SIMULASI MONITORING OBJEK YANG MASUK DAN KELUAR UNTUK MENGONTROL KETERSEDIAAN LAHAN MENGGUNAKAN VIDEO PROCESSING

MONITORING SIMULATION OF INCOMING AND OUTGOING OBJECT TO CONTROL FIELD AVAILABILITY WITH VIDEO PROCESSING

RANGGI SISTAMA[1] RATRI DWI ATMAJA[2] NUR ANDINI[3]

[1, 2, 3]Prodi S1 Teknik Telekomunikasi, Fakultas Teknik Elektro, Universitas Telkom

[1]Rsistama24@gmail.com [2]ratridwiatmaja@telkomuniversity.ac.id [3]nurandini@telkomuniversity.ac.id

Abstrak

Sistem *monitoring* kendaraan yang masuk dan keluar merupakan suatu program yang dirancang untuk mengontrol ketersedian suatu lahan parkir kendaraan dengan kapasitas lahan parkir tertentu. Pengerjaan Tugas Akhir ini adalah akan membuat salah satu sistem *monitoring* berbasis *video processing*. Sistem ini menggunakan hardware berupa 2 unit *webcam* dengan resolusi 960 x 720 piksel. SIstem yang dibuat bersifat *real-time* di mana sistem akan memproses data secara langsung dan akan langsung memperbarui *database* serta memberi pemberitahuan kepada user mengenai status lahan parkir. Sistem ini menghasilkan nilai akurasi terbaik ketika parameter Difference Threshold bernilai 40000 dan parameter Move Threshold bernilai 0.

Kata kunci: video processing, sistem monitoring, real-time

Abstract

Incoming and outgoing vehicle monitoring system is a program which is designed to control the availability of the parking area with its certain capacity. This Final Project's purpose is to create one of the monitoring system with video processing. The system will use 2 webcams as supporting hardwares with 960 x 720 pixels of video resolution. This system will work in real-time condition where the data samples will directly processed live and the system will also update the database and inform the user about the field condition directly. This system produces the good accuracy value when the Difference Threshold parameter's value is 40000 and Move Threshold parameter's value is 0.

Keywords: video processing, monitoring system, real-time

Pendahuluan

Kota-kota besar sekarang ini telah banyak dibangun tempat umum baik tempat berbelanja, rumah sakit, kantor, atau tempat umum lainnya. Di dalam tempat-tempat tersebut terdapat tempat penitipan kendaraan sementara atau tempat parkir bagi orang berkepentingan tertentu. Sayangnya, pengelolaan tempat parkir ini masih bersifat manual dan kadang masih tidak beraturan. Yang dimaksud tidak beraturan adalah ketika kendaraan tiba di pintu masuk, penjaga masih sering menyatakan bahwa terdapat lahan parkir tanpa memeriksa yang sebenarnya. Akibatnya, ketika kendaraan tersebut masuk ke area parkir dan ternyata tidak menemukan adanya lahan, maka kendaraan tersebut harus mengantri berputar mengelilingi area dalam waktu tertentu untuk mencari parkir yang tersedia, atau dapat pula mengantri dengan kendaraan yang ingin keluar maupun dengan kendaraan lain yang juga tidak mendapatkan lahan untuk parkir. Hal ini mengakibatkan terjadinya penumpukan kendaraan yang menyebabkan kemacetan lokal karena dapat menghalangi mobil yang ingin keluar dari lahan parkirnya sehingga dapat merugikan waktu dan biaya bagi tempat parkir yang harus membayar biaya parkir setiap jamnya.

Dengan permasalahan yang seperti telah dipaparkan di atas, pada tugas akhir ini akan dibuat simulasi suatu sistem yang dapat menjadi fasilitas yang dibutuhkan sebagai solusi dari masalah tersebut. Sistem yang ditawarkan berupa gabungan beberapa hardware dan software. Hardware yang digunakan adalah berupa dua buah kamera yang masing-masing terpasang di pintu masuk dan pintu keluar area parkir. Software yang dibuat nantinya akan mendeteksi dan memproses kendaraan yang masuk atau keluar, serta secara otomatis dan simultan memperbarui status ketersediaan lahan pada area tersebut.

1. Konsep Dasar *Video Processing*

2.1 Citra Digital

Secara garis besar, pengolahan citra digital berlandaskan pada pemrosesan gambar 2 dimensi menggunakan komputer. Citra digital adalah sebuah larik (*array*) yang berisikan atas nilai – nilai *real* maupun kompleks yang direpresentasikan dalam deretan bit [5].

Suatu citra dapat diwakili oleh fungsi *f(x,y)* berukuran M baris dan N kolom, dimana nilai x dan y merupakan koordinat spasial. Nilai pada suatu irisan antara baris dan kolom (pada posisi x,y) disebut *picture elements, image elements*, atau *pixels*. Namun, yang lebih sering digunakan pada citra digital adalah piksel. Citra digital dapat ditulis dalam bentuk matriks sebagai berikut :

$$f(x,y) = \begin{bmatrix} f(1,1) & f(1,2) & f(1,3) \\ f(2,1) & f(2,2) & f(2,3) \\ \vdots & \cdots & \vdots \\ f(M\ 1,1) & f(M\ 1,2) & f(M\ 1,3) \end{bmatrix} \quad (1)$$

$$[\quad f(1) \quad f(2) \quad f(N) \quad]$$

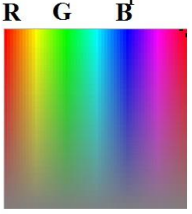
2.2 Citra RGB

RGB merupakan 3 buah warna dasar dalam sistem pengolahan citra. Dalam pengolahan citra warna dipresentasikan dengan nilai hexadesimal dari 0x00000000 sampai 0x00ffffff. Warna hitam adalah 0x00000000 dan warna putih adalah 0x00ffffff. Variabel 0x00 menyatakan angka dibelakangnya adalah *hexadecimal* [6] .

Contoh : 0x00 BB GG RR

BB = Nilai warna *Blue* (biru)
GG = Nilai warna *Green* (hijau)
RR = Nilai warna *Red* (merah)

Range nilai yang akan mengisi BB, GG, dan RR adalah 00 sampai dengan FF dalam skala bilangan hexadesimal, yang apabila kita jadikan desimal yaitu 0 sampai 255. Dengan demikian citra ini memiliki nilai keabuan berjumlah $2^8 = 256$. Range warna yang digunakan adalah $(2^8)\ (2^8)\ (2^8) = 2^{24}$ (atau yang dikenal dengan istilah True Color pada Windows). Nilai warna yang digunakan di atas merupakan gabungan warna cahaya merah, hijau dan biru seperti yang terlihat pada gambar 2. Sehingga untuk menentukan nilai dari suatu warna yang bukan warna dasar digunakan gabungan skala kecerahan dari setiap warnanya [6].



Gambar 1. Komposisi warna RGB [2]

2.3 Citra *Grayscale*

Gambar *grayscale* adalah gambar yang hanya terdiri dari beberapa tingkat warna dari putih hingga hitam. Gambar *grayscale* 8 bit memiliki 256 tingkat warna abu-abu mulai dari putih hingga hitam [6].

Cara mengubah gambar RGB menjadi *grayscale* adalah dengan cara mengganti seluruh nilai RGB pixel-pixelnya menjadi rata-rata jumlah nilai RGB tiap pixel tersebut tersebut. Jika kita memiliki sebuah pixel RGB dengan nilai RGB: 200, 40, 0, maka nilai RGB untuk pixel *grayscale*-nya adalah: 80, 80, 80. Nilai 80 diperoleh dari persamaan berikut

$$\frac{200+40+0}{3} = 80 \quad (2)$$

Selain persamaan di atas, terdapat persamaan yang juga dapat merepresentasikan proses konversi gambar RGB menjadi grayscale yang dapat ditunjukkan pada perasamaan 3 [3] berikut.

$$0.2989 * R + 0.5870 * G + 0.1140 * B \quad (3)$$

2.4 Citra YCbCr

Citra YCbCr banyak digunakan dalam video digital. Dalam format ini terdapat beberapa elemen yaitu Y, Cb, dan Cr. Elemen Y mewakili *luminance* atau tingkat kecerahan dari citra tersebut, sedangkan Cb (*Chrominance blue)* dan Cr (*Chrominance red)* mewakili selisih masing-masing antara komponen warna biru dengan nilai referensinya, serta warna merah dengan nilai referensinya [7]. Dalam proses konversi citra dari RGB ke YCbCr merupakan proses yang *reversible* atau dapat dikonversi dari RGB ke YCbCr atau sebaliknya. Menurut standar ITU CCIR, rumus konversi dari citra RGB ke YCbCr dapat dilakukan dengan menghitung persamaan 3 [4] berikut.

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.257 & 0.504 & 0.098 \\ -0.148 & -0.291 & 0.439 \\ 0.439 & -0.368 & -0.071 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} \quad (4)$$

2.5 Citra Biner

Dalam sebuah citra biner, setiap piksel hanya mempunyai dua kemungkinan dua nilai, seperti *on* dan *off*. Sebuah citra biner disimpan dalam matriks dengan nilai 0 dan 1. Sebuah citra biner disebut juga dengan citra yang hanya berisi hitam dan putih (BW).

Sebuah citra biner dapat disimpan dengan tipe *double* atau *uint8*. Sebuah *array* bertipe *uint8* lebih banyak digunakan daripada *double*, karena tipe *uint8* menggunakan lebih sedikit memori [7].

2.6 Transformasi Citra

Transformasi citra merupakan perubahan bentuk citra untuk mendapatkan suatu informasi tertentu, yang dibagi menjadi dua :

1. Transformasi piksel atau transformasi geometris
   Transformasi piksel ini masih bermain di domain yang sama (domain spasial), hanya posisi piksel yang kadang diubah. Contoh adalah rotasi, translasi, *scaling*, invers dan lain – lain [7].
2. Transformasi Warna
   Merupakan proses perubahan warna suatu citra, baik transformasi dalam ruang warna yang sama (misalkan dalam ruang warna RGB), seperti operasi komplemen warna, pemotongan citra, pengolahan citra dari citra berwarna menjadi citra *grayscale*, citra *black and white* atau sebaliknya [7].

2.7 Video Digital

Menurut The Lycos Tech Glossary, 1999, video digital merupakan manipulasi, pengambilan, serta penyimpanan beberapa gambar yang bergerak yang dapat ditampilkan pada layar komputer. Video digital memiliki beberapa karakteristik yaitu :

1. Resolusi
   Resolusi atau dimensi *frame* merupakan ukuran sebuah *frame*. Resolusi merupakan jumlah piksel yang terdapat pada sebuah *frame* dalam suatu video. Resolusi berpengaruh terhadap detail tampilan warna pada setiap frame pada video tersebut. Jika resolusi *frame* bernilai tinggi, tampilan video tersebut akan semakin baik, namun resolusi yang tinggi membutuhkan jumlah bit yang besar sehingga berpengaruh terhadap ukuran *file* video tersebut.
2. Laju *Frame*
   Laju *frame* (*frame rate*) adalah kecepatan suatu video menampilkan jumlah frame selama video berjalan. Karakteristik ini mendukung tingkat kehalusan suatu video. Semakin cepat laju *frame* pada suatu video akan membuat kualitas video tersebut semakin halus [1].

3 Pembahasan Sistem *Monitoring*

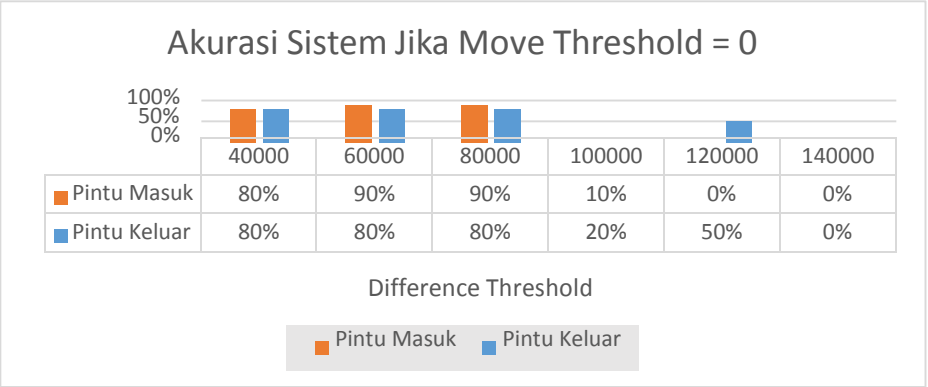
Pada penelitian ini akan dilakukan 3 skenario pengujian untuk mengukur nilai akurasi sistem serta waktu rata-rata yang dibutuhkan sistem untuk melakukan satu kali proses *loop*. Proses *loop* sendiri adalah inti dari proses ini di mana terdapat beberapa perintah yang berjalan secara terus menerus sampai sistem ini dihentikan oleh user atau terjadi kesalahan tertentu pada sistem.

Skenario pertama adalah dengan menjalankan kedua sistem lalu objek pada akan melewati pintu masuk dengan jumlah objek = 10. Kemudian, setelah semua objek masuk, proses selanjutnya adalah kesepuluh objek tersebut akan melewati pintu keluar. Percobaan ini dilakukan untuk mengetahui akurasi sistem pada pintu masuk maupun pada pintu keluar secara terpisah dengan nilai Difference Threshold dan Move Threshold sebagai tolak ukur akurasi sistem.

Skenario kedua adalah dengan menambahkan data percobaan dari skenario pertama namau hanya berfokus pada 1 nilai parameter yaitu Difference Threshold yang bernilai 40000 dan Move Threshold yang bernilai 0. Tujuan dari percobaan ini adalah untuk mengetahui tingkat kestabilan sistem terhadap parameter Difference Threshold yang bernilai 40000 dan Move Threshold yang bernilai 0

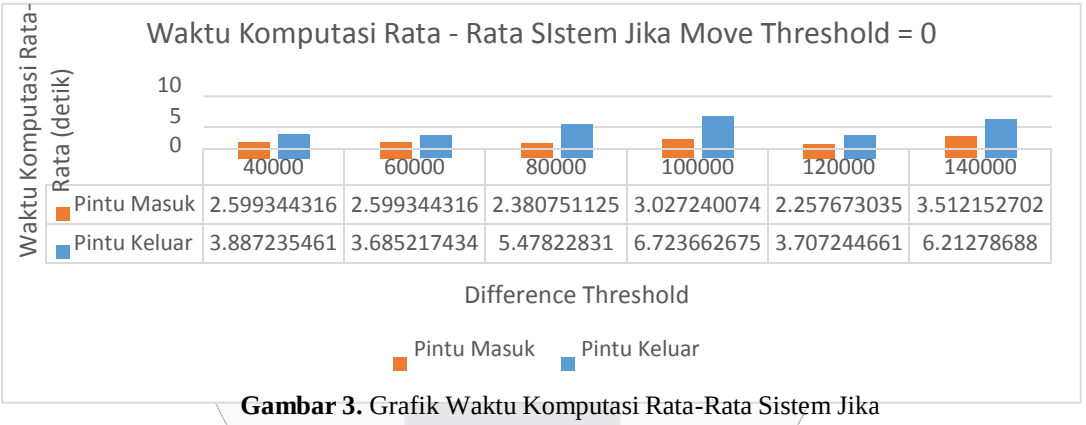
Skenario ketiga adalah dengan menjalankan program pada pintu masuk serta program pada pintu keluar secara bersamaan. Kemudian objek akan melewati pintu masuk lalu kemudian langsung melewati pintu keluar. Tolak ukur keberhasilan dari skenario ini adalah jika tidak terjadi kesalahan di salah satu atau kedua program pada pintu masuk maupun pintu keluar.

a. Skenario 1



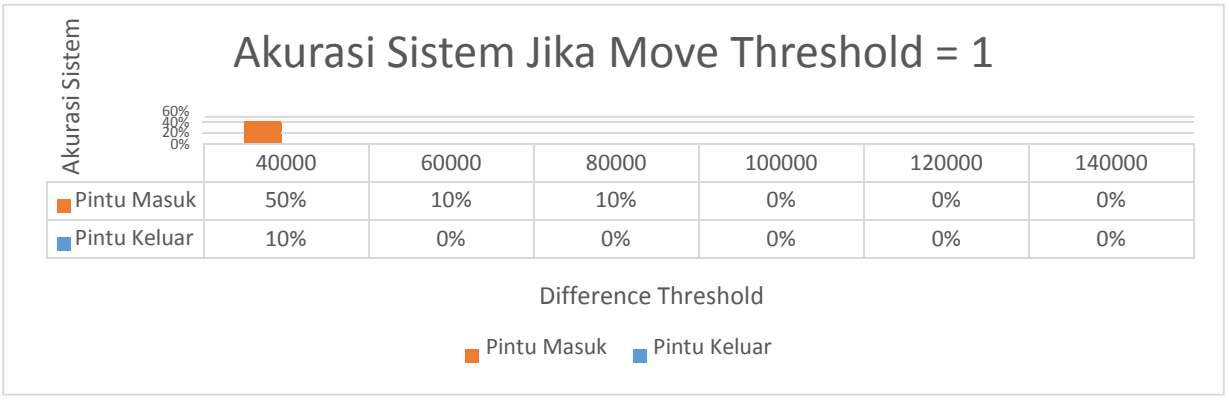
Gambar 2. Grafik Akurasi Sistem Jika Move threshold = 0

Berdasarkan Gambar 2 di atas, akurasi sistem bernilai cenderung tinggi ketika nilai *difference threshold* berada pada nilai kurang dari 100000. Hal ini disebabkan karena nilai maksimum dari *difference* pada *frame* hasil monitoring objek bernilai berkisar antara 80000 sampai 140000. Oleh karena itu, ketika nilai *difference threshold* ditetapkan pada nilai di bawah 100000, sistem akan cenderung lebih sensitif terhadap perubahan yang terjadi pada *webcam* yang dapat berpengaruh pada meningkatnya akurasi. Namun jika memperhatikan grafik akurasi pada parameter *difference threshold* 40000, terdapat nilai akurasi yang bernilai 90%. Hal ini terjadi karena sistem mengalami kesalahan di mana sistem menjadi terlalu sensitif sehingga membuat objek dapat terdeteksi 2 kali. Berdasarkan grafik di atas, sistem mencapai akurasi tertinggi pada kedua sisi yaitu pada saat *difference threshold* yang bernilai 60000. Karena sistem menjadi tidak terlalu sensitif, serta nilai akurasi bernilai baik yaitu 90% pada pintu masuk.dan 80% pada pintu keluar.



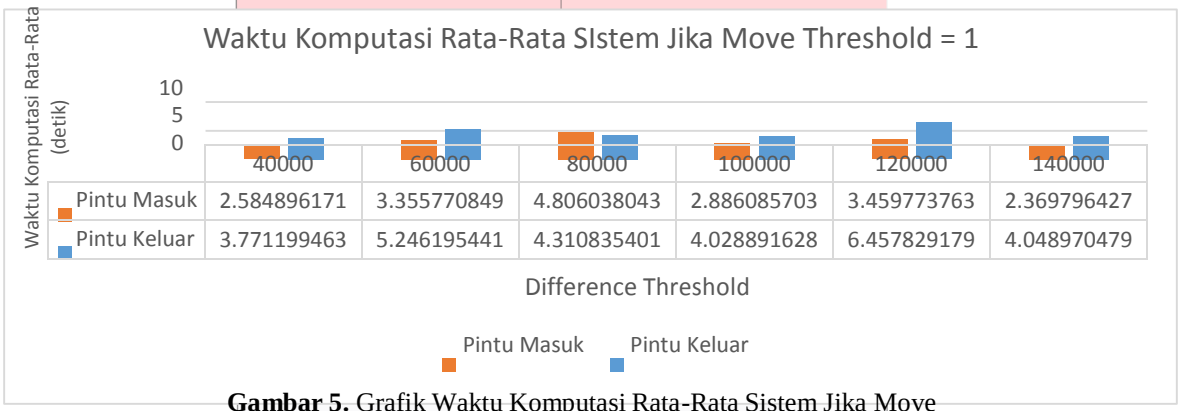
Gambar 3. Grafik Waktu Komputasi Rata-Rata Sistem Jika Move Threshold = 0

Berdasarkan Gambar 3, waktu komputasi pada pintu masuk selalu bernilai lebih rendah daripada pada pintu keluar. Hal ini disebabkan karena adanya proses sinkronisasi *database* dari program pada pintu keluar. Proses membaca database ini dapat memakan waktu yang lebih lama karena terdapat 2 buah variabel yang akan dibaca pada *database* yaitu variabel *count* dan parameter *upper threshold*. Keuntungan dari model sistem seperti ini adalah, keakuratan sinkronisasi *database* dapat terjamin karena pada pintu keluar tidak perlu memasukkan nilai *upper threshold* lagi sehingga dapat meminimalisir terjadinya kesalahan. Selain itu, spesifikasi *hardware* juga dapat berpengaruh terhadap waktu komputasi sistem.



Gambar 4. Grafik Akurasi Sistem Jika Move threshold = 1

Berdasarkan Gambar 4 terlihat bahwa Move threshold sangat berpengaruh terhadap akurasi. Namun pada nilai *difference threshold* dengan nilai 40000 terlihat bahwa pada nilai tersebut sistem memiliki nilai akurasi paling tinggi. Hal ini menunjukkan bahwa jika ingin menggunakan parameter *difference threshold* dengan nilai 40000, maka harus disertai dengan move threshold dengan nilai 1 karena sifat sistem dengan nilai *difference threshold* 40000 yang sangat sensitif terhadap perubahan frame. Dengan adanya move threshold, maka akan meminimalisir kesalahan sistem mendeteksi 2 kali suatu objek.



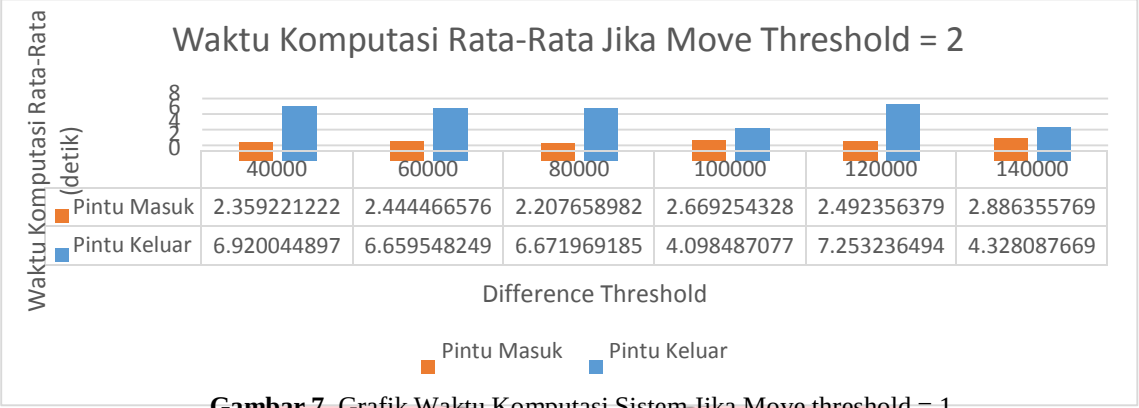
Gambar 5. Grafik Waktu Komputasi Rata-Rata Sistem Jika Move Threshold = 1

Nilai waktu komputasi jika move threshold bernilai 1 ini masih memiliki kesamaan ciri dengan jika move threshold bernilai 0. Kesamaan ciri tersebut adalah waktu komputasi pada pintu keluar tetap lebih besar daripada waktu komputasi pada pintu masuk. Hal ini masih disebabkan oleh hal yang sama, karena faktor waktu tambahan yang dibutuhkan sistem pada pintu keluar untuk melakukan sinkronisasi antara sistem dengan variabel yang terdapat pada database.

Gambar 6. Grafik Akurasi Sistem Jika Move threshold = 1

Berdasarkan Gambar 6, terlihat bahwa akurasi sistem mencapai titik terendah yaitu 0%. Hal ini disebabkan karena perilaku objek yang tidak memenuhi syarat *difference threshold* dalam rentang nilai *move threshold* yang telah ditentukan. Sehingga ketika objek melewati *webcam*, sistem mendeteksi adanya perubahan namun tidak melakukan pembaruan *database*. Namun untuk nilai *difference threshold* pada pintu keluar, terdapat akurasi

bernilai 10% yang berarti terdapat 1 buah objek terdeteksi. Hal ini terjadi karena terjadi kesalahn pada saat proses pengambilan data di mana terdapat perubahan nilai variabel *difference* yang sangat besar dan melewati nilai *move threshold*, sehingga sistem melakukan pembaruan *database* yang salah.



Gambar 7. Grafik Waktu Komputasi Sistem Jika Move threshold = 1

Jika membandingkan 3 grafik waktu komputasi secara keseluruhan terhadap parameter move threshold dan difference threshold, maka dapat ditarik kesimpulan bahwa move threshold dan difference threshold tidak berpengaruh terhadap waktu komputasi sistem. Karena nilai waktu komputasi dapat berubah secara tidak tentu tanpa dipengaruhi oleh kedua parameter tersebut. Nilai waktu komputasi rata-rata terbesar terletak pada pengujian sistem dengan difference threshold 120000 dan move threshold yang bernilai 2 yaitu 7.253236 detik . Hal ini kemungkinan besar dipengaruhi oleh kemampuan komputer dalam memproses data selama program berlangsung

b. Skenario 2

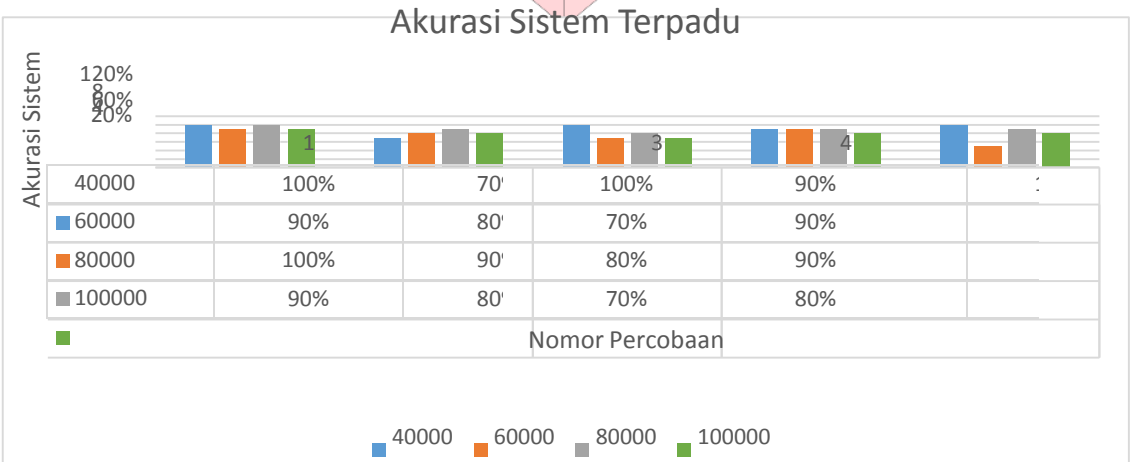
Gambar 8 Grafik Akurasi Sistem Untuk 10 Percobaan dengan Parameter Difference Threshold bernilai 40000 dan Move Threshold bernilai 0

Berdasarkan grafik pada Gambar 8, nilai akurasi sistem bernilai di atas 70% untuk setiap percobaan. Hal ini disebabkan karena piksel yang dihasilkan dari hasil operasi absolute difference pada proses loop pada sistem ini memenuhi syarat parameter Difference Threshold pada percobaan ini yaitu 40000, sehingga sistem dapat mendeteksi objek dengan benar untuk setiap frame yang terdapat objek yang dimaksud. Kesalahan yang terjadi disebabkan karena terdapat frame yang memiliki nilai piksel objek yang tidak memenuhi nilai Difference Threshold sehingga objek pada frame tersebut tidak terdeteksi oleh sistem. Hal ini disebabkan karena proses pengambilan frame tidak tepat pada frame yang memiliki nilai frame objek yang memenuhi syarat Difference Threshold. Dengan dilakukannya pengujian ini, maka parameter Difference Threshold yang bernilai 40000 dan Move Threshold yang bernilai 0, merupakan parameter yang menghasilkan nilai akurasi yang cukup baik dan cukup stabil bagi sistem ini.

Gambar 9 Grafik Waktu Komputasi Rata-Rata Untuk 10 Percobaan dengan Parameter Difference Threshold bernilai 40000 dan Move Threshold bernilai 0

Berdasarkan grafik pada Gambar 9 terlihat bahwa waktu komputasi pada pintu masuk bernilai lebih kecil daripada waktu komputasi pada pintu keluar. Hal ini disebabkan karena sistem pada pintu keluar membutuhkan waktu tambahan untuk proses sinkronisasi database yang terdapat pada sistem pada pintu masuk.

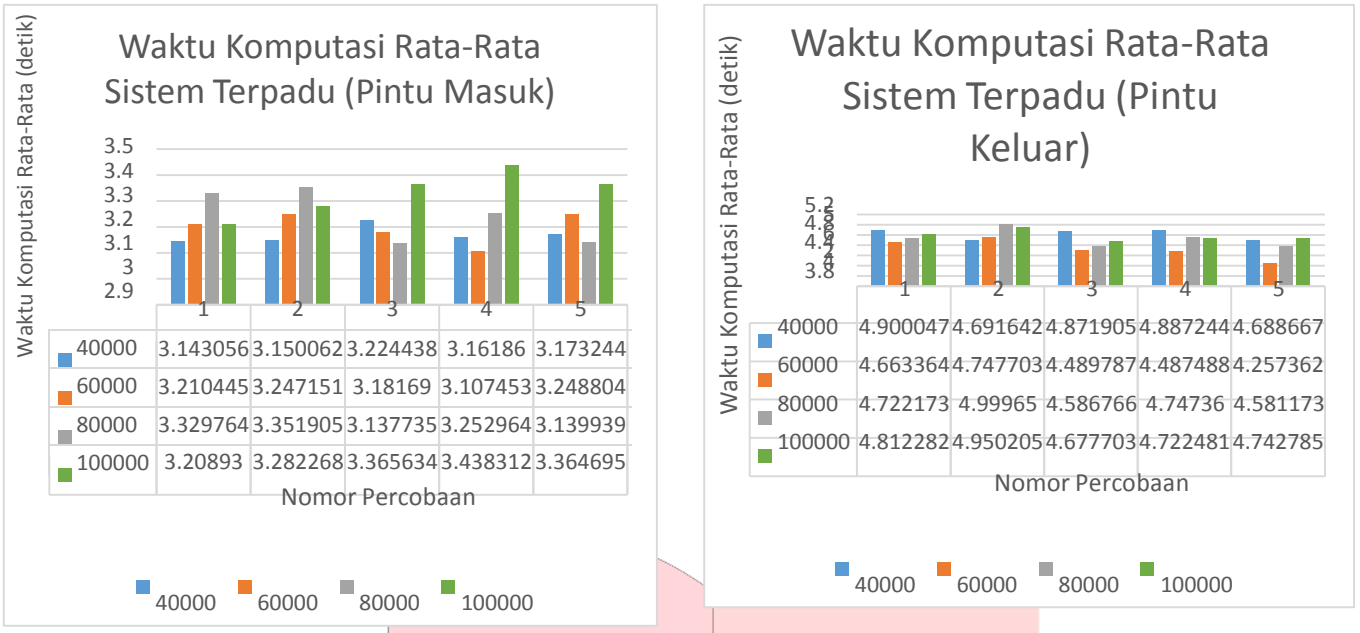
c. Skenario 3



Gambar 10 Grafik Akurasi Sistem Terpadu Untuk 5 Percobaan dengan Parameter Difference Threshold bernilai 40000, 60000, 80000, 100000 dan Move Threshold bernilai 0

Berdasarkan grafik pada Gambar 10, hasil akurasi terbaik terletak pada percobaan pertama. Pada percobaan ini, nilai akurasi mencapai 100% untuk parameter Difference Threshold yang bernilai 80000 dan 40000. Hal ini disebabkan karena pada parameter tersebut, nilai hasil dari operasi absolute difference dari nilai piksel frame yang diinginkan dengan frame before dan background pada proses loop masih dalam syarat nilai Difference Threshold, sehingga sistem dapat mendeteksi frame yang terdapat objek di dalamnya dengan benar.

Berdasarkan grafik pada Gambar 10, kesalahan pada sistem ini banyak terjadi pada percobaan kedua, Kesalahan yang terjadi adalah objek terdeteksi 2 kali atau objek tidak terdeteksi sama sekali. Objek terdeteksi 2 kali disebabkan karena perilaku objek tersebut yang bersifat random untuk semua percobaan. Kesalahan ini terjadi karena ketika pemrosesan frame suatu objek, objek tersebut tidak langsung meninggalkan kamera, namun masih menyisakan sedikit bagian dari objek tersebut sehingga terdeteksi sebagai frame objek baru. Sedangkan kesalahan objek yang tidak terdeteksi sama sekali disebabkan karena ketepatan pengambilan frame objek yang diinginkan bukan merupakan frame yang mencakup banyak piksel objek, sehingga syarat Difference Threshold tidak terpenuhi.



Gambar 11 Grafik Waktu Komputasi Rata-Rata Sistem Terpadu Untuk 5 Percobaan dengan Parameter Difference Threshold bernilai 40000, 60000, 80000, 100000 dan Move Threshold bernilai 0

Berdasarkan grafik pada Gambar 11, terlihat bahwa waktu komputasi pada pintu keluar bernilai lebih besar daripada waktu komputasi pada pintu masuk. Hal ini disebabkan karena terdapat proses sinkronisasi database pada sistem pada pintu keluar. Informasi pada database tersebut harus ditentukan terlebih dahulu pada sistem pada pintu masuk. Sehingga proses komputasi pada pintu keluar memakan waktu lebih lama daripada waktu komputasi pada pintu masuk.

4 Kesimpulan Penelitian

Berdasarkan yang telah direalisasikan dan diamati, maka dari penelitian sistem simulasi monitoring objek yang masuk dan keluar untuk mengontrol ketersediaan lahan menggunakan *video processing* ini dapat ditarik beberapa kesimpulan antara lain:

1. Sistem simulasi monitoring objek dengan video processing ini memiliki keterbatasan mengenai sensitivitas pada parameter tertentu terhadap perubahan situasi pada lapangan. Hal ini akan berpengaruh pada nilai akurasi dari sistem itu sendiri.
2. Nilai akurasi rata-rata sistem bernilai cukup baik ketika parameter Difference Threshold bernilai 40000 dan Move Threshold bernilai 0.
3. Waktu komputasi rata-rata sistem pada pintu keluar selalu lebih besar daripada sistem pada pintu masuk. Hal ini disebabkan karena adanya proses sinkronisasi database dengan sistem pada pintu masuk.
4. Sistem monitoring objek ini hanya mendeteksi 1 buah objek dalam 1 kali pemrosesan, sehingga rentan terjadi kesalahan jika kondisi lahan padat.

Daftar Pustaka

[1] Anonim, *Operation Manual : SONY VEGAS PRO 11.0*, SONY Corporation, Japan, 2011
[2] Anonim, *Edit Colors : Paint 6.2(Build 9200)*, Microsoft Corporation, United States, 2012
[3] Anonim, *Product Help :MATLAB 7.8.0 (R2009a)*, The MathWorks, United States, 2009
[4] Kuo, Sen, M dkk. 2006. *Real Time Signal Processing*. Edisi ke 2. London.
[5] Prasetyo, Eko. 2012. *Pengolahan Citra Digital dan Aplikasinya Menggunakan Matlab*. Yogyakarta. Penerbit Andi.
[6] Ryanto. *"Praktikum 2 Dasar Pengolahan Citra (1)"*. diunduh pada 30, November, 2013. http://lecturer.eepis-its.edu/~riyanto/**citra**-bab2.pdf .
[7] Wijaya, Marvin Ch dan Agus Priyono. 2007. *Pengolahan Citra Digital Menggunakan MatLab Image Processing Toolbox*. Bandung. Informatika.