

## BAB IV

### PENGUJIAN SISTEM

Pengujian sistem yang dilakukan merupakan pengujian terhadap program mendeteksi tempat parkir yang telah selesai dibuat. Dimulai dari pengambilan citra dari webcam, pengolahan citra yang dimulai dengan streaming Citra, update citra kondisi parkir, menentukan koordinat pada tiap-tiap area, akses *pixel*, Background Subtraction, *thresholding*, penghitungan *pixel* putih dan identifikasi benda. kemudian yang terakhir adalah pengujian sistem secara keseluruhan yaitu Program akan mengeluarkan output list nomor parkir yang kosong sesuai kondisi setiap 5 menit sekali.

#### 4.1 Pengujian *Streaming* Citra Melalui webcam

Pengujian *streaming* ini dilakukan dengan mengintegrasikan Microsoft Visual C++ melalui library OpenCV. Yaitu untuk memanggil serta menjalankan *console Webcam* tersebut.

##### 4.1.1 Tujuan

Tujuan pengujian ini yaitu untuk mengetahui apakah aplikasi sudah mampu menampilkan data citra dari webcam ke aplikasi pada Visual C++ dan apakah dapat langsung diproses oleh program.

#### 4.1.2 Alat yang Digunakan

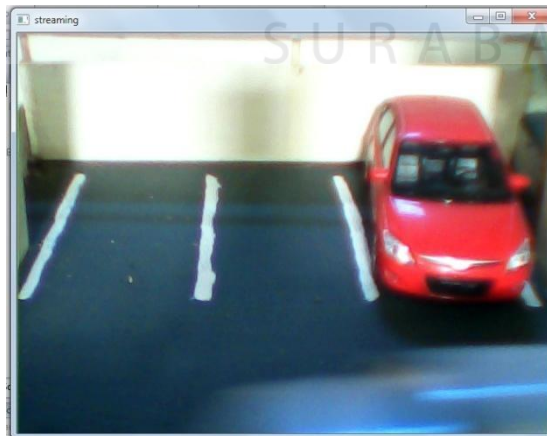
1. Microsoft Visual C++ 2008
2. Webcam
3. Personal Computer (PC)

#### 4.1.3 Prosedur Pengujian

1. Menjalankan program *console* pada Microsoft Visual C++ 2008
2. Menjalankan program untuk mengakses data citra pada kamera PC tersebut
3. Melihat hasil data citra pada *window*

#### 4.1.4 Hasil Pengujian

Setelah melakukan pengujian sesuai dengan prosedur diatas berikut adalah gambar yang didapatkan dari Webcam pada Gambar 4.1.



Gambar 4.1 Capture streaming dengan Webcam

## 4.2 Pengujian *Update* Citra Kondisi Parkir

Pengujian update citra kondisi parkir dilakukan dengan melihat perubahan gambar yang tersimpan pada direktori D:\\Pict\_TA\\update.jpg dan melihat time sistem sebagai waktu acuan.

### 4.2.1 Tujuan

Tujuan pengujian ini yaitu untuk mengetahui apakah aplikasi sudah mampu mengambil gambar secara otomatis setiap 5 menit sekali.

### 4.2.2 Alat yang Digunakan

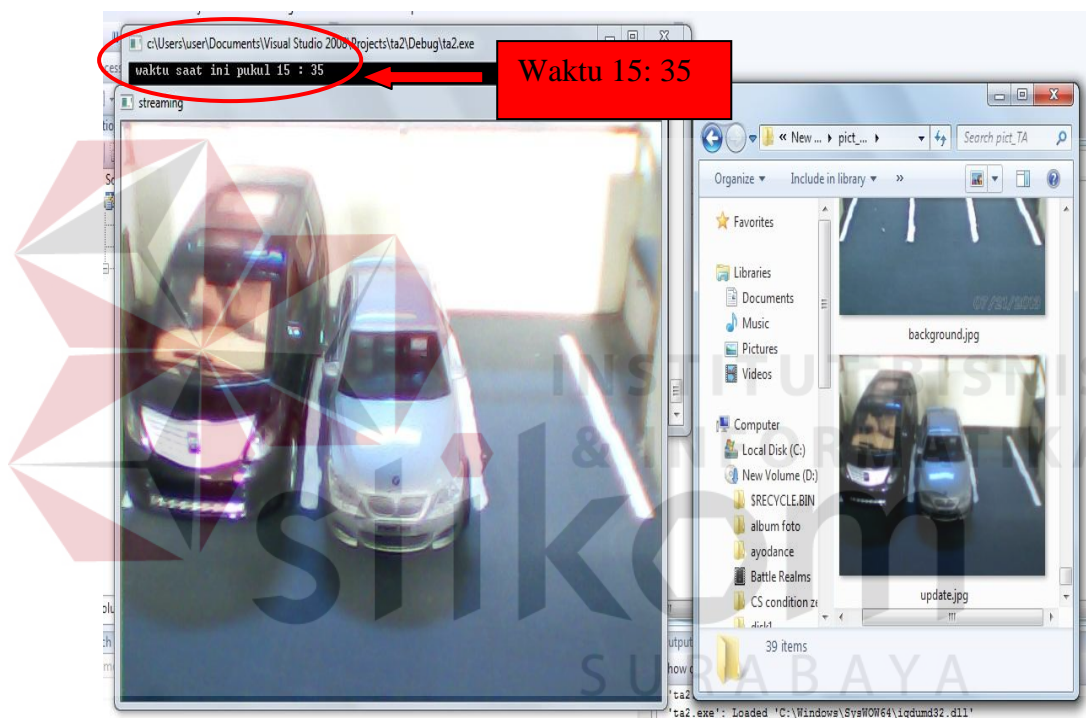
1. Microsoft Visual C++ 2008
2. *Personal Computer* (PC)
3. *Webcam*
4. Miniatur parkir mobil

### 4.2.3 Prosedur Pengujian

1. Menjalankan program *console* pada Microsoft Visual C++ 2008
2. Menjalankan program untuk mengakses data citra pada *Webcam*
3. Menjalankan program untuk *streaming* citra
4. Menjalankan program untuk mengakses *time system*
5. Membuka direktori D:\\Pict\_TA, melihat file bernama update.jpg
6. Melihat perubahan gambar pada file update.jpg

#### 4.2.4 Hasil Pengujian

Hasil dari pengujian ini adalah mengetahui perubahan nilai pada file update.jpg yaitu berupa perubahan gambar yang tersimpan. Karena file tersebut akan dikurangkan dengan citra sampel. Setelah melakukan pengujian sesuai dengan prosedur diatas berikut adalah gambar hasil proses update citra kondisi parkir .



Gambar 4.2 Gambar Citra Kondisi Awal



Gambar 4.3 Gambar Citra Kondisi Parkir Setelah Beberapa Menit

Pada Gambar 4.3 menunjukkan bahwa data telah di update yaitu kondisi awal pada blok tersebut terdapat 2 mobil pada pukul 15:35 (Gambar 4.2). Kemudian terdapat 1 mobil lagi parkir pada blok tersebut sehingga pada saat update data maka file image “update.jpg” pada direktori D:\\Pict\_TA juga akan berubah mengikuti kondisi saat proses update terakhir.

### 4.3 Pengujian Penentuan Koordinat Pada Tiap Area

Pengujian penentuan koordinat pada tiap area dilakukan dengan melihat posisi tanda persegi pada gambar yang telah ditampilkan.

#### 4.3.1 Tujuan

Tujuan pengujian ini yaitu untuk mengetahui apakah koordinat *pixel* yang menjadi acuan dari 3 area sudah tepat.

#### 4.3.2 Alat yang Digunakan

- 1 Microsoft Visual C++ 2008
- 2 *Personal Computer* (PC)
- 3 Citra sampel
- 4 Citra update

#### 4.3.3 Prosedur Pengujian

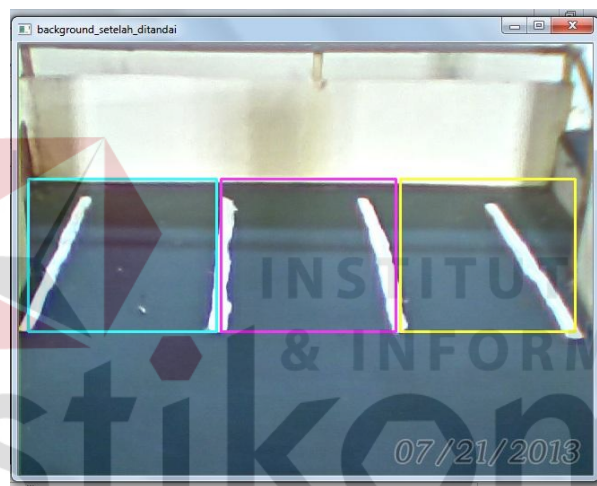
1. Menjalankan program *console* pada Microsoft Visual C++ 2008
2. Menjalankan program untuk mengakses data citra pada *Webcam*
3. Menjalankan program untuk memuat gambar
4. Menjalankan program untuk membuat tanda persegi
5. Menjalankan program untuk menampilkan window image citra sampel dan update

#### 4.3.4 Hasil Pengujian

Hasil dari pengujian ini adalah mengetahui apakah posisi koordinat yang menjadi acuan proses dari masing-masing area yang telah ditentukan sudah berada di posisi yang benar. Berikut adalah gambar hasil penentuan koordinat *pixel*.



Gambar 4.4 *Background Awal*



Gambar 4.5 *Background Setelah Diberi Tanda*

Pada Gambar 4.5 menunjukkan bahwa proses menentukan koordinat dari masing-masing area berhasil dilakukan dengan member tanda persegi pada masing-masing area. Persegi biru merupakan koordinat area 1, persegi ungu merupakan koordinat area 2 dan persegi kuning merupakan koordinat area 3.

#### 4.4 Pengujian Akses *Pixel*

Pengujian akses *pixel* dilakukan dengan menampilkan window image yang berisi *pixel* bernilai merah, window image bernilai hijau dan window image



bernilai biru. Selain itu ditampilkan juga output nilai dari masing-masing image tersebut.

#### 4.4.1 Tujuan

Tujuan pengujian ini yaitu untuk mengetahui apakah aplikasi dapat mengakses *pixel* merah, hijau dan biru pada sebuah gambar.

#### 4.4.2 Alat yang Digunakan

1. Microsoft Visual C++ 2008
2. *Personal Computer* (PC)
3. Citra sampel

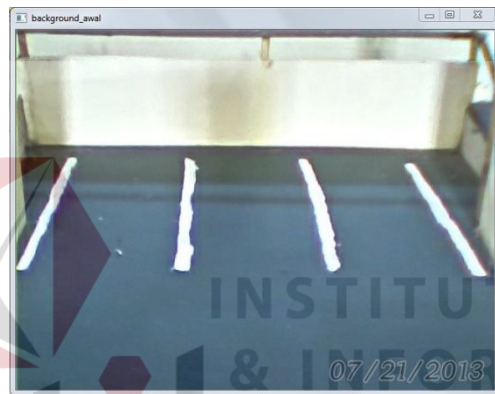
#### 4.4.3 Prosedur Pengujian

4. Menjalankan program *console* pada Microsoft Visual C++ 2008
5. Menjalankan program untuk mengakses data citra pada *Webcam*
6. Menjalankan program untuk memuat citra sampel
7. Menjalankan program untuk mengakses *pixel*
8. Menjalankan program untuk mengubah kumpulan *pixel* menjadi file berformat image
9. Menjalankan program untuk menampilkan window image *pixel* merah, hijau dan biru



#### 4.4.4 Hasil Pengujian

Hasil dari pengujian ini adalah mengetahui apakah program berhasil akses tiap-tiap *pixel* sesuai dengan jenis warnanya yaitu merah, hijau dan biru. Indikasi ketika program berhasil mengakses tiap-tiap *pixel* sesuai jenis warnanya adalah hasil tampilan program berupa image kumpulan warna merah, hijau dan biru. Berikut adalah gambar hasil pengujian akses *pixel*.



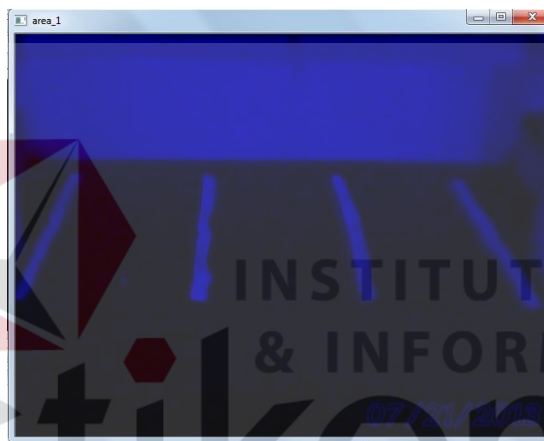
Gambar 4.6 Gambar Sampel



Gambar 4.7 Gambar Hasil Akses *Pixel* Merah



Gambar 4.8 Gambar Hasil Akses *Pixel* Hijau



Gambar 4.9 Gambar Hasil Akses *Pixel* Biru

#### 4.5 Pengujian *Background Subtraction*

Pengujian Background Subtraction dilakukan dengan menampilkan window image yang berisi hasil subtraksi dari masing-masing area.

##### 4.5.1 Tujuan

Tujuan pengujian ini yaitu untuk mengetahui apakah aplikasi dapat melakukan subtraksi/pengurangan *pixel* citra sampel dengan citra update.

#### 4.5.2 Alat yang Digunakan

1. Microsoft Visual C++ 2008
2. *Personal Computer* (PC)
3. Citra sampel
4. Citra update





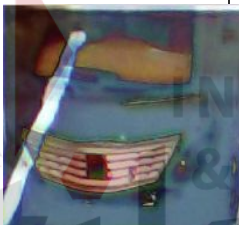



#### 4.5.3 Prosedur Pengujian


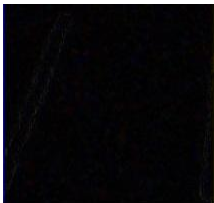

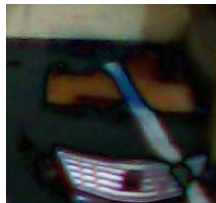

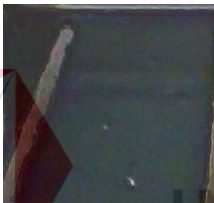




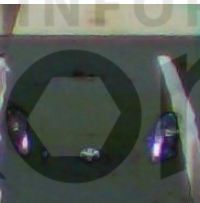

1. Menjalankan program *console* pada Microsoft Visual C++ 2008
2. Menjalankan program untuk mengakses data citra pada *Webcam*
3. Menjalankan program untuk memuat citra sampel dan citra update
4. Menjalankan program untuk subtraksi
5. Menjalankan program untuk menampilkan window image hasil subtraksi

#### 4.5.4 Hasil Pengujian

Hasil dari pengujian ini adalah mengetahui apakah program dapat mengurangi antara *pixel* RGB dari citra sampel dengan *pixel* RGB citra update. Berikut merupakan tabel hasil proses *Background Subtraction*.

Tabel 4.1 Hasil *Subtraction Background*

Kondisi Citra update	Area 1 Setelah subtraksi	Area 2 Setelah subtraksi	Area 3 Setelah subtraksi
 Kondisi kosong			
 Area 1 terisi			
 Area 1 & 2 terisi			
 Area 1 & 3 terisi			
 Area 2 terisi			

Kondisi Citra update	Area 1 Setelah subtraksi	Area 2 Setelah subtraksi	Area 3 Setelah subtraksi
 Area 2 & 3 terisi			
 Area 3 terisi			
 Parkir terisi semua			

Pada Tabel 4.1 merupakan hasil subtraksi dalam beberapa kondisi. Area 1, area 2 dan area 3 merupakan keterangan untuk mewakili nomor dari masing-masing slot parkir. Seperti area 1 untuk slot parkir nomor 1, area 2 untuk slot parkir nomor 2, dsb. Dari hasil subtraksi masing-masing area akan di *thresholding* agar dapat melakukan penghitungan *pixel* sehingga dapat melakukan proses identifikasi benda.

## 4.6 Pengujian *Thresholding*

Pengujian *thresholding* dilakukan dengan menampilkan window image yang berisi gambar hasil subtraksi berformat *grayscale* dan *thresholding* dari masing-masing area.

### 4.6.1 Tujuan

Tujuan pengujian ini yaitu untuk mengetahui apakah program dapat mengkonversi format gambar ke bentuk *grayscale* dan biner.

### 4.6.2 Alat yang Digunakan

1. Microsoft Visual C++ 2008
2. *Personal Computer* (PC)
3. Citra hasil subtraksi area 1, area 2 dan area 3

### 4.6.3 Prosedur Pengujian

1. Menjalankan program *console* pada Microsoft Visual C++ 2008
2. Menjalankan program untuk mengakses data citra pada *Webcam*
3. Menjalankan program untuk memuat citra hasil subtraksi.
4. Menjalankan program untuk konversi RGB ke bentuk *grayscale*
5. Menjalankan program untuk konversi RGB ke bentuk biner
6. Menjalankan program untuk menampilkan window image *grayscale* dan biner

#### 4.6.4 Hasil Pengujian

Hasil dari pengujian ini adalah mengetahui apakah program dapat mengubah gambar dari format RGB menjadi format *grayscale* dan biner .



Gambar 4.10 Gambar Hasil Subtraksi



Gambar 4.11 Gambar Hasil Konversi *Grayscale*





Gambar 4.12 Gambar Hasil *Thresholding*

#### **4.7 Pengujian Penghitungan *Pixel***

Pengujian penghitungan *pixel* dilakukan dengan menampilkan output berupa nilai total *pixel* putih dari tiap-tiap area.

##### **4.7.1 Tujuan**

Tujuan pengujian ini yaitu untuk mengetahui apakah program dapat melakukan penghitungan *pixel* putih pada masing-masing area. Tujuan lainnya adalah untuk mengetahui nilai total *pixel* putih dari masing-masing area.

##### **4.7.2 Alat yang Digunakan**

1. Microsoft Visual C++ 2008
2. *Personal Computer* (PC)
3. Citra hasil subtraksi yang telah dibinerkan

#### 4.7.3 Prosedur Pengujian

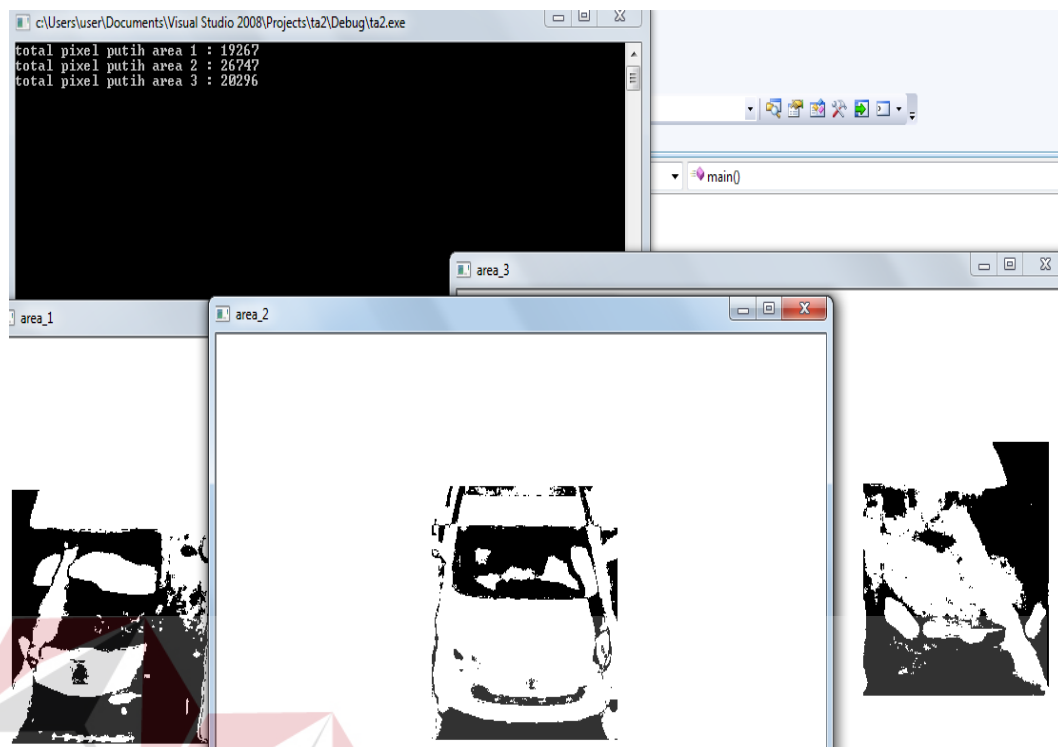
1. Menjalankan program *console* pada Microsoft Visual C++ 2008
2. Menjalankan program untuk mengakses data citra pada *Webcam*
3. Menjalankan program untuk memuat gambar hasil subtraksi yang dibinerkan
4. Menjalankan program untuk penghitungan *pixel*
5. Menjalankan program untuk menampilkan output nilai

#### 4.7.4 Hasil Pengujian

Hasil dari pengujian ini adalah mengetahui nilai total *pixel* warna putih yang terdapat pada area 1, area 2 dan area 3. Berikut adalah gambar hasil pengujian penghitungan *pixel*.



Gambar 4.13 Gambar Citra *Update*



Gambar 4.14 Hasil Penghitungan *Pixel* Putih

#### 4.8 Pengujian Identifikasi Benda

Pengujian penghitungan *pixel* dilakukan dengan menampilkan output berupa list nomor parkir yang kosong, window image citra update, window image hasil subtraksi yang telah dibinerkan dan output nilai hasil pembagian *pixel* putih dengan total *pixel* pada masing-masing area.

##### 4.8.1 Tujuan

Tujuan pengujian ini yaitu untuk mengetahui apakah program dapat membedakan objek yang berada di tempat parkir merupakan mobil atau tidak.

#### 4.8.2 Alat yang Digunakan

4. Microsoft Visual C++ 2008
5. *Personal Computer* (PC)
6. Citra hasil subtraksi yang telah dibinerkan
7. Citra update

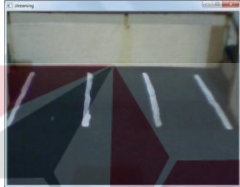
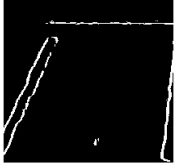


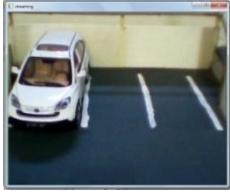

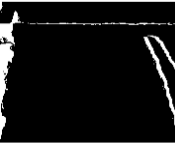

#### 4.8.3 Prosedur Pengujian

6. Menjalankan program *console* pada Microsoft Visual C++ 2008
7. Menjalankan program untuk mengakses data citra pada *Webcam*
8. Menjalankan program untuk memuat citra update.
9. Menjalankan program untuk memuat gambar hasil subtraksi yang dibinerkan
10. Menjalankan program untuk mencari prosentasi *pixel* putih
11. Menjalankan program untuk menampilkan output prosentasi *pixel* putih
12. Menjalankan program untuk klasifikasi objek
13. Menjalankan program untuk menampilkan list nomor parkir yang kosong

#### 4.8.4 Hasil Pengujian

Hasil dari pengujian ini adalah mengetahui bahwa program telah mampu membedakan tempat yang kosong dan tempat yang terdapat mobil. Berikut adalah gambar hasil identifikasi benda.

Tabel 4.2 Hasil Identifikasi Benda

Kondisi	Area	Total <i>white pixel</i>	Total <i>pixel</i>	Persentase (hasil)	Status
	 Area 1	1787 1634 1631 1609 1615	42411 42411 42411 42411 42411	0,042171 0,038550 0,038463 0,037916 0,038074	Kosong Kosong Kosong Kosong Kosong
	 Area 2	2002 1926 1906 1897 1906	39396 39396 39396 39396 39396	0,050823 0,048801 0,043827 0,048198 0,043834	Kosong Kosong Kosong Kosong Kosong
	 Area 3	1546 1532 1498 1484 1528	39396 39396 39396 39396 39396	0,039265 0,038812 0,380289 0,037734 0,038767	Kosong Kosong Kosong Kosong Kosong
	 Area 1	28715 28649 28408 28588 28689	42411 42411 42411 42411 42411	0,677132 0,675545 0,669874 0,67412 0,676433	Terisi Terisi Terisi Terisi Terisi
	 Area 2	3123 3022 3022 3032 3068	39396 39396 39396 39396 39396	0,079276 0,076701 0,076701 0,076977 0,077834	Kosong Kosong Kosong Kosong Kosong
	 Area 3	1694 1620 1615 1652 1677	39396 39396 39396 39396 39396	0,042921 0,041134 0,040956 0,041978 0,042654	Kosong Kosong Kosong Kosong Kosong

 Area 1	18831	42411	0,444018	Terisi
	18379	42411	0,433571	Terisi
	18330	42411	0,432245	Terisi
	18270	42411	0,430898	Terisi
	18161	42411	0,428212	terisi
 Area 2	31560	39396	0,801135	Terisi
	30221	39396	0,767121	Terisi
	30145	39396	0,765155	Terisi
	29713	39396	0,754276	Terisi
	29467	39396	0,747963	Terisi
 Area 3	2339	39396	0,056414	Kosong
	2253	39396	0,05567	Kosong
	2189	39396	0,056658	Kosong
	2223	39396	0,056341	Kosong
	2192	39396	0,057122	Kosong
 Area 1	18638	42411	0,439461	Terisi
	18077	42411	0,426234	Terisi
	17982	42411	0,423994	Terisi
	17869	42411	0,421329	Terisi
	17801	42411	0,419726	Terisi
 Area 2	7226	39396	0,18342	Kosong
	6941	39396	0,176185	Kosong
	6893	39396	0,174967	Kosong
	6928	39396	0,175855	Kosong
	6948	39396	0,176363	Kosong
 Area 3	22608	39396	0,573865	Terisi
	22582	39396	0,573205	Terisi
	22421	39396	0,569119	Terisi
	22328	39396	0,566758	Terisi
	23745	39396	0,602726	Terisi

	 <p>Area 1</p>	1921 1949 1924 1898 1833	42411 42411 42411 42411 42411	0,0452949 0,0459551 0,0453656 0,0447525 0,0457429	Kosong Kosong Kosong Kosong Kosong
	 <p>Area 2</p>	20172 20298 20313 20347 20290	39396 39396 39396 39396 39396	0,512032 0,51523 0,515611 0,516474 0,515027	Terisi Terisi Terisi Terisi Terisi
	 <p>Area 3</p>	1824 1815 1864 1871 1864	39396 39396 39396 39396 39396	0,0462737 0,0460707 0,0473144 0,0474921 0,0473144	Kosong Kosong Kosong Kosong Kosong
	 <p>Area 1</p>	1858 1797 1780 1793 1726	42411 42411 42411 42411 42411	0,0438094 0,0423711 0,0419702 0,0422768 0,040697	Kosong Kosong Kosong Kosong Kosong
	 <p>Area 2</p>	23903 23514 23313 23421 23458	39396 39396 39396 39396 39396	0,606737 0,596863 0,591761 0,594756 0,595441	Terisi Terisi Terisi Terisi Terisi
	 <p>Area 3</p>	19821 19173 18958 18964 19036	39396 39396 39396 39396 39396	0,503122 0,486674 0,481216 0,481369 0,483196	Terisi Terisi Terisi Terisi Terisi



	 <p>Area 1</p>	1970	42411	0,0464502	Kosong
		1849	42411	0,0435972	Kosong
		1804	42411	0,0425361	Kosong
		1784	42411	0,0420646	Kosong
		1773	42411	0,0418052	Kosong
	 <p>Area 2</p>	4980	39396	0,126409	Kosong
		4686	39396	0,118946	Kosong
		4664	39396	0,118388	Kosong
		4578	39396	0,116205	Kosong
		4608	39396	0,116966	Kosong
	 <p>Area 3</p>	21392	39396	0,542999	Terisi
		20541	39396	0,519114	Terisi
		20031	39396	0,508453	Terisi
		19902	39396	0,505178	Terisi
		19705	39396	0,500178	Terisi
	 <p>Area 1</p>	19174	42411	0,4521	Terisi
		18865	42411	0,444814	Terisi
		18911	42411	0,445898	Terisi
		18897	42411	0,445568	Terisi
		18794	42411	0,44314	Terisi
	 <p>Area 2</p>	20243	39396	0,513834	Terisi
		19768	39396	0,501777	Terisi
		19558	39396	0,496446	Terisi
		19512	39396	0,495279	Terisi
		19440	39396	0,493451	Terisi
	 <p>Area 3</p>	21319	39396	0,541146	Terisi
		20664	39396	0,52452	Terisi
		20338	39396	0,516245	Terisi
		20314	39396	0,515636	Terisi
		20295	39396	0,510382	Terisi

	18994	42411	0,447856	Terisi	
	18881	42411	0,445191	Terisi	
	18901	42411	0,445663	Terisi	
	18911	42411	0,445898	Terisi	
	18866	42411	0,444837	Terisi	
	Area 1				
		20188	39396	0,512438	Terisi
		19344	39396	0,491014	Terisi
		19371	39396	0,4917	Terisi
		19610	39396	0,497766	Terisi
19441		39396	0,492715	Terisi	
Area 2					
	8983	39396	0,228018	Kosong	
	6684	39396	0,169662	Kosong	
	6581	39396	0,167047	Kosong	
	6678	39396	0,16951	Kosong	
	6432	39396	0,163265	Kosong	
Area 3					
	1836	42411	0,0432907	Kosong	
	1804	42411	0,0425361	Kosong	
	1877	42411	0,0442574	Kosong	
	1856	42411	0,0437622	Kosong	
	1839	42411	0,0433614	Kosong	
Area 1					
	22395	39396	0,568459	Terisi	
	22263	39396	0,565108	Terisi	
	1839	39396	0,56554	Terisi	
	22735	39396	0,577089	Terisi	
	22678	39396	0,575642	Terisi	
Area 2					
	9499	39396	0,241116	Kosong	
	9473	39396	0,240456	Kosong	
	9500	39396	0,241141	Kosong	
	9499	39396	0,241116	Kosong	
	9473	39396	0,240456	Kosong	
Area 3					

Pada Tabel 4.2 merupakan hasil pengujian yang dilakukan dengan berbagai kondisi parkir. Pada tabel tersebut terdapat area, nilai total *white pixel* , nilai total pixel, nilai presentase (hasil) dan status. Untuk penjelasan yang dimaksud area di Tabel 4.2 adalah hasil akhir citra masing-masing area (slot tempat parkir) yang diperoleh dari proses *Background Subtraction* yang telah dikonversi ke *grayscale* dan biner. Sedangkan yang dimaksud total *white pixel* merupakan nilai hasil penghitungan jumlah keseluruhan *pixel* berwarna putih pada masing-masing area. Untuk penjelasan total *pixel* adalah jumlah *pixel* secara keseluruhan pada masing-masing area. Sedangkan yang dimaksud nilai presentase (hasil) adalah hasil pembagian total *white pixel* dengan total *pixel*. Pada tabel tersebut juga terdapat status slot parkir tersebut kosong atau terisi dari hasil perhitungannya.

Pada sistem ini telah ditentukan batas minimal presentase warna *pixel* putih adalah 35% dari nilai total pixel keseluruhan. Nilai 35% sebagai batas minimal presentase warna *pixel* putih didapatkan dari hasil beberapa kali pengujian (pengujian untuk mendapatkan hasil yang akurat). Sehingga apabila pada area tersebut terdapat objek yang mempunyai presentase pixel putih 35% atau lebih maka area tersebut dinyatakan terdapat mobil. Sebaliknya apabila presentase pixel putih dibawah 35% maka dinyatakan area tersebut masih kosong.

Pada Tabel 4.2 terlihat bahwa ketika proses update citra dan ketika ada benda lain atau manusia yang berada di tempat parkir yang kosong (pada 2 kondisi citra yang terakhir pada Tabel 4.2 ) maka tidak dianggap mobil dan tetap dianggap kosong tempat tersebut. Hal tersebut terlihat dari nilai presentase pixel

putih pada area yang terdapat benda lain yang berada dibawah batas minimal yang telah ditentukan yaitu 0,35 atau 35 %.

#### **4.9 Evaluasi Sistem Secara Keseluruhan**

Pengujian terakhir adalah pengujian sistem secara keseluruhan dari awal hingga akhir, dimana pengujian dilakukan dengan menjalankan aplikasi secara keseluruhan. Mengubah-ubah kombinasi posisi mobil pada miniatur, kemudian ketika waktu sistem sudah 5 menit atau kelipatannya maka secara otomatis list nomor parkir yang kosong akan diperbarui sesuai. Dan secara bersamaan juga, Webcam mengirimkan citra yang disorot *streaming* dan ditampilkan pada layar PC.

##### **4.9.1 Tujuan**

Tujuan evaluasi sistem ini adalah untuk mengetahui sistem pada aplikasi apakah sudah dapat berjalan sesuai dengan yang diharapkan. Dimulai dari memperbarui citra update, hingga output berupa list nomor parkir yang kosong dapat sesuai.

##### **4.9.2 Alat yang Digunakan**

1. Miniatur tempat parkir mobil
2. Microsoft Visual C++ 2008
3. *Personal Computer* (PC)
4. Webcam

##### **4.9.3 Prosedur Pengujian**

1. Menghubungkan *Webcam* dengan PC
2. Menjalankan program *console* pada Microsoft Visual C++ 2008

3. Menjalankan aplikasi program secara keseluruhan
4. Memberikan inputan perubahan kondisi parkir
5. Melihat list nomor parkir yang kosong
6. Melihat pada layar PC, apakah citra streaming dari *Webcam* telah ditampilkan pada layar PC

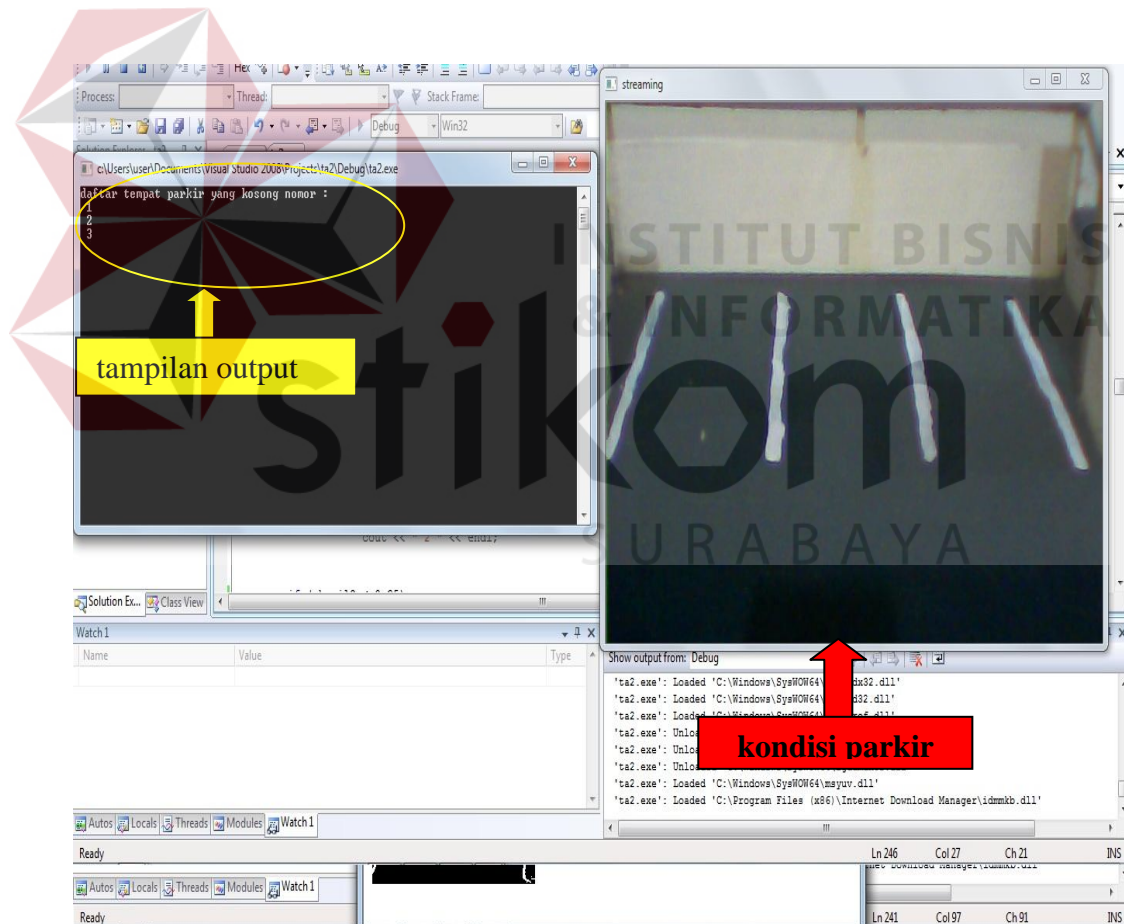
#### 4.9.4 Hasil Pengujian

Setelah melalui seluruh prosedur pengujian diatas didapatkan hasil tingkat keberhasilan 100% dari 20 kali pengujian, ketika diberikan input perubahan kondisi parkir pada *Webcam* maka output yang ditampilkan telah sesuai dengan kondisi saat itu. Tabel 4.3 merupakan hasil pendeteksian tahan parkir dengan berbagai kondisi.

Tabel 4.3 Hasil Pendeteksian Tempat Parkir

kondisi	Area 1	Area 2	Area 3
Kosong semua	Kosong	Kosong	Kosong
Kosong nomor 2 dan 3	Terisi	Kosong	Kosong
Kosong nomor 3	Terisi	Terisi	Kosong
Kosong nomor 1	Kosong	Terisi	Terisi
Kosong nomor 2	Terisi	Kosong	Terisi
Penuh	Terisi	Terisi	Terisi
Kosong nomor 1 dan 2	Kosong	Kosong	Terisi
Kosong nomor 1 dan 3	Kosong	Terisi	Kosong
Kosong terdapat benda selain mobil (misal pada area 2)	Kosong	Kosong	Kosong

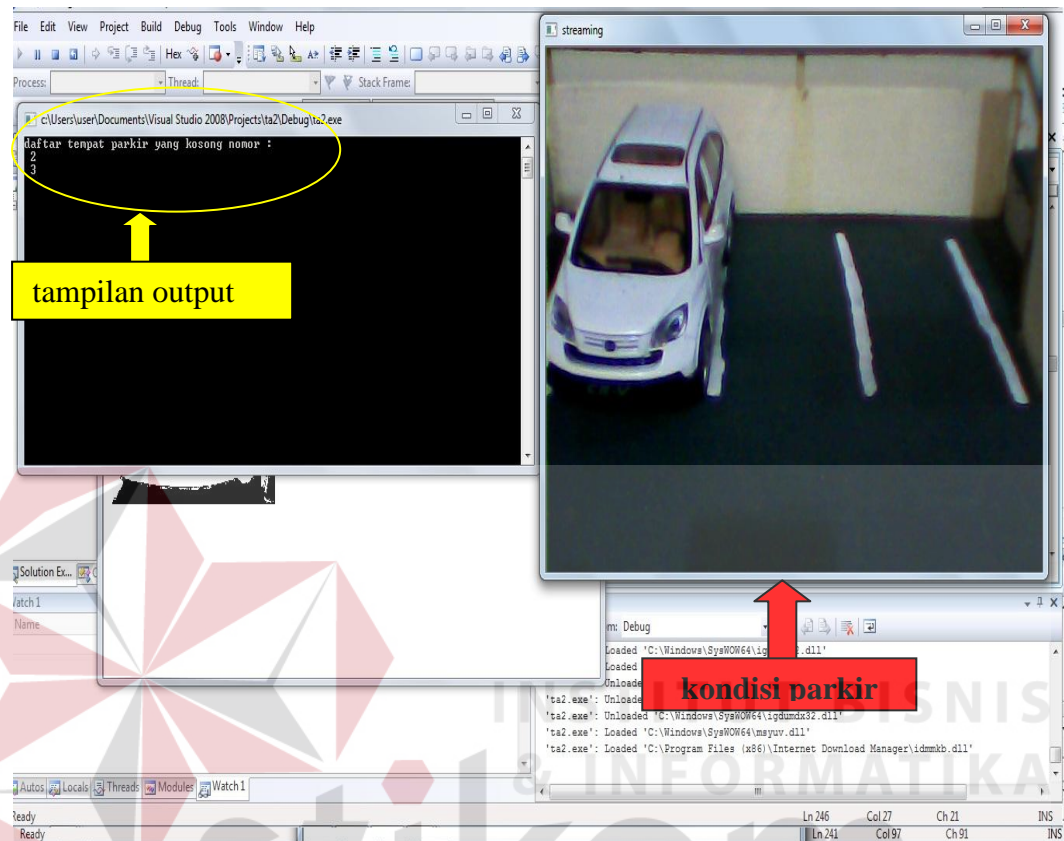
Tabel 4.3 menunjukkan bahwa program telah mampu mengenali objek mobil dan dapat membedakan area parkir yang kosong serta menghasilkan data yang valid. Jadi ketika diberi beberapa kondisi seperti pada tabel maka status dari masing-masing slot tempat parkir (status area 1 mewakili slot nomor 1, area 2 untuk slot nomor 2 dan area 3 untuk slot nomor 3). Ketika diberi kondisi maka dari masing-masing area akan menunjukkan statusnya kosong atau terisi yang sesuai dengan kondisi. Untuk memperjelas Tabel 4.3 disertakan juga hasil capture pengujian pada Gambar 4.15 sampai dengan Gambar 4.23 seperti berikut.



Gambar 4.15 Hasil Pendeteksi Tempat Kosong Semua

Pada Gambar 4.15 ketika sistem diberi kondisi slot/tempat parkir kosong semua (seperti ditunjukkan panah warna merah) maka sistem akan mendeteksi

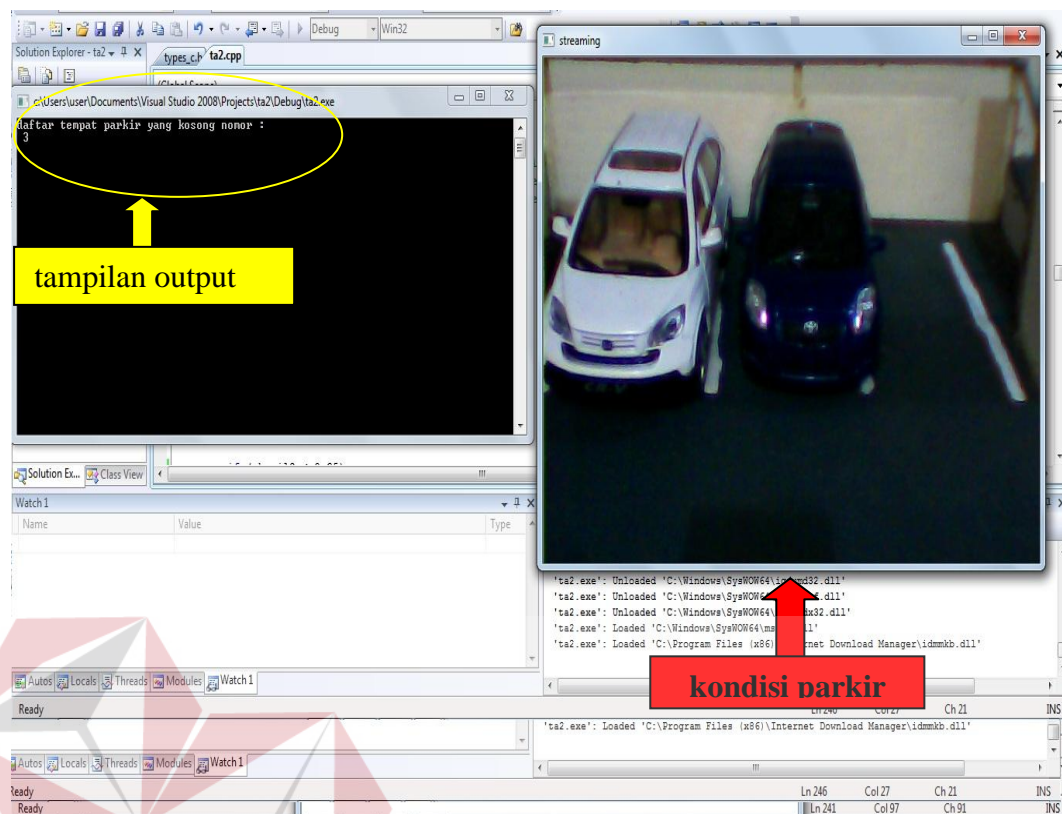
dan menampilkan nomor parkir yang statusnya kosong adalah 1,2 dan 3 (seperti ditunjukkan panah warna kuning).



Gambar 4.16 Hasil Pendeteksi Kosong Nomor 2 dan 3

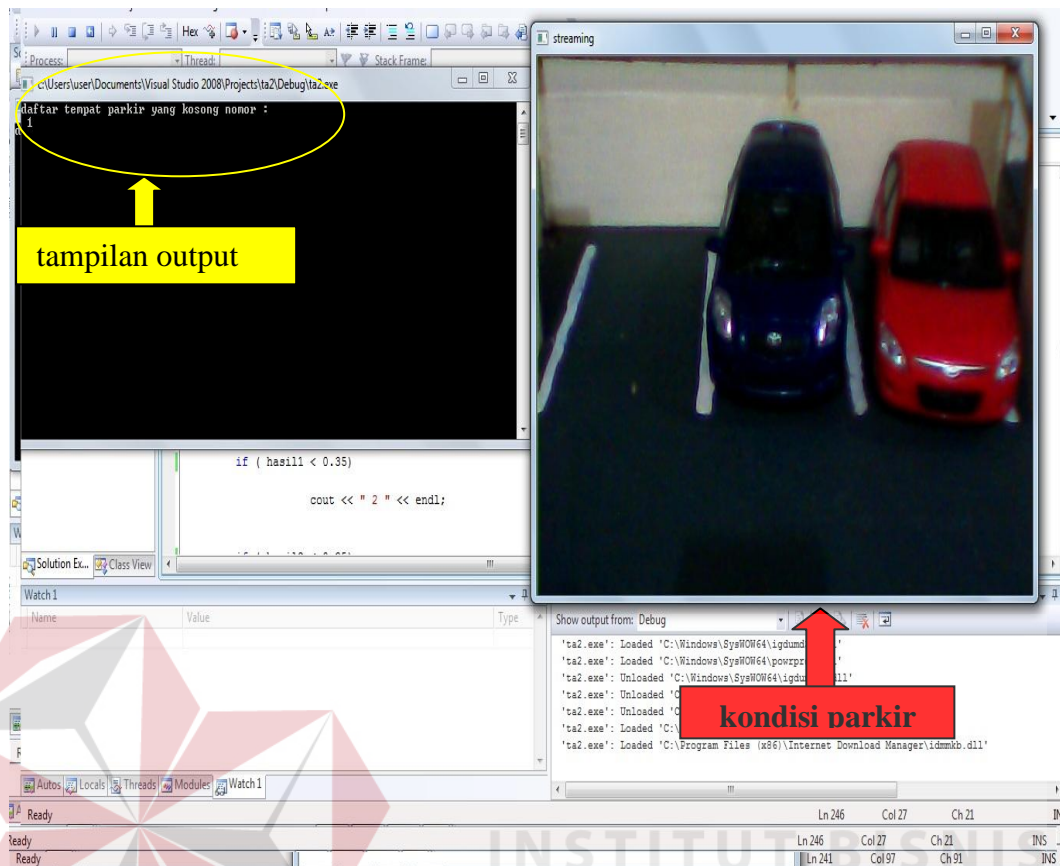
Pada Gambar 4.16 ketika sistem diberi kondisi slot/tempat parkir kosong pada nomor 2 dan 3 (seperti ditunjukkan panah warna merah) maka sistem akan mendeteksi dan menampilkan nomor parkir yang statusnya kosong adalah 2 dan 3 (seperti ditunjukkan panah warna kuning).





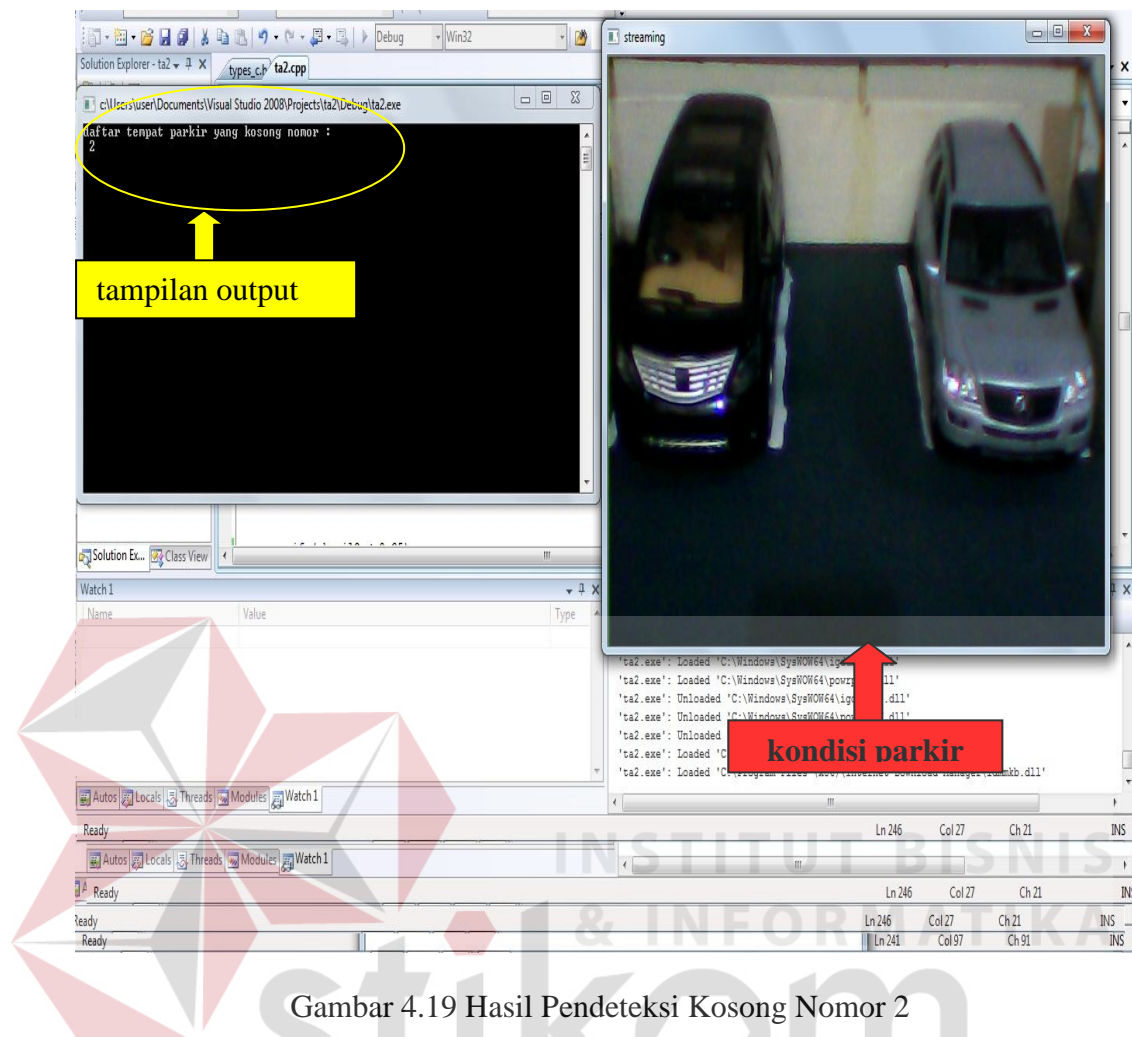
Gambar 4.17 Hasil Pendeteksi Kosong Nomor 3

Pada Gambar 4.17 ketika sistem diberi kondisi slot/tempat parkir kosong pada nomor 3 (seperti ditunjukkan panah warna merah) maka sistem akan mendeteksi dan menampilkan nomor parkir yang statusnya kosong adalah 3 saja (seperti ditunjukkan panah warna kuning).



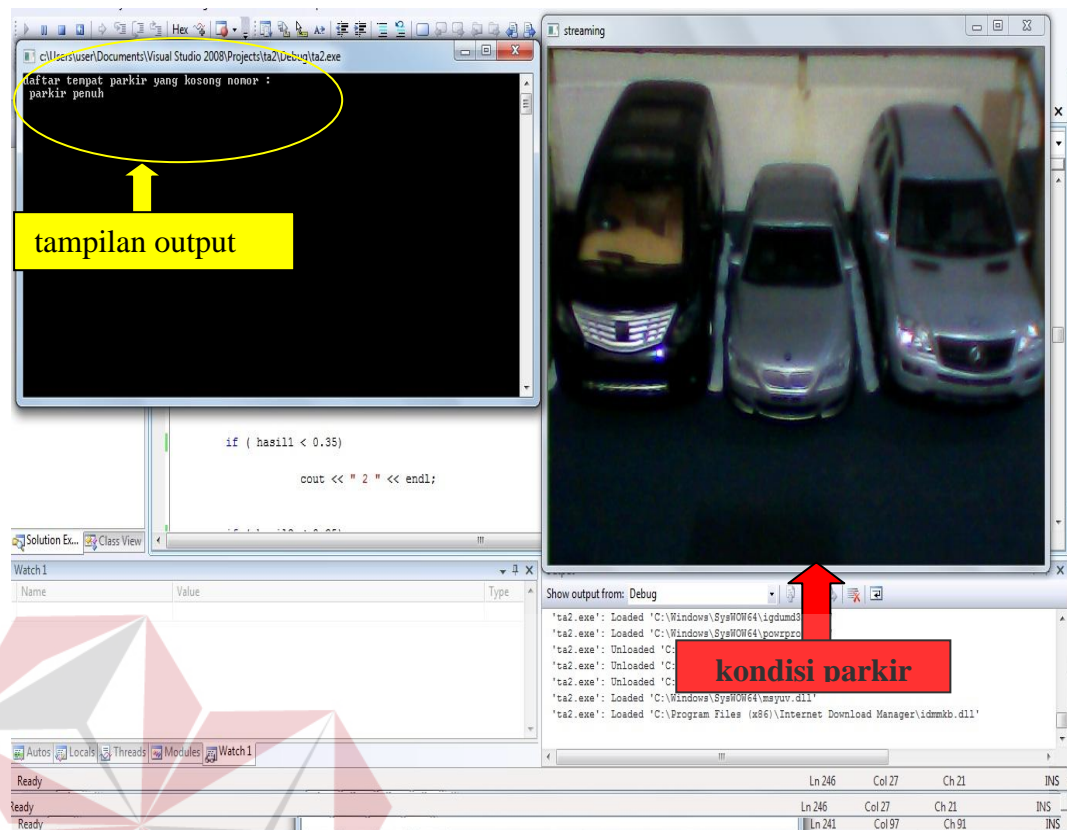
Gambar 4.18 Hasil Pendeteksi Kosong Nomor 1

Pada Gambar 4.18 ketika sistem diberi kondisi slot/tempat parkir kosong pada nomor 1 (seperti ditunjukkan panah warna merah) maka sistem akan mendeteksi dan menampilkan nomor parkir yang statusnya kosong adalah 1 saja (seperti ditunjukkan panah warna kuning).



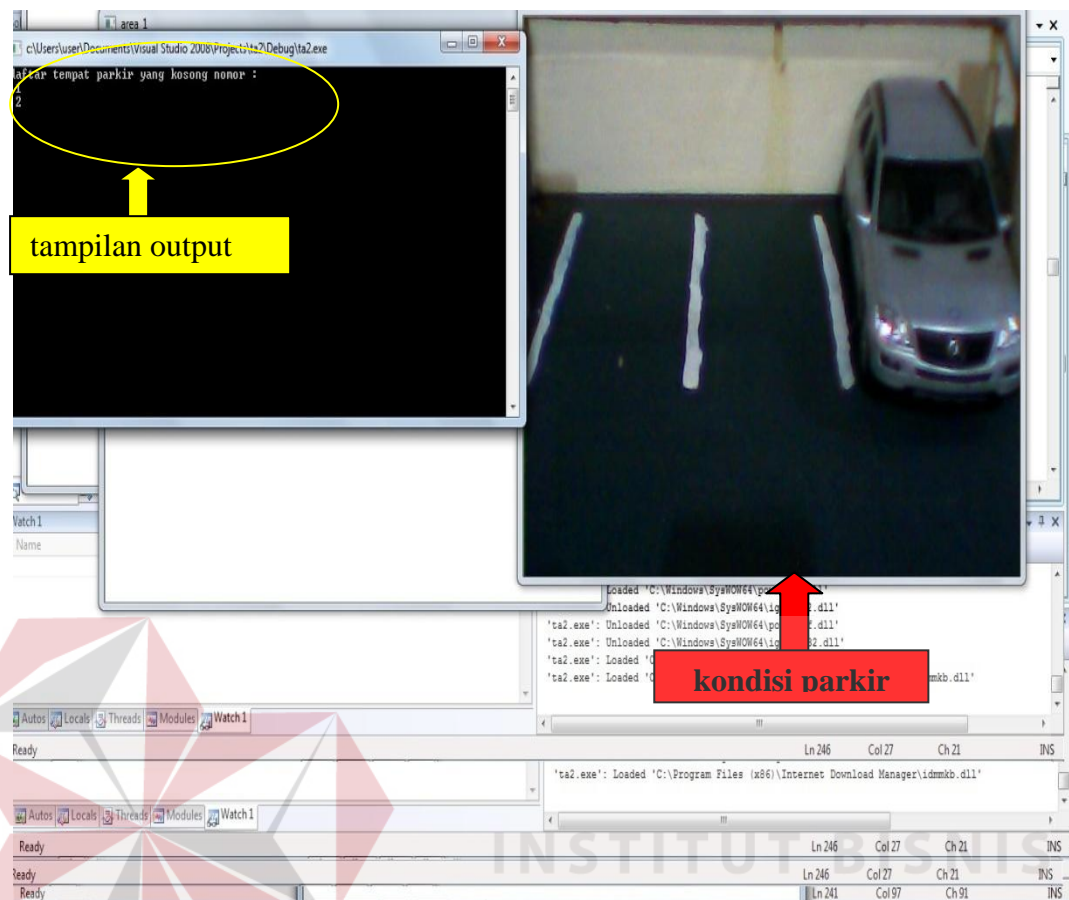
Gambar 4.19 Hasil Pendeteksi Kosong Nomor 2

Pada Gambar 4.19 ketika sistem diberi kondisi slot/tempat parkir kosong pada nomor 2 (seperti ditunjukkan panah warna merah) maka sistem akan mendeteksi dan menampilkan nomor parkir yang statusnya kosong adalah 2 saja (seperti ditunjukkan panah warna kuning).



Gambar 4.20 Hasil Pendeteksi Parkir Penuh

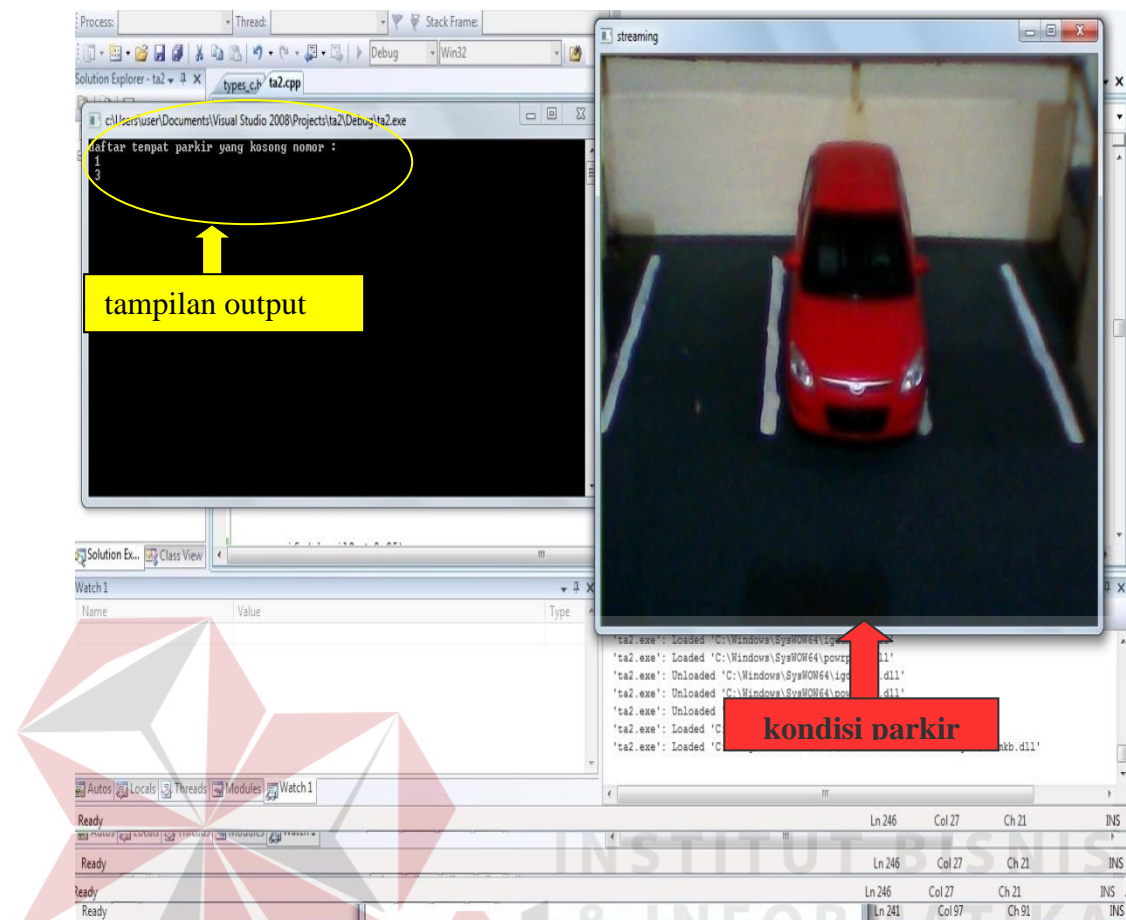
Pada Gambar 4.20 ketika sistem diberi kondisi slot/tempat parkir yang terisi mobil semua (seperti ditunjukkan panah warna merah) maka sistem akan mendeteksi dan menginformasikan bahwa tempat parkir penuh (seperti ditunjukkan panah warna kuning).



Gambar 4.21 Hasil Pendeteksi Kosong Nomor 1 dan 2

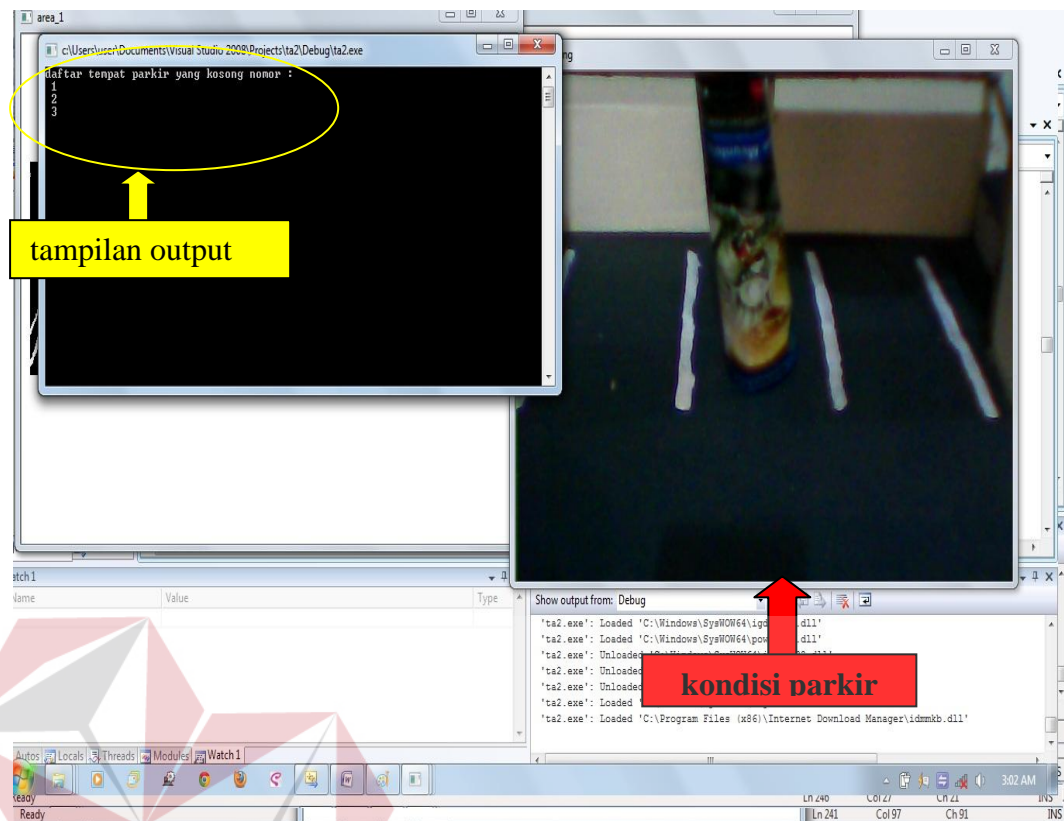
Pada Gambar 4.21 ketika sistem diberi kondisi slot/tempat parkir kosong pada nomor 1 dan 2 (seperti ditunjukkan panah warna merah) maka sistem akan mendeteksi dan menampilkan nomor parkir yang statusnya kosong adalah 1 dan 2 (seperti ditunjukkan panah warna kuning).





Gambar 4.22 Hasil Pendeteksi Kosong Nomor 1 dan 3

Pada Gambar 4.22 ketika sistem diberi kondisi slot/tempat parkir kosong pada nomor 1 dan 3 (seperti ditunjukkan panah warna merah) maka sistem akan mendeteksi dan menampilkan nomor parkir yang statusnya kosong adalah 1 dan 3 (seperti ditunjukkan panah warna kuning).



Gambar 4.23 Hasil Pendeteksi Benda Selain Mobil

Pada Gambar 4.23 ketika sistem diberi kondisi slot/tempat parkir kosong semua dan diletakkan benda lain pada slot nomor 2 yang dimisalkan manusia berdiri di slot tersebut (seperti ditunjukkan panah warna merah) maka sistem akan tetap mendeteksi slot yang terdapat manusia sebagai slot yang kosong dan menampilkan nomor parkir yang statusnya kosong adalah 1,2 dan 3 (seperti ditunjukkan panah warna kuning).

Dari hasil-hasil pengujian seperti pada Gambar 4.15 sampai dengan Gambar 4.23 , input perubahan kondisi parkir dari yang kondisi kosong, penuh, berbagai kombinasi posisi mobil dan ketika terdapat benda selain mobil program telah mampu mendeteksi sesuai kondisi.