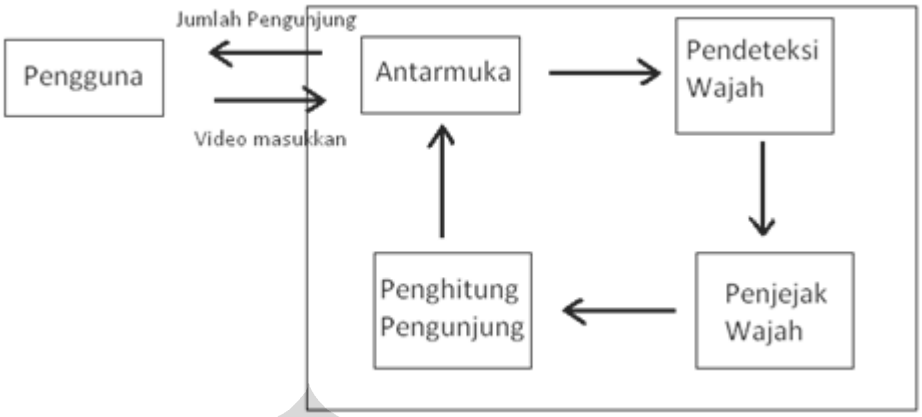
Sesuai dengan yang telah disampaikan pada bagian latar belakang dari laporan tugas akhir ini, para pengusaha tempat umum membutuhkan data jumlah orang yang mengunjungi tempat yang dikelolanya. Penghitungan jumlah pengunjung juga diharapkan dapat dilakukan secara otomatis, karena akan dilakukan dalam jangka waktu yang lama. Oleh karena itu, dibutuhkan sebuah sistem yang dapat melakukan proses penghitungan pengunjung secara otomatis. Dalam melakukan proses penghitungan pengunjung, deteksi wajah merupakan fitur yang berperan penting, karena sistem hanya akan menghitung pengunjung yang merupakan objek manusia, sehingga sistem hanya perlu mengenali manusia.

Dengan adanya tahap deteksi wajah, maka kini memungkinkan untuk membangun tahap penjejakan wajah. Penjejakan wajah akan menjejaki wajah yang terdeteksi antar frame pada video. Dengan proses penjejakan tersebut, sistem akan mengunci pengunjung antar frame. Setelah itu sistem dapat memutuskan apakah pengunjung yang dijejaki wajahnya tersebut akan digolongkan sebagai pengunjung datang atau tidak berdasarkan pada proses penjejakan wajah sebelumnya.

Berdasarkan hal – hal yang telah disebutkan sebelumnya, maka penulis akan mengidentifikasi kebutuhan – kebutuhan yang perlu disediakan solusinya dalam sistem yang akan dikembangkan. Kebutuhan – kebutuhan tersebut antara lain:

- Sistem yang akan dibangun adalah sistem yang dapat menerima data masukan berupa video maupun rekaman secara waktu nyata (*realtime*).
- Sistem harus memiliki fitur deteksi wajah, karena pengunjung tempat umum yang dimaksud adalah manusia, dimana wajah merupakan elemen utama yang mewakili seorang manusia.
- Sistem yang akan dibangun memiliki kemampuan untuk menjejaki pengunjung. Penjejakan akan dilakukan pada bagian wajah.
- Sistem dapat menghitung jumlah pengunjung yang terdapat pada rekaman secara langsung.

Sistem Penghitung Pengunjung (SiPP) yang akan dikembangkan diilustrasikan pada Gambar 7.



Gambar 7. Arsitektur Sistem

Pada Gambar 7 dapat dilihat arsitektur Sistem Penghitung Pengunjung secara keseluruhan. Dari gambar tersebut, terlihat empat unit yang berperan penting dalam membangun sistem ini secara keseluruhan. Keempat unit itu adalah unit Antarmuka, unit Pendeteksi Wajah, unit Penjejak Wajah, dan unit Penghitung Pengunjung.

Subsistem pendeteksi wajah berguna untuk mendeteksi wajah dari suatu frame pada video. Jika pada suatu frame terdapat sejumlah wajah, maka subsistem ini akan melakukan deteksi wajah terhadap setiap *sub-image* dari frame tersebut berdasarkan pengklasifikasi wajah yang telah dilatih sebelumnya. Jika *sub-image* yang dicocokkan memang wajah, maka akan diberikan lingkaran seukuran wajah yang terdeteksi.

Subsistem penjejak wajah akan menjejaki wajah antar frame, berfungsi untuk mengetahui apakah suatu wajah pada frame saat ini merupakan wajah yang sama pada frame sebelumnya. Subsistem ini melakukan penjejakan wajah tanpa

mengetahui identitas dari wajah, tetapi hanya menggunakan data koordinat posisi wajah dan ukuran wajah.

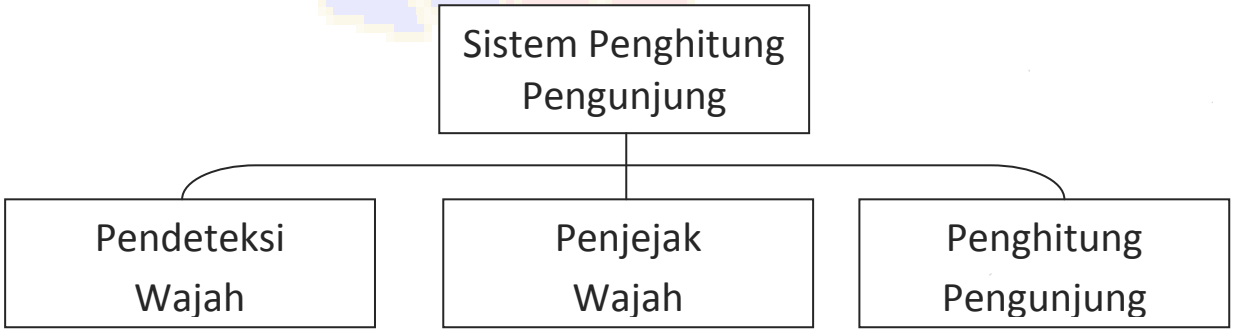
Subsistem penghitung pengunjung berguna untuk menghitung pengunjung pada video berdasarkan pada penjejakan wajah. Setiap informasi penjejakan wajah digunakan untuk mengetahui apakah seorang pengunjung telah memasuki ruangan.

4.2 RANCANGAN SISTEM

Bagian ini akan menjelaskan tentang perancangan sistem yang akan dikembangkan berdasarkan hasil analisis kebutuhan yang telah dilakukan sebelumnya. Rancangan sistem terbagi menjadi tiga bagian yang terdiri dari desain modul, desain basis data, dan desain antar muka yang akan digunakan oleh sistem ini.

4.2.1 Desain Modul

Berdasarkan kebutuhan dari sistem yang telah dianalisis sebelumnya, diperoleh tiga modul yang akan menyusun Sistem Penghitung Pengunjung, yaitu modul Pendeteksi Wajah, modul Penjejak Wajah, dan modul Penghitung Pengunjung. Desain modul tersebut diilustrasikan pada Gambar 8 berikut.



Gambar 8. Desain Modul

Berikut adalah penjelasan dari masing – masing modul yang digambarkan pada Gambar 8,

1. Modul Pendeteksi Wajah

   Modul ini berguna untuk melakukan pendeteksian wajah pada suatu citra. Untuk data masukan berupa video, setiap frame pada video akan diperlakukan sebagai citra untuk kemudian dideteksi wajah yang terdapat pada citra tersebut. Objek yang dianggap wajah pada citra akan diberikan tanda berupa lingkaran tepat pada posisi wajah berada.

2. Modul Penjejak Wajah

   Modul ini bertanggung jawab untuk melakukan penjejakan wajah pada suatu data masukan berupa video. Modul ini bekerja dengan data yang didapat dari modul pendeteksi wajah. Data dari wajah yang terdeteksi yang terdiri dari posisi koordinat piksel titik berat wajah dan ukuran radius wajah akan dijadikan sebagai masukan bagi modul ini. Modul ini sendiri dikembangkan dengan dua metode yang berbeda untuk dibandingkan. Metode jarak *Euclidian* akan menghitung jarak perpindahan objek yang sama dari suatu frame ke frame berikutnya. Sementara metode pengukuran *fuzzy* akan menghitung besarnya derajat kepercayaan apakah objek yang terdeteksi di frame saat ini adalah objek yang sama di frame sebelumnya. Modul ini bekerja dengan cara membandingkan dua frame berurutan.

3. Modul Penghitung Pengunjung

   Modul ini bertanggung jawab melakukan penghitungan pengunjung berdasarkan data yang diperoleh dari dua modul sebelumnya. Data – data tersebut adalah informasi koordinat piksel terakhir pengunjung yang wajahnya

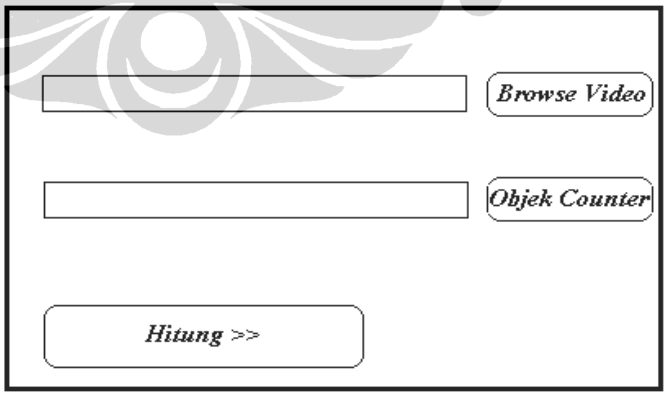
terdeteksi beserta dengan kemunculannya dalam beberapa frame terakhir. Jika memenuhi kedua kriteria ini, maka penghitungan akan menambahkan satu jumlah pengunjung.

4.2.2 Desain Basis Data

Pada sistem ini, basis data yang digunakan adalah sebuah *file* dalam format XML yang berisi basis data wajah hasil pelatihan. Basis data wajah ini akan digunakan oleh modul Pendeteksi Wajah untuk mendeteksi keberadaan wajah pada suatu citra.

4.2.3 Desain Antarmuka

Sistem yang akan dikembangkan merupakan sistem yang membutuhkan interaksi dengan pengguna, oleh sebab itu sistem ini akan berbasiskan GU (*Graphical User* Interface). Berdasarkan kebutuhan yang telah diidentifikasi sebelumnya, maka desain GUI yang dapat memfasilitasi kebutuhan tersebut akan digambarkan dibawah ini. Gambar 9 dan 10 merupakan rancangan GUI yang dapat memfasilitasi kebutuhan tersebut.



Gambar 9. Antarmuka Sistem Penghitung Pengunjung

Gambar 10. Antarmuka Proses Pemutaran Video Beserta Penghitungannya

BAB V

IMPLEMENTASI SISTEM

Bab ini menjelaskan tentang implementasi dari sistem yang akan dikembangkan. Penjelasan yang diberikan meliputi komponen perangkat keras, komponen perangkat lunak, dan implementasi prosedur yang digunakan dalam membangun sistem ini.

5.1 SPESIFIKASI SISTEM

Pada bagian spesifikasi sistem ini akan dijelaskan mengenai komponen – komponen yang digunakan untuk membangun sistem. Komponen – komponen yang dijelaskan meliputi komponen perangkat keras dan perangkat lunak.

5.1.1 Perangkat Keras

Sistem Penghitung Pengunjung (SiPP) ini dikembangkan pada computer desktop dengan spesifikasi perangkat keras sebagai berikut:

Prosesor : Intel ® Pentium 4 CPU 2,4 GHz

Memori : 256 MB RAM

Harddisk : 40 GB

5.1.2 Perangkat Lunak

Perangkat lunak yang digunakan untuk mengembangkan sistem ini adalah sebagai berikut:

Sistem Operasi	: Microsoft Windows XP Professional SP 2
IDE	: Microsoft Visual C++ 6.0
Library	: OpenCV [19].

5.2 IMPLEMENTASI PROSEDUR

Subbab implementasi prosedur akan merinci *pseudocode* yang digunakan dalam implementasi Sistem Penghitung Pengunjung. Pada jarak Euclidian, toleransi jarak maksimum adalah 50 piksel. Nilai ini diukur berdasarkan eksperimen yang dilakukan oleh penulis. Sementara dengan metode pengukuran *fuzzy*, nilai *total confidence value* minimum agar dapat diikutsertakan dalam proses penjejakan adalah 0,5.

5.2.1 Prosedur Pengambilan Frame dari Video

```
openVideo(filename)//Ambil file input
{
    if(exist(filename))//Memastikan keberadaan input
    {
        video =getVideo(filename)//Mengambil file dalam format video
        for(;;)
        {
          if(!grabFrame(video))// tidak ada frame yang dapat diambil, keluar dari loop
              break
          else//Jika frame dapat diambil lakukan dibawah ini
          {
            frame = retrieveFrame(video)//ambil frame
            if(!frame)// frame atau video sudah habis diputar seluruhnya, keluar dari loop
               break
            else
            {
             do_Detect_Trace_Count(frame)// deteksi, jejaki, dan menghitung wajah
```

```
            }
            display(frame)//menampilkan frame (akan tampak sebagai  video)
          }
        }
    }
}
```

5.2.2 Prosedur Pendeteksian, Penjejakan, dan Penghitungan Pengunjung

```
do_Detect_Trace_Count(frame)//Mendeteksi, menjejaki, dan menghitung pengunjung
{
    faces = detectFace(frame)
    for(i = 0 ; i  < faces.total ; i++)
    {
      x = getLocationFace(faces,i).x//mendapatkan koordinat x dari wajah ke-i
      y = getLocationFace(faces,i).y//mendapatkan koordinat y dari wajah ke-i
      radius = getRadiusFace(faces,i)//mendapatkan ukuran radius wajah
      drawCircle(x,y,radius)//gambar lingkaran pada wajah terdeteksi di koordinat
    (x,y) dengan radius sebesar RADIUS
      saveLocation(x,y,i,radius,CURRENTFRAME)//Menyimpan lokasi wajah yang
    terdeteksi beserta dengan radius dalam kaitannya dengan nomor frame
    }
    track_And_Count_Object()//Untuk menjejaki objek
}
```

5.2.3 Prosedur Penyimpanan Lokasi Wajah dengan jarak Euclidian

```
saveLocation(x,y,i,radius,CURRENTFRAME)
{
    if(noPointSaved())//Jika sama sekalibelum ada koordinat wajah yang disimpan
    {
       point[i][CURRENTFRAME].x = x
       point[i][CURRENTFRAME].y = y
       radius[i][CURRENTFRAME] = radius
    }
    else
    {
       for(i = 0 ; i < JUMLAHMAKSIMUMWAJAH ; i++)
       {
        euclidianDistance = euclidianDistance(point[i][CURRENTFRAME
    - 1] ,point(x,y))
       }
       insertIntoArrayPointBasedOnMinDistance(x,y)
    }
}
```

5.2.4 Prosedur Penyimpanan Lokasi Wajah dengan Pengukuran *Fuzzy*

```
saveLocation(x,y,i,radius,CURRENTFRAME)
```

```
{
  if(noPointSaved())//Jika sama sekali belum ada koordinat wajah yang disimpan
  {
   point[i][CURRENTFRAME].x = x
   point[i][CURRENTFRAME].y = y
   radius[i][CURRENTFRAME] = radius
  }
  else
  {
    for(i = 0 ; i < JUMLAHMAKSIMUMWAJAH ; i++)
    {
     euclidianDistance = euclidianDistance(point[i][CURRENTFRAME - 1]
,point(x,y))
     deltaRadius = abs(radius - radius[i][CURRENTFRAME - 1])
     totalConfidenceValues = totalConfidenceValue(euclidianDistances,
deltaRadius);
    }       insertIntoArrayPointBasedOnMaxTotalConfidenceValue(x,y)
  }
}
```

5.2.5 Prosedur Penjejakan dan Penghitungan Pengunjung

```
track_And_Count_Object()//Untuk menjejaki objek
{
    //Tracking objek berdasarkan data pada array Point.
    //Pergerakan objek digambarkan dengan menggunakan lingkaran kecil.
    for(i=0; i<JUMLAHMAKSIMUMWAJAH; i++)
    {
      for(ii = 0 ; ii < CURRENTFRAME; ii++)
      {
        if(Point[i][ii].x !=0 && Point[i][ii].y !=0 )
        {
         drawCircle(Point[i][ii].x,Point[i][ii].y,SMALLRADIUS)
          countObject();//untuk menghitung objek
        }
      }
    }
}
```

5.2.6 Prosedur Penghitungan Pengunjung

```
countObject()//untuk menghitung objek
{
  if(//Sebuah Objek terdeteksi pada 4 frame sebelumnya && telah menghilang dari layar saat
ini && terakhir kali berada di zona penghitungan)
  {
    //tambah jumlah pengunjung
    //hapus semua koordinat pada slot array yang baru saja di counter
  }
  else if(//tracking terputus)
  {
     //hapus semua koordinat pada slot array yang trackingnya terputus
  }
}
```

5.2.7 Prosedur Penghitungan Nilai Kepercayaan

```
Lambda = -0.428;
totalConfidenceValue(velocity, deltaRadius)//Velocity atau euclidian distance
memiliki besaran yang sama
{
    sameObject = rule1(velocity, deltaRadius)//Fuzzy Rule #1
    probablySameObject = rule2(velocity, deltaRadius)//Fuzzy Rule #2
    notSameObject = rule3(velocity, deltaRadius)//Fuzzy Rule #3
    g[1] = 0.9;
    g[2] = 0.9;
    g[3] = 0.9;

    h[1] = sameObject;
    h[2] = probablySameObject;
    h[3] = notSameObject;

    sortDescending(g[])
    sortDescending(h[])

    gLamda[1] = gLambda(1);
    gLamda[2] = gLambda(2);
    gLamda[3] = gLambda(3);

    totalConfidenceValue = maximum(minimum(gLamda[1],h[1]) ,
minimum(gLamda[2],h[2]) , minimum(gLamda[3],h[3]));
    return totalConfidenceValue;
}
```

5.2.8 Prosedur Penghitungan g_λ

```
gLambda(i)
{
    if(i == 1)
    {
        return g[1];
    }
    else
    {
        return g[i] + gLambda(i-1) + Lambda * g[i] * gLambda(i-1);
    }
}
```

5.2.9 Prosedur Aturan *Fuzzy*

```
rule1(velocity, deltaRadius)//Fuzzy Rule #1 : if velocity is slow and deltaRaius is small
then same object
{
    rule1 = min(slow(velocity) , smallRadius(deltaRadius))
    return rule1
}

rule2(velocity, deltaRadius)//Fuzzy Rule #2 : if velocity is Medium and deltaRadius is
medium then probably same object
{
    rule2 = min(medium(velocity) , mediumRadius(deltaRadius))
    return rule2
```

```
}

rule3(velocity, deltaRadius) //Fuzzy Rule #3 : if velocity is tooFast or deltaRaius is
tooBig then not the same object
{
    rule3 = max(fast(velocity) , bigRadius(deltaRadius));
    return 1 - rule3
}

slow(x)
{
    slow = trapmf( x, -0.000000000001, 0, 10,20);
    return slow;
}

medium(x)
{
    medium = trapmf( x, 10, 20, 40,60);
    return medium;
}

fast(x)
{
    fast = trapmf( x, 50, 70, 1000,1000);
    return fast;
}

smallRadius(x)
{
    smallRad = trapmf( x, -0.000000000001, 0, 1,2);
    return smallRad;
}

mediumRadius(x)
{
    mediumRad = trapmf( x, 1, 2, 3,4);
    return mediumRad;
}

bigRadius(x)
{
    bigRad = trapmf(x, 3, 6, 13, 100);
    return bigRad;
}
```

5.2.10 Prosedur Jarak Euclidian

```
euclidianDistance(point1, point2)
{
    distance = sqrt((point1.x-point2.x)^2 + (point1.y-point2.y)^2);
    return distance;
}
```

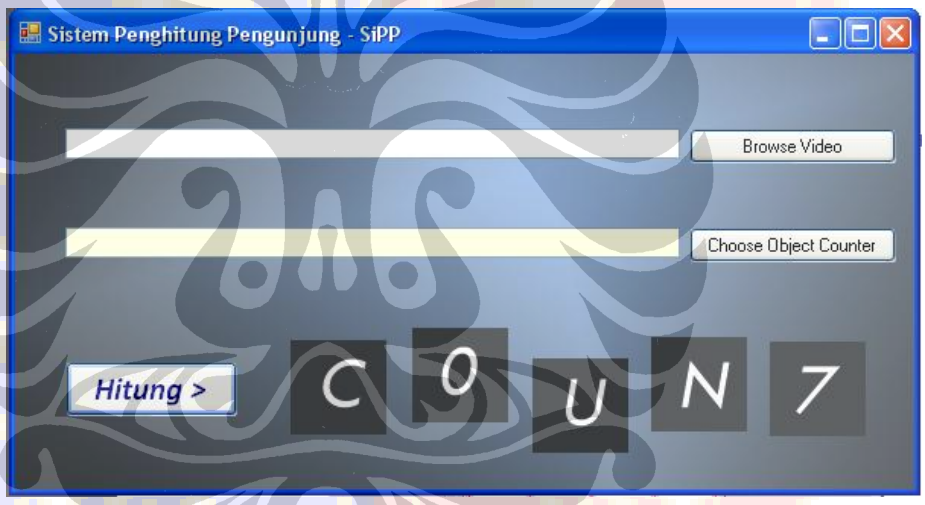
5.2.11 Prosedur Fungsi Keanggotaan Trapesium

```
/*
Method ini adalah fungsi keanggotaan trapesoidal yang akan dipakai
pada fuzzy rule untuk mengukur perubahan centroid dan radius wajah
```

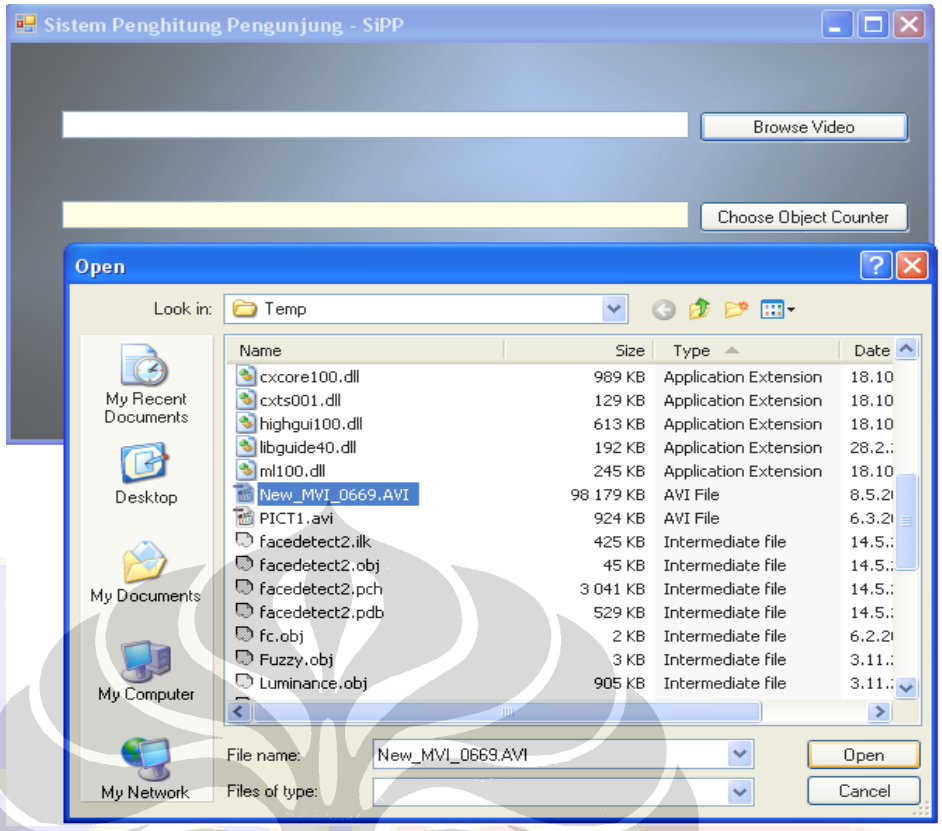
```
*/
trapmf(x, a,  b,  c,  d)
{
    if(x < a) return 0;
    else if(x>=a && x <= b) return (x-a)/(b-a);
    else if(x>=b && x <= c) return 1;
    else if(x>=c && x <= d) return (d-x)/(d-c);
    else return 0;
}
```

5.3 ANTAR MUKA

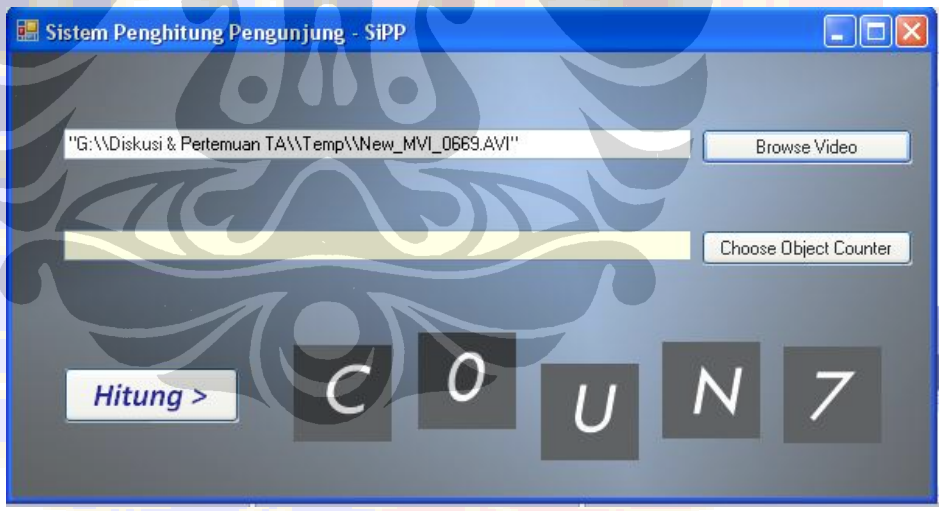
Antar muka dari Sistem Penghitung Pengunjung dibuat dengan menggunakan Visual C# 2005 *Express Edition.* Berikut ini tampilan antar muka dari SiPP yang berbasiskan *Graphical User Interface* (GUI).



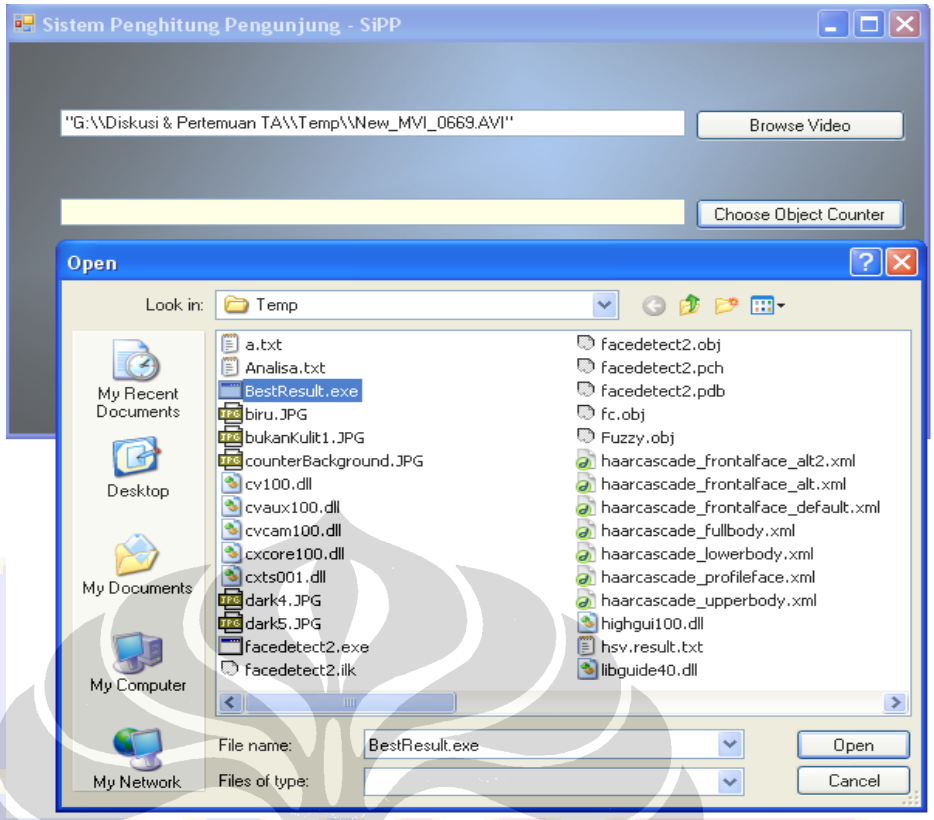
Gambar 11. Antar Muka Utama Sistem



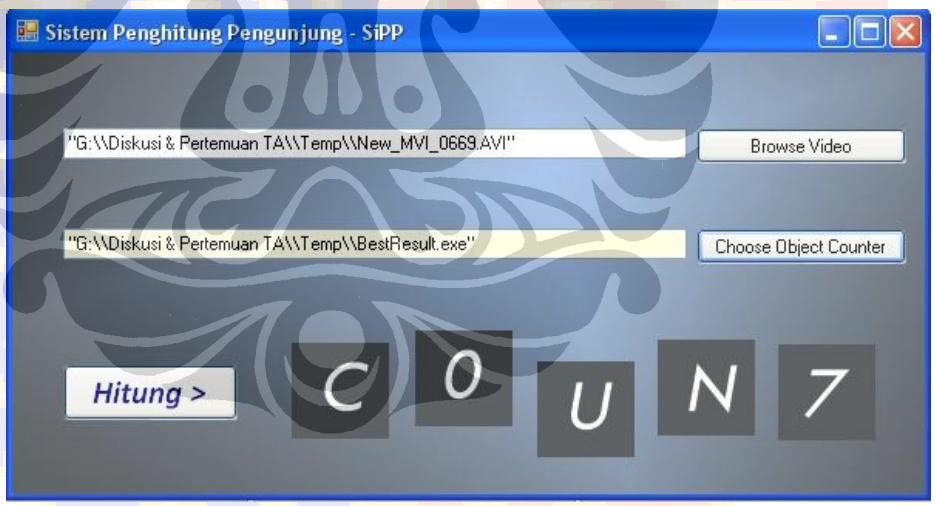
Gambar 12. Antar Muka Pemasukan Video



Gambar 13. Antar Muka Setelah Pemasukan Video



Gambar 14. Antar Muka Pemasukan Engine Human Counter



Gambar 15. Antar Muka Setelah Pemasukan Engine Human Counter

Gambar 16. Pemutaran Video Masukan

Gambar 17. Tampilan Angka Jumlah Pengunjung

BAB VI

UJI COBA DAN ANALISIS

Bab ini tersusun atas penjelasan hasil uji coba terhadap Sistem Penghitung Pengunjung yang dikembangkan beserta analisis hasil uji coba tersebut. Pembahasan meliputi data, skenario, dan hasil uji coba beserta analisisnya.

6.1 DATA UJI COBA

Penelitian ini menggunakan data video rekaman yang direkam oleh penulis. Video rekaman diambil di dua lokasi berbeda, yaitu lab *Multimedia Understanding* (MMU) yang terdiri atas 13 *file* video berdurasi pendek dan stasiun UI yang terdiri atas sebuah *file* video berdurasi panjang. Pengambilan perlu dilakukan di dua tempat berbeda karena video rekaman yang dilakukan di lab MMU adalah rekaman yang telah diskenariokan oleh penulis, sehingga berdurasi pendek dan pergerakan pengunjung kurang alami (natural), namun berguna untuk mengambil pergerakan – pergerakan tertentu, seperti dua orang yang berjalan paralel ataupun dua orang yang berjalan bersilangan. Sementara rekaman yang dibuat di stasiun UI adalah rekaman yang dilakukan tanpa skenario, sehingga pergerakan pengunjung yang terekam memang merupakan pergerakan yang alami (natural) dan merepresentasikan keadaan sesungguhnya.

6.2 SKENARIO UJI COBA

Dalam melakukan pengujian, penulis membuat Sistem Penghitung Pengunjung dengan menggunakan dua metode berbeda untuk menjejaki pengunjung. Metode pertama adalah metode jarak *Euclidian*, sementara metode kedua adalah metode pengukuran *fuzzy*. Kedua metode ini akan dibandingkan kinerjanya dengan membandingkan kemampuan penjejakan beserta dengan akurasinya dalam menghitung jumlah pengunjung.

Dalam ujicoba, terdapat empat kasus utama yang akan dijadikan data masukan bagi sistem. Keempat kasus berikut dipilih karena mewakili berbagai kemungkinan yang akan terjadi. Keempat kasus tersebut adalah:

1. Seorang pengunjung berjalan sendiri.
2. Dua pengunjung berjalan parallel (sejajar).
3. Dua pengunjung berjalan saling bersilangan.
4. Sejumlah pengunjung berjalan dalam formasi tak beraturan.

Keempat kasus tersebut terdapat dalam rekaman di laboratorium MMU yang telah diskenariokan sebelum pengambilan video. Dalam pengujian juga disertakan rekaman di stasiun UI, dimana pergerakan pengunjung tidak beraturan (keempat kasus di atas terdapat pada rekaman ini) dan mewakili kondisi nyata. Tabel 1 menunjukkan skenario ujicoba yang akan dilakukan.

Tabel 1. Skenario Uji Coba Sistem

No Rekaman	Jumlah Pengunjung	Durasi (menit : detik)	Dimensi (piksel)	Jenis Kasus	Diskenariokan
1	1	00 : 12	320 x 240	1	ya
2	2	00 : 17		2	
3	2	00 : 16		3	
4	3	00 : 20		4	
5	21	04 : 16	640 x 480	1,2,3,4	tidak

6.3 HASIL UJI COBA

Pada subbab ini akan dijelaskan hasil uji coba yang telah dilakukan berdasarkan skenario uji coba yang telah dibuat. Dari uji coba juga diketahui, baik metode pengukuran *fuzzy* maupun jarak Euclidian memiliki akurasi kinerja yang sama, sehingga hasil penghitungannya sama akurat. Adapun sedikit perbedaan hanya dari segi waktu komputasi. Oleh karena itu, baik *screenshot* dan tabel hanya dituliskan hasil uji coba untuk SiPP, tanpa merinci metode yang digunakan, karena kedua metode yang digunakan menghasilkan hasil penghitungan yang sama (kecuali untuk tabel waktu komputasi).

Rekaman 1: Pengunjung berjalan seorang diri

Hasil uji coba untuk pengunjung yang berjalan sendiri dapat dilihat pada Lampiran 1. Penjelasan akan diberikan melalui 3 gambar *screen shot* yang paling representatif, yaitu saat pengunjung berjalan sebelum bagian wajah memasuki zona penghitungan, ketika wajah berada dalam zona penghitungan, dan ketika wajah telah melewati zona penghitungan.

Dari hasil percobaan, SiPP berhasil menghitung pengunjung yang berjalan seorang diri. Pengunjung yang terdapat pada rekaman ini berjumlah satu orang, dan SiPP berhasil menghitungnya dengan benar, sehingga angka yang tertera pada counter adalah satu.

Rekaman 2: Dua pengunjung berjalan paralel

Screenshot dari rekaman ini dapat dilihat pada Lampiran 2. Rekaman ini terdiri dari dua orang yang berjalan sejajar (paralel). Pengunjung tersebut berjalan paralel sejak awal kemunculannya sampai dengan memasuki ruangan. Pada rekaman ini SiPP juga berhasil melakukan penjejakan terhadap keduanya dengan baik serta dan menghitung jumlah pengunjung masuk dengan benar, yaitu dua.

Rekaman 3: Dua pengunjung berjalan bersilangan

Pada rekaman ini, terdapat dua pengunjung yang berjalan bersama pada awal kemunculannya, namun menjelang memasuki pintu masuk, seorang pengunjung mengambil jalan pengunjung yang lain, sehingga laku orang tersebut bersilangan terhadap orang lain. Kasus ini menjadi perhatian, karena menimbulkan potensi salah penjejakan. Pada hasil uji coba yang dilakukan, tidak terdapat kesalahan penjejakan maupun penghitungan. Hasil uji coba dapat dilihat pada Lampiran 3.

Rekaman 4: Tiga pengunjung berjalan dalam formasi tak beraturan

Pada rekaman ini terdapat tiga pengunjung yang hendak memasuki ruangan. Rekaman ini menjadi perhatian, karena juga menimbulkan adanya potensi kesalahan penjejakan dan penghitungan. Dari uji coba, SiPP berhasil menghitung pengunjung dengan benar. *Screenshot* dari rekaman ini dapat dilihat di Lampiran 4.

Rekaman 5: Pintu masuk peron Stasiun UI

Penyajian hasil uji coba untuk rekaman di Stasiun UI berbeda dibandingkan dengan penyajian sebelumnya. Panjangnya durasi rekaman tidak memungkinkan untuk dilakukannya pengambilan *screenshot* seperti pada 4 penyajian sebelumnya (terdapat di lampiran 1 - 4). Contoh *screenshot* untuk rekaman ini dapat dilihat pada Gambar 14.

Berdasarkan rekaman video, ada 21 pengunjung yang memasuki peron Stasiun UI. Berbeda dengan empat uji coba sebelumnya, dalam uji coba kali ini SiPP gagal menghitung pengunjung secara sempurna. Dalam melakukan proses penghitungan pengunjung, SiPP berhasil menghitung 13 orang secara benar. 8 orang sisanya gagal dihitung dengan benar dengan jenis kesalahan *false negative*, yaitu dalam keadaan sesungguhnya memang ada pengunjung yang memasuki peron, namun SiPP tidak menghitung orang tersebut (dijelaskan pada Tabel 2). SiPP juga melakukan tiga kali *false positive*, yaitu dalam kondisi sebenarnya tidak ada yang perlu dihitung, namun sistem mendeteksi adanya pengunjung masuk dan menaikkan angka pada *counter* (dijelaskan pada Tabel 3). Rangkuman hasil penghitungan disajikan pada Tabel 4.

Tabel 2. Kesalahan False Negative Untuk Rekaman Stasiun UI

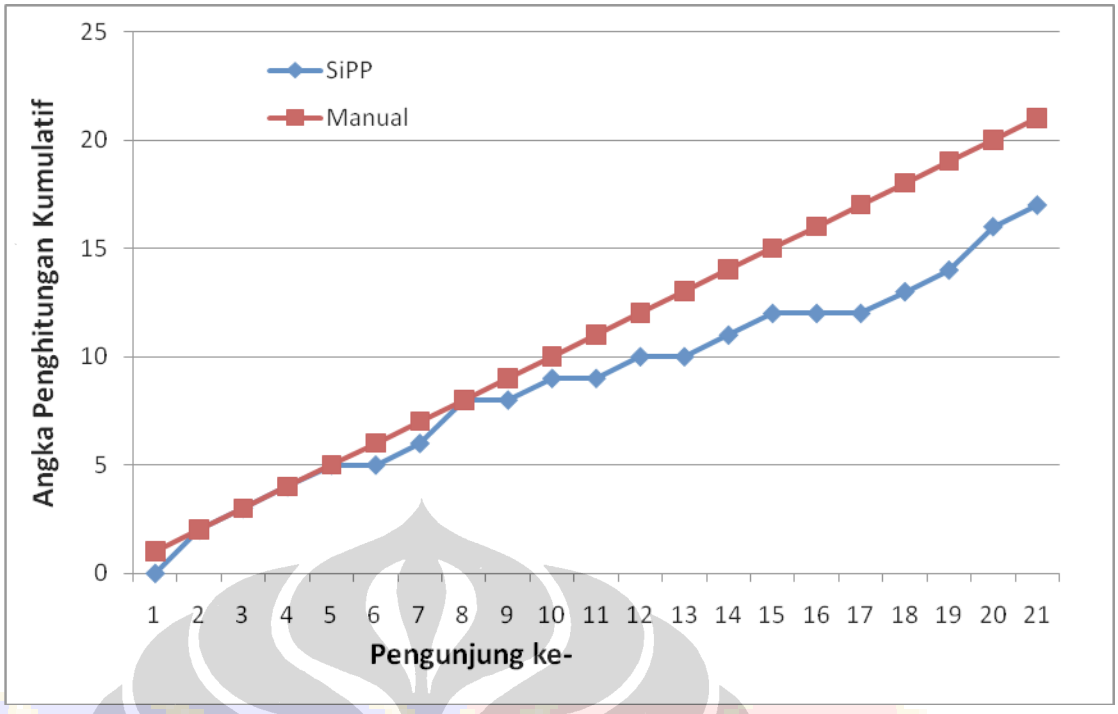
# Pengunjung	Status Penghitungan	Deskripsi
1	Gagal (*false negative*)	Wajah tidak terdeteksi
2		Pengunjung berbalik di pintu masuk
6		Wajah tidak terdeteksi
9		Wajah tidak terdeteksi
11		Tubuh pengunjung terlalu tinggi, sehingga wajah keluar dari zona penghitungan
13		Wajah tidak terdeteksi
16		Wajah tidak terdeteksi
17		Tubuh pengunjung terlalu tinggi, wajah keluar dari zona penghitungan

Tabel 3. Kesalahan False Positive Untuk Rekaman Stasiun UI

Angka Pada Counter	Status Penghitungan	Deskripsi
1	Gagal (*false positive*)	Ada false positive pada proses pendeteksian wajah di zona penghitungan
7		
15		Pengunjung ke-19 dihitung ulang, karena penjejakan padanya terputus tepat di zona penghitungan dan dijejaki ulang untuk kedua kalinya.

Tabel 4. Hasil Uji Coba Rekaman Stasiun UI

# Pengunjung	Status Penghitungan	Angka Pada Counter
1	*false negative*	0
	false positive	1
2	*false negative*	2
3	Sukses	3
4	Sukses	4
5	Sukses	5
6	*false negative*	5
7	Sukses	6
	false positive	7
8	Sukses	8
9	*false negative*	8
10	Sukses	9
11	*false negative*	9
12	Sukses	10
13	*false negative*	10
14	Sukses	11
15	Sukses	12
16	*false negative*	12
17	*false negative*	12
18	Sukses	13
19	Sukses	14
	false positive	15
20	Sukses	16
21	Sukses	17



Gambar 18. Grafik Perbandingan Antara Penghitungan Manual dengan SiPP

Gambar 18 di atas menggambarkan hubungan antara angka perhitungan kumulatif yang dihitung dengan menggunakan SiPP dan manual. Dari grafik di atas, memang tampak adanya kecenderungan, bahwa penghitungan kumulatif yang dilakukan SiPP akan semakin memiliki kesalahan yang membesar untuk jumlah pengunjung yang lebih banyak.

Berdasarkan hasil uji coba yang dilakukan, dapat disimpulkan, terjadinya kesalahan penghitungan disebabkan hal-hal berikut:

1. *False negative* pada proses deteksi wajah
2. *False positive* pada proses deteksi wajah
3. Wajah Pengunjung tidak memasuki zona penghitungan
4. Pengunjung membalikkan tubuhnya di pintu masuk.

Dari hasil uji coba juga diketahui bahwa baik untuk metode pengukuran *fuzzy* maupun jarak Euclidian menghasilkan akurasi yang sama, oleh karena itu penulis

akan membandingkan selisih waktu komputasi yang digunakan oleh metode pengukuran *fuzzy* dengan jarak Euclidian dalam Tabel 5 dan Tabel 6.

Tabel 5. Perbandingan Waktu Komputasi

Ukuran Frame	Parameter acuan	Pengukuran *Fuzzy*	Jarak Euclidian	Selisih
320 X 240	Rata – rata (ms)	0,003454886	0,000496779	0,002958108
	Minimum (ms)	0,0023093	0,000140058	0,002134227
	Maksimum (ms)	0,0535523	0,00163568	0,053087106
640 X 480	Rata – rata (ms)	0,00293653	0,000326645	0,00261
	Minimum (ms)	0,00224594	0,000126719	0,002064
	Maksimum (ms)	0,0215773	0,00160233	0,021417

Tabel 5 merupakan perbandingan waktu komputasi untuk menghitung prosedur pengukuran *fuzzy* dengan jarak Euclidian. Jarak Euclidian memiliki waktu komputasi yang lebih singkat. Hal ini memang seharusnya terjadi, karena untuk melakukan prosedur pengukuran *fuzzy*, sebelumnya harus menghitung jarak Euclidian terlebih dahulu (jarak Euclidian dijadikan input untuk fungsi pengukuran *fuzzy*).

Tabel 6. Perbandingan Kecepatan Penjejakan Wajah

Ukuran Frame	Parameter acuan	Pengukuran *Fuzzy*	Jarak Euclidian	Selisih
320 X 240	Rata – rata (ms)	75,11624	73,43745	1,678791
	Minimum (ms)	70,7146	69,069	1,6456
	Maksimum (ms)	79,7919	79,4814	0,3105
640 X 480	Rata – rata (ms)	334,9573	333,5488	1,408559
	Minimum (ms)	316,721	275,339	41,382
	Maksimum (ms)	435,492	432,64	2,852

Pada Tabel 6 merupakan perbandingan waktu yang dibutuhkan untuk menjejaki wajah seorang pengunjung antar frame. Dari tabel tersebut, juga tetap diketahui, bahwa metode jarak Euclidian masih lebih singkat dibandingkan dengan metode pengukuran *fuzzy*.

6.4 ANALISIS

Dari uji coba yang telah dilakukan, baik metode pengukuran *fuzzy* maupun jarak Euclidian memiliki akurasi penjejakan yang sama. Walaupun lebih lambat dari segi waktu komputasi, namun metode pengukuran *fuzzy* memiliki keunggulan dibandingkan dengan metode jarak Euclidian. Keunggulan tersebut adalah:

1. Metode pengukuran *fuzzy* memperhitungkan ukuran radius wajah.

   Metode pengukuran *fuzzy* menjejaki wajah seseorang berdasarkan derajat nilai kepercayaan yang dihitung dengan menggunakan parameter input jarak Euclidian dan perubahan radius wajah. Sementara jarak Euclidian hanya menggunakan jarak untuk menjejaki seseorang. Metode pengukuran *fuzzy* dapat mengantisipasi jika terjadi perubahan ukuran wajah seseorang secara tak wajah, sehingga tidak dihitung sebagai pengunjung yang sama. Sementara metode jarak Euclidian tetap akan menganggap itu sebagai pengunjung yang sama. Namun keunggulan ini tidak tampak pada uji coba, karena sangat sulit untuk membuat kondisi ini pada saat melakukan rekaman.

2. Metode pengukuran *fuzzy* lebih mudah disesuaikan.

   Telah diketahui sebelumnya, jika posisi kamera dipindahkan, maka perlu dilakukan penyesuaian ulang. Metode pengukuran *fuzzy* akan lebih mudah disesuaikan, karena menggunakan fungsi keanggotaan yang lebih intuitif, sehingga mudah dalam melakukan pengaturan terhadap *rule*.