Coding : Jurnal Komputer dan Aplikasi
Volume 08, No. 01 (2020), Hal 132-141 ISSN 2338-493X

APLIKASI DETEKSI OBJEK BERGERAK BERBASIS CITRA DENGAN METODE *BACKGROUND SUBTRACTION* dan *BLOB DETECTION* (STUDI KASUS: MAMI MART KUBU RAYA)

[1]Putri Apriani, [2]Ikhwan Ruslianto, [3]Uray Ristian
[1] [2] [3] Jurusan Rekayasa Sistem Komputer, Fakultas MIPA Universitas Tanjungpura

Jl. Prof. Dr. H. Hadari Nawawi, Pontianak
Telp./Fax.: (0561) 577963
e-mail: [1]putriaprianiptri@gmail.com, [2]ikhwanruslianto@siskom.untan.ac.id,
[3]eristian@siskom.untan.ac.id

ABSTRAK

Video adalah rekaman gambar hidup atau program televisi untuk ditayangkan lewat pesawat televisi. Video juga bisa dikatakan sebagi gambar-gambar mati yang ditangkan secara berurutan dalam satuan waktu *frame per second(fps)*. Banyak sekali informasi yang didapat melaui sebuah video, seperti aktor didalam video, tempat pembuatan video dan waktu pembuatan video. Video banyak dimanfaatkan untuk kepentingan usaha yaitu sebagai sistem deteksi atau pemantauan yang memanfaatkan data dari *cctv (closed circuit television)*. Salah satunya adalah untuk mendeteksi adanya objek yang bergerak atau yang berjalan disekitar tempat usaha, dengan tujuan untuk melihat banyaknya pergerakkan dari objek yang bergerak. Penelitian ini bertujuan untuk membangun sebuah sistem deteksi objek bergerak berbasis citra. Deteksi merupakan proses untuk memeriksa atau melakukan pemeriksaan terhadap suatu benda dengan menggunakan cara dan teknik tertentu. Penelitian ini menggunakan metode *Background Subtraction* dan *Blob Detection*. Metode *Background Subtraction* digunakan sebagai pemisah antara objek dan *background*. Metode *Blob Detection* digunakan sebagai deteksi citra bergerak. Sistem deteksi objek bergerak ini dilakukan berdasarkan nilai aspek *ratio*. Berdasarkan hasil pengujian menggunakan 40 data uji dengan 3 kali pengujian menghasilkan persentase keberhasilan dalam deteksi objek bergerak pada saat keadaan terang sebesar 70% dan deteksi objek bergerak dalam keadaan gelap sebesar 75 %.

Kata Kunci : Deteksi, *Background Subtraction, Blob Detection.*

1. PENDAHULUAN

Video adalah rekaman gambar hidup atau program televisi untuk ditayangkan lewat pesawat televisi [1]. Banyak sekali informasi yang kita dapat melalui sebuah video seperti aktor di dalam video, *background* tempat pembuatan video, waktu pembuatan video, dan lain- lain. Video banyak dimanfaatkan untuk kepentingan usaha yaitu sebagai sistem deteksi atau pemantauan yang memanfaatkan data dari *cctv (closed circuit television)*. Salah satunya adalah untuk mendeteksi adanya objek yang bergerak atau yang berjalan disekitar tempat usaha, dengan tujuan untuk melihat banyaknya pergerakkan dari objek yang bergerak.

Berdasarkan uraian diatas maka perlu dirancang aplikasi untuk deteksi objek yang bertujuan untuk bertujuan untuk memonitoring atau memantau suatu tempat. Berapa penelitian yang telah dilakukan yang membahas tentang sistem deteksi melalui video.

Penelitian pertama yaitu tentang Implementasi Metode *Background Subtraction* Dalam Sistem Analisis Trayektori Hasil Latihan Lompat Jauh Berbasis Pengolahan Citra Digital [2]. Pada penelitian ini metode *background subtraction* diterapkan untuk melihat adanya gerak trayektori dan data tersebut disimpan pada database, kemudian untuk deteksi objek peneliti menggunakan metode Kalman Filter. Penelitian yang kedua tentang Aplikasi Pendeteksi Objek Bergerak Pada *Image Sequence* dengan Metode *Background Subtraction* [3]. Penelitian ini mendeteksi adanya mobil yang bergerak dan mobil yanag bergerak tersebut ditandai dengan *boundingbox* kemudian dideteksi dengan

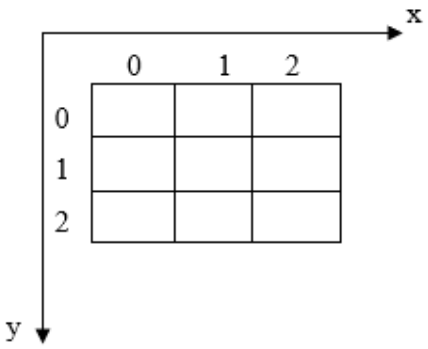
metode *Background Subtraction*. Penelitian yang ketiga tentang Aplikasi Penghitung Kendaraan Pada Jalur Pantura Menggunakan *Blob Detection* dan Kalman Filter [4]. Penelitian ini mendeteksi dan menghitung jumlah kendaraan yang lewat pada jalur pantura, kemudian data disimpan di perangkat kemudian dideteksi dengan metode *Blob Detection*.

Berdasarkan penelitian sebelumnya tentang deteksi objek bergerak dan berdasarkan penelitian-penelitian yang telah menerapkan metode *Background Subtarction* dan *Blob Detection*, akan dibuat penelitian yang dapat mendeteksi objek bergerak berbasis citra dengan menerapkan metode *Background Subtraction* dan *Blob Detection*. Penelitian ini bertujuan untuk mendeteksi banyaknya pergerakkan pada objek yang bergerak. Adapun penelitian yang akan dibuat yaitu Aplikasi Deteksi Objek Bergerak dengan Metode *Background Subtraction* dan *Blob Detection*. Aplikasi yang dibuat adalah aplikasi berbasis *desktop*.

2. TINJAUAN PUSTAKA

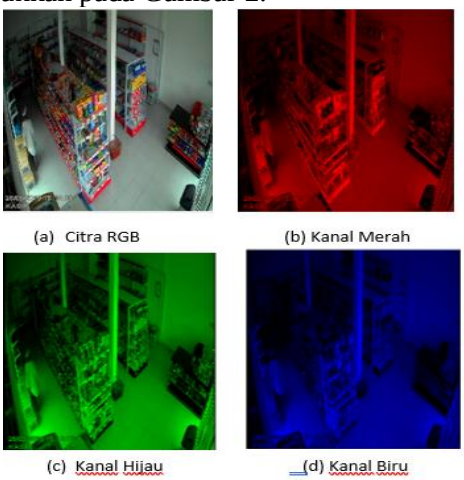
2.1 Pengolahan Citra

Pengolahan citra merupakan bidang ilmu yang mempelajari tentang bagaimana suatu citra itu dibentuk, diolah, dan dianalisis sehingga menghasilkan informasi yang dapat dipahami oleh manusia. Citra tersusun oleh sekumpulan *pixel* yang memiliki koordinat (x, y) dan amplitudo $f(x, y)$ [5]. Representasi koordinat citra digital ditunjukkan pada Gambar 1.



Gambar 1. Representasi Koordinat Citra Digital

Pada umumnya, berdasarkan kombinasi warna pada *pixel*, citra dibagi menjadi tiga jenis yaitu citra RGB, citra *grayscale*, dan citra biner. Citra pada Gambar 2.3 termasuk dalam jenis citra *RGB truecolor* 24-*bit*. Citra tersebut tersusun oleh kombinasi warna yaitu merah, hijau, dan biru. Representasi citra RGB dan masing-masing kanal warna penyusunnya ditunjukkan pada Gambar 2.



Gambar 2. Representasi Citra RGB

Jenis citra yang kedua adalah citra *grayscale*. Citra *grayscale* merupakan citra yang nilai intensitas *pixel* nya didasarkan pada derajat keabuan. Pada citra *grayscale* 8-*bit*, derajat warna hitam sampai dengan putih dibagi ke dalam 256 derajat keabuan di mana warna hitam sempurna direpresentasikan dengan nilai 0 dan putih sempurna dengan nilai 255. Persaman untuk mencari nilai *grayscale* pada citra dapat dilihat pada Persamaan 1.

$$Gray = 0.299R + 0.5870G + 0.1140B \quad (1)$$

Citra RGB dapat dikonversi menjadi citra *grayscale* sehingga dihasilkan hanya satu warna yaitu abu-abu. Citra hasil konversi RGB menjadi *grayscale* ditunjukkan pada Gambar 3.

Gambar 3. Hasil Konversi RGB ke *Grayscale*

Jenis citra yang ketiga adalah citra biner. Citra biner adalah citra yang *pixel* nya memiliki kedalaman *bit* sebesar 1 *bit* sehingga hanya memiliki dua nilai intensitas warna yaitu 0 (hitam) dan 1 (putih). Citra *grayscale* dapat dikonversi menjadi citra biner melalui proses *thresholding*. *Thresholding* adalah metode yang menetapkan suatu nilai, kemudian hanya mengambil nilai diatasnya saja/dibawahnya saja.

Sedangkan nilai selainnya diabaikan [6]. Berdasarkan nilai ambang *threshold* (T) dapat ditunjukkan dengan Persamaan 2.

$$f_{(x,y)} = \begin{cases} 1, jika\ I_{(x,y)} > T \\ 0, jika\ \ I_{(x,y)} \leq T \end{cases} \quad (2)$$

2.2 *Background Subtraction*

Background Subtraction, yang juga dikenal sebagai *foreground detection*, adalah salah satu teknik pada bidang pengolahan citra dan *computer vision* yang bertujuan untuk mendeteksi/mengambil *foreground* dari *background* untuk diproses lebih lanjut (seperti pada proses *object recognition*). Umumnya *foreground* yang diinginkan adalah berupa objek manusia, hewan, mobil, dan objek lain. Penarapan metode *Background Subtraction* pada penelitian ini adalah untuk memisahkan *background* dengan objek atau yang disebut dengan proses *foreground segmentation* [7]. Persamaan yang digunakan untuk proses *foreground segmentation* dapat dilihat pada Persamaan 3.

$$f_{(x,y)} = \begin{cases} 1, jika\ \left|I_{(x,y)} - I'_{(x,y)} - 1\right| > T \\ 0, jika\ \left|I_{(x,y)} - I'_{(x,y)} - 1\right| \leq T \end{cases} \quad (3)$$

Keterangan:

$I_{(x,y)}$: *Frame background* model
$I'_{(x,y)}$ *: Frame* objek bergerak
T : Nilai *Threshold*

Pemodelan dari hasil *foreground segmentation* dapat dilihat pada Gambar 4.

Gambar 4. Hasil *Foreground Segmentation*

2.3 *Blob Detection*

Blob Detection adalah metode yang digunakan untuk menentukan suatu grup dari *pixel* saling berhubungan satu sama lain atau tidak. Pada penelitian ini, *Blob Detection* diterapkan untuk mendeteksi adanya objek bergerak pada video, dan memberikan nilai aspek *ratio* pada proses *Blob Detection*. Aspek *ratio* adalah angka yang menunjukkan perbandingan panjang dan lebar suatu bidang gambar. Nilai aspek *ratio* yang digunakan pada penelitian ini sebanding yaitu lebar *pixel* x >20 dan panjang *pixel* y >20. Metode *Blob Detection* sangat berguna untuk mengidentifikasi objek yang terpisah-pisah pada suatu citra atau menghitung jumlah suatu objek pada suatu citra [8]. Adapun hasil *Blob Detection* dapat dilihat pada Gambar 5.

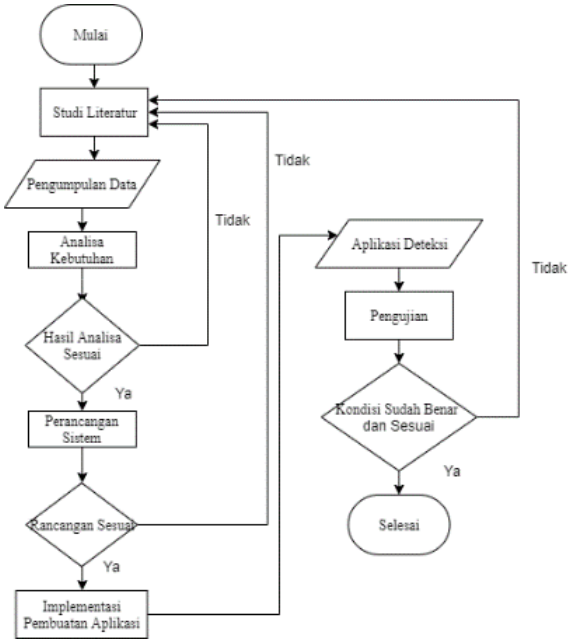
Gambar 5. Hasil *Blob Detection*

2.4 OpenCV

OpenCV (*Open Source Computer Vision Library*) adalah sebuah *library open source* yang dikembangkan oleh intel yang fokus untuk menyederhanakan *programing* terkait citra digital. OpenCV bersifat *open source*, bebas digunakan untuk hal-hal yang bersifat akademis maupun komersial. Di dalam OpenCV, terdapat *interface* untuk bahasa pemrograman C, C++, phyton, dan java yang nantinya dapat berjalan pada windows, linux, android, dan mac [9].

3. METODOLOGI PENELITIAN

Metode penelitian merupakan rencana penelitian untuk menerapkan sistem yang dibuat. Penelitian dilakukan dengan 7 tahap yaitu studi literatur, pengumpulan data, analisis kebutuhan, perancangan, implementasi, dan pengujian yang ditunjukkan pada Gambar 6.

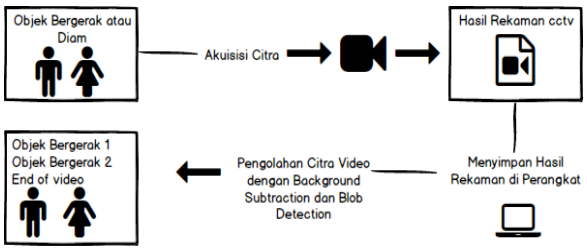


Gambar 6. *Flowchart* Penelitian

4. PERANCANGAN

4.1 Rancangan Sistem

Rancangan sistem yang akan dibangun merupakan sistem deteksi objek bergerak berbasis citra dengan menggunakan metode *Background Subtraction* dan *Blob Detection* dan berbasis *desktop* aplikasi dibangun dengan Visual Studio. Rancangan sistem secara umum ditunjukkan pada Gambar 7.



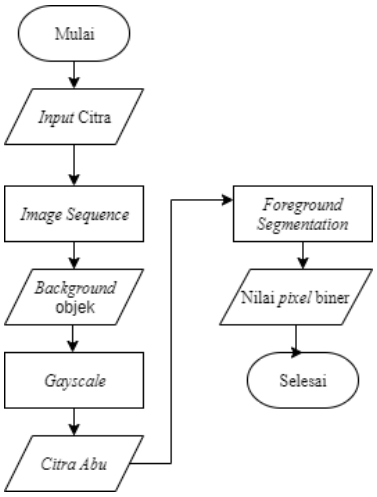
Gambar 7. Rancangan Sistem Secara Umum

Adapun penjelasan lebih lanjut dari sistem ini adalah sebagai berikut:

1. Video yang diambil merupakan video objek bergerak yang melewati/berjalan disuatu tempat.
2. Setelah video diambil menggunakan *cctv*/kamera statis, citra video tersebut akan disimpan ke dalam perangkat dengan format *file*.mp4 dan sudah di *resize* dengan ukuran ≤ 5Mb, agar mempercepat kerja sistem. Kemudian *file* video yang telah disimpan dan di *resize* tersebut dimasukkan ke dalam sistem aplikasi.
3. Pada tahap ini citra video telah berada di dalam aplikasi. Citra akan diproses oleh Visual Studio dengan tahapan metode *Background Subtraction*. Selanjutya proses mengubah citra asli RGB menjadi citra *grayscale* dan mengubah citra *grayscale* menjadi citra biner. Selanjutnya dilakukan proses deteksi menggunakan metode *Blob Detection*.

4.2 Flowchart Metode *Background Subtraction*

Flowchart metode *Background Subtraction* adalah urutan proses yang digunakan dalam membuat sistem aplikasi deteksi objek bergerak. *Flowchart* metode *Background Subtraction* dapat dilihat pada Gambar 8.



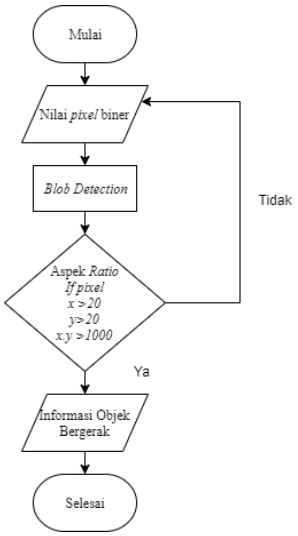
Gambar 8. *Flowchart Background Subtraction*

Penjelasan rincian dari masing-masing tahapan dijelaskan sebagai berikut:

1. Proses dimulai dengan menginputkan video.
2. Citra video akan masuk ke proses *image sequence* yaitu pembacaan antara *frame* objek dengan *frame background*.
3. Setelah proses *image sequence*, citra video akan diubah warnanya menjadi citra abu atau *grayscale*.
4. Selanjutnya masuk ke tahap *foreground segmentation* yaitu proses pemisahan antara objek dan *background* dengan mengubah nilai citra *grayscale* menjadi citra biner melalui proses *thresholding* dengan nilai *threshold 30* dan dengan kondisi seperti pada Persamaan 3.
5. Proses selesai.

4.3 Flowchart Metode *Blob Detection*

Flowchart metode *Blob Detection* menunjukkan proses deteksi objek bergerak dengan perbandingan nilai aspek *ratio*. Nilai aspek *ratio* yang digunakan pada penelitian ini adalah lebar *pixel* x>20 dan panjang *pixel* y>20. Flowchart metode *Blob Detection* dapat dilihat pada Gambar 9.



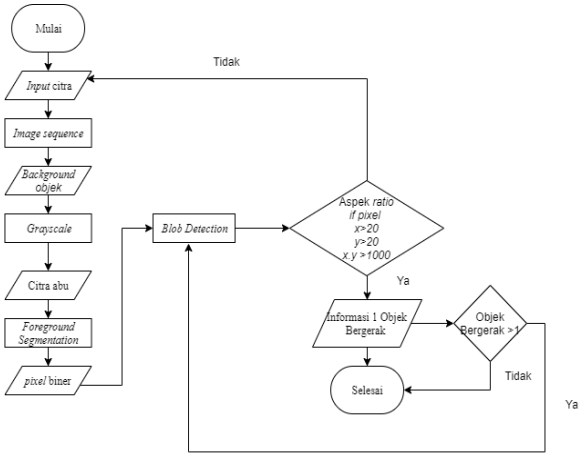
Gambar 9. *Flowchart Blob Detection*

Penjelasan rincian dari masing-masing tahapan dijelaskan sebagai berikut:

1. Proses dimulai dengan menginputkan nilai *pixel* biner.
2. Nilai *pixel* biner yang didapat akan melalui proses *Blob Detection* dimana nilai *pixel* akan disesuaikan dengan nilai aspek *ratio* yang ditelah ditentukan sebelumnya.
3. Setelah proses penyesuaian dengan nilai aspek *ratio*, dan jika nilai *pixel* tersebut sesuai, paka citra akan terdeteksi sebagai citra objek bergerak.
4. Sistem memberikan informasi berupa tulisan adanya objek bergerak yang berhasil terdeteksi.
5. Proses selesai.

4.4 *Flowchart* Pengujian

Flowchart pengujian adalah urutan proses sistem aplikasi dalam melakukan pengujian data. *Flowchart* data dapat dilihat pada Gambar 10.



Gambar 10. *Flowchart* Pengujian

Dengan penjelasan rincian dari masing-masing tahapan akan dijelaskan sebagai berikut:

1. Proses dimulai dengan melakukan *input* citra yang akan diuji ke aplikasi.
2. Proses dilanjutkan dengan proses *image sequence* yaitu pembacaan citra oleh sistem, jika citra memiliki *frame* lebih dari 1 maka dapat menuju proses selanjutnya, tetapi jika citra hanya memiliki 1 *frame* maka alur akan kembali ke proses sebelumnya.
3. Proses selanjutnya *grayscale* yaitu mengubah citra RGB menjadi citra keabuan.
4. Proses selanjutnya adalah *foreground segmentation* yaitu mengubah citra hasil *grayscale* menjadi citra hitam putih dengan menggunakan nilai *threshold* sebagai pembanding.
5. Data nilai *pixel* yang didapat dari proses *foreground segmentation* selanjutnya akan melewati proses *Blob Detection* dengan menyesuaikan nilai aspek *ratio* yang telah ditentukan sebelumnya.
6. Jika nilai *pixel* hasil *foreground segmentation* tersebut sesuai dengan kondisi aspek *ratio*, maka sistem akan memberikan informasi bahwa ada objek bergerak yang berhasil dideteksi.
7. Jika terdapat objek lebih dari satu, maka sistem akan melakukan pengecekan terhadap nilai *pixel* dan disesuaikan dengan nilai aspek *ratio* pada proses *Blob Detection.*
8. Proses selesai.

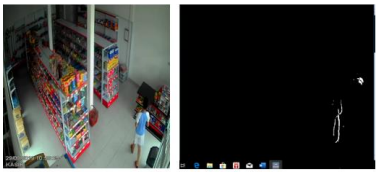
5. IMPLEMENTASI, PENGUJIAN, DAN PEMBAHASAN

5.1 Implementasi Perangkat Lunak

Tahap implementasi sistem ini merupakan tahap awal untuk melakukan deteksi objek bergerak. Halaman tampilan video dapat dilihat pada Gambar 11.

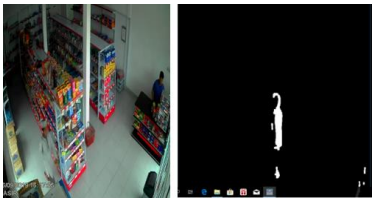
Gambar 11. Halaman Tampilan Video

Halaman ini merupakan halaman yang menampilkan video yang sudah diinputkan oleh pengguna di halaman utama. Hasil *thresholding* citra objek bergerak dan *background* dapat dilihat pada Gambar 12 berikut ini.



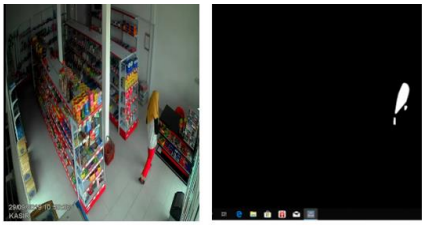
Gambar 12. Halaman *Thresholding*

Gambar 12 menunjukkan halaman *thresholding.* Selanjutnya data akan melalui proses perbaikan citra, yaitu kontur citra. Tampilan kontur citra dapat dilihat pada Gambar 13.



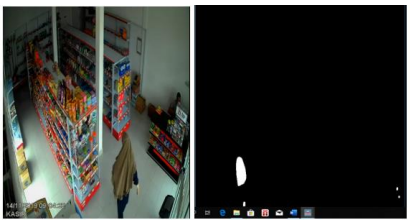
Gambar 13. Halaman Kontur Citra

Setelah melalui proses kontur citra, data selanjutnya dimonitor pada proses *currentblob.* Tampilan *currentblob* dapat dilihat pada Gambar 14.



Gambar 14. Halaman *Current Blob*

Setelah melalui proses *currentblob*, citra akan melalaui proses *Blob Detection* untuk hasil akhir proses deteksi. Proses ini menggunakan nilai aspek *ratio* untuk menentukan adanya objek bergerak. Proses *Blob Detection* dapat dilihat pada Gambar 15.



Gambar 15. Halaman *Blob Detection*

Setelah itu, sistem akan menampilkan informasi bahwa ada objek bergerak yang berhasil dideteksi dalam bentuk tulisan, gambar dan jumlah objek bergerak. Tampilan hasil informasi deteksi dapat dilihat pada Gambar 16 di bawah ini.

Gambar 16. Halaman Informasi Deteksi

5.2 Pengujian

Pengujian dilakukan untuk mengukur tingkat keberhasilan aplikasi dalam mendeteksi objek bergerak berdasarkan aspek *ratio* yang telah ditentukan. Terdapat 20 data video yang akan diuji dengan ukuran yang berbeda dan dari 20 data video dibagi menjadi 2 bagian, 10 data video dalam keadaan terang (*cctv* RGB) dan 10 video dalam keadaan gelap (*cctv infrared)*.

Pengujian dianggap berhasil jika citra pada pengujian dapat dideteksi sebagai citra objek bergerak yang bentuk matriksnya saling berhubungan dan sesuai dengan nilai aspek *ratio* pada prose *Blob Detection*. Sebaliknya pengujian dianggap gagal apabila citra uji yang bergerak tidak dapat terdeteksi oleh aplikasi. Hasil pengujian terhadap video objek bergerak dalam keadaan terang pada aplikasi ini dapat dilihat pada Tabel 1, hasil pengujian terhadap video objek bergerak dalam keadaan *grayscale* dapat dilihat pada Tabel 2 dan hasil pengujian terhadap video objek bergerak dalam keadaan gelap dapat dilihat pada Tabel 3 dibawah ini.

Tabel 1. Hasil Pengujian Video Terang

No	*File*	Pergerakkan	Hasil	Penjelasan
1	Terang1	2	2	Semua pergerakkan terdeteksi
2	Terang2	5	5	Semua pergerakkan terdeteksi
3	Terang3	6	6	Semua pergerakkan terdeteksi
4	Terang4	3	3	Semua pergerakkan terdeteksi
5	Terang5	2	2	Semua pergerakkan terdeteksi
6	Terang6	4	2	Karena objek bergerak bersamaan
7	Terang7	2	2	Semua pergerakkan terdeteksi
8	Terang8	4	4	Semua pergerakkan terdeteksi
9	Terang9	3	2	Karena objek bergerak bersamaan
10	Terang10	3	1	Karena objek bergerak bersamaan

Tabel 2. Hasil Pengujian Video *Grayscale*

No	*File*	Pergerakkan	Hasil	Penjelasan
1	Terang1	2	2	Semua pergerakkan terdeteksi
2	Terang2	5	1	Karena salah satu objek terlalu gelap
3	Terang3	6	6	Semua pergerakkan terdeteksi
4	Terang4	3	2	Karena salah satu objek terlalu gelap
5	Terang5	2	2	Semua pergerakkan terdeteksi
6	Terang6	4	4	Semua pergerakkan terdeteksi
7	Terang7	2	2	Semua pergerakkan terdeteksi
8	Terang8	4	4	Semua pergerakkan terdeteksi
9	Terang9	3	3	Semua pergerakkan terdeteksi
10	Terang10	3	2	Karena salah satu objek terlalu gelap

Tabel 3. Hasil Pengujian Video Gelap

No	*File*	Pergerakkan	Hasil	Penjelasan
1	Gelap1	2	2	Semua pergerakkan terdeteksi
2	Gelap2	1	1	Semua pergerakkan terdeteksi
3	Gelap3	1	1	Semua pergerakkan terdeteksi
4	Gelap4	2	2	Semua pergerakkan terdeteksi
5	Gelap5	3	3	Semua pergerakkan terdeteksi
6	Gelap6	2	1	Karena salah satu objek terlalu gelap
7	Gelap7	1	1	Semua pergerakkan terdeteksi
8	Gelap8	3	0	Karena objek terlalu gelap
9	Gelap9	1	0	Karena objek terlalu gelap
10	Gelap10	4	4	Semua pergerakkan terdeteksi

5.3 Perhitungan *Background Subtraction* dan *Blob Detection*

Proses Perhitungan citra *Background Subtraction* secara manual berisi perhitungan setiap tahap pemrosesan *citra Background Subtraction.* Tahapan ini meliputi perhitungan mengubah citra RGB menjadi citra *grayscale,* dan mengubah citra *grayscale* menjadi citra hitam putih dengan proses *thresholding.* Hasil nilai *pixel* RGB pada *background* dapat dilihat pada Tabel 4.

Tabel 4. Nilai *Pixel* RGB *Background*

	R			G			B		
x \ y	1	2	3	1	2	3	1	2	3
1	181	176	177	186	180	179	154	133	122
2	181	181	180	186	185	184	156	140	125
3	183	182	180	187	186	184	161	145	130

Tabel 4 merupakan contoh potongan nilai *pixel* RGB yang diambil dari *background* yang memiliki objek bergerak. Selanjutnya menentukan nilai *grayscale* menggunakan Persamaan 1.

Grayscale = 0.299 (181) + 0.5870 (186) + 0.1140 (154)

Grayscale = 54.11 + 106.02 + 17.55

Grayscale = 177.68 (x_1, y_1)

Menggunakan cara yang sama, dilakukan perhitungan pada semua nilai *pixel* RGB *background* model. Hasil untuk nilai *grayscale* yang telah dihitung dapat dilihat pada Tabel 5.

Tabel 5. Nilai *Pixel Grayscale Background*

x / y	1	2	3
1	181	173	172
2	181	179	176
3	183	180	177

Setelah menentukan nilai RGB dan *grayscale* dari *background* model, selanjutnya mencari nilai RGB dari citra objek bergerak, dan menghitung nilai *grayscale*nya. Untuk mrndapatkan nilai *pixel* RGB dari *background* dan objek bergerak dari suatu citra peneliti menggunakan aplikasi Matlab. Hasil nilai *pixel* RGB pada *background* dapat dilihat pada Tabel 6.

Tabel 6. Nilai *Pixel* RGB Objek

	R			G			B		
x / y	1	2	3	1	2	3	1	2	3
1	142	141	133	157	154	142	103	92	82
2	143	139	132	157	150	140	102	87	79
3	137	139	140	149	149	140	94	85	87

Tabel 6 merupakan contoh potongan nilai *pixel* RGB yang diambil dari objek bergerak. Setelah mendapatkan nilai RGB dari objek bergerak, maka selanjutnya menentukan nilai *grayscale* menggunakan Persamaan 1:

Grayscale = 0.299 (142) + 0.5870 (157) + 0.1140 (103)

Grayscale = 42.45 + 92.15 + 11.74

Grayscale = 146.07 (x_1, y_1)

Menggunakan cara yang sama, dilakukan perhitungan pada semua nilai *pixel* RGB citra objek bergerak, hasil nilai *grayscale* dapat dilihat pada Tabel 7.

Tabel 7. Nilai *Pixel Grayscale Objek*

x / y	1	2	3
1	146	143	132
2	147	140	131
3	139	139	138

Setelah didapat nilai *grayscale* masing-masing dari citra *background* dan objek bergerak, selanjutnya hasil kedua citra *grayscale* tersebut mengalami proses *foreground segementation.* Proses *foreground segementation* dilakukan dengan cara pengurangan nilai *pixel grayscale background* dengan *pixel grayscale* objek bergerak. Kemudian hasilnya akan dibandingkan dengan nilai *threshold*. Nilai *threshold* yang digunakan pada perhitungan ini adalah 30. Setiap *pixel* citra yang bernilai dibawah 30 akan diubah menjadi warna hitam (*background*), sedangkan *pixel* citra yang bernilai diatas 30 akan diubah menjadi warna putih (objek). Proses perhitungan dapat dilakukan dengan menggunakan Persamaan 3 dibawah ini dan hasil perhitungan dapat dilihat pada Tabel 8.

1. $I_{(x1,y1)}$ = 181 $I'_{(x1,y1)}$ = 146

$$f_{(x1,y1)} = |I_{(x1,y1)} - I'_{(x1,y1)} - 1|$$
$$= |181\text{-}146\text{-}1|$$
$$= 34$$

$f_{(x1,y1)}$ = 1, karena 34 >30

2. $I_{(x1,y2)}$ = 181 $I'_{(x1,y2)}$ = 147

$$f_{(x1,y2)} = |I_{(x1,y2)} - I'_{(x1,y2)} - 1|$$
$$= |181\text{-}147\text{-}1|$$
$$= 33$$

$f_{(x1,y2)}$ = 1, karena 33 >30

3. $I_{(x1,y3)}$ = 183 $I'_{(x1,y3)}$ = 139

$$f_{(x1,y3)} = |I_{(x1,y3)} - I'_{(x1,y3)} - 1|$$
$$= |183\text{-}139\text{-}1|$$
$$= 43$$

$f_{(x1,y3)}$ = 1, karena 43 >30

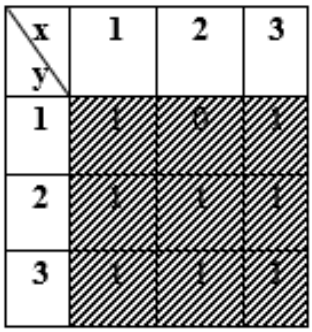
Tabel 8. Hasil *Foreground Segmentation*

x / y	1	2	3
1	1	0	1
2	1	1	1
3	1	1	1

Setelah melalui beberapa tahap pada proses *Background Subtraction*, citra akan melalui proses deteksi objek bergerak dengan metode *Blob Detection* dan aspek *ratio*. Pada tahapan ini akan dilakukan *masking* pada nilai-nilai *pixel* yang sama dan saling berhubungan. *Masking* adalah proses pelapisan objek bergerak yang berhasil dideteksi dimana nilai-nilai *pixel* yang

sama akan diberi lapisan warna putih oleh sistem. Hasil dari *masking foreground* manual objek bergerak dapat dilihat pada Tabel 9.

Tabel 9. Hasil *Masking Foreground*



x / y	1	2	3
1	1	0	1
2	1	1	1
3	1	1	1

5.4 Pembahasan

Pembahasan dalam penelitian ini mencakup tiga bagian, yang pertama yaitu bagaimana kerja sistem dalam mendeteksi objek bergerak saat keadaan terang, bagaimana mendeteksi objek bergerak pada video *grayscale* dan bagaimana mendeteksi objek bergerak dalam keadaan gelap.

5.4.1 Video Keadaan Terang

Aplikasi deteksi objek bergerak pada penelitian ini menggunakan metode *Background Subtraction* dan *Blob Detection*. Data yang digunakan berupa video objek bergerak yang sedang melewati atau berjalan di suatu tempat. Selanjutnya video tersebut akan diproses beberapa tahap dalam metode *Background Subtraction*. Setelah didapatkan nilai *pixel* dari citra objek bergerak tersebut, nilai *pixel* yang sama dan saling berhubungan akan di proses dengan metode *Blob Detection* untuk dilakukan *masking foreground*. Setelah citra mengalami *masking foreground*, citra objek bergerak tersebut akan disesuaikan bentuk *pixel*nya dengan nilai aspek *ratio*. Jika bentuk *pixel* citra tersebut memenuhi syarat dari nilai aspek *ratio*, maka citra tersebut dianggap sebagai citra objek bergerak.

5.4.2 Video Keadaan *Grayscale*

Deteksi untuk objek *grayscale* prosesnya sama seperti objek dalam kedaan terang. Secara sistem deteksi objek *grayscale* lebih cepat dibandingkan dengan deteksi objek terang, karena *background* dan objek bergerak hanya memiliki 1 warna yang terdiri dari warna abu atau skala keabuan dan tidak memiliki banyak warna yang berbeda.

5.4.3 Video Keadaan Gelap

Aplikasi deteksi objek bergerak dalam keadaan gelap sama dengan deteksi objek bergerak dalam keadaan terang yaitu menggunakan metode *Background Subtraction* dan *Blob Detection*. Data yang digunakan berupa video objek bergerak yang sedang melewati atau berjalan di suatu tempat. Selanjutnya video tersebut akan diproses beberapa tahap dalam metode *Background Subtraction* namun video dalam keadaan gelap ini tidak melalui proses *grayscale* karena video sudah berwarna keabuan. Proses selanjutnya adalah mencari nilai *pixel* biner, setelah didapatkan nilai *pixel* dari citra objek bergerak tersebut, nilai *pixel* yang sama dan saling berhubungan akan di proses dengan metode *Blob Detection* untuk dilakukan *masking foreground*. Setelah citra mengalami *masking foreground*, citra objek bergerak tersebut akan disesuaikan bentuk *pixel*nya dengan nilai aspek *ratio* yang telah ditentukan sebelumnya. Jika bentuk *pixel* citra tersebut memenuhi syarat dari nilai aspek *ratio*, maka citra tersebut dianggap sebagai citra objek bergerak.

6. KESIMPULAN DAN SARAN

6.1 Kesimpulan

Adapun Kesimpulan dari penelitian ini adalah sebagai berikut:

1. Deteksi objek bergerak dalam kedaan terang dilakukan dengan metode *Background Subtraction* dan *Blob Detection* yang menggunakan aspek *ratio* yaitu untuk *pixel* x dan y harus lebih dari 20:20. Hasil yang didapat adalah terdeteksinya pergerakkan dari setiap objek.
2. Deteksi objek bergerak dalam kedaan gelap dilakukan dengan metode *Background Subtraction.* Selanjutnya melewati proses *Blob Detection* yang menggunakan nilai aspek *ratio* yaitu untuk *pixel* x dan y harus lebih dari 20:20. Hasil yang didapat adalah terdeteksinya pergerakkan dari setiap objek.
3. Tingkat akurasi metode *Background Subtraction* dan *Blob Detection* dalam penelitian deteksi video objek bergerak dalam keadaan terang adalah sebesar 70% dengan 20 data uji terdapat 14 data yang hasil pengujiannya benar dan 6 data yang hasil pengujiannya salah. Pada penelitian ini peneliti juga menguji data video objek bergerak dalam keadaan terang yang dirubah menjadi objek bergerak dalam

keadaan *grayscale*. Tingkat akurasi pada pengujian video objek bergerak dalam keadaan *grayscale* yaitu 75% dengan 20 data uji terdapat 15 data yang hasil pengujiannya benar dan 5 data yang hasil pengujiannya salah. Hasil pengujian yang salah diakibatkan oleh banyaknya kombinasi warna pada *background* serta banyaknya pergerakkan berulang dari objek bergerak itu sendiri Tingkat akurasi pada pengujian video objek bergerak dalam keadaan gelap yaitu 75% dengan 20 data uji terdapat 15 data yang hasil pengujiannya benar dan 5 data yang hasil pengujiannya salah. Hasil pengujian yang salah diakibatkan oleh video yang terlalu gelap sehingga aplikasi menganggap objek bergerak itu sebagai *background*.

6.2 Saran

Adapun saran untuk pengembangan Aplikasi Deteksi Objek Bergerak Berbasis Citra dengan Metode *Background Subtraction* dan *Blob Detection* agar menjadi lebih baik kedepannya adalah sebagai berikut:

1. Pada penelitian selanjutnya aplikasi dapat dibuat menjadi aplikasi deteksi *realtime* dan waktu untuk melakukan pendeteksian bisa lebih cepat sehingga lebih efektif dan mudah dalam mendeteksi objek bergerak.
2. Pada penelitian selanjutnya bisa dibuat menjadi aplikasi pendeteksian objek bergerak yang lebih spesifik, misalnya pendeteksian hewan atau pendeteksian kendaraan.

DAFTAR PUSTAKA

[1] Effendy, M. (2016). Kamus Besar Bahasa Indonesia.

[2] Aditia, K., & Anifah, L. (2019). Implementasi metode *background substraction* dalam sistem analisis trayektori hasil latihan atlet lompat jauh berbasis pengolahan citra digital. Surabaya: Universitas Negeri Surabaya.

[3] Putri, N. N. (2016). Aplikasi Pendeteksi Objek Bergerak Pada *Image Sequence* dengan Metode *Background Subtraction.* Depok: Universitas Gunadarma.
Sumarudin, & teman-teman. (2019). Aplikasi penghitung kendaraan pada jalur pantura menggunakan blob detection dan kalman filter. Indramayu: Politeknik Negeri R. JIndramayu.
Pamungkas, A. (2017). Pengolahan Citra.

[4] *Retrieved from* Pengolahan Citra: https://pemrogramanmatlab.com/2017/07/26/pengolahan-citra-digital/.
Darshak, & kawan-kawan. (2014). *A*

[5] *Survey on Object Detection and Tracking Methods*.
Pamungkas, A. (2015). Pemrograman

[6] Matlab. *Retrieved from* Pemrograman Matlab: https://pemrogramanmatlab.com/2015/10/12/*background-subtraction-foreground-detection/*.
Kaspers, & Anne. (2011). *Blob Detection*

[7] *Biomedical Image Sciences*. *Image Science Institute*.
Sidharta, H. A. (2017). *Introduction To*

[8] *OpenCV*. Retrieved from Introduction ToOpenCV: http://binus.ac.id/malang/2017/10/introduction-to-open-cv/.