

**Deteksi Lahan Parkir Kosong di UKDW Menggunakan Perbandingan Posisi  
Objek**

**Tugas Akhir**



**Diajukan oleh :**

**Billy Fanino B.**

**71130126**

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI  
UNIVERSITAS KRISTEN DUTA WACANA  
YOGYAKARTA  
2016**

**PERNYATAAN KEASLIAN SKRIPSI**

## **HALAMAN PERSETUJUAN**

## **HALAMAN PENGESAHAN**

## **KATA PENGANTAR**

Puji dan syukur penulis panjatkan kepada Tuhan yang Maha Esa karena telah melimpahkan rahmat dan kasih karunia-Nya, sehingga penulis dapat menyelesaikan laporan tugas akhir ini yang berjudul “Deteksi Lahan Parkir Kosong di UKDW Menggunakan Perbandingan Posisi Objek” dengan baik dan tepat pada waktunya. Penulisan laporan ini merupakan salah satu syarat untuk mencapai gelar sarjana (S1) pada Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Duta Wacana Yogyakarta.

Dalam menyelesaikan Tugas Akhir ini, penulis telah banyak menerima bantuan berupa motivasi, bimbingan dan masukan dari berbagai pihak baik secara langsung maupun tidak langsung. Untuk itu penulis ingin mengucapkan terima kasih kepada semua pihak yang telah membantu penulis dalam penggeraan Tugas Akhir ini.

Penulisan Tugas Akhir ini diajukan sebagai salah satu syarat untuk memenuhi syarat memperoleh gelar Sarjana Komputer. Penulis menyadari bahwa Tugas Akhir ini masih jauh dari kata sempurna. Oleh karena itu, penulis sangat menghargai dan menerima masukan dan kritik yang membangun dari pembaca.

Akhir kata penulis memohon maaf bila ada kata-kata yang kurang berkenan dan kesalahan selama penyusunan Tugas Akhir. Penulis berharap Tugas Akhir yang telah dibuat oleh penulis dapat bermanfaat bagi para pembaca.

Yogyakarta, 12 Mei 2017

Penulis

## **INTISARI**

### **DETEKSI LAHAN PARKIR KOSONG DI UKDW MENGGUNAKAN PERBANDINGAN POSISI OBJEK**

Saat ini, pegawai dan mahasiswa di Universitas Kristen Duta Wacana mengalami masalah dalam mencari tempat untuk memparkirkan kendaraannya, terutama untuk mobil. Hal ini dikarenakan sempitnya lahan parkir dan semakin banyak pegawai dan mahasiswa Universitas yang pergi ke kampus dengan menggunakan kendaraan roda 4. Masalah yang muncul pada parkiran mobil adalah waktu yang dihabiskan untuk mencari ketersediaan lahan parkir. Pengendara akan mencari lahan parkir sampai mereka mendapatkan lahan parkir yang tersedia. Tidak adanya sistem informasi parkir otomatis membuat pengendara mobil harus mencari sendiri lahan parkir yang kosong, dan pengendara juga tidak mengetahui apakah masih terdapat tempat kosong atau tidak ketika pengendara mobil baru memasuki area parkir.

Dari solusi di atas masih ditemukan beberapa kendala. Dengan berkembangnya kemampuan komputer untuk melakukan *image processing*, permasalahan ini dapat diselesaikan dengan menerapkan *Background Subtraction*. Aplikasi yang melakukan *image processing* untuk memantau kondisi lahan parkir apakah tempat tersebut tersedia atau tidak berdasarkan kondisi terakhir lahan parkir.

Penelitian ini akan membuat prototype aplikasi untuk mengolah gambar dari kondisi lahan parkir di Universitas Kristen Duta Wacana menggunakan metode *Background Subtraction* sehingga diperoleh hasil apakah ada lahan parkir yang tersedia di tempat parkir. Berdasarkan hasil pengujian dan analisis, deteksi lahan parkir kosong di gedung B1 UKDW dengan 65 data uji keadaan parkir dapat menghasilkan presentase keberhasilan sebesar 93.69%.

Kata Kunci: *Background Subtraction*, Pengolahan Citra.

## DAFTAR ISI

PERNYATAAN KEASLIAN SKRIPSI.....	iii
HALAMAN PERSETUJUAN.....	iv
HALAMAN PENGESAHAN .....	v
KATA PENGANTAR .....	vi
INTISARI .....	vii
DAFTAR ISI.....	viii
DAFTAR GAMBAR .....	x
DAFTAR TABEL.....	xi
DAFTAR LAMPIRAN.....	xii
BAB 1 .....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Batasan Masalah .....	2
1.4 Tujuan Penelitian.....	2
1.5 Metodologi Penelitian.....	3
BAB 2 .....	5
2.1 Tinjauan Pustaka.....	5
2.2 Landasan Teori .....	6
2.2.1     Citra Biner.....	6
2.2.2     Citra <i>Grayscale</i> .....	7
2.2.3     Pengolahan Citra.....	8
2.2.4     Confusion Matrix .....	11
BAB 3 .....	13
3.1.3. Spesifikasi Sistem.....	14
3.2 Rancangan Fungsionalitas .....	15
3.2.1.1 Flowchart Sistem Secara Umum.....	15
3.2.2.2 Flowchart Proses Background Subtraction .....	16
3.3 Metode Pengujian .....	17
3.4 Metode Evaluasi .....	18

3.5	Perancangan Struktur Data .....	19
3.6	Perancangan Antar Muka.....	19
BAB 4 .....		21
4.1	Implementasi Sistem.....	21
4.1.1	Implementasi Antarmuka.....	21
4.1.2	Implementasi Algoritma .....	22
4.1	Implementasi Background Subtraction.....	23
4.1.2	Implementasi Connected Component Labelling.....	24
4.2	Pengujian dan Analisis Sistem.....	25
4.2.1.	Pengujian Sistem.....	25
4.2.1.1	Penentuan Threshold Background Subtraction.....	25
4.2.1.2	Pengujian Menggunakan Threshold 1.....	25
4.2.1.3.	Pengujian Menggunakan Threshold 2.....	28
4.2.1.4	Pengujian Menggunakan Threshold 3.....	30
4.2.3	Analisis Sistem .....	33
BAB 5 .....		36
5.1	Kesimpulan .....	36
5.2	Saran .....	36
DAFTAR PUSTAKA .....		38
LAMPIRAN .....		40

## **DAFTAR GAMBAR**

Gambar 2.1 kiri : citra RGB , kanan : citra biner.....	6
Gambar 2.2 citra berwarna.....	7
Gambar 2.3 citra gray scale.....	8
Gambar 2.4 citra background.....	9
Gambar 2.4 citra keadaan sekarang.....	9
Gambar 2.6 contoh Background Subtraction.....	10
Gambar 2.7 operator untuk scanning.....	10
Gambar 2.8. Hasil dari labeling .....	11
Gambar 3.1 Blok Diagram Sistem.....	13
Gambar 3.2 Flowchart Sistem pencarian lahan parkir kosong di UKDW secara umum.....	15
Gambar 3.3 Flowchart Background Subtraction.....	17
Gambar 3.4 Perancangan antar muka sistem.....	20
Gambar 4.1 Antarmuka sistem.....	21
Gambar 4.2 Antarmuka Sistem setelah dijalankan.....	22
Gambar 4.3 pengubahan citra awal menjadi citra grayscale.....	23
Gambar 4.4 Hasil Background Subtraction .....	23
Gambar 4.5 Hasil pemotongan dari Background Subtraction .....	24
Gambar 4.6 hasil implementasi Connected Component Labelling .....	24
Gambar 4.7 Hasil akhir sistem .....	25
Gambar 4.8 contoh gambar .....	32
Gambar 4.9 contoh gambar .....	33
Gambar 4.10 contoh gambar .....	34

## **DAFTAR TABEL**

Tabel 2.1 Tabel confusion matrix .....	11
Tabel 3.1 Confusion matrix .....	18
Tabel 4.1 Hasil uji sistem menggunakan threshold 1.....	26
Tabel 4.2 Hasil uji sistem menggunakan threshold 2.....	28
Tabel 4.3 Hasil Uji sistem menggunakan threshold 3.....	31

## **DAFTAR LAMPIRAN**

Lampiran A: Citra Pengujian

Lampiran B: Data.txt

Lampiran C: Source Code

Lampiran D: Dokumen

## **BAB 1**

### **PENDAHULUAN**

#### **1.1 Latar Belakang**

Saat ini, pegawai dan mahasiswa di Universitas Kristen Duta Wacana mengalami masalah dalam mencari tempat untuk memparkirkan kendaraannya, terutama untuk mobil. Hal ini dikarenakan sempitnya lahan parkir dan semakin banyak pegawai dan mahasiswa Universitas yang pergi ke kampus dengan menggunakan kendaraan roda 4.

Masalah yang muncul pada parkiran mobil adalah waktu yang dihabiskan untuk mencari ketersediaan lahan parkir. Pengendara akan mencari lahan parkir sampai mereka mendapatkan lahan parkir yang tersedia. Tidak adanya sistem informasi parkir otomatis membuat pengendara mobil harus mencari sendiri lahan parkir yang kosong, dan pengendara juga tidak mengetahui apakah masih terdapat tempat kosong atau tidak ketika pengendara mobil baru memasuki area parkir. Menurut Choudekar (2011) Tempat parkir mobil adalah objek penting bagi pengendara dan lalu lintas. Dengan masalah kemacetan lalu lintas kota dan semakin berkurangnya tempat, tempat parkir perlu dilengkapi dengan informasi parkir otomatis dan sistem panduan parkir.

Masalah seperti ini dapat diselesaikan dengan memasang sensor seperti inframerah disetiap bagian lahan parkir, akan tetapi cara ini memerlukan penataan ulang lahan parkir sehingga cara ini kurang efisien. Selain itu jumlah alat sensor inframerah yang dibutuhkan sejumlah lahan parkir yang ada, jadi ketika lahan parkir berkapasitas 100 mobil, maka jumlah sensor inframerah yang dibutuhkan juga sejumlah 100 buah. (Idris, dkk, 2009)

Dari solusi di atas masih ditemukan beberapa kendala. Dengan berkembangnya kemampuan komputer untuk melakukan *image processing*, permasalahan ini dapat diselesaikan dengan menerapkan *Background Subtraction*. Aplikasi yang melakukan *image processing* untuk memantau kondisi lahan parkir

apakah tempat tersebut tersedia atau tidak berdasarkan kondisi terakhir lahan parkir.

Penelitian ini akan membuat prototype aplikasi untuk mengolah gambar dari kondisi lahan parkir di Universitas Kristen Duta Wacana sehingga diperoleh hasil apakah ada lahan parkir yang tersedia di tempat parkir dan di mana letak lahan parkir tersebut berada, sehingga nantinya pengendara bisa langsung diarahkan menuju lahan parkir yang tersedia dan diharapkan waktu pencarian lahan parkir oleh pengendara di Universitas Kristen Duta Wacana menjadi lebih singkat daripada sebelumnya.

## **1.2 Rumusan Masalah**

Rumusan masalah dalam penelitian ini adalah bagaimana implementasi dan tingkat keakuratan metode *Background Subtraction* untuk mencari ketersediaan lahan parkir mobil di UKDW.

## **1.3 Batasan Masalah**

Penelitian yang akan dilakukan memiliki batasan sebagai berikut:

1. Implementasi akan dilakukan di *basement 1* gedung Agape Universitas Kristen Duta Wacana khususnya untuk mobil.
2. Pengujian dilakukan pada jam 07.30 – 15.00 saat aktivitas di tempat parkir sedang banyak

## **1.4 Tujuan Penelitian**

Tujuan dari penelitian ini adalah untuk membuat aplikasi berbasis sensor kamera yang dapat mencari ketersediaan lahan parkir mobil di Universitas Kristen Duta Wacana dengan menggunakan metode *background subtraction*.

## **1.5 Metodologi Penelitian**

Metodologi yang akan digunakan dalam penelitian guna mencari lahan parkir kosong di Universitas Kristen Duta Wacana adalah sebagai berikut:

1) Studi Pustaka

Studi Pustaka dilakukan dengan mempelajari teori-teori yang berkaitan dengan Pengolahan Citra Digital, Segmentasi Citra, dan *Computer Vision* melalui beberapa buku, jurnal, artikel, dan bahan lain yang mendukung.

2) Penentuan Standar Struktur Data

Menentukan standar struktur data dan format data ketersediaan lahan parkir. Standar struktur dan format data tersebut akan diterapkan pada Aplikasi pencari lahan parkir kosong pada penelitian ini, Aplikasi pendekripsi tempat parkir, dan data yang disimpan pada server.

3) Pengumpulan Data

Pengumpulan data dilakukan dengan cara memasang kamera di lahan parkir mobil Universitas Kristen Duta Wacana. Yang pada saat tertentu akan mengambil data citra sebanyak 3-6 citra yang nantinya akan dikirim ke server.

4) Pembuatan Sistem

Pembuatan sistem pengolahan citra akan dibuat dengan bahasa pemrogramman C++.

5) Metode Pengujian

Pengujian dilakukan dengan memasukan data citra ke server, lalu server akan mengolahnya dan memberikan hasil. Jika hasil yang diberikan sudah tepat dan sesuai keadaan di lahan parkir maka pengujian dianggap berhasil.

## **1.6 Sistematika Penulisan**

BAB 1 PENDAHULUAN membahas tentang latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, metode penelitian serta sistematika penulisan penelitian ini.

BAB 2 TINJAUAN PUSTAKA. Berisi tinjauan pustaka dan landasan teori yang menjadi dasar penelitian ini. Pada bab ini dijelaskan secara detail seluruh informasi dan studi pustaka yang diperoleh oleh peneliti berkatian dengan

penelitian ini. Bab ini akan menjadi acuan peneliti untuk melakukan tahapan penelitian.

**BAB 3 ANALISIS DAN PERANCANGAN PENELITIAN.** Berisi rancangan sistem *monitoring* ketersediaan jumlah parkir mobil, alur kerja sistem, hardware dan software yang dibutuhkan untuk mendukung penelitian.

**BAB 4 IMPLEMENTASI SISTEM DAN ANALISIS SISTEM.** Berisi uraian detail implementasi sistem dan uraian detail hasil analisis sistem yang didapatkan dari hasil uji coba yang dilakukan.

**BAB 5 SARAN DAN KESIMPULAN.** Berisi kesimpulan dari hasil analisis yang didapat, saran dan rekomendasi yang dapat dilakukan untuk penelitian lebih lanjut.

## **BAB 2**

### **LANDASAN TEORI**

#### **2.1 Tinjauan Pustaka**

Penelitian serupa tentang sistem pencarian lahan parkir dan *object detection* sudah pernah dibuat sebelumnya, tetapi tempat penerapan dan metode yang digunakan berbeda – beda. Adapun penelitian sebelumnya yang berkaitan dengan pencarian lahan parkir yang pernah dibuat adalah sebagai berikut :

Penelitian G Nithinya & Kumar (2016) yang berjudul “*Design and Implementation of an Intelligent Parking Management System Using Image Processing*” dilakukan pendektsian kendaraan dan mengklasifikasikan ke tiga jenis yaitu bus, mobil, dan kendaraan roda 2 dengan menggunakan teknik *grayscale* pada citra yang didapat, lalu mengklasifikasikannya menggunakan metode ANN dan membandingkan data latih dengan data sample. Berdasarkan hasil yang didapat jika kendaraan yang terdeteksi adalah bus akan diarahkan ke parkiran barat, mobil ke arah timur dan kendaraan roda 2 akan diarahkan ke bagian utara.

Penelitian Yusnita dkk. (2012) yang berjudul “*Intelligent Parking Space Detection System Based on Image Processing*” dilakukan pendektsian lahan parkir menggunakan marker yang ditempelkan pada masing - masing slot lahan parkir yang ada, ketika ada yang mencari lahan parkir sistem akan mengambil gambar terakhir lahan parkir, lalu mengubahnya menjadi *grayscale*, menggunakan 4 morfologi dasar untuk *noise removal* lalu melakukan pencarian marker. Sistem akan mencatat berapa banyak tempat parkir yang tersedia.

Pada penelitian Vinay(2015) yang berjudul “*Object Tracking Using Background Subtraction Algorithm*” dilakukan pendektsian objek menggunakan *Background Subtraction*, menghilangkan citra *background* untuk mendapatkan citra *foreground* yang dapat disimpulkan adalah sebuah objek. Penelitian ini ditujukan untuk menghitung jumlah mobil di jalan raya, akurasi yang didapatkan dalam menghitung mobil mencapai 94%.

Penelitian Oji (2012) yang berjudul “*An Automatic Algorithm for Object Recognition and Detection Based on ASIFT*” dilakukan pendekripsi objek penggunaan metode *ASHIFT* untuk mendekripsi objek berdasarkan *online dataset* untuk mendekripsi objek pada gambar yang diambil. Presentase dari pendekripsi objek ini cukup tinggi dengan tingkat akurasi diatas 85% untuk setiap objek yang didekripsi keberadaannya.

Menarik adanya perbedaan yang ada pada penelitian sebelumnya. Gambar yang akan diolah akan diubah ke dalam bentuk *grayscale* untuk mempermudah komputer untuk mengolah citra. Penulis menggunakan *Background Subtraction* untuk mendekripsi citra *foreground*, menggunakan *Connected Component Labeling* untuk mendapatkan tiap – tiap objek yang ada dalam gambar, lalu menghitung jumlah node dalam tiap - tiap objek yang didapat guna mendapatkan area luas tiap objek, yang lalu akan digunakan untuk penentuan apakah objek tersebut berupa mobil atau tidak dengan melihat luas objek yang didapat, jika luas objek tersebut cukup besar, sistem akan menganggap objek tersebut adalah mobil.

## 2.2 Landasan Teori

### 2.2.1 Citra Biner

Menurut Kumar (2010) citra biner adalah citra dengan kemungkinan warna hitam dan putih saja, sekumpulan *array* yang hanya berisi 0 dan 1, mewakilkan sebagai hitam dan putih secara berurutan. Contoh perbedaan citra berwarna dan citra biner dapat dilihat pada gambar 2.1



Gambar 2.1 kiri : citra RGB , kanan : citra biner

### 2.2.2 Citra *Grayscale*

Menurut Kumar (2010) citra *gray scale* dikenal juga sebagai intensitas, dan *gray level image*. Memiliki jarak nilai dari 0 - 255 untuk kedalaman 8 bit, untuk *single* atau *double array* memiliki jarak 0-1, dan 0 - 65535 untuk kedalaman 16 bit. Berikut adalah persamaan untuk mengubah citra dari citra berwarna menjadi citra *gray scale*:

$$\text{Gray} = 0.229R + 0.587G + 0.114B \quad [2.1]$$

Dengan:

Gray = hasil citra *gray scale*

R = intensitas piksel merah

G = intensitas piksel hijau

B = intensitas piksel biru

Adapun contoh perbedaan gambar berwarna dan gambar *Gray scale* dapat dilihat pada gambar 2.2 dan 2.3



Gambar 2.2 citra berwarna



Gambar 2.3 citra *gray scale*

### 2.2.3 Pengolahan Citra

Pengolahan citra adalah suatu proses penting dalam mengenali suatu citra. Pengolahan citra biasa digunakan untuk pendekripsi objek, pengenalan objek dan bentuk. Dapat juga digunakan untuk menemukan objek pada citra lain, melacak dan mengenali objek menggunakan sebuah algoritma tertentu. (Oji, 2012)

Pengolahan citra sering dipakai untuk mengambil informasi tertentu dari sebuah citra input. Seperti mengenali wajah seseorang dari citra wajah. Mengambil fitur – fitur wajah yang kemudian dicocokan dengan data latih untuk dapat menentukan wajah tersebut cocok dengan fitur – fitur wajah yang ada pada sistem.

### 2.2.4 *Background Subtraction*

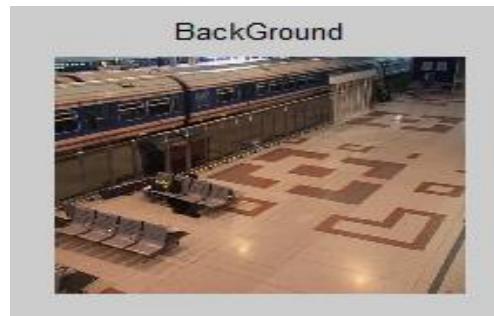
Menurut Vinay(2015) Salah satu pendekatan untuk mengidentifikasi objek yang bergerak dari rangkaian citra(video) adalah dengan melakukan background subtraction. Mendeteksi gerakan atau perubahan signifikan didalam rangkaian citra (video), ketika dibandingkan dengan data sample, dan untuk menghilangkan semua komponen non-signifikan. *Background subtraction* diterapkan pada banyak tempat, seperti sistem pengawasan (secara efektif hanya melihat objek yang bergerak).

Menurut Kuihe(2013) perbedaan frame adalah cara yang paling efektif untuk mendapatkan objek yang bergerak dengan melakukan pengurangan antara citra, 2 citra yang berdekatan yang diambil pada saat ini. Misalkan video frame pada waktu ke t dinyatakan dalam  $f(x,y,t)$ , maka frame selanjutnya dinyatakan dengan

$f(x,y,t+1)$ . Operasi citra biner hasil perbedaan frame dapat didefinisikan seperti berikut:

$$D(x, y, t + 1) = \begin{cases} 1 & \text{jika } |f(x, y, t) - f(x, y, t + 1)| > Th \\ 0 & \text{selain itu} \end{cases} \quad [2.2]$$

Dimana Th menjadi *threshold* untuk keputusan. Jika perbedaan nilai citra lebih besar daripada Th, maka letakkan titik itu sebagai piksel *foreground*, jika nilai perbedaan lebih kecil daripada Th, maka titik itu dianggap sebagai piksel *background*. Contoh proses *Background Subtraction* dapat dilihat pada gambar 2.4, 2.5 dan 2.6 dibawah ini.



Gambar 2.4 citra background

Sumber gambar : <https://www.pantechsolutions.net/blog/matlab-code-for-background-subtraction/>



Gambar 2.5 citra keadaan sekarang

Sumber gambar : <https://www.pantechsolutions.net/blog/matlab-code-for-background-subtraction/>  
Setelah gambar 2.4 dan 2.5 dimasukan kedalam fungsi background subtraction, maka hasil yang didapatkan dapat dilihat pada gambar 2.6



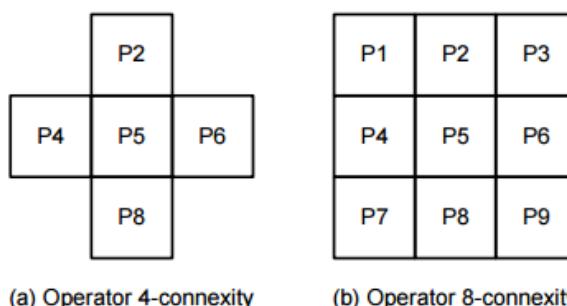
Gambar 2.6 contoh *Background Subtraction*

Sumber gambar : <https://www.pantechsolutions.net/blog/matlab-code-for-background-subtraction/>

### 2.2.5 *Connected Component labeling*

*Connected Component labeling* adalah suatu proses pemberian label yang sama pada sekumpulan pixel pembentuk objek yang saling berdekatan pada suatu citra. Objek yang berbeda memiliki label yang berbeda pula. Labeling termasuk pemrosesan citra tahap *inter- mediate level*. Labeling memiliki peran yang sangat penting pada pengolahan citra untuk mempermudah proses penganalisaan bentuk dan pengenalan pola pada tahap *high level*.

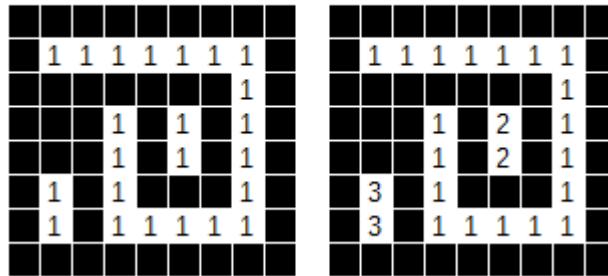
Proses labeling dilakukan dengan *scanning* terhadap seluruh piksel. Mencari bagian piksel yang saling terhubung(*connected*). Untuk mencari hubungan antar piksel dapat digunakan 2 operator lokal 4-*connexity* atau operator local 8-*connexity*. Bila menggunakan prinsip 4-*connexity* maka 2 pixel yang bersinggungan secara diagonal dianggap 2 objek, sedangkan pada 8- *connexity* dianggap 1 objek (Mozef, 2003). Untuk contoh 4-*connexity* dan 8-*connexity* dapat dilihat pada gambar 2.7.



Gambar 2.7 operator untuk scanning

Contoh citra sebelum dan sesudah di *labeling* dapat dilihat pada gambar 2.8

Thresholded image:      Labelled image:



Gambar 2.8. Hasil dari labeling

Sumber gambar: <https://www.codeproject.com/KB/recipes/825200/Sample.png>

#### 2.2.4 Confusion Matrix

Menurut Santra (2012) Confusion matrix mengandung informasi tentang data prediksi dan data aktual sebuah klasifikasi yang dilakukan oleh sistem. Performa sistem semacam itu biasa di evaluasi menggunakan data pada matriks. Untuk contoh tabel confusion matrix dapat dilihat pada tabel 2.1

Tabel 2.1 tabel confusion matrix

		Prediksi	
		Negative	Positive
Aktual	Negative	A	B
	Positive	C	D

Penjelasan dari tabel 2.1 adalah sebagai berikut:

1. A adalah jumlah prediksi negatif yang aktualnya adalah negatif.
2. B adalah jumlah prediksi positif yang aktualnya adalah negatif.
3. C adalah jumlah prediksi negatif aktualnya adalah positif.
4. D adalah jumlah prediksi positif yang aktualnya adalah positif.

Akurasi dari sistem didapat dari jumlah prediksi yang tepat dibagi dengan total data yang ada. Berikut adalah persamaan akurasi sistem:

$$AC = \frac{A+D}{A+B+C+D} \quad [2.3]$$

*True Positive* adalah jumlah kasus positif yang diprediksi benar. Berikut adalah persamaan tingkat *true positive*:

$$TP = \frac{D}{C+D} \quad [2.4]$$

*True Negative* adalah jumlah kasus negatif yang diprediksi benar. Berikut adalah persamaan tingkat *true negative*:

$$TN = \frac{A}{A+B} \quad [2.5]$$

## BAB 3

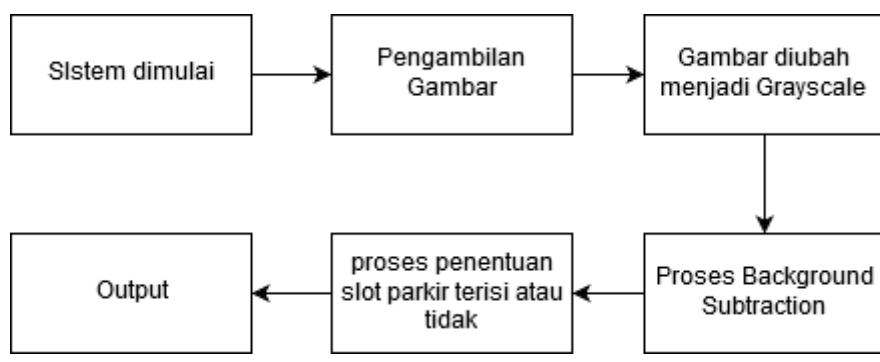
### PERANCANGAN SISTEM

#### 3.1 Deskripsi Umum

##### 3.1.1 Deskripsi Sistem

Sistem ini merupakan sebuah sistem yang menggabungkan sebuah embedded system yang dapat terhubung ke server menggunakan jaringan. Sistem ini mengambil beberapa frame yang akan dikirimkan ke server dan lalu diolah menggunakan *background subtraction* untuk mencari mana yang merupakan citra *background* dan mana yang merupakan objek bergerak atau *foreground*. Setelah itu objek bergerak akan dideteksi apakah objek tersebut merupakan sebuah mobil atau bukan, jika mobil yang dideteksi berada di lahan parkir, maka lahan parkir tersebut dianggap tidak tersedia/sudah terpakai, bila tidak ada objek yang berada di lahan parkir maka lahan parkir dianggap tersedia/belum dipakai.

##### 3.1.2 Blok Diagram Sistem



Gambar 3.1 Blok Diagram Sistem

##### 3.1.3 Kebutuhan Pengembangan Sistem

Perangkat yang digunakan untuk mengembangkan sistem ini adalah sebagai berikut:

1. Laptop dengan spesifikasi: Ram 8GB, Processor Intel i5, Sistem Operasi Windows 8.1.

### **3.1.4 Kebutuhan Perangkat Keras**

Perangkat keras yang dibutuhkan untuk menjalankan perangkat ini adalah sebagai berikut:

1. Tp-link camera NC200
2. UPS

### **3.1.5 Kebutuhan Perangkat Lunak**

Perangkat lunak yang dibutuhkan untuk menjalankan sistem ini adalah sebagai berikut:

1. Sistem Operasi Windows 8.1
2. Bahasa pemrogramman C++

### **3.1.6 Batasan Sistem**

Sistem yang dikembangkan memiliki batasan tertentu. Berikut adalah batasan sistem tersebut.

1. Input berupa gambar beresolusi 640 x 480.

### **3.1.3. Spesifikasi Sistem**

Sistem yang akan dibuat di dalam Tugas Akhir ini memiliki spesifikasi sebagai berikut:

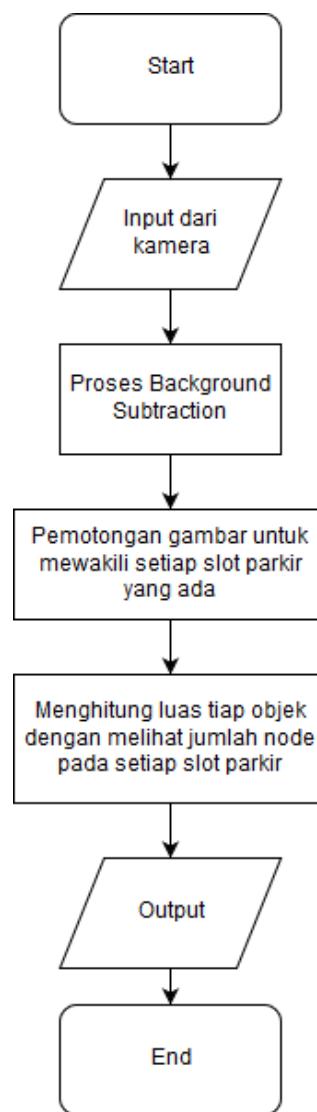
1. Algoritma yang akan digunakan adalah algoritma *Background Subtraction* dan *Connected Component Labelling*.
2. Input terdiri dari 2 macam, yaitu citra background dan citra keadaan aktual parkir.
3. Output berupa teks yang menuliskan keadaan tentang slot parkir (terisi / kosong).

## 3.2 Rancangan Fungsionalitas

### 3.2.1 Proses

#### 3.2.1.1 Flowchart Sistem Secara Umum

Flowchart berikut secara umum menggambarkan sistem pencari lahan parkir kosong di area *Basement 1* gedung Agape Universitas Kristen Duta Wacana yang akan dirancang. Flowchart rancangan sistem secara umum dapat dilihat pada gambar 3.2



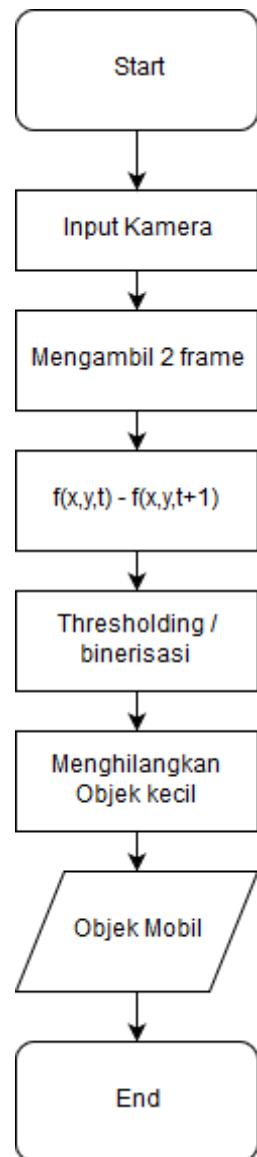
Gambar 3.2 Flowchart Sistem pencarian lahan parkir kosong di UKDW secara umum

Pada awalnya gambar keadaan tempat parkir diubah terlebih dahulu ke dalam bentuk gambar *grayscale* kemudian dilakukan proses *Background Subtraction* yang akan menghasilkan citra mobil didalamnya. Setelah itu proses penghilangan noise akan dilakukan untuk menghilangkan objek- objek kecil yang didapatkan dari proses *Background Subtraction*.

Setelah *pre-processing* selesai, gambar akan dipotong sejumlah slot parkir yang ada untuk masing – masing dilihat apakah ada objek didalamnya atau tidak dengan menghitung besaran luas objek pada masing – masing gambar dengan cara menghitung jumlah *connected component* pada suatu objek. Ketika luas objek sudah didapat maka objek akan diklasifikasi dengan melihat besar luas objek yang didapat, jika objek tersebut cukup besar luasnya, sistem akan menganggap objek tersebut adalah sebuah mobil.

### 3.2.2.2 Flowchart Proses Background Subtraction

Flowchart pendeksiian mobil (*Background Subtraction*) adalah seperti dijelaskan pada gambar 3.3



Gambar 3.3 Flowchart *Background Subtraction*

### 3.3 Metode Pengujian

Pengujian akan dilakukan di area parkir mobil B1 gedung Agape Universitas Kristen Duta Wacana, kamera akan diletakan di beberapa tempat untuk mengambil data pengujian pada pukul 07.30 – 15.00 saat aktivitas parkiran mobil sedang banyak. Pengujian juga melakukan peletakan kamera dengan posisi yang berbeda sehingga dapat diketahui posisi seperti apa yang terbaik untuk meletakan kamera di area B1 gedung Agape.

### 3.4 Metode Evaluasi

Metode Evaluasi yang digunakan untuk mengevaluasi data uji adalah *confusion matrix*. *Confusion matrix* adalah matrix yang berisikan informasi aktual dan prediksi dari hasil pengujian pada sistem. Tabel *confusion matrix* untuk menunjukkan hasil pengenalan lahan parkir apakah tersedia atau terisi dapat dilihat pada tabel 3.1.

Tabel 3.1 Confusion matrix

n (jumlah data)		Prediksi	
		Kosong	Terisi
Aktual	Kosong	TN	FN
	Terisi	FP	TP

Keterangan:

- True Negative (TN): jumlah data uji yang diprediksi sebagai slot parkir kosong dan data aktualnya sebagai slot parkir kosong.
- True Positive (TP): jumlah data uji yang diprediksi sebagai slot parkir terisi dan data aktualnya sebagai slot parkir terisi.
- False Positive (FP): jumlah data uji yang diprediksi sebagai slot parkir terisi, namun pada data aktualnya sebagai slot parkir kosong.
- False Negative (FN): jumlah data uji yang diprediksi sebagai slot parkir kosong, namun pada data aktualnya sebagai slot parkir terisi.

Tingkat keberhasilan (*Success Rate*) dapat dihitung dari jumlah TP + TN dibagi dengan jumlah TP + TN + FP + FN.

$$\text{Success Rate} = \frac{TP+TN}{TP+TN+FP+FN} \quad [3.1]$$

### **3.5 Perancangan Struktur Data**

Pada sub bab ini akan dijelaskan tipe data yang akan dipakai untuk sarana penyimpanan pada sistem, serta penjelasan mengenai fungsi apa saja yang digunakan didalam sistem deteksi lahan parkir kosong gedung B1 UKDW.

#### 1. Tipe Data

##### a. Bitmap

Bitmap adalah salah satu tipe data digunakan untuk menampung citra. Pada sistem ini tipe Bitmap digunakan untuk menyimpan citra RGB , citra grayscale, ataupun citra hasil thresholding *Background Subtraction*.

##### b. Struct titikSlot

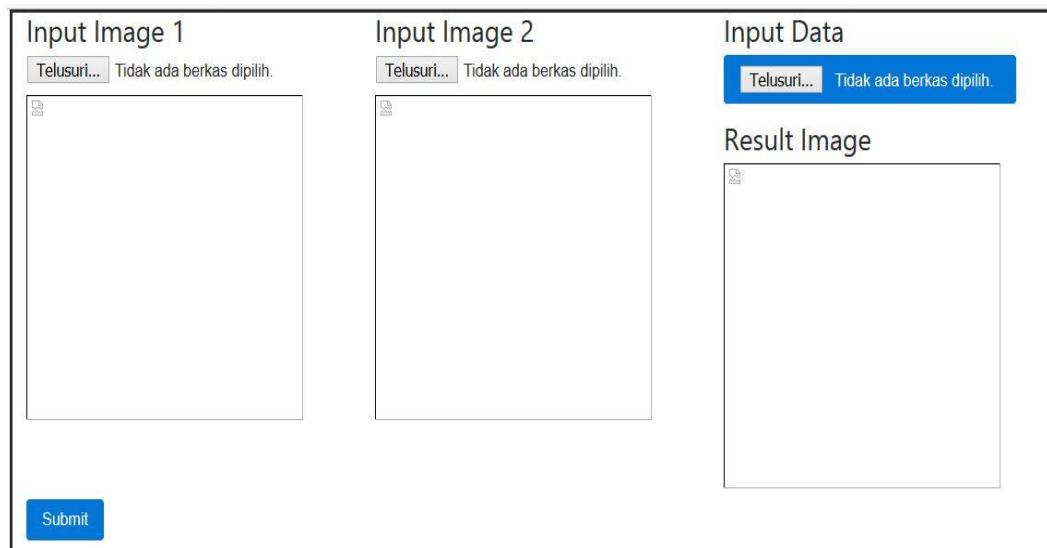
Tipe data titikSlot pada sistem ini digunakan untuk menyimpan 4 titik koordinat (x1,x2,y1,y2). Titik – titik tersebut adalah titik dimana masing – masing slot parkir berada.

##### c. Ifstream data

Tipe data input file stream akan menyimpan file .txt yang nantinya berisi info tentang masing – masing slot parkir berada.

### **3.6 Perancangan Antar Muka**

Antar Muka dalam sistem pendekslan tempat parkir kosong memiliki 1 form yang berisi input untuk gambar *background* dan *foreground* serta input file .txt serta keluaran dari keadaan tempat parkir. Mock up sistem dapat dilihat pada gambar 3.4 dibawah ini.



Gambar 3.4 Perancangan antar muka sistem

Penjelasan komponen – komponen yang ada didalam halaman utama dapat dilihat pada tabel 3.2

Komponen	Fungsi
Tombol Input Image 1	Tombol untuk memasukan citra ke dalam sistem sebagai citra <i>background</i> .
Tombol input Image 2	Tombol untuk memasukan citra ke dalam sistem sebagai citra <i>foreground</i> .
Tombol input Data	Tombol Untuk memasukan data .txt sebagai tempat yang akan di cek ketersediaan slot parkirnya.
Tombol Submit	Tombol untuk mengolah semua input yang ada dan sistem akan mengeluarkan output.

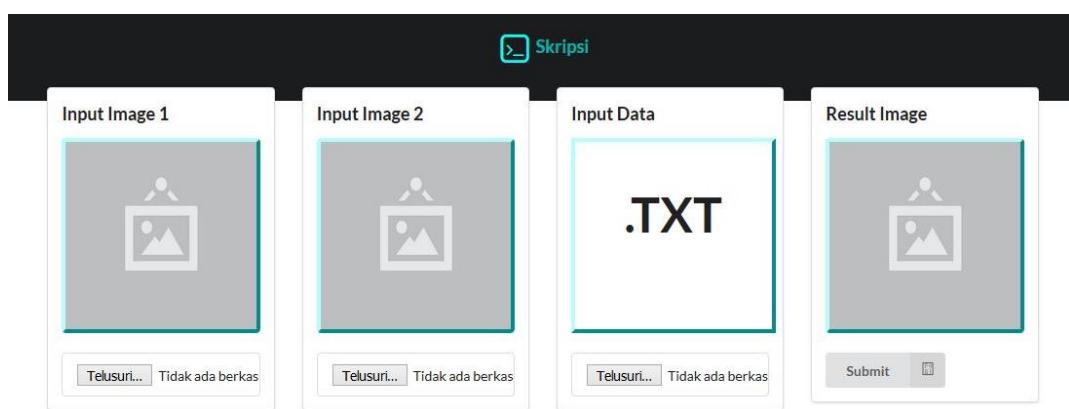
## BAB 4

### IMPLEMENTASI DAN ANALISIS SISTEM

#### 4.1 Implementasi Sistem

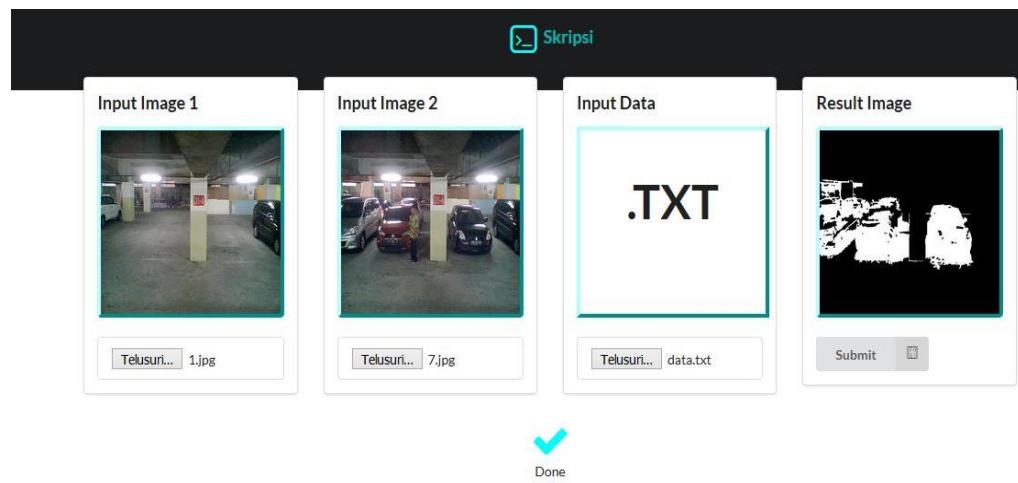
##### 4.1.1 Implementasi Antarmuka

Dalam sub bab ini akan dijelaskan implementasi antarmuka sistem yang digunakan untuk menguji algoritma sistem, antarmuka yang dibuat menggunakan basis website yang menjalankan program C++ ketika input telah dimasukan. Untuk implementasi antarmuka sistem dapat dilihat pada gambar 4.1



Gambar 4.1 Antarmuka sistem

Dalam antarmuka diatas terdapat 3 input yaitu input citra *background*, citra *foreground*, serta data.txt untuk memasukan data letak slot parkir yang akan dicek ketersediaannya. Setelah input dimasukan, sistem akan mengeluarkan output yang menunjukkan ketersediaan slot parkir pada area tersebut. Untuk contoh output dapat dilihat pada gambar 4.2



- slot ke 1 terisi
- slot ke 2 terisi
- slot ke 3 terisi

Gambar 4.2 Antarmuka Sistem setelah dijalankan

#### 4.1.2 Implementasi Algoritma

Dalam sub bab ini akan dijelaskan proses implementasi algoritma pada sistem dimulai dari proses *grayscale*, *Background Subtraction*, hingga penentuan terisi / tidaknya slot parkir. Untuk contoh pengujian dapat dilihat pada gambar 4.3

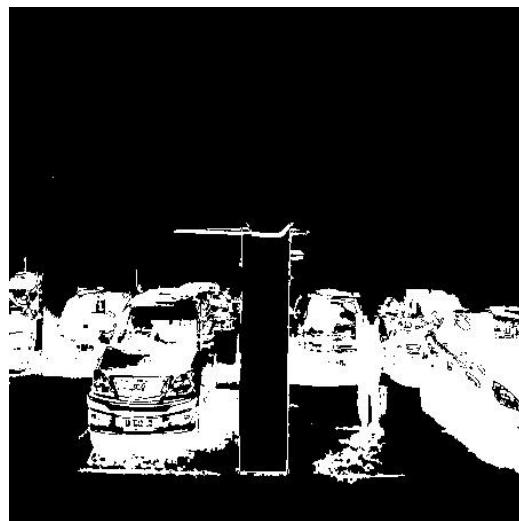




Gambar 4.3 pengubahan citra awal menjadi citra *grayscale*

#### 4.1 Implementasi Background Subtraction

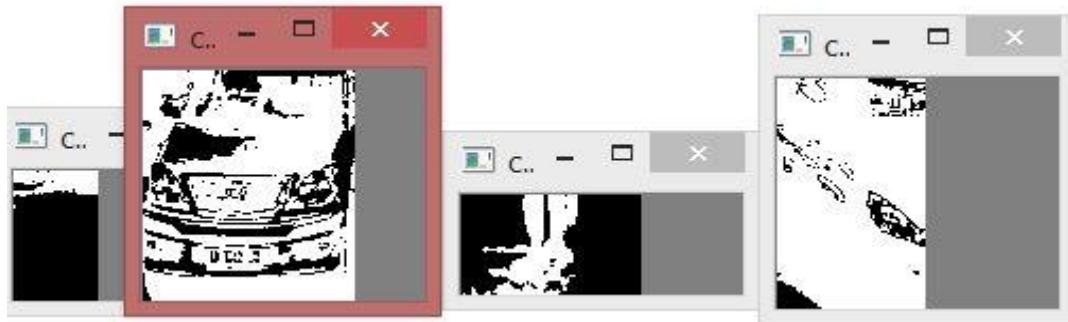
Pada contoh pertama dapat dilihat citra awal diubah menjadi citra *grayscale*, proses selanjutnya adalah melakukan *Background Subtraction* untuk mendapatkan objek yang ada pada gambar diatas, untuk hasil *Background Subtraction* dapat dilihat pada gambar 4.4



Gambar 4.4 Hasil *Background Subtraction*

Ketika *Background Subtraction* dilakukan, maka objek *Background* akan dihilangkan dan objek *Foreground* akan tetap ada, setelah melakukan Thresholding maka citra menjadi citra biner seperti pada gambar 4.3. proses selanjutnya adalah memotong gambar kedalam beberapa gambar sesuai jumlah mobil yang tercakup dalam gambar tersebut, letak titik dan besarnya potongan gambar telah ditentukan

sebelumnya sesuai dengan keadaan tempat parkir. Untuk hasil pemotongan gambar dapat dilihat pada gambar 4.5



Gambar 4.5 Hasil pemotongan dari *Background Subtraction*

#### 4.1.2 Implementasi Connected Component Labelling

Selanjutnya pada setiap gambar yang ada akan dihitung luasnya dengan *Connected Component Labelling*, jika luasnya melebihi threshold maka objek tersebut dianggap mobil yang mengisi slot parkir. Threshold sendiri didapat dari besar area yang dipotong\*0.6. Jika ada objek tetapi luasnya dibawah threshold, maka objek tersebut dianggap bukan mobil dan tidak menggunakan slot parkir. Untuk contoh output dapat dilihat pada gambar 4.6.

```
threshold ccl : 1560
size objek 1 : 450 - Tidak Memenuhi
size objek 1 : 3 - Tidak Memenuhi
threshold ccl : 6900
size objek 2 : 9052 - Memenuhi
threshold ccl : 2550
size objek 3 : 1298 - Tidak Memenuhi
threshold ccl : 4830
size objek 4 : 7731 - Memenuhi
slot ke 1 kosong
slot ke 2 terisi
slot ke 3 kosong
slot ke 4 terisi
```

Gambar 4.6 hasil implementasi *Connected Component Labelling*

Hasil perhitungan tersebut tidak ditampilkan secara langsung oleh sistem akan tetapi jika besaran luas memenuhi syarat threshold maka hasil tersebut akan ditampung oleh satu variable Boolean yang menyimpan keadaan slot parkir tersebut. Untuk hasil akhir sistem dapat dilihat pada gambar 4.7.



```
slot ke 1 kosong
slot ke 2 terisi
slot ke 3 kosong
slot ke 4 terisi
```

Gambar 4.7 Hasil akhir sistem

## 4.2 Pengujian dan Analisis Sistem

### 4.2.1. Pengujian Sistem

#### 4.2.1.1 Penentuan Threshold Background Subtraction

Metode Background Subtraction menggunakan selisih intensitas piksel antara citra background dan citra foreground yang kemudian akan di thresholding dengan satu angka threshold, ketika hasil selisih tersebut lebih kecil dari angka threshold maka nilai intensitas pada piksel tersebut menjadi 0, jika lebih besar akan menjadi 255. Karena angka threshold yang cocok untuk setiap kasus berbeda – beda maka digunakan cara dengan mencari terlebih dahulu rata – rata selisih intensitas tiap piksel yang lalu akan digunakan sebagai angka threshold pada proses *Background Subtraction*. Pada pengujian ini, peneliti mencoba menerapkan 3 jenis threshold yang berdasar pada rata – rata selisih intensitas piksel yaitu:

1. Threshold 1: rata – rata + 10
2. Threshold 2: rata – rata + 0
3. Threshold 3: rata – rata - 5

#### 4.2.1.2 Pengujian Menggunakan Threshold 1

Pengujian dilakukan dengan data uji sebanyak 65 citra, terdapat 10 citra background yang mewakili 4-8 citra foreground. Masing – masing tempat parkir memiliki satu data .txt yang mencatat letak slot parkir yang ada didalam tempat

parkir tersebut. Thresholding pada *Background Subtraction* sebesar rata – rata selisih intensitas piksel + 10. Untuk hasil pengujian sistem yang telah dilakukan dapat dilihat pada tabel 4.1

Tabel 4.1 Hasil uji sistem menggunakan threshold 1

No	TH	TP	TN	FP	FN	Total	Presentase
1	33	2	2	0	0	4	100%
2	26	1	3	0	0	4	100%
3	18	0	4	0	0	4	100%
4	32	2	2	0	0	4	100%
5	17	0	4	0	0	4	100%
6	37	3	1	0	0	4	100%
7	38	3	0	0	1	4	75%
8	41	3	0	0	1	4	75%
9	20	0	4	0	0	4	100%
10	26	0	4	0	0	4	100%
11	24	0	4	0	0	4	100%
12	40	2	1	0	1	4	75%
13	34	1	2	0	1	4	75%
14	36	1	1	0	0	2	100%
15	42	1	1	0	0	2	100%
16	25	1	1	0	0	2	100%
17	27	2	0	0	0	2	100%
18	38	2	2	0	0	4	100%
19	21	0	4	0	0	4	100%
20	24	0	4	0	0	4	100%
21	26	1	3	0	0	4	100%
22	27	1	3	0	0	4	100%
23	28	1	3	0	0	4	100%
24	29	2	2	0	0	4	100%
25	37	1	2	0	1	4	75%
26	36	1	2	0	1	4	75%
27	21	0	3	0	0	3	100%
28	29	1	2	0	0	3	100%
29	36	1	1	0	1	3	66.67%
30	42	2	1	0	0	3	100%
31	43	2	0	0	0	2	100%
32	27	1	1	0	0	2	100%
33	34	1	0	1	0	2	50%
34	25	1	1	0	0	2	100%

Tabel 4.1 (lanjutan)

35	46	2	0	0	0	2	100%
36	41	2	0	0	0	2	100%
37	15	0	3	0	0	3	100%
38	16	0	3	0	0	3	100%
39	16	0	3	0	0	3	100%
40	20	0	3	0	0	3	100%
41	22	1	2	0	0	3	100%
42	28	2	1	0	0	3	100%
43	31	3	0	0	0	3	100%
44	30	3	0	0	0	3	100%
45	45	4	0	0	0	4	100%
46	43	3	1	0	0	4	100%
47	40	2	2	0	0	4	100%
48	38	1	2	1	0	4	75%
49	28	0	4	0	0	4	100%
50	26	0	4	0	0	4	100%
51	27	0	3	0	0	3	100.00%
52	24	1	2	0	0	3	100.00%
53	27	1	2	0	0	3	100.00%
54	29	2	1	0	0	3	100.00%
55	40	1	1	0	1	3	66.67%
56	37	2	1	0	0	3	100.00%
57	32	1	1	0	1	3	66.67%
58	31	1	1	0	1	3	66.67%
59	36	0	2	0	0	2	100.00%
60	48	0	1	1	0	2	50.00%
61	27	0	2	0	0	2	100.00%
62	29	1	1	0	0	2	100.00%
63	39	1	0	0	1	2	50.00%
64	34	1	0	0	1	2	50.00%
65	34	1	0	0	1	2	50.00%
Total		76	114	3	13	206	92.23%

Pada tabel 4.1 menunjukkan total slot parkir yang dideteksi ada 206 slot parkir dengan TP (*True Positive*) sebanyak 76, TN (*True Negative*) sebanyak 114, FP (*False Positive*) sebanyak 3, dan FN (*False Negative*) sebanyak 13. Dengan

*confusion matrix* dapat dihitung *success rate* dari implementasi sistem, berikut adalah penghitungan *success rate*:

$$\begin{aligned}
 \text{Success Rate} &= \frac{TP + TN}{TP + TN + FP + FN} \\
 &= \frac{76 + 114}{76 + 114 + 3 + 13} \\
 &= \frac{190}{206} = 0.922330
 \end{aligned}$$

Berdasarkan perhitungan *confusion matrix* didapatkan *Success rate* Sebesar 0.922330 atau 92.2330%.

#### 4.2.1.3. Pengujian Menggunakan Threshold 2

Pengujian dilakukan dengan data uji yang sama tetapi pada tahap *Background Subtraction* menggunakan threshold sebesar rata – rata selisih intensitas antar piksel dikurangi 5. Untuk hasil pengujian menggunakan threshold kecil dapat dilihat pada tabel 4.2.

Tabel 4.2 Hasil uji sistem menggunakan threshold 2

No	TH	TP	TN	FP	FN	Total	Presentase
1	23	2	2	0	0	4	100%
2	16	1	3	0	0	4	100%
3	8	0	4	0	0	4	100%
4	22	2	2	0	0	4	100%
5	7	0	4	0	0	4	100%
6	27	3	1	0	0	4	100%
7	28	4	0	0	0	4	100%
8	31	4	0	0	0	4	100%
9	10	0	4	0	0	4	100%
10	16	0	4	0	0	4	100%
11	14	0	4	0	0	4	100%
12	30	2	1	0	1	4	75%
13	24	2	2	0	0	4	100%
14	26	1	1	0	0	2	100%

Tabel 4.2 (lanjutan)

15	32	1	1	0	0	2	100%
16	15	1	1	0	0	2	100%
17	17	2	0	0	0	2	100%
18	28	2	2	0	0	4	100%
19	11	0	3	1	0	4	75%
20	14	0	3	1	0	4	75%
21	16	1	3	0	0	4	100%
22	17	1	3	0	0	4	100%
23	18	1	3	0	0	4	100%
24	19	2	2	0	0	4	100%
25	27	1	2	0	1	4	75%
26	26	2	2	0	0	4	100%
27	11	0	3	0	0	3	100%
28	19	1	2	0	0	3	100%
29	26	2	1	0	0	3	100%
30	32	2	1	0	0	3	100%
31	33	2	0	0	0	2	100%
32	17	1	1	0	0	2	100%
33	24	1	0	1	0	2	50%
34	15	1	1	0	0	2	100%
35	36	2	0	0	0	2	100%
36	31	2	0	0	0	2	100%
37	5	0	3	0	0	3	100%
38	6	0	3	0	0	3	100%
39	6	0	3	0	0	3	100%
40	10	0	3	0	0	3	100%
41	12	1	2	0	0	3	100%
42	18	2	1	0	0	3	100%
43	21	3	0	0	0	3	100%
44	20	3	0	0	0	3	100%
45	35	4	0	0	0	4	100%
46	33	3	1	0	0	4	100%
47	30	2	2	0	0	4	100%
48	28	1	2	1	0	4	75%
49	18	0	4	0	0	4	100%
50	16	0	4	0	0	4	100%
51	17	0	3	0	0	3	100%
52	14	1	1	1	0	3	67%
53	17	1	1	1	0	3	67%

54	19	2	1	0	0	3	100%
55	30	2	1	0	0	3	100%
56	27	2	1	0	0	3	100%
57	22	3	0	0	0	3	100%
58	21	3	0	0	0	3	100%
59	26	0	2	0	0	2	100%
60	38	0	1	1	0	2	50%
61	17	0	2	0	0	2	100%
62	19	1	0	1	0	2	50%
63	29	1	0	0	1	2	50%
64	24	1	0	0	1	2	50%
65	24	1	0	0	1	2	50%
Total	86	107	8	5	206	93.69%	

$$\begin{aligned}
 Success Rate &= \frac{TP + TN}{TP + TN + FP + FN} \\
 &= \frac{86 + 107}{86 + 107 + 8 + 5} \\
 &= \frac{193}{206} = 0.93689
 \end{aligned}$$

Berdasarkan perhitungan *confusion matrix* pada table 4.2 didapatkan *Success rate* Sebesar 0.93689 atau 93.689%. Hasil ini sedikit lebih besar daripada pengujian sebelumnya yang menggunakan threshold lebih besar, akan tetapi pengujian ini menyebabkan kemungkinan *False Positive* ditemukan lebih banyak karena semakin kecil threshold pada *Background Subtraction*. Semakin banyak pula citra yang dianggap sebagai citra *background* dianggap sebagai objek oleh sistem karena lebih besar dari threshold yang ada.

#### 4.2.1.4 Pengujian Menggunakan Threshold 3

Pengujian dilakukan dengan data uji yang sama tetapi pada tahap *Background Subtraction* menggunakan threshold yang digunakan adalah sebesar rata – rata selisih intensitas antar piksel dikurangi 5. Untuk hasil pengujian menggunakan threshold kecil dapat dilihat pada tabel 4.3.

Tabel 4.3 Hasil Uji sistem menggunakan threshold 3

No	TH	TP	TN	FP	FN	Total	Presentase
1	18	1	2	1	0	4	75%
2	11	1	3	0	0	4	100%
3	3	0	0	4	0	4	0%
4	17	2	2	0	0	4	100%
5	2	0	0	4	0	4	0%
6	22	3	1	0	0	4	100%
7	23	4	0	0	0	4	100%
8	26	4	0	0	0	4	100%
9	5	0	3	1	0	4	75%
10	11	0	2	2	0	4	50%
11	9	0	4	0	0	4	100%
12	25	3	1	0	0	4	100%
13	19	2	2	0	0	4	100%
14	21	1	1	0	0	2	100%
15	27	1	1	0	0	2	100%
16	10	1	1	0	0	2	100%
17	12	2	0	0	0	2	100%
18	23	2	2	0	0	4	100%
19	6	0	0	4	0	4	0%
20	9	0	1	3	0	4	25%
21	11	1	3	0	0	4	100%
22	12	1	3	0	0	4	100%
23	13	1	3	0	0	4	100%
24	14	2	2	0	0	4	100%
25	22	1	2	0	1	4	75%
26	21	2	2	0	0	4	100%
27	6	0	1	2	0	3	33%
28	14	1	2	0	0	3	100%
29	21	2	1	0	0	3	100%
30	27	2	1	0	0	3	100%
31	28	2	0	0	0	2	100%
32	12	1	1	0	0	2	100%
33	19	1	0	1	0	2	50%
34	10	1	1	0	0	2	100%
35	31	2	0	0	0	2	100%
36	26	2	0	0	0	2	100%
37	0	0	0	3	0	3	0%

Tabel 4.3 (lanjutan)

38	1	0	0	3	0	3	0%
39	1	0	0	3	0	3	0%
40	5	0	3	0	0	3	100%
41	7	1	1	1	0	3	67%
42	13	2	1	0	0	3	100%
43	16	3	0	0	0	3	100%
44	15	3	0	0	0	3	100%
45	30	4	0	0	0	4	100%
46	28	3	1	0	0	4	100%
47	25	2	2	0	0	4	100%
48	23	1	2	1	0	4	75%
49	13	0	4	0	0	4	100%
50	11	0	4	0	0	4	100%
51	12	0	2	1	0	3	67%
52	9	1	1	1	0	3	67%
53	12	1	1	1	0	3	67%
54	14	2	1	0	0	3	100%
55	25	2	1	0	0	3	100%
56	22	2	1	0	0	3	100%
57	17	3	0	0	0	3	100%
58	16	3	0	0	0	3	100%
59	21	0	2	0	0	2	100%
60	33	0	1	1	0	2	50%
61	12	0	2	0	0	2	100%
62	14	1	0	1	0	2	50%
63	24	1	0	0	1	2	50%
64	19	1	0	0	1	2	50%
65	19	1	0	0	1	2	50%
Total		86	78	38	4	206	79.61%

$$\begin{aligned}
 \text{Success Rate} &= \frac{TP + TN}{TP + TN + FP + FN} \\
 &= \frac{86 + 78}{86 + 78 + 38 + 4} \\
 &= \frac{164}{206} = 0.7961
 \end{aligned}$$

Berdasarkan perhitungan *confusion matrix* pada table 4.3 didapatkan *Success rate* Sebesar 0.7961 atau 79.61%. Hasil ini jauh lebih kecil daripada pengujian sebelumnya yang menggunakan threshold 1 dan threshold 2. Ujian ini juga menyebabkan kemungkinan *False Positive* ditemukan lebih banyak karena semakin kecil threshold pada *Background Subtraction*. Semakin banyak pula citra yang dianggap sebagai citra *background* dianggap sebagai objek oleh sistem karena lebih besar dari threshold yang ada.

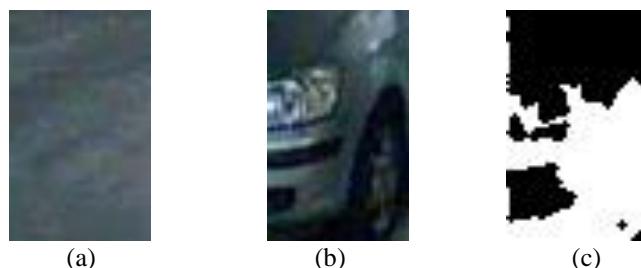
#### 4.2.3 Analisis Sistem

Sesuai dengan hasil pengujian yang telah dilakukan, dengan menggunakan metode *Background Subtraction* untuk mencari lahan parkir kosong. Sistem dapat mengenali tempat parkir kosong atau terisi dengan tingkat keberhasilan 93.6% dari data uji sebanyak 65 citra keadaan lahan parkir di gedung B1 Universitas Kristen Duta Wacana. Dari 65 data citra yang diuji. Beberapa berhasil mencapai tingkat keberhasilan sebesar 100%, tetapi ada juga data uji yang mencapai tingkat keberhasilan 75%, dan 50%.

Disamping memiliki persentase keberhasilan yang cukup tinggi, sistem ini memiliki beberapa keterbatasan. Tidak semua citra uji berhasil dikenali. Masih terdapat beberapa hasil yang tidak sesuai dengan keadaan aktual. Gagalnya sistem dalam mendeteksi ketersediaan lahan parkir disebabkan oleh beberapa hal akan dijelaskan dibawah ini.

1. Metode *Background Subtraction* adalah metode yang mengurangkan citra *background* dengan citra *foreground* untuk mendapat keadaan yang berubah dari citra *background*. Dengan kata lain mencari objek yang ada selain objek pada citra *background*. Selisih intensitas piksel antara citra *background* dan citra *foreground* sangatlah berpengaruh. Seperti halnya pada data uji ke-7. Pada data uji ke-7 terjadi *False Negative* sebanyak 1 yaitu pada slot pertama. Warna mobil yang berada di slot pertama mendekati warna background sehingga objek yang didapat oleh sistem cenderung kecil dan tidak melebihi threshold sehingga objek yang berada di slot 1 tidak dianggap mobil dan sistem

mengatakan bahwa slot tersebut kosong. Gambar 4.8 merupakan contoh citra *background* dan *foreground* yang memiliki kesamaan warna.



Gambar 4.8 contoh gambar (a) citra *background* (b) citra *foreground* (c) citra hasil

Dari gambar diatas dapat dilihat jika warna pada citra *background* dan citra *foreground* cenderung mirip. Terutama pada bagian atas yang cenderung gelap sehingga selisih dari kedua citra tersebut kecil dan citra tersebut dianggap sebagai citra *background*.

2. Metode *Background Subtraction* digunakan untuk mengambil objek pada citra *foreground*, akan tetapi objek yang didapat tidak diketahui jenisnya, untuk mengenali apakah objek tersebut sebuah mobil atau bukan. Sistem yang dibuat menggunakan metode threshold luas objek yang berada di masing – masing slot parkir. Jika besar objek memenuhi threshold maka objek tersebut dianggap sebuah mobil. Sistem tetap bisa memprediksi dengan tepat jika objek non-mobil (manusia) berada di slot parkir karena besarnya masih dibawah threshold yang ada. Tetapi jika ada beberapa orang yang berdekatan sehingga saat metode *Background Subtraction* dilakukan sehingga sekumpulan orang yang ada seperti menjadi sebuah objek tunggal yang besarnya menyerupai mobil. Sehingga saat penghitungan luas oleh sistem. Gambar 4.9 merupakan citra *foreground* yang terdapat objek besar selain mobil.



(a) (b) (c)

Gambar 4.9 contoh gambar 2 (a) citra *background* (b) citra *foreground* (c) citra hasil

Dari contoh gambar 4.8 dapat dilihat bahwa slot parkir yang seharusnya kosong dikenali terisi oleh sistem dikarenakan objek yang berada di slot tersebut besarnya memenuhi syarat thresholding. Hal ini bisa terjadi karena 2 atau lebih objek kecil yang berdekatan sehingga tidak terhapus saat sub-proses *Background Subtraction* yang menghilangkan objek – objek kecil pada citra hasil.

Selain contoh kasus diatas ada contoh data uji lain yang mendeteksi ada orang di slot parkir. Akan tetapi karena besaran luas objek yang berada pada slot tersebut tidak memenuhi threshold maka objek tersebut tidak dikenali sebagai mobil dan output sistem tetap menyatakan slot tersebut kosong. Untuk contoh gambar dapat dilihat pada gambar 4.10 dibawah ini.



Gambar 4.10 contoh gambar 3 (a) citra *background* (b) citra *foreground* (c) citra hasil

## **BAB 5**

### **KESIMPULAN DAN SARAN**

#### **5.1 Kesimpulan**

Berdasarkan hasil pengujian dan analisis sistem yang dibuat, maka dapat disimpulkan bahwa:

1. Sistem berhasil dalam mengimplementasikan *Background Subtraction* untuk mendeteksi slot parkir yang terisi / kosong pada gedung parkiran. Implementasi dilakukan dengan mengubah citra *background* dan *foreground* menjadi ukuran 450 x 450 dan diubah dalam bentuk *grayscale* yang digunakan untuk proses *Background Subtraction*. Pengujian menggunakan 10 citra background yang mewakili 4-8 citra foreground, total citra sebanyak 65 memiliki tingkat kesuksesan sebesar 92.2330% untuk penggunaan threshold 1, sebesar 93.689% untuk penggunaan threshold 2, dan sebesar 79.61% untuk penggunaan threshold 3.
2. Implementasi *Background Subtraction* memiliki beberapa kelemahan dalam mengenali objek yang didapat setelah proses *Background Subtraction* itu sendiri. Hal ini disebabkan tidak ada metode khusus yang digunakan untuk mengenali objek dari citra *foreground*. Pengenalan hanya mengandalkan besaran luas dari objek yang didapat sehingga tidak diketahui dengan pasti apakah objek tersebut merupakan mobil atau bukan. Selain itu *Background Subtraction* sangat bergantung pada kesamaan citra background dan foreground sehingga sangat sensitive terhadap perubahan yang terjadi.

#### **5.2 Saran**

Sistem deteksi lahan parkir kosong di UKDW ini masih dapat dilakukan pengembangan lebih lanjut lagi. Penulis memberikan saran sebagai berikut:

1. Pemotongan slot parkir yang dapat dilakukan secara otomatis oleh sistem. Sehingga tidak perlu lagi adanya konfigurasi ulang ketika sistem di implementasikan pada tempat lain.
2. Penyempurnaan metode normalisasi sehingga threshold yang didapatkan dapat lebih menyesuaikan keadaan tempat parkir di UKDW
3. Perlu ditambahkan process *pre-processing* sehingga ketika citra *foreground* mengalami perubahan seperti terkena angin sistem tetap dapat mendapat hasil yang optimal.

## DAFTAR PUSTAKA

- Choudekar, M. P. (2011). Implementation of Image Processing in Real Time Traffic Light Control. *Image (Rochester, N.Y.)*, 2(1), 94–98.  
<http://doi.org/10.1109/ICECTECH.2011.5941662>
- Eril Mozef. (2003). Algoritma Labeling Citra Biner Dengan Performansi Optimal Processor-Time. *Jurnal Informatika*, 5(1), 67–77. Retrieved from  
<http://puslit2.petra.ac.id/ejournal/index.php/inf/article/view/15841>
- Idris, M. Y. I., Leng, Y. Y., Tamil, E. M., Noor, N. M., & Razak, Z. (2009). Car park system: A review of smart parking system and its technology. *Information Technology Journal*. <http://doi.org/10.3923/itj.2009.101.113>
- Kumar, T., & Verma, K. (2010). A Theory Based on Conversion of RGB image to Gray image. *International Journal of Computer Applications*, 7(2), 5–12.  
<http://doi.org/10.5120/1140-1493>
- Nithinya, & Kumar, S. (2016). Design and Implementation of an Intelligent Parking Management System Using Image Processing. *International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)*, 5(1), 95–100. [http://doi.org/10.1007/978-3-642-31020-1\\_17](http://doi.org/10.1007/978-3-642-31020-1_17)
- Oji, R. (2012). An Automatic Algorithm for Object Recognition and Detection Based on ASIFT. *Signal & Image Processing : An International Journal (SIPIJ)*, 3(5), 29–39. <http://doi.org/10.5121/sipij.2012.3503>
- Santra, A. K., & Christy, C. J. (2012). Genetic Algorithm and Confusion Matrix for Document Clustering 1, 9(1), 322–328.
- Vinay, & Kumar, N. L. (2015). Object Tracking Using Background Subtraction Algorithm. *International Journal of Engineering Research and General Science*, 3(1), 237–243.
- Yang, K., Cai, Z., & Zhao, L. (2013). Algorithm Research on Moving Object Detection of Surveillance Video Sequence \*. *Optics and Photonics Journal*, 3(20121213), 308–312. <http://doi.org/10.4236/opj.2013.32B072>
- Yusnita, R., Fariza, N., & Norazwinawati, B. (2012). Intelligent Parking Space Detection System Based on Image Processing. *International Journal of Innovation, Management and Technology*, 3(3), 232–235. Retrieved from

<http://www.ijimt.org/papers/228-G0038.pdf>

## LAMPIRAN

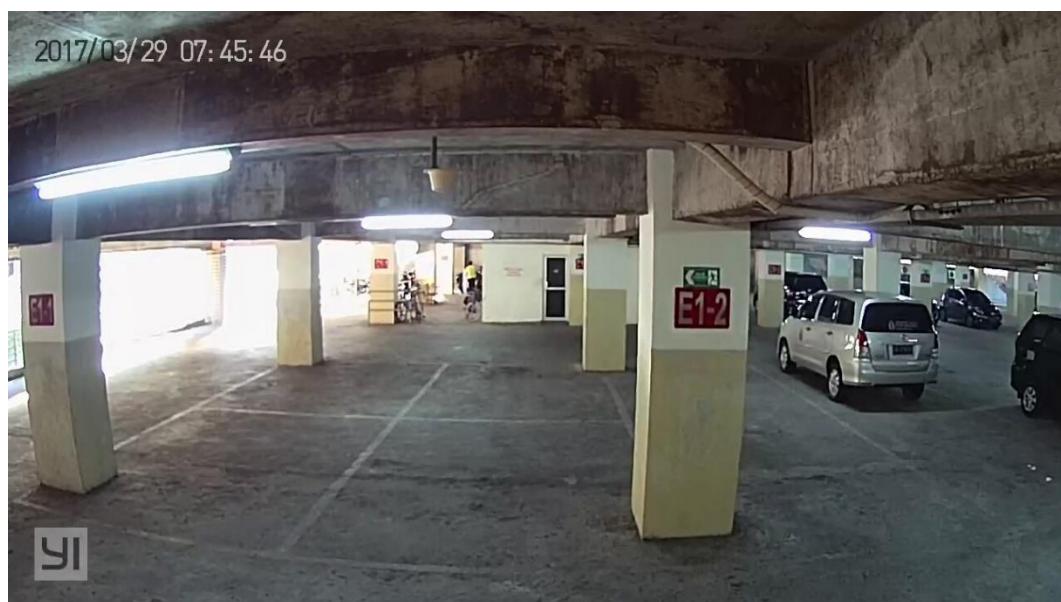
### Lampiran A - Citra Background dan Foreground

Citra Bakground

Background 1



Background 2



Background 3



Background 4



Background 5



Background 6



Background 7



Background 8



Background 9



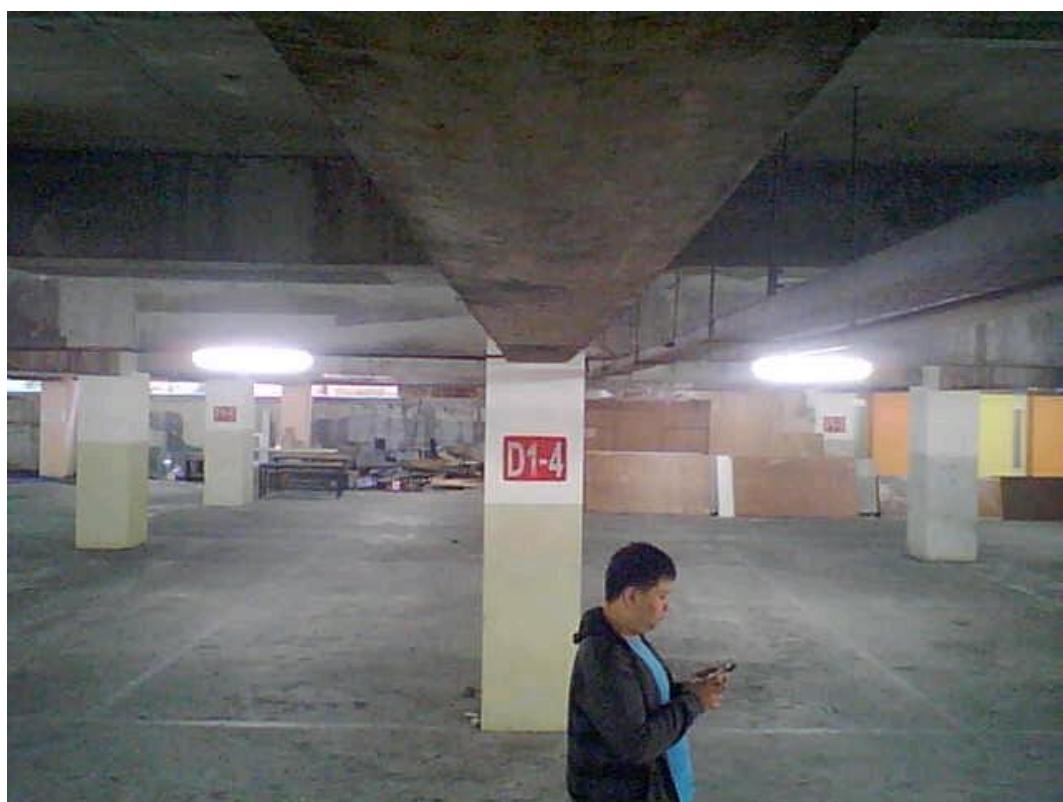
Background 10



## Citra Foreground

Foreground 1



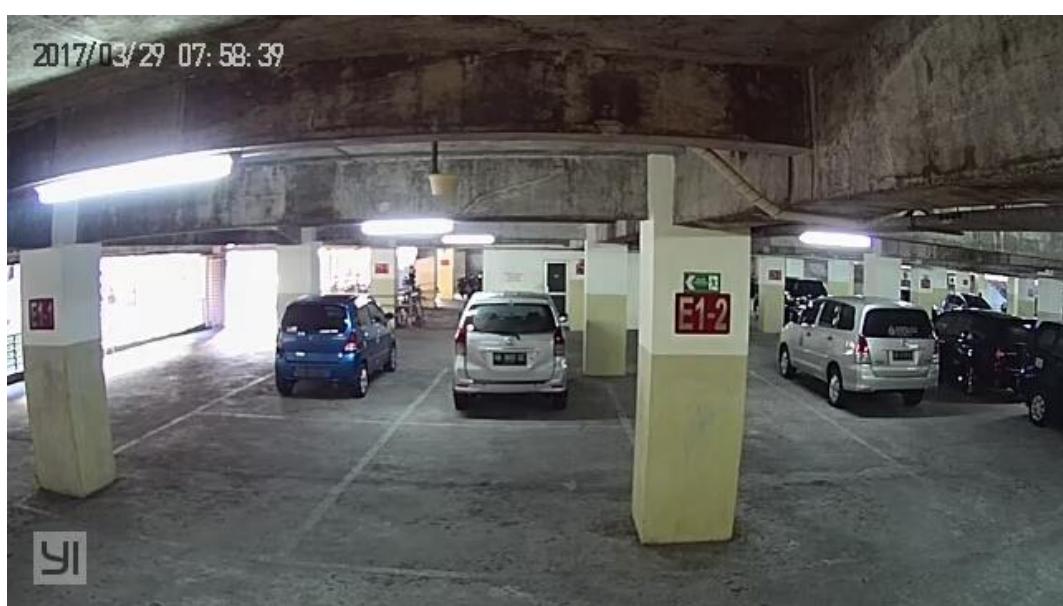
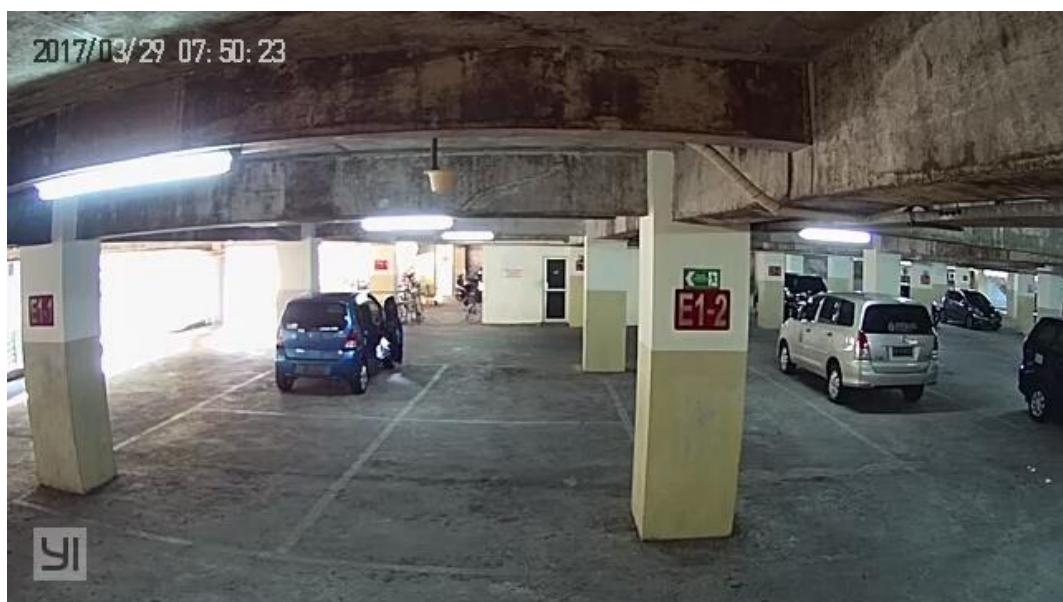


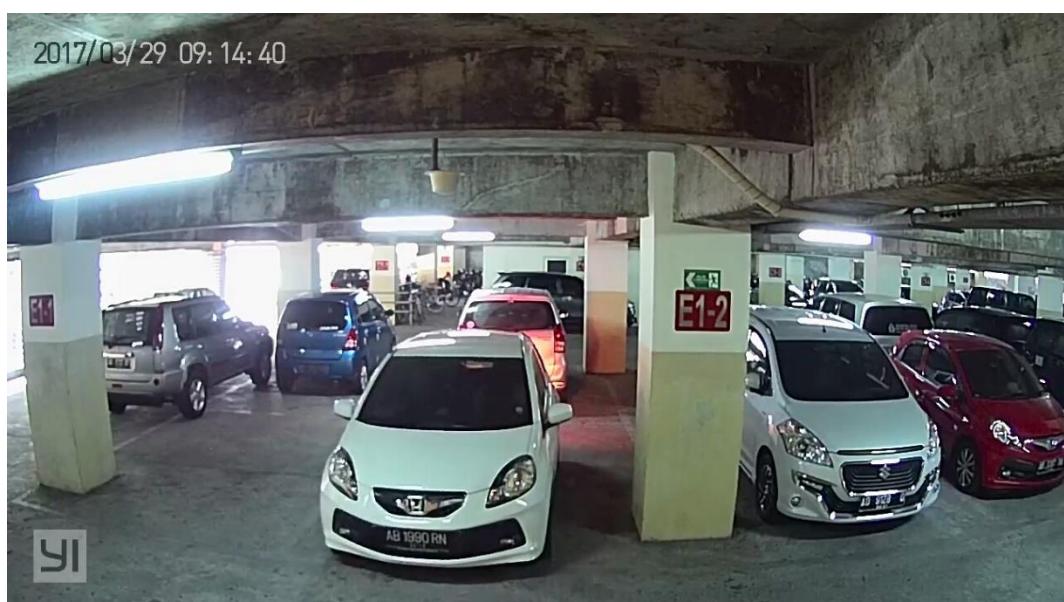
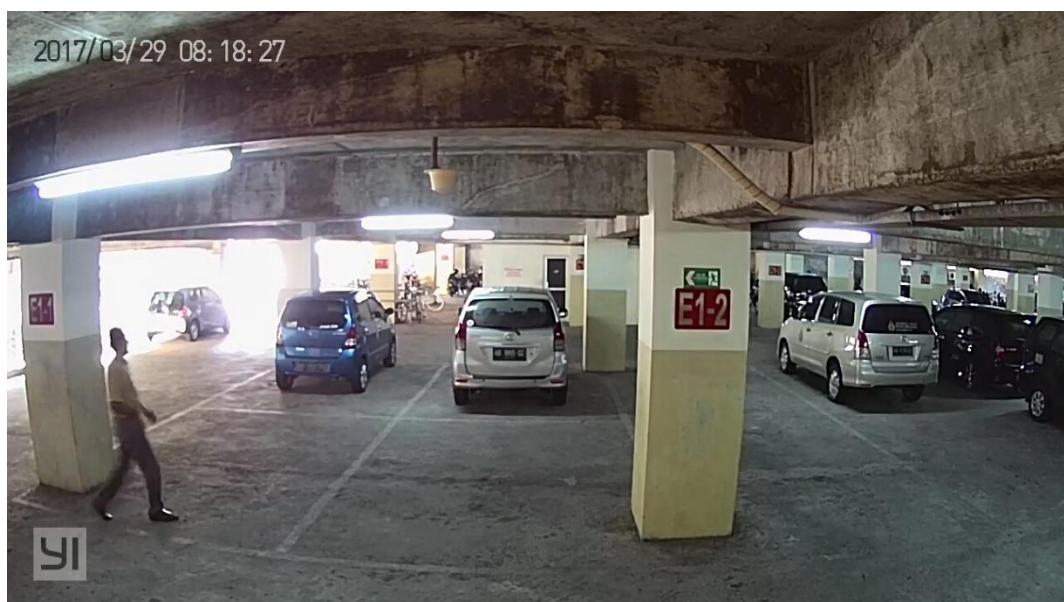


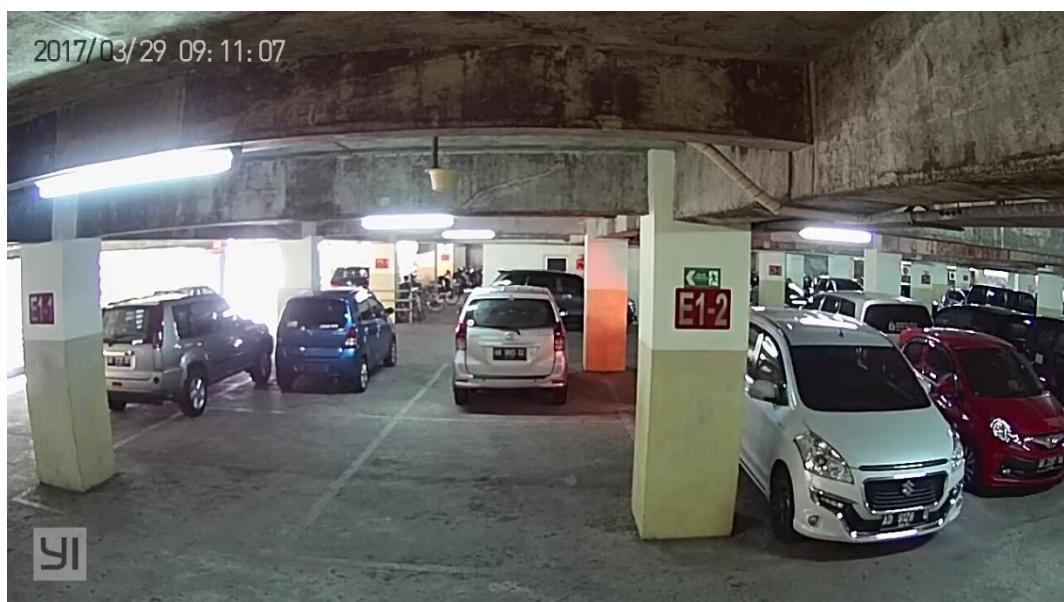


Foreground 2

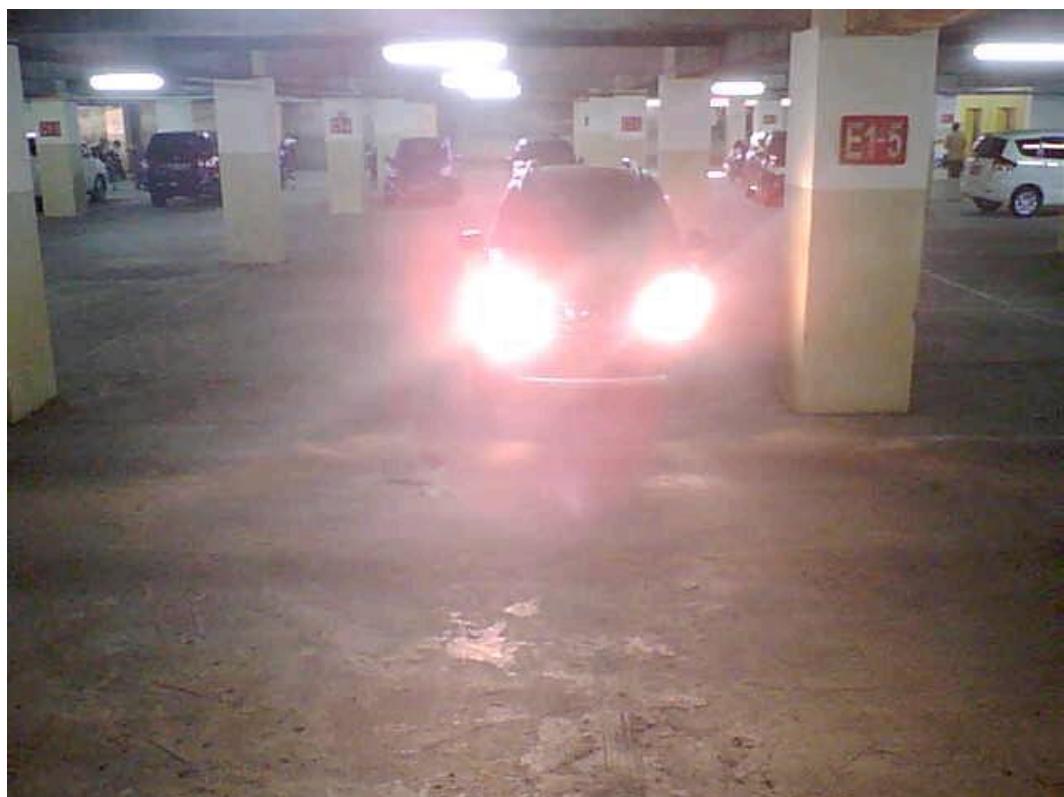








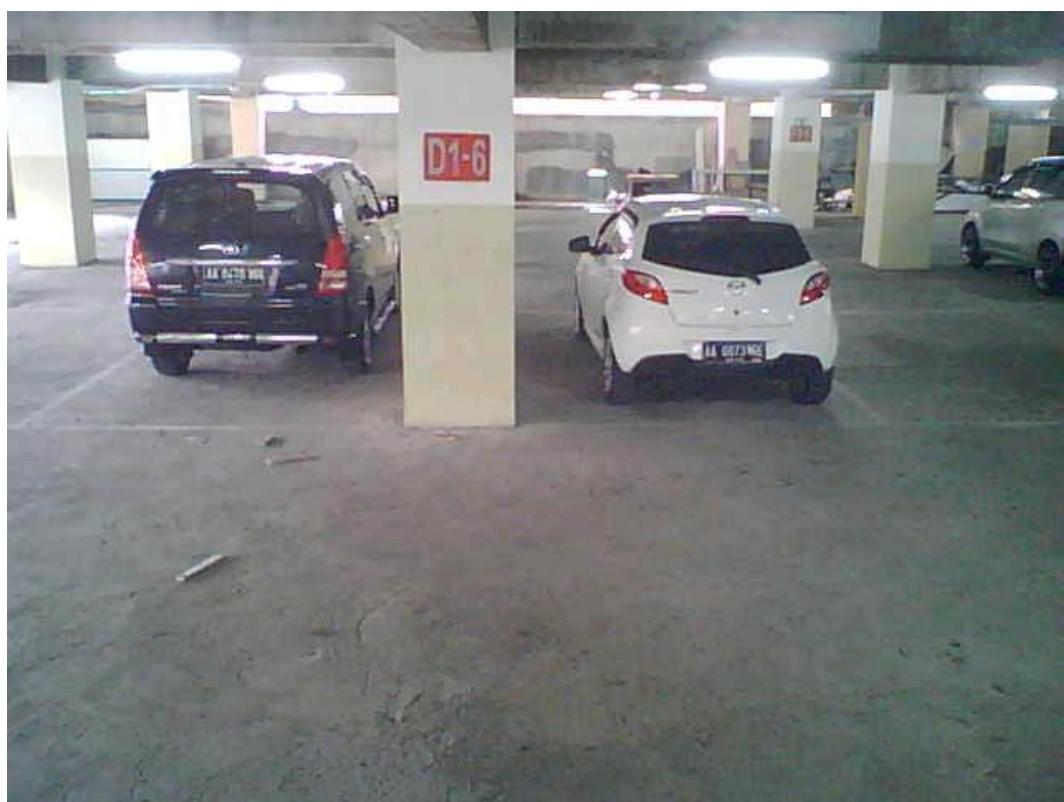
Foreground 3



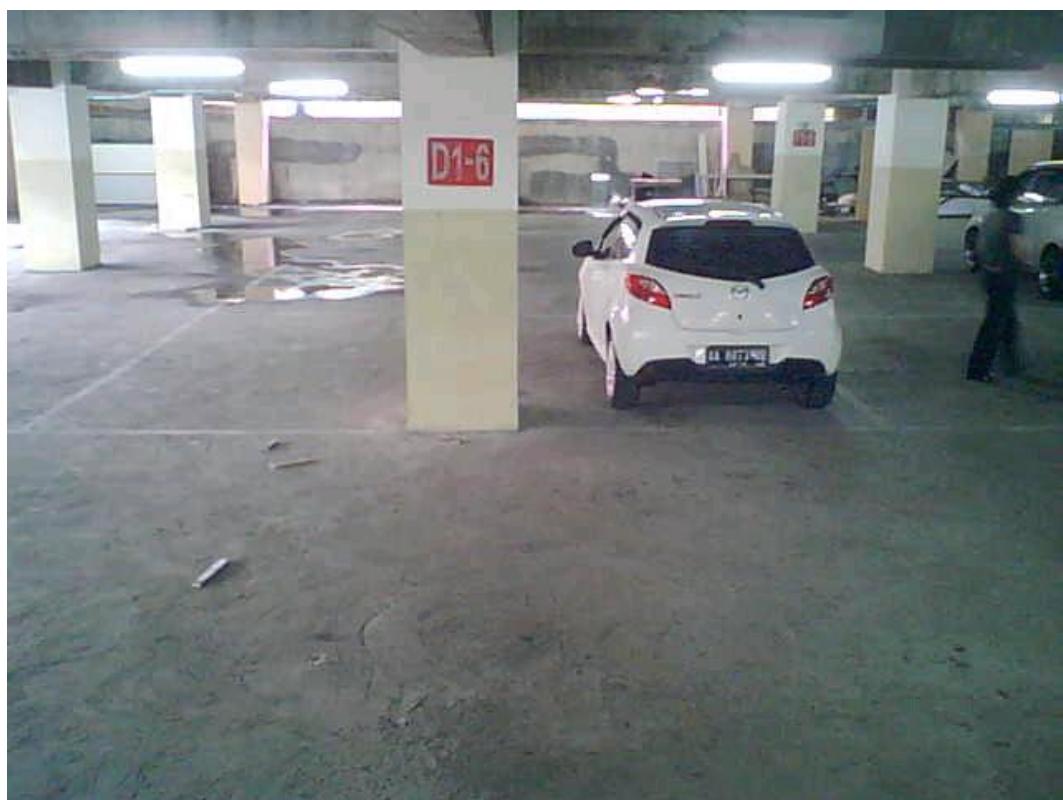
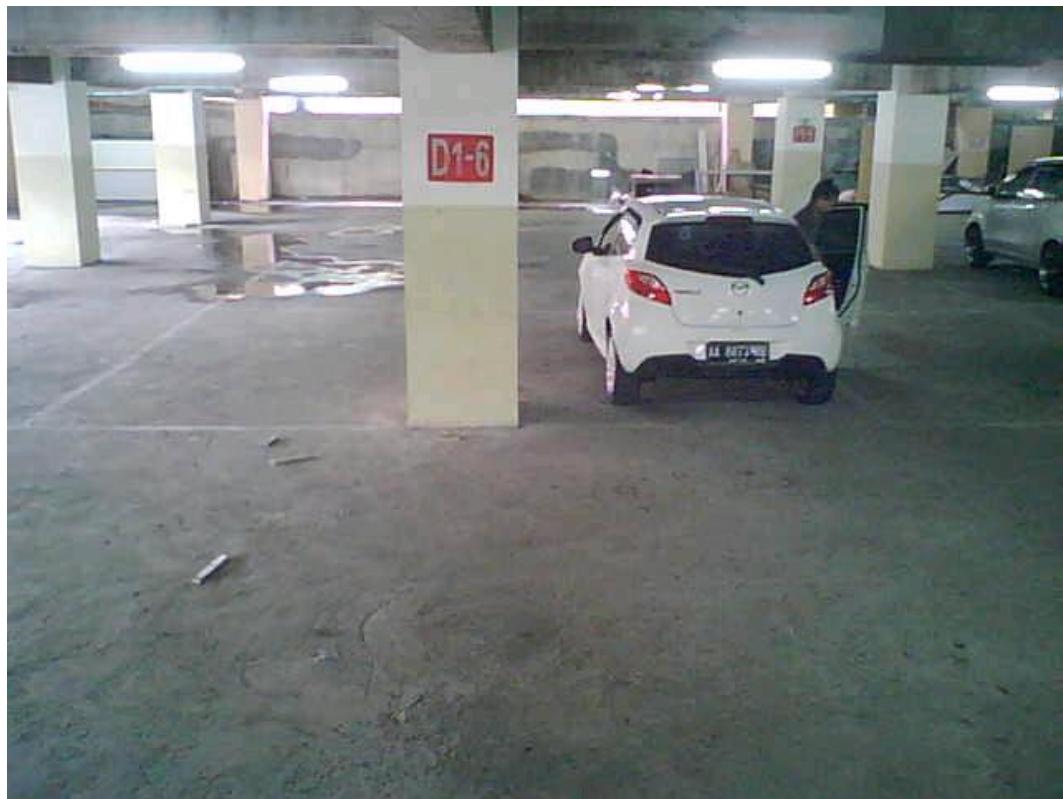


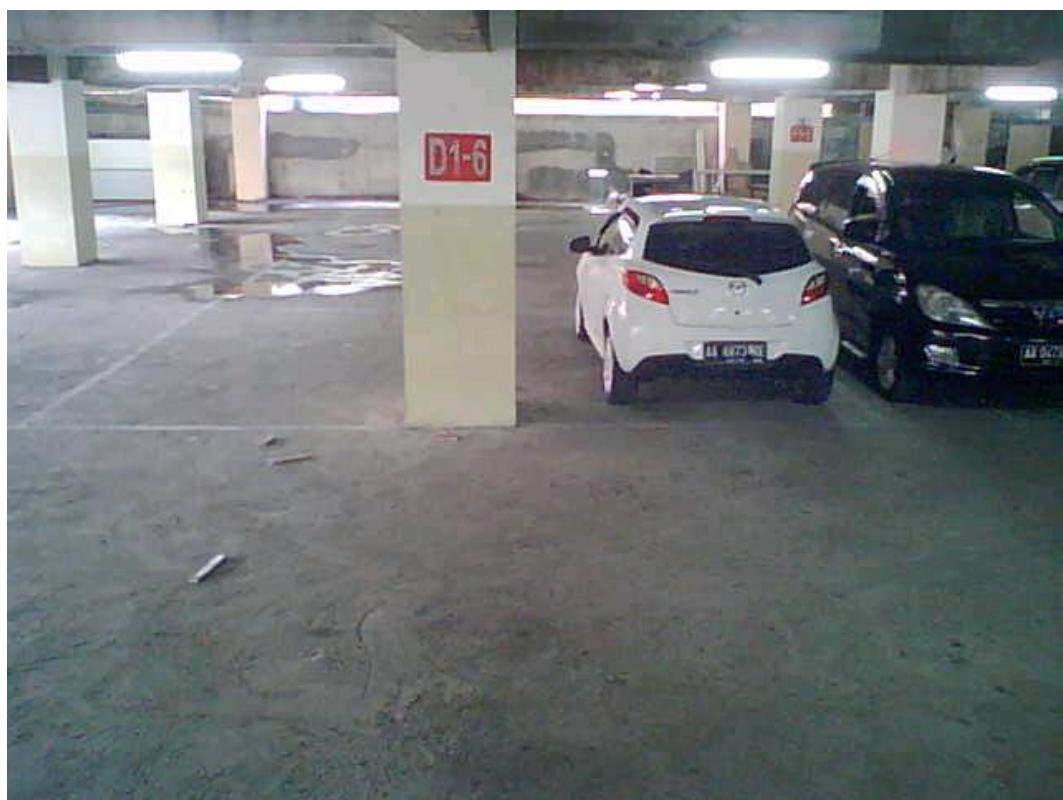
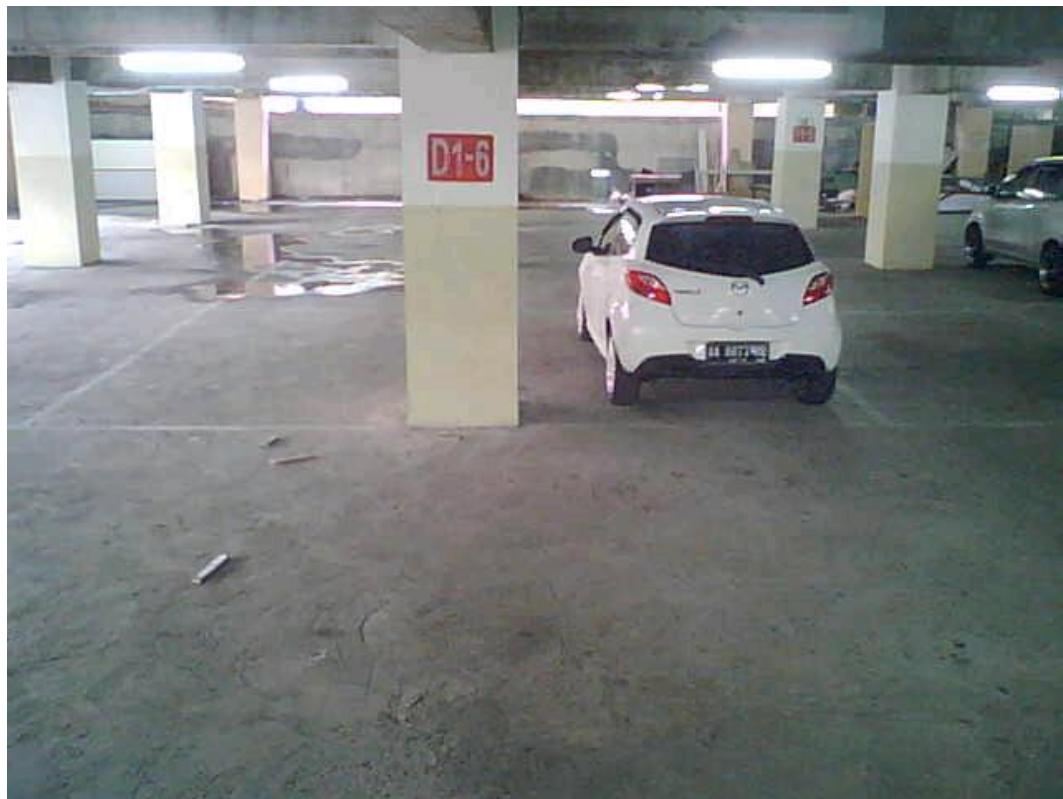


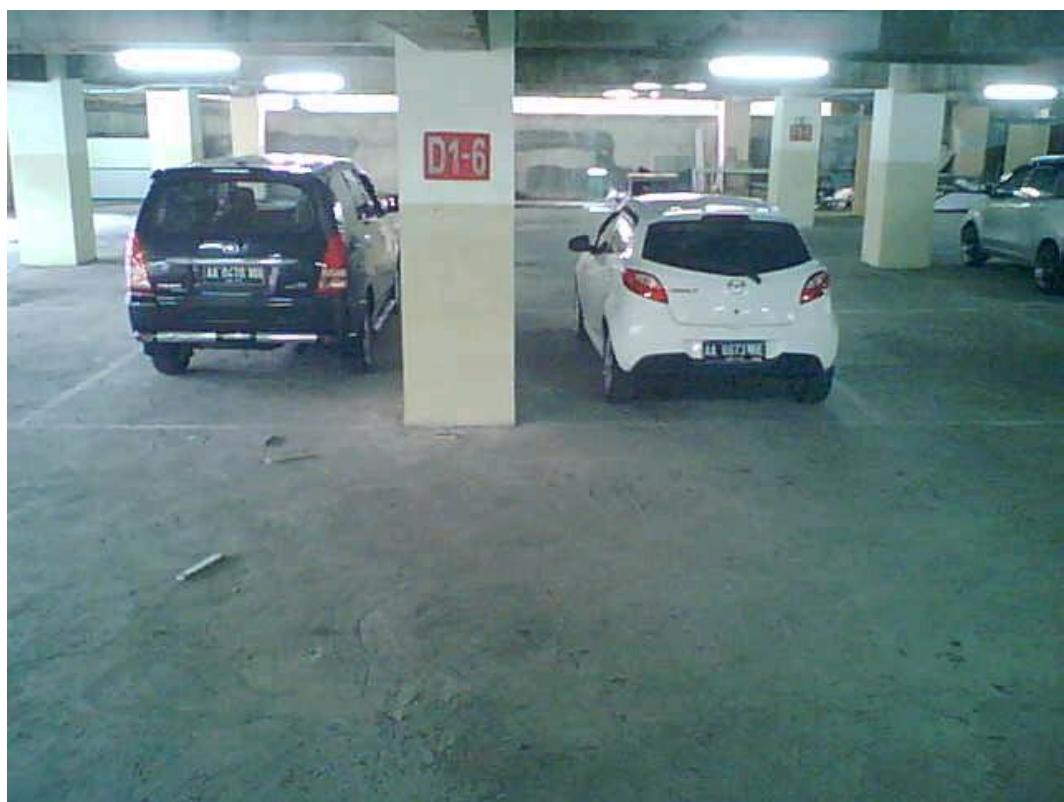
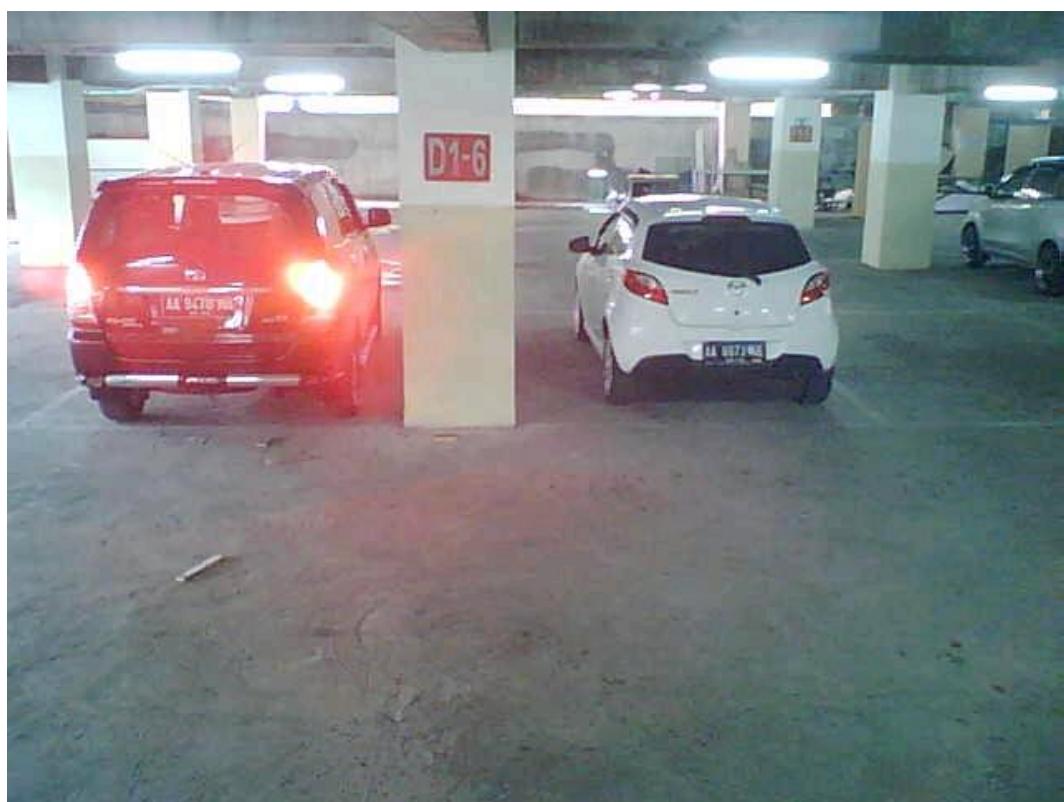
Foreground 4



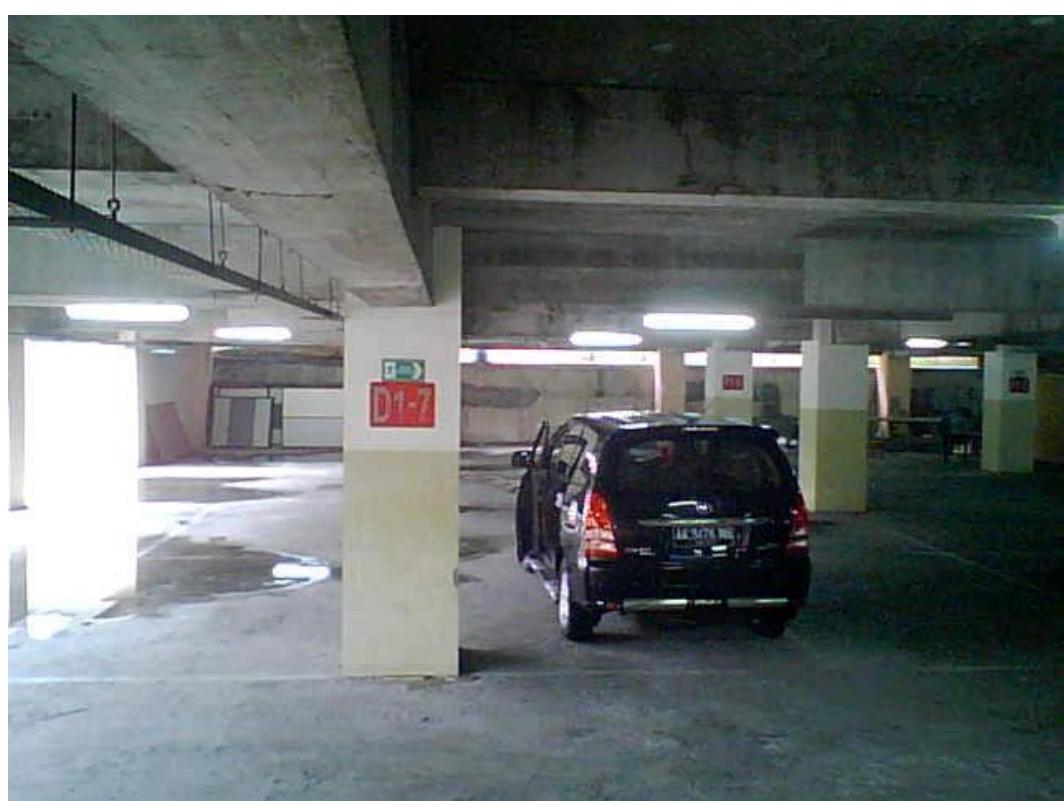


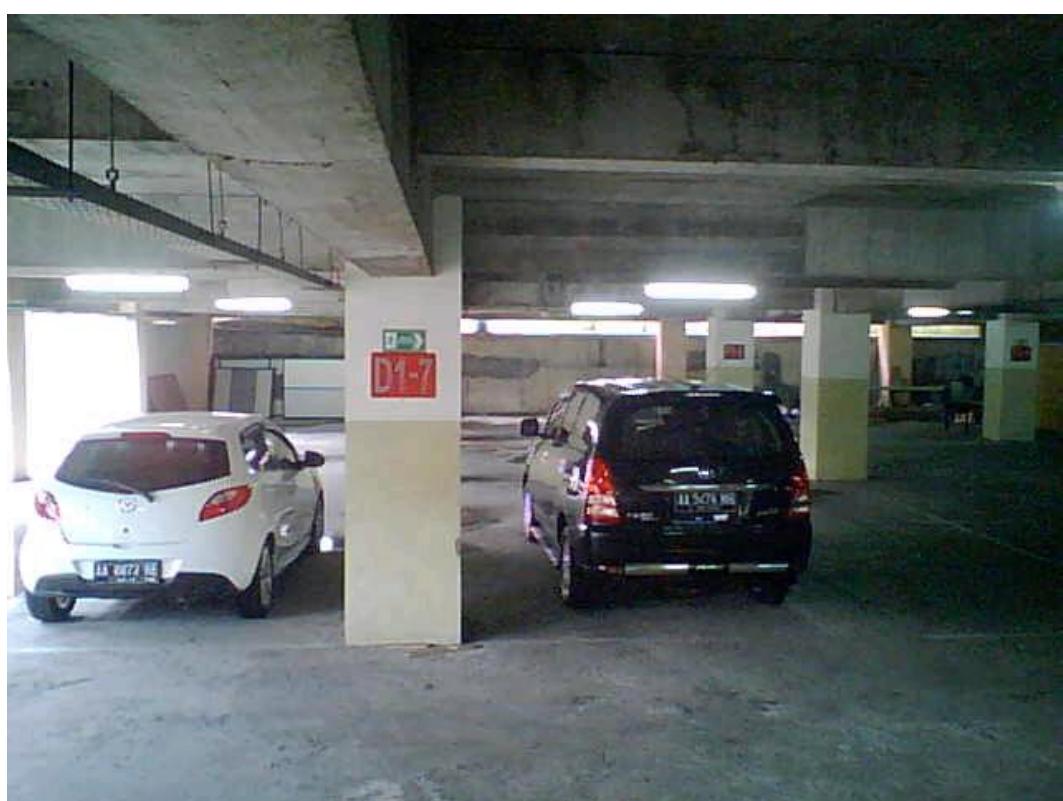




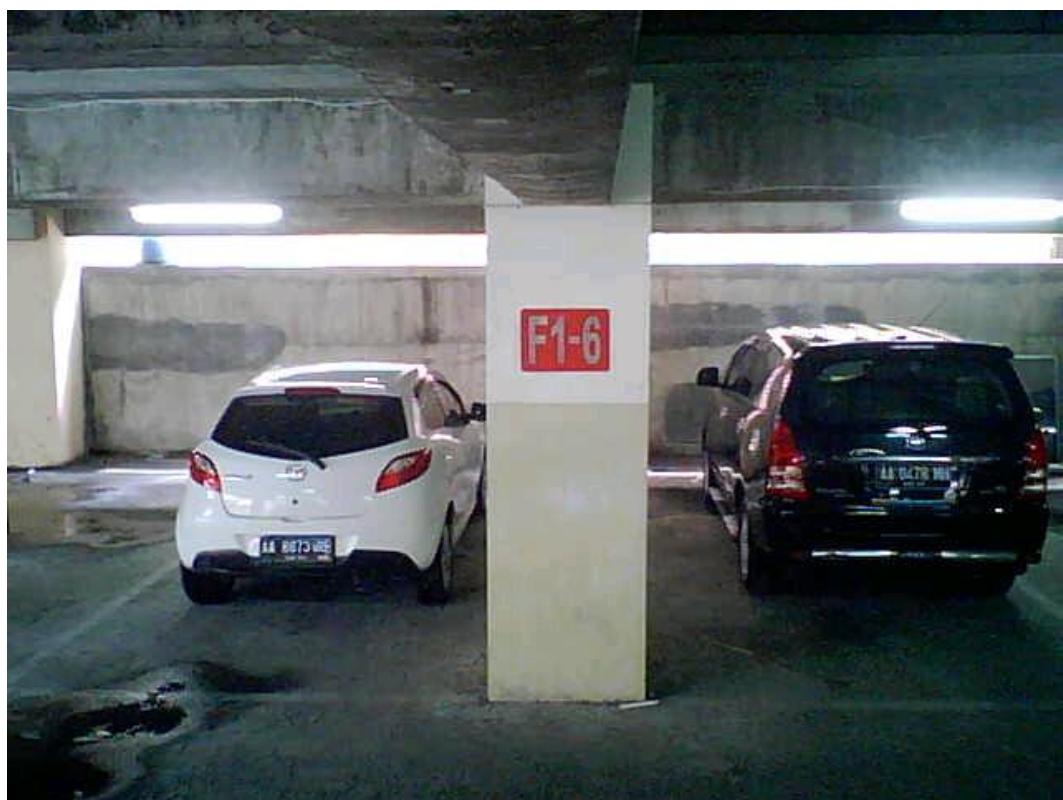


Foreground 5

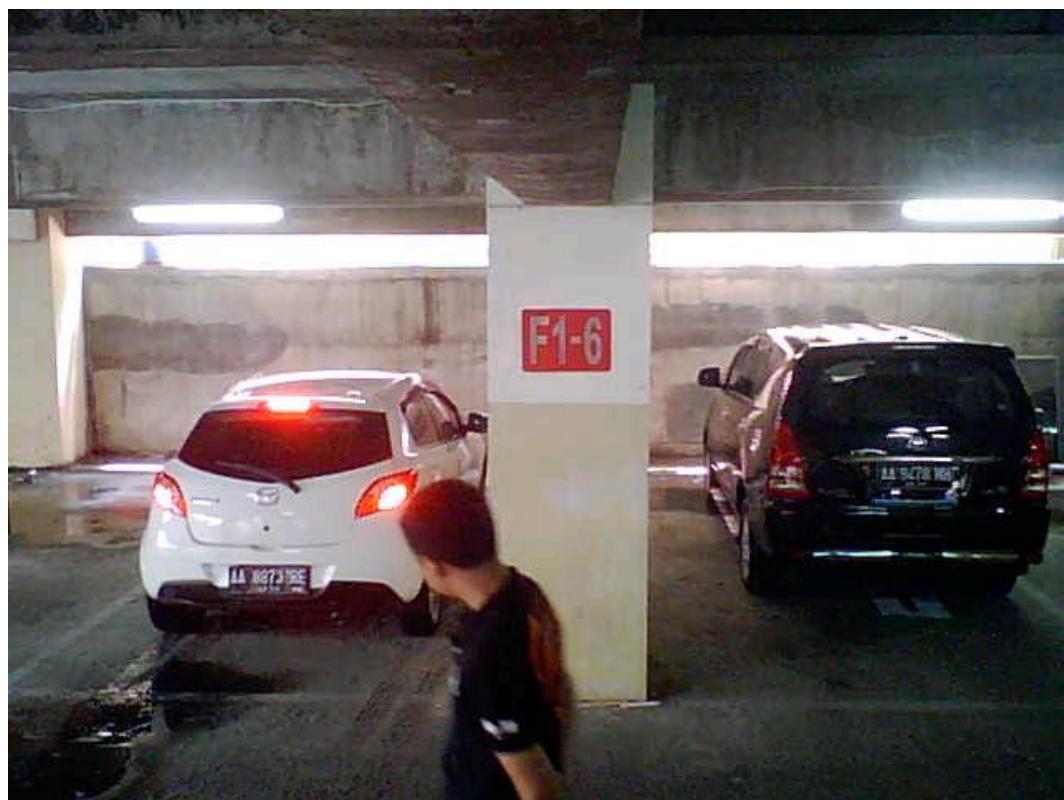




Foreground 6

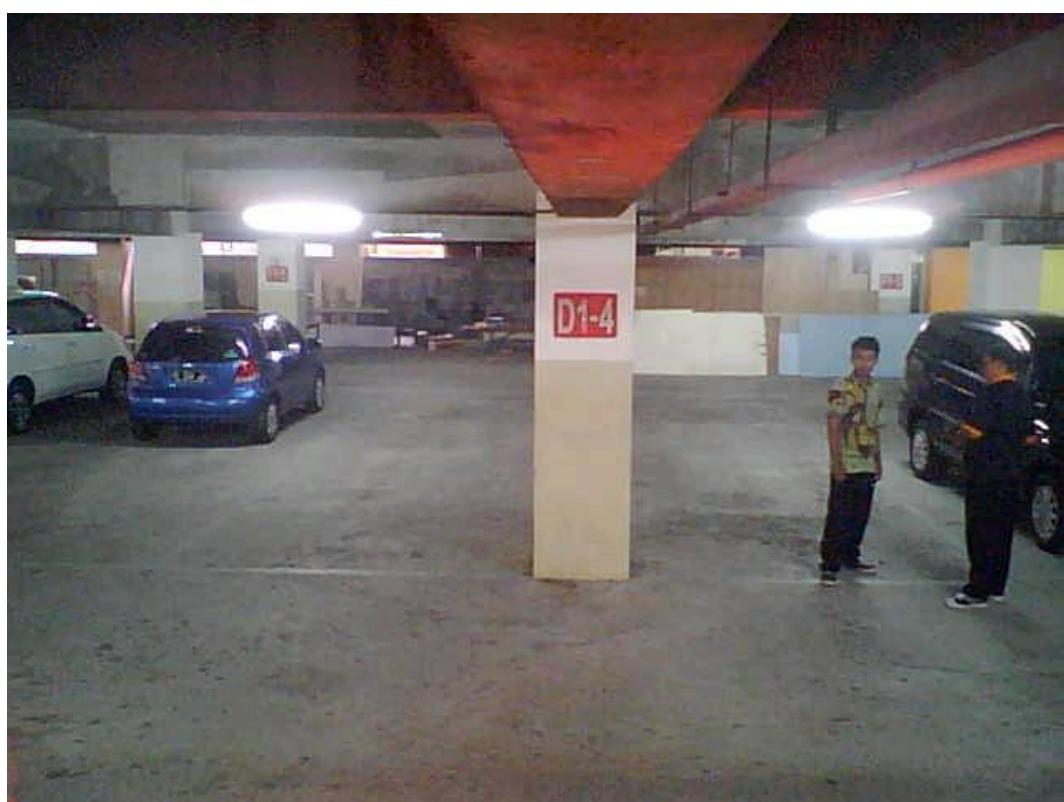






Foreground 7









Foreground 8

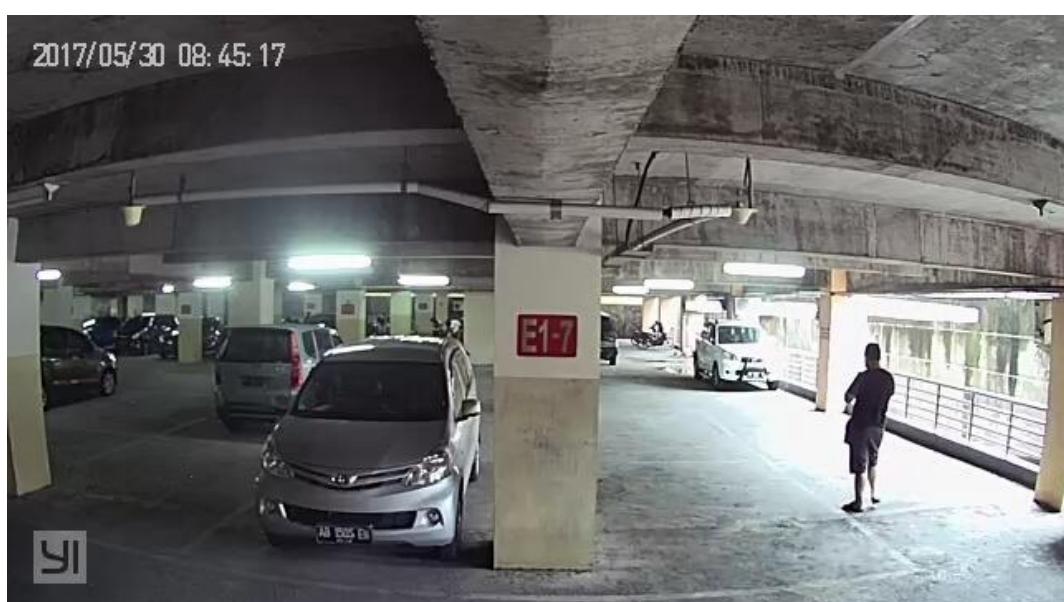
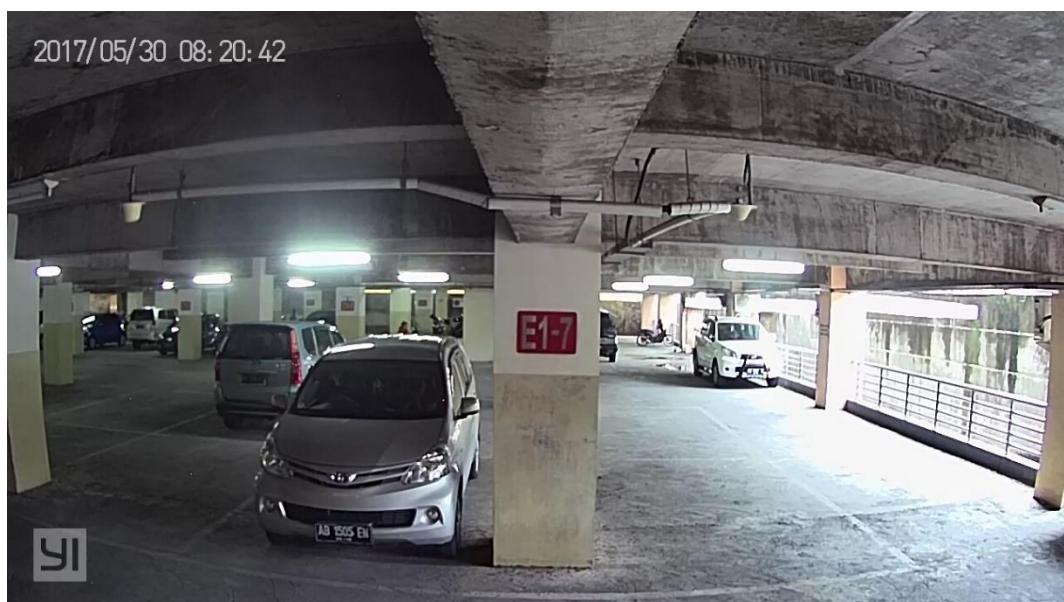






Foreground 9



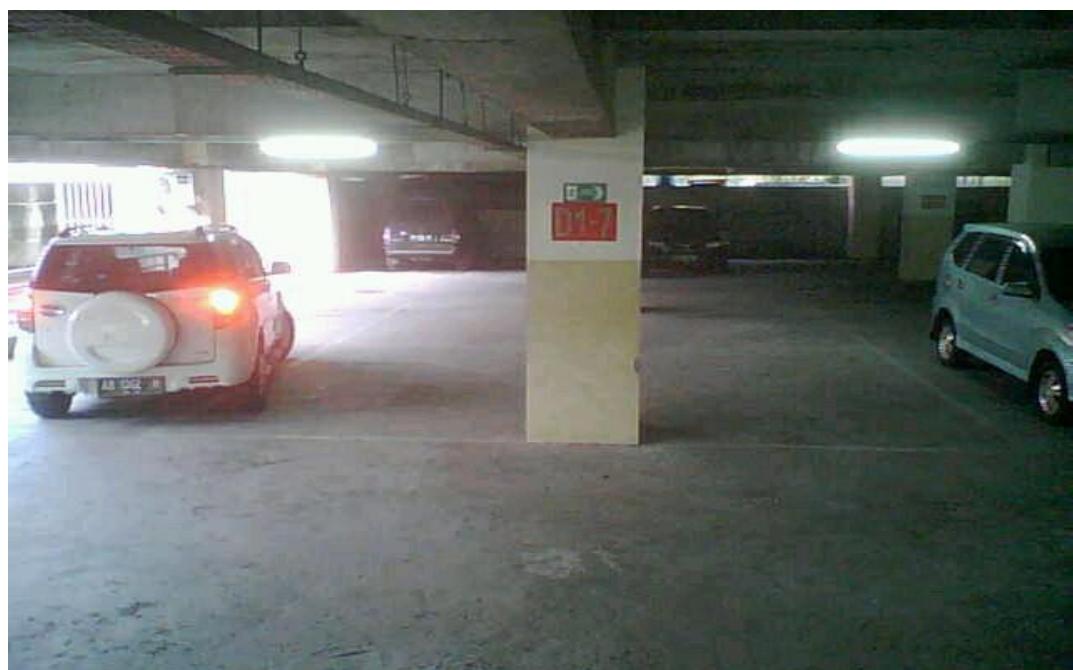








Foreground 10









## Lampiran B – Data.txt

### Data.txt

Data 1

x	y	panjang	lebar
0	310	40	65
70	270	100	115
275	340	85	50
380	270	70	115

Data 2

x	y	panjang	lebar
65	270	65	95
155	285	75	90
320	280	75	110
400	285	50	85

Data 3

x	y	panjang	lebar
75	130	90	90
205	125	90	90

Data 4

x	y	panjang	lebar
0	155	50	70
55	120	95	100
255	145	75	80
360	155	85	70

Data 5

x	y	panjang	lebar
60	290	70	70
220	270	100	85
350	270	80	80

Data 6

x	y	panjang	lebar
90	220	95	110
295	225	105	130

Data 7

x	y	panjang	lebar
0	220	60	75
100	220	100	75
280	225	100	75

Data 8

x	y	panjang	lebar
60	170	80	90
160	175	80	90
325	100	70	100
415	200	30	80

## Lampiran C – Source Code

Fungsi grayscale

```
Mat grayscale(Mat img){  
    Mat grayImage = Mat(img.size(),CV_8UC1);  
    Vec3b* rgbPtr;  
    uchar* grayPtr;  
    for(int y = 0; y<img.size().height;y++){  
        rgbPtr = img.ptr<Vec3b>(y);  
        grayPtr = grayImage.ptr<uchar>(y);  
        for(int x = 0;x<img.size().width;x++){  
            int r = rgbPtr[x][2];  
            int g = rgbPtr[x][1];  
            int b = rgbPtr[x][0];  
            grayPtr[x] = 0.299*r + 0.587*g + 0.114 * b;  
        }  
    }  
    return grayImage;  
}
```

Fungsi Normalisasi

```
void normalisasi(Mat gray1,Mat gray2,int &mean){  
    Mat image1 = gray1.clone();  
    Mat image2 = gray2.clone();  
    ostringstream ai;  
    uchar* ptr1;  
    uchar* ptr2;  
    for(int y=0;y<image1.size().height;y++){  
        ptr1 = image1.ptr<uchar>(y);  
        ptr2 = image2.ptr<uchar>(y);  
        for(int x=0;x<image1.size().width;x++){  
            int a = abs(ptr1[x] - ptr2[x]);  
            if(x>=0 && x<=40 && y>=310 && y<=375){  
                ai << a << " ";  
            }  
            if(a<0)  
                a = 0;  
        }  
    }  
}
```

```

        mean += a;
    }
}

mean = mean/(image1.size().height*image1.size().width);
}

Fungsi Background Subtraction

Mat backgroundSubtraction(Mat image1,Mat image2,int mean){
    uchar* ptr1;
    uchar* ptr2;
    uchar* sub;
    int threshold= mean+10;
    cout << "Threshold : " << threshold << endl;
    Mat subtractImg = Mat(image1.size(),CV_8UC1);
    for(int y=0;y<image1.size().height;y++){
        ptr1 = image1.ptr<uchar>(y);
        ptr2 = image2.ptr<uchar>(y);
        sub = subtractImg.ptr<uchar>(y);
        for(int x=0;x<image1.size().width;x++){
            int a = abs(ptr1[x] - ptr2[x]);
            if(a>=threshold){ //threshold subtraction
                a = 255;
            }else{
                a = 0;
            }
            sub[x] = a;
        }
    }
    return subtractImg;
}

```

### Fungsi ccl 1 (penghitung luas)

```

void ccl(Mat subtractImg,vector<vector<Point>> &objek,int i,bool status[]){
    uchar* point;
    int threshold = subtractImg.size().height*subtractImg.size().width*0.6;
    for(int y = 0;y<subtractImg.size().height;y++){
        point = subtractImg.ptr<uchar>(y);
        for(int x = 0;x<subtractImg.size().width;x++){
            if(point[x]==255){
                queue<Point> antrian;

```

```

vector<Point> daftarKoordinat;
antrian.push(Point(x,y));
daftarKoordinat.push_back(Point(x,y));
point[x] = 0;
do{
    //image.at<Vec3b>(slot1.y1,slot1.x1)[2] = 255;
    int j = antrian.front().x;
    int i = antrian.front().y;
    antrian.pop();
    //atas
    if(i<subtractImg.size().height && subtractImg.at<uchar>(i+1,j)==255){
        antrian.push(Point(j,i+1));
        daftarKoordinat.push_back(Point(j,i+1));
        subtractImg.at<uchar>(i+1,j)=0;
    }
    //atas kanan
    if(i<subtractImg.size().height && j<subtractImg.size().width &&
subtractImg.at<uchar>(i+1,j+1)==255){
        antrian.push(Point(j+1,i+1));
        daftarKoordinat.push_back(Point(j+1,i+1));
        subtractImg.at<uchar>(i+1,j+1)=0;
    }
    //kanan
    if(j<subtractImg.size().width && subtractImg.at<uchar>(i,j+1)==255){
        antrian.push(Point(j+1,i));
        daftarKoordinat.push_back(Point(j+1,i));
        subtractImg.at<uchar>(i,j+1)=0;
    }
    //kanan atas
    if(j<subtractImg.size().width && i>0 && subtractImg.at<uchar>(i-
1,j+1)==255){
        antrian.push(Point(j+1,i-1));
        daftarKoordinat.push_back(Point(j+1,i-1));
        subtractImg.at<uchar>(i-1,j+1)=0;
    }
    //tengah atas
    if(i>0 && subtractImg.at<uchar>(i-1,j)==255){
        antrian.push(Point(j,i-1));
        daftarKoordinat.push_back(Point(j,i-1));
        subtractImg.at<uchar>(i-1,j)=0;
    }
    //atas kiri
    if(i>0 && j>0 && subtractImg.at<uchar>(i-1,j-1)==255){
        antrian.push(Point(j-1,i-1));
        daftarKoordinat.push_back(Point(j-1,i-1));
        subtractImg.at<uchar>(i-1,j-1)=0;
    }
}

```

```

        }
        //kiri
        if(j>0 && subtractImg.at<uchar>(i,j-1)==255){
            antrian.push(Point(j-1,i));
            daftarKoordinat.push_back(Point(j-1,i));
            subtractImg.at<uchar>(i,j-1)=0;
        }
        //atas kanan
        if(j>0 && i<subtractImg.size().height && subtractImg.at<uchar>(i+1,j-1)==255){
            antrian.push(Point(j-1,i+1));
            daftarKoordinat.push_back(Point(j-1,i+1));
            subtractImg.at<uchar>(i+1,j-1)=0;
        }
    }while(!antrian.empty());
    if(daftarKoordinat.size()>threshold){
        cout << "size objek " << i+1 << " : " << daftarKoordinat.size() << endl;
        objek.push_back(daftarKoordinat);
        status[i] = true;
    }
}
}
}
}

Fungsi ccl 2 (menghilangkan objek kecil)

```

```

void ccl(Mat subtractImg){
    uchar* point;
    Mat backup;
    subtractImg.copyTo(backup);
    unsigned int threshold = 1100;
    stack<Point> kecil;
    for(int y = 0;y<subtractImg.size().height;y++){
        point = subtractImg.ptr<uchar>(y);
        for(int x = 0;x<subtractImg.size().width;x++){
            if(point[x]==255){
                queue<Point> antrian;
                vector<Point> daftarKoordinat;
                antrian.push(Point(x,y));
                daftarKoordinat.push_back(Point(x,y));
                point[x] = 0;
                do{
                    //image.at<Vec3b>(slot1.y1,slot1.x1)[2] = 255;
                    int j = antrian.front().x;
                    int i = antrian.front().y;

```

```

        antrian.pop();
        //atas
        if(i<subtractImg.size().height && subtractImg.at<uchar>(i+1,j)==255){
            antrian.push(Point(j,i+1));
            daftarKoordinat.push_back(Point(j,i+1));
            subtractImg.at<uchar>(i+1,j)=0;
        }
        //atas kanan
        if(i<subtractImg.size().height && j<subtractImg.size().width &&
subtractImg.at<uchar>(i+1,j+1)==255){
            antrian.push(Point(j+1,i+1));
            daftarKoordinat.push_back(Point(j+1,i+1));
            subtractImg.at<uchar>(i+1,j+1)=0;
        }
        //kanan
        if(j<subtractImg.size().width && subtractImg.at<uchar>(i,j+1)==255){
            antrian.push(Point(j+1,i));
            daftarKoordinat.push_back(Point(j+1,i));
            subtractImg.at<uchar>(i,j+1)=0;
        }
        //kanan atas
        if(j<subtractImg.size().width && i>0 && subtractImg.at<uchar>(i-
1,j+1)==255){
            antrian.push(Point(j+1,i-1));
            daftarKoordinat.push_back(Point(j+1,i-1));
            subtractImg.at<uchar>(i-1,j+1)=0;
        }
        //tengah atas
        if(i>0 && subtractImg.at<uchar>(i-1,j)==255){
            antrian.push(Point(j,i-1));
            daftarKoordinat.push_back(Point(j,i-1));
            subtractImg.at<uchar>(i-1,j)=0;
        }
        //atas kiri
        if(i>0 && j>0 && subtractImg.at<uchar>(i-1,j-1)==255){
            antrian.push(Point(j-1,i-1));
            daftarKoordinat.push_back(Point(j-1,i-1));
            subtractImg.at<uchar>(i-1,j-1)=0;
        }
        //kiri
        if(j>0 && subtractImg.at<uchar>(i,j-1)==255){
            antrian.push(Point(j-1,i));
            daftarKoordinat.push_back(Point(j-1,i));
            subtractImg.at<uchar>(i,j-1)=0;
        }
        //atas kanan
    }
}

```

```

        if(j>0 && i<subtractImg.size().height && subtractImg.at<uchar>(i+1,j-
1)==255){
            antrian.push(Point(j-1,i+1));
            daftarKoordinat.push_back(Point(j-1,i+1));
            subtractImg.at<uchar>(i+1,j-1)=0;
        }
    }while(!antrian.empty());
if(daftarkoordinat.size()<=threshold){
    for(unsigned int i = 0;i<daftarkoordinat.size();i++){
        kecil.push(daftarkoordinat[i]);
    }
}
}
}

backup.copyTo(subtractImg);
if(!kecil.empty()){
    while(!kecil.empty()){
        subtractImg.at<uchar>(kecil.top().y,kecil.top().x) = 0;
        kecil.pop();
    }
}
}
}

```

## Lampiran D – Dokumen

Kartu Konsultasi Tugas Akhir		
Program Studi Teknik Informatika Fakultas Teknologi Informasi Universitas Kristen Duta Wacana Yogyakarta Dr. Wahidin Sudirahusada 5-25 Yogyakarta, 55224. Telp. (0274)563929		
NIM	: BILLY FANINO BAGYO	
Judul	: Deteksi Lahan Parkir Kosong di UKDW Menggunakan Perbandingan Posisi Objek	
Dosen Pembimbing I	: Kristian Adi Nugraha, S.Kom., M.T.	
1	Tanggal:	Paraf:
	7 Februari 2017	
<ul style="list-style-type: none"><li>- konsultasi program</li><li>- gunakan normalisasi</li></ul>		
2	Tanggal:	Paraf:
	19 Februari 2017	
<ul style="list-style-type: none"><li>- konsultasi program</li></ul>		
3	Tanggal:	Paraf:
	16 Februari 2017	
<ul style="list-style-type: none"><li>- Kongkretisasi program</li><li>- konsultasi laporan</li></ul>		
4	Tanggal:	Paraf:
	16 maret 2017	
<ul style="list-style-type: none"><li>- Demo program</li></ul>		
5	Tanggal:	Paraf:
	30 maret 2017	
<ul style="list-style-type: none"><li>- Demo program</li><li>- konsultasi Bab 3</li></ul>		
6	Tanggal:	Paraf:
	20 April 2017	
<ul style="list-style-type: none"><li>- Demo tampilan</li></ul>		
7	Tanggal:	Paraf:
8	Tanggal:	Paraf:



### Kartu Konsultasi Tugas Akhir

Program Studi Teknik Informatika Fakultas Teknologi Informasi  
Universitas Kristen Duta Wacana Yogyakarta  
Dr. Wahidin Sudirahusada 5-25 Yogyakarta, 55224. Telp. (0274)563929

NIM : BILLY FANINO BAGYO  
Judul : Deteksi Lahan Parkir Kosong di UKDW Menggunakan Perbandingan Posisi Objek  
Dosen Pembimbing II : Dra. Widi Hapsari, M.T.

1	Tanggal: <u>9 Februari 2017</u>	Paraf:
<p>- <del>tanda</del> beri data asli</p>		
3	Tanggal: <u>3 April 2017</u>	Paraf:
<p>- konsultasi laporan - Demo program</p>		
5	Tanggal: <u>20 April 2017</u>	Paraf:
<p>- Demo tampilan</p>		
7	Tanggal:	Paraf:
2	Tanggal: <u>2 Maret 2017</u>	Paraf:
<p>- konsultasi program - bersihkan hasil subtract</p>		
4	Tanggal: <u>20 April 2017</u>	Paraf:
<p>- konsultasi laporan - Demo program</p>		
6	Tanggal: <u>12 Mei - 2017</u>	Paraf:
<p>- ACC Pendadaran</p>		
8	Tanggal:	Paraf:



Program Studi Teknik Informatika  
Fakultas Teknologi Informasi  
Universitas Kristen Duta Wacana Yogyakarta  
Dr. Wahidin Sudirahusada 5-25 Yogyakarta, 55224. Telp. (0274)563929

**FORMULIR PERBAIKAN (REVISI) SKRIPSI**

Strata-1 Program Studi Teknik Informatika

Yang bertanda tangan di bawah ini:

Nama : BILLY FANINO BAGYO  
N I M : 71130126  
Judul Skripsi : DETEKSI LAHAN PARKIR KOSONG DI UKDW MENGGUNAKAN  
PERBANDINGAN POSISI OBJEK  
Tanggal Pendadaran : 26 Mei 2017 13:00 WIB

Telah melakukan perbaikan tugas akhir dengan lengkap.

Demikian pernyataan kami agar dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 4 Juni 2017

Dosen Pembimbing I

Kristian Adi Nugraha, S.Kom., M.T.

Dosen Pembimbing II

Widi Hapsari, Dra. M.T.

Dicetak tanggal: 4 Juni 2017 21:31 WIB



Program Studi Teknik Informatika Fakultas Teknologi Informasi  
Universitas Kristen Duta Wacana Yogyakarta  
Dr. Wahidin Sudirahusada 5-25 Yogyakarta, 55224. Telp. (0274)563929

### FORMULIR CATATAN UJIAN SKRIPSI

(Diisi oleh Ketua Tim Pengaji)

Pada hari ini : **Jumat, 26 Mei 2017**, telah dilakukan Ujian Skripsi untuk mahasiswa tersebut dibawah ini:

Nama Mahasiswa : **BILLY FANINO BAGYO**  
No. Induk Mahasiswa : **71130126**  
Judul Skripsi : **DETEKSI LAHAN PARKIR KOSONG DI UKDW MENGGUNAKAN PERBANDINGAN POSISI OBJEK**  
Dosen Pembimbing I : **Kristian Adi Nugraha, S.Kom., M.T.**  
Dosen Pembimbing II : **Widi Hapsari, Dra. M.T.**  
Keterangan : **LULUS / TIDAK LULUS**  
(coret yang tidak terpilih)

Beberapa perubahan/catatan yang harus dilakukan oleh mahasiswa tersebut diatas terkait dengan skripsi yang dikerjakannya:

NO.	CATATAN PERBAIKAN
1	Tambah sampel data
2	
3	
4	
5	
6	
7	
8	
9	
10	

Perubahan diatas harus sudah diselesaikan paling lambat tanggal : **Senin, 26 Juni 2017**

Yogyakarta, 26 Mei 2017  
Ketua Tim Pengaji

*Dra. Widi Hapsari, MT*

**Catatan:**

- \* 1 (satu) lembar untuk mahasiswa
- \* 1 (satu) lembar untuk arsip