IMPLEMENTASI APLIKASI *TOUR GUIDE* DI KEBUN BINATANG MENGGUNAKAN METODE *MOTION TRACKING* BERBASIS *ANDROID*

IMPLEMENTATION OF TOUR GUIDE APPLICATION ANDROID BASED IN THE ZOO USING MOTION TRACKING METHOD

Franksye Octafyan Sipangkar[1], R Rumani M[2], Anton Siswo Raharjo Ansori[3]

[123] Prodi S1 Teknik Komputer, Fakultas Teknik Elektro, Universitas Telkom

[1]frank.sipangkar@gmail.com, [2] rumani@telkomuniversity.ac.id, [3] raharjo@telkomuniversity.ac.id

Abstrak

Kebun binatang adalah objek wisata atau tempat rekreasi yang didalamnya terdapat tempat perlindungan berbagai macam binatang seperti yang telah masuk dalam kategori dilindungi ataupun tidak dilindungi. Sebagai objek wisata yang sering dikunjungi wisatawan, kebun binatang seharusnya terus meningkatkan fasilitas-fasilitas yang ada dan memberikan inovasi sehingga minat wisatawan untuk datang semakin bertambah.

Oleh karena itu, melalui tugas akhir ini, telah diimplementasikan aplikasi *mobile Tour Guide* berbasis Android untuk menambah minat wisatawan untuk datang. Aplikasi *Tour Guide* ini menggunakan fitur HOG+SVM untuk menangkap/mendeteksi gerakan (*motion tracking*) dan melakukan pengenalan (*recognition*) objek (binatang/hewan) sehingga dapat menampilkan informasi mengenai binatang.

Kata Kunci : *computer vision, tour guide,* aplikasi *mobile,* HOG+SVM, *motion tracking, animal detection.*

Abstract

Zoo is a tourist attraction or a recreation area which in there are a wide range of animal protection as it has been entered in the category of protected or unprotected. As a tourist attraction frequented by tourists, the zoo should continue to improve existing facilities and provide innovation so that the interest of tourists to come is growing.

Therefore, through this final project, has been implemented Android mobile application's Tour Guide for increase the interest of tourists to come. Tour Guide's app is using features HOG + SVM to detect motion (motion tracking) and perform recognition of the object (animal) so that app can display information about the animals.

Keywords : computer vision, tour guide, mobile application, HOG+SVM, motion tracking, animal detection.

1. Pendahuluan

Pada era globalisasi saat ini, perkembangan teknologi yang semakin pesat seiring dengan tingginya tuntutan hidup membuat kebutuhan akan hiburan dan rekreasi. Oleh karena itu, kebun binatang menjadi salah satu tempat rekreasi yang dapat memberikan hiburan untuk melupakan sejenak tuntutan hidup sehari-hari. Selain itu kebun binatang dapat dijadikan sebagai tempat pembelajaran yang baik dan menyenangkan, karena terdapat berbagai macam fasilitas yang telah disediakan oleh pihak pengelola kebun binatang dan juga aktifitas dari binatang yang ada. Namun seiring berjalannya waktu, fasilitas yang telah tersedia belum banyak memberikan informasi yang tepat dikarenakan adanya papan informasi yang terbengkalai dan tidak terawat sehingga dapat menurunkan minat dari pengunjung.

2. Dasar Teori

2.1. *Tour Guide*

Tour Guide atau sering disebut pramuwisata merupakan jenis profesi pemandu wisatawan yang berkunjung ke tempat rekreasi seperti museum, kebun binatang dan lain sebagainya. Menurut [6], Pramuwisata adalah petugas pariwisata yg berkewajiban memberi petunjuk dan informasi yg diperlukan wisatawan; pemandu wisata. Menurut [5], *Tour Guide* atau pramuwisata bukanlah jenis pekerjaan yang mudah, hal ini dikarenakan seorang *Tour Guide* harus selalu *stand by* (siap siaga) demi memastikan *customer*/wisatawan puas akan jasa

yang diberikan, terlebih lagi seorang *Tour Guide* harus menjelaskan semua informasi mengenai binatang/hewan yang ada di kebun binatang.

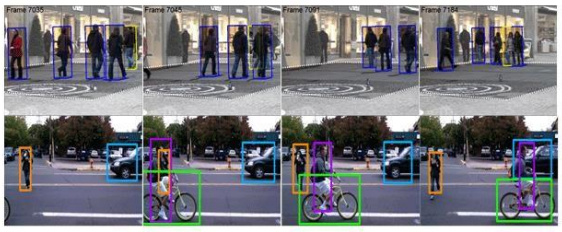
2.2. *Computer Vision*

Computer Vision atau sering dikenal dengan visi komputer merupakan suatu bidang ilmu yang mempelajari bagaimana suatu komputer dapat meniru dan melakukan sistem visual yang dilakukan pada manusia. Secara lebih jelasnya, *computer vision* meniru sistem visual manusia, yang dimana manusia melihat suatu objek dengan mata lalu diteruskan ke otak sehingga manusia dapat mengetahui objek yang sedang dilihatnya [2].

2.3. *Motion Tracking*

Metode *Motion tracking* merupakan metode yang dapat mendeteksi, melacak atau mencari objek yang bergerak pada suatu citra *static* atau *dynamic*. Untuk mengadopsi teknik ini, diperlukan sebuah sistem visi komputer (*computer vision)* yang dapat melakukan proses dimana komputer dapat mendeteksi setiap pergerakan objek yang terekam dari sumber citra seperti video yang sudah terekam sebelumnya ataupun secara *real-time.* Sistem visi komputer (computer vision) yang digunakan adalah *library* dari *Open Source Computer Vision Library (OpenCV).*

Jika ditelaah lebih lanjut, pengertian *motion* merupakan pergerakan dari objek yang terjadi akibat pergantian frame dan *Tracking* merupakan istilah yang digunakan untuk mendeteksi, melacak atau mencari suatu objek pada citra [3].

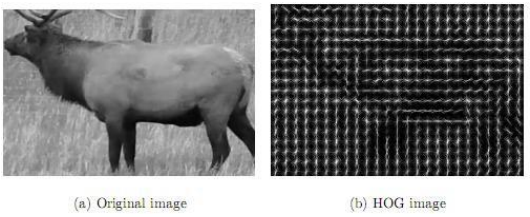


Gambar 1. Tracking Obyek Melalui Urutan Video [1]

2.4. *Histograms of Oriented Gradients (HOG)*

Histogram of Oriented Gradients (HOG) merupakan fitur ekstrasi untuk melakukan proses *image processing* dengan tujuan deteksi target objek, terlebih lagi objek tersebut difokuskan untuk deteksi *pedestrian* [1]. Fitur ini dikenal sebagai *descriptor* yakni HOG *descriptor* yang memiliki keunggulan dapat menggambarkan kontur (*contour*) dan sisi tepi (*edge*) dari objek selain untuk manusia, seperti deteksi binatang dan alat transportasi kendaraan [9].

Pada implementasinya, *Histogram of Oriented Gradients* (HOG) melakukan normalisasi *gamma* dan warna terhadap suatu citra, yang kemudian citra tersebut dibagi menjadi banyak *cell* yang membentuk histogram, membentuk blok dari setiap histogram, dan akhirnya melakukan normalisasi terhadap blok yang dihasilkan.



Gambar 2. Hasil dari ekstrasi fitur HOG [9]

2.5. *Support Vector Machine* (SVM)

Support Vector Machine atau sering dikenal *Support Vector Network* merupakan salah satu teknik pengenalan pola (*pattern recognition*) algorima *classification*. *Support Vector Machine* juga merupakan suatu teknik untuk melakukan prediksi, baik dalam kasus klasifikasi ataupun regresi [7].

[1] Sumber gambar dari https://cvl.gist.ac.kr/wp-content/uploads/2012/11/m1.gif

Pada penelitian tugas akhir ini, *pattern* atau pola yang dikenali adalah pola dari tubuh binatang/hewan melalui data latih atau *train* yang sudah disiapkan terlebih dahulu yang kemudian diproses dalam proses *training* menggunakan fitur HOG dengan dikombinasikan algoritma SVM.

2.6. *Confusion Matrix*

Confusion matrix (matriks konfusi) merupakan tabel kemungkinan (*contingency table)* untuk mencatat berbagai hasil kemungkinan dari hasil klasifikasi. Menurut [8], misalkan elemen matriks konfusi untuk data klasifikasi dengan dua kelas dinyatakan dengan fij, maka setiap sel fij menyatakan jumlah *record*/data yang sebenarnya masuk dalam kelas i, tetapi hasil prediksinya mengklasifikasikan data tersebut pada kelas j. Tabel 1 menunjukkan penjelasan mengenai matriks konfusi.

Tabel 1. Matriks Konfusi

Hasil Observasi (*actual class)*	Hasil Prediksi (*predicted class)*	
	Kelas 1	Kelas 2
Kelas 1	f_{11}	f_{12}
Kelas 2	f_{21}	f_{22}

Setelah didapatkan hasil klasifikasi dari tabel tersebut, kemudian hasil ini dihitung untuk mendapatkan tingkat akurasi klasifikasi. Akurasi dihitung menggunakan formula :

$$\text{Akurasi} = \frac{\text{Jumlah data yang diklasifikasikan dengan benar}}{\text{Jumlah seluruh data}} = \frac{f_{11} + f_{22}}{f_{11} + f_{12} + f_{21} + f_{22}}$$

3. Perancangan Sistem

3.1. Deskripsi Umum Kebutuhan Sistem

Aplikasi *Tour Guide* ini dibuat dengan menggunakan bahasa pemrograman *Java* dengan pendekatan *Native C/C++ Development*, yang dimana dalam melakukan *image processing* secara *real-time* dilakukan di lapisan *Native*, yakni menggunakan bahasa C/C++ yang kemudian untuk hasil dari proses sebelumnya dilanjutkan kembali ke *layer applications* dengan bahasa pemrograman *Java*. Dengan menggunakan pendekatan *Native C/C++ Development*, dapat meningkatkan kompleksitas aplikasi dalam segi akurasi dan waktu deteksi [4].

Dalam membangun aplikasi *Tour Guide* ini digunakan spesifikasi perangkat lunak sebagai berikut :

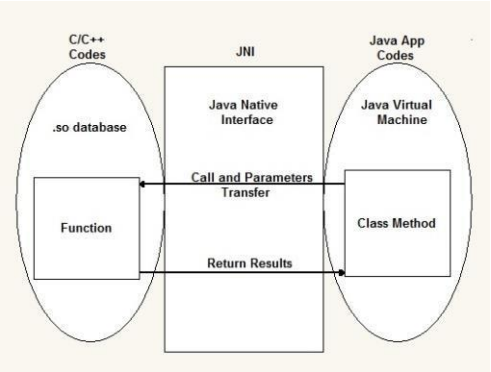
a. Sistem Operasi Windows 8.1.
b. Sistem Operasi Android *KitKat* versi 4.4.2.
c. Android Studio versi 1.2.2.
d. *Java Development Kit* (JDK) versi 8.0.
e. OpenCV Android SDK versi 2.4.10.
f. *Android Native Development Kit* (NDK) r10d *for Windows.*
g. *Classifier Tool For OpenCV and FANN* versi 4.11.8.

Dalam melakukan penelitian tugas akhir ini, berikut merupakan spesifikasi perangkat keras yang digunakan :

a. Notebook PC Acer Aspire E1-471G, *Intel Core i3 Processor.*
b. Notebook PC RAM 4GB.
c. *Smartphone* K Touch Nibiru Mars One H1, *Octa-Core Processor.*
d. *Smartphone* RAM 2GB.

3.2. Perancangan Sistem Aplikasi *Tour Guide*

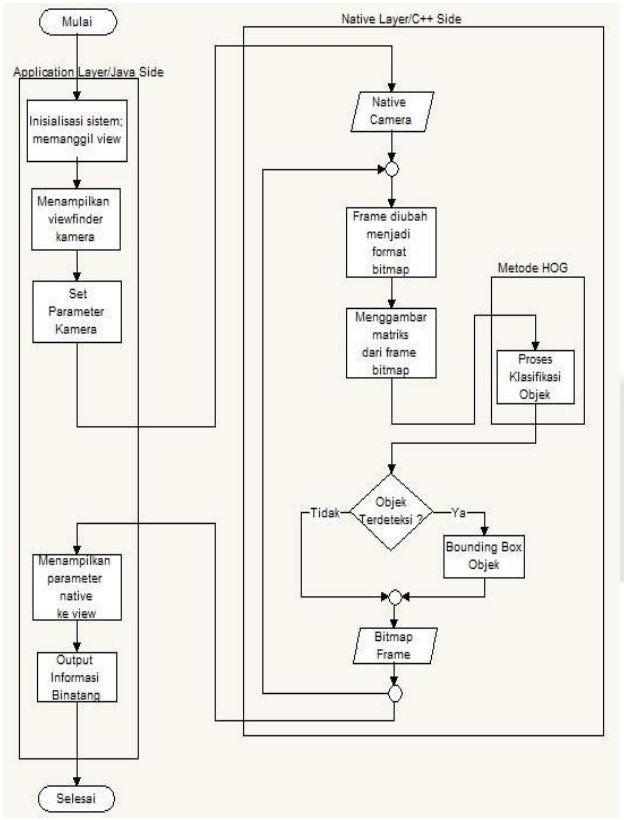
Setelah dijelaskan sebelumnya, pada perancangan sistem aplikasi *Tour Guide* menggunakan bahasa pemrograman *Java* dengan pendekatan *Native C/C++ Development* dengan tujuan meningkatkan kompleksitas ataupun performansi aplikasi dalam hal deteksi objek sehingga dapat melakukan *processing* yang cepat dan *real-time*.



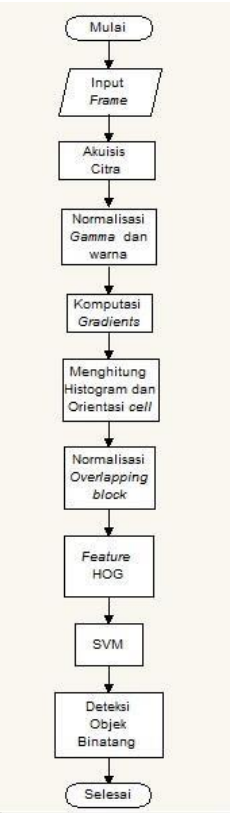
Gambar 3. Cara Kerja Aplikasi dengan pendekatan *Native Development*

Sistem kerja aplikasi *Tour Guide* dengan pendekatan *Native Development* pada gambar 3 memiliki dua sisi layer program utama, yakni *Application Layer/Java Side* dan *Native Layer/C++ Side*. Layer *Application* merupakan lapisan pertama (paling atas) di arsitektur *Android*, yang mengatur *life cycle* dari jalannya aplikasi seperti proses menginialisasi sistem sehingga menampilkan *viewfinder* kamera, menghentikan proses berjalannya aplikasi secara sementara ataupun permanen ketika menutup aplikasi dan lain sebagainya.Pada sisi layer program *Application* ditulis dengan bahasa *Java.*

Pada gambar 4, merupakan *flowchart* Sistem Aplikasi *Tour Guide*, yang menjelaskan sisi *Native Layer/C++* merupakan proses dalam mendeteksi objek yang bergerak. Proses deteksi ini dimulai dari inialisasi parameter-parameter dari kamera yang digunakan seperti resolusi *frame* kamera pada *viewfinder* yang kemudian hasil dari *frame* yang tertangkap di *viewfinder* diubah menjadi format bitmap sehingga dapat digambarkan representasi matriks yang dihasilkan.Setelah mendapatkan representasi matriks-nya, terdapat tahap proses klasifikasi objek yang dimana di dalam tahap ini merupakan pemrosesan citra dengan metode HOG. Setelah objek terdeteksi, *bounding box* akan tergambar diatas *frame* yang yang terindikasi objek yang dicari.Jika target objek ditemukan, sistem akan menjalankan *output* yang berisi informasi mengenai objek tersebut.



Gambar 4. *Flowchart* Sistem Aplikasi *Tour Guide*.



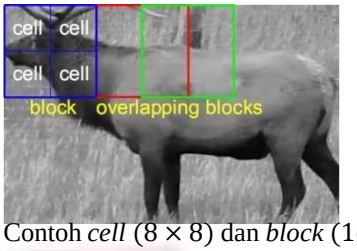
Gambar 5. Alur proses deteksi objek dengan metode HOG+SVM

Pada gambar 5, merupakan alur proses klasifikasi objek untuk mendeteksi objek (binatang/hewan) yang bergerak, *Histogram of Oriented Gradient* (HOG) diimplementasikan ke dalam aplikasi *Tour Guide*, yang dikombinasikan bersama dengan *Support Vector Machine* (SVM) untuk proses data *learning*.

Pada tahap awal, input dari *frame* ditransformasi dari citra asli menjadi citra digital di dalam proses akuisisi citra yang kemudian dirubah menjadi citra *grayscale* dengan tujuan untuk mengurangi beban performansi aplikasi secara *real-time* dalam melakukan *processing*. Kemudian citra *grayscale* ini dinormalisasi terhadap masukan citra digital, sehingga dalam tahap ini dapat dihitung akar kuadrat dari setiap *channel* warna yang dimana hasil dari tahap normalisasi merupakan *edge* atau garus tepi antara objek-objek di *frame* yang sedang di *processing*. Pada gambar 6, merupakan contoh *edge* saat normalisasi.

Selanjutnya pada tahap komputasi *gradient*, *gradient* dari citra digital ini dihitung dan kemudian hasil dari *gradient* ini dibagi menjadi *block* yang dimana didalamnya terdapat *cell*. Pada gambar 7, merupakan contoh distribusi *cell* dan *block*. *Block* merupakan kelompok gabungan histogram *cell* yang berukuran 8×8 (piksel) sehingga *block* berukuran 16×16 (piksel). Setiap *block* dinormalisasi terhadap histogram yang dimana didalam proses normaliasi ini, setiap *block* saling *overlapping* (tumpang tindih) terhadap *block* lainnya. Kemudian hasil

Gambar 6. Contoh *edge* saat normalisasi [9]



Gambar 7. Contoh *cell* (8×8) dan *block* (16×16) [9]

dari semua normalisasi ini digabung untuk mendapatkan data fitur HOG.

Contoh hasil fitur HOG yang dihasilkan dibagi menjadi dua, yakni nilai +1 dan nilai -1. Nilai +1 untuk citra positif adalah nilai fitur vektor positif yang merupakan objek target sedangkan nilai -1 untuk citra negatif adalah nilai fitur vektor negatif yang merupakan bukan objek target. Setelah mendapatkan data fitur HOG, data fitur tersebut dirubah menjadi fitur vektor (*feature vector*) sebagai data masukan dalam proses *learning* SVM. Proses *learning* SVM ini digunakan untuk mencari *hyperplane* (garis pemisah) yang optimal terhadap 2 *class*, yakni +1 dan -1 yang dimana proses *learning* SVM ini dapat membantu proses deteksi objek lebih cepat. Setelah proses *learning*, maka objek yang terdeteksi akan terindikasi apakah objek tersebut sesuai dengan data latih (*training*) atau tidak.

Proses *learning* SVM digunakan untuk mencari *hyperplane* (garis pemisah) 2 *class* dari data *training* (latih). Data *training* (latih) berupa citra (gambar) positif objek hewan dan citra (gambar) negatif bukan objek hewan yang difoto secara langsung di kebun binatang. Pada saat pengambilan foto untuk data *training*, foto diambil di hari, jam, dan kecerahan cuaca yang berbeda-beda. Dari keseluruhan hewan di kebun binatang, hewan yang menjadi objek terdiri dari 2 hewan yakni

- Banteng
  - Data latih : 310 citra positif dan 200 citra negatif.
- Unta
  - Data latih : 556 citra positif dan 300 citra negatif.

Setiap gambar positif dan negatif memiliki rasio lebar : tinggi = $7 : 5$ dengan ukuran 56×40 piksel sesuai *detector window* yang disarankan untuk objek hewan.

Gambar 8. Beberapa contoh citra data latih positif.

Gambar 9. Beberapa contoh citra data latih negatif

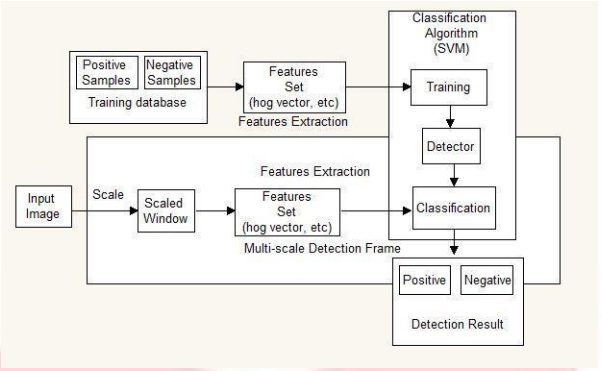
Untuk data gambar positif, ada dua kategori citra (gambar) hewan yang digunakan adalah kategori tampak samping *(side*-view) HTL (*Head to Left*) *shape* dan kategori tampak samping *(side*-view) HTR (*Head to Right*) *shape* [9].

Gambar 10.Kategori HTL (*Head to* Left) *shape*.

Gambar 11. Kategori HTR (*Head to Right*) *shape*.

3.3. *Classiier Training SVM*

Untuk mendapatkan vektor dari data latih (*training*) yang telah dipersiapkan sebelumnya, dibutuhkan *classifier training. Classifier training* yang digunakan adalah *Classifier Tool For OpenCV and FANN.Classifier*[2] ini mengunakan algoritma *machine learning SVMlight*[3] untuk melakukan prediksi, klasifikasi dan regresi terhadap data *training*. Pada gambar 12, menjelaskan secara umum proses *training SVM* yang dilakukan dan proses deteksi *multi-scale* dalam sistem deteksi.



Gambar 12. *Flowchart training* dan deteksi *multi-scale* dalam sistem deteksi objek [9].

Pada *flowchart* diatas, data latih (*training*) yang telah disiapkan dirubah/diekstrak menjadi vektor yang sesuai dengan fitur HOG menggunakan *classifier* untuk proses deteksi. Kemudian, vektor dari data latih diklasifikasi dengan vektor yang berasal dari *input image* (*input frame*) yang telah diekstrak sebelumnya menggunakan fitur HOG. Dalam proses klasifikasi, vektor dari data latih dan *input image* dicocokan sehingga dapat menentukan apakah objek terdeksi ataupun tidak terdeteksi.

4. Pengujian dan Analisis

4.1 Pengujian Akurasi Deteksi

Pengujian aplikasi ini dilakukan untuk mengetahui tingkat akurasi dari metode HOG+SVM dengan menggunakan dua data *training* berbeda yang telah dipersiapkan sebelumnya, yakni data *training* banteng berjumlah 310 citra positif, 200 citra negatif dan data *training* unta berjumlah 556 citra positif, 300 citra negatif.

Pengujian akurasi deteksi dilakukan untuk mengetahui tingkat akurasi deteksi objek binatang dengan menggunakan data *training*.Nilai akurasi deteksi (%) menggunakan parameter klasifikasi, yaitu :

- *True Positive* : Objek binatang terdeteksi (objek binatang ada dan berada didalam kotak deteksi).
- *True Negative* : Objek binatang tidak ada dan kotak deteksi juga tidak ada.
- *False Positive* : Objek binatang tidak terdeteksi dengan benar (Objek binatang ada dan kotak deteksi ada namun mendeteksi *background* ataupun objek lain).
- *False Negative* : Objek binatang tidak ada dan kotak deteksi ada namun mendeteksi *background* ataupun objek lain.

Setelah mendapatkan hasil klasifikasi, kemudian hasil tersebut dihitung untuk memperoleh akurasi dengan menggunakan formula :

$$\text{Akurasi} = \frac{\text{Jumlah data yang terdeteksi dengan benar}}{\text{Jumlah seluruh data}} \times 100\%$$

$$= \frac{Tp + Tn}{Tp + Tn + Fp + Fn} \times 100\%$$

Dimana dalam formula akurasi diatas, Tp adalah *True Positive*, Tn adalah *True Negative*, Fp adalah *False Positive*, dan Fn adalah *False Negative.*Dengan melakukan pengujian akurasi deteksi, didapatkan analisis perbedaan tingkat akurasi dari dua data *training* yang telah disiapkan sebelumnya.

4.2 Hasil Pengujian Deteksi dan Analisis Akurasi Deteksi

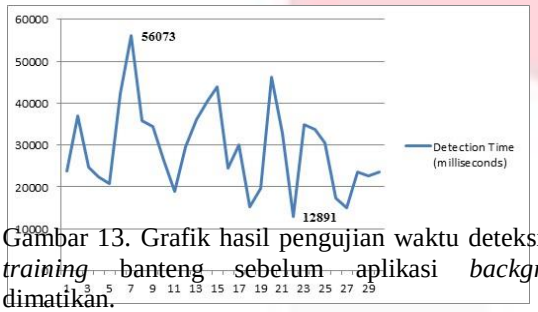
Pada pengujian deteksi binatang, aplikasi ini melakukan deteksi dua objek binatang, yakni banteng dan unta.Aplikasi *Tour Guide* menggunakan resolusi *frame* 640×480 piksel, parameter *HOGDescriptor* (win_size=(56,40)), data *training* banteng yang telah disiapkan adalah 310 citra positif dan 200 citra negatif dan data *training* unta adalah 556 citra positif, 300 citra negatif.

[2] Dapat diunduh di http://classifieropencv.codeplex.com/
[3] Baca selengkapnya di http://svmlight.joachims.org/

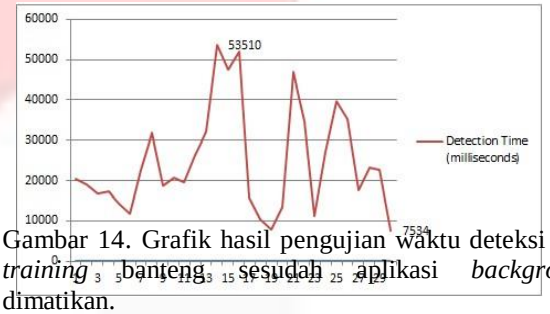
Tabel 2. Hasil pengujian data keseluruhan

Data Pengujian	Akurasi	Minimal waktu yang diperlukan (*milliseconds*)	Maksimal waktu yang diperlukan (*milliseconds*)	Total waktu yang diperlukan (*milliseconds*)
1	**16.67%**	**12891**	**56073**	**875591**
2	**13.33%**	**7534**	**53510**	**738222**
3	**36.67%**	**25027**	**107788**	**1840859**
4	**30%**	**33568**	**97094**	**1795981**

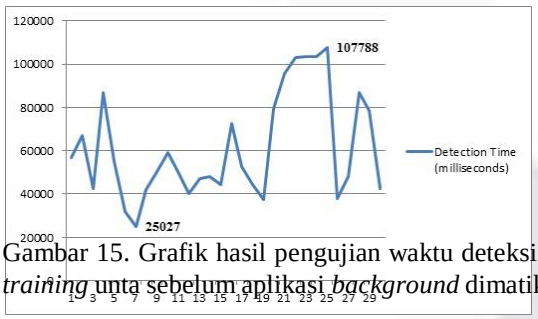
Pada tabel 2, pengambilan data pengujian 1 dan 3 diambil pada hari selasa, 2 februari 2016 sedangkan pengambilan data pengujian 2 dan 4 diambil pada hari sabtu, 20 februari 2016. Data pengujian 1 adalah skenario pengujian data *training* banteng sebelum aplikasi *background* dimatikan, data pengujian 2 adalah skenario pengujian data *training* banteng sesudah aplikasi *background* dimatikan, data pengujian 3 adalah skenario pengujian data *training* unta sebelum aplikasi *background* dimatikan dan data pengujian 4 adalah skenario pengujian data *training* unta sesudah aplikasi *background* dimatikan. Dari pengujian data *training* banteng dan unta diatas, perolehan persentase tingkat akurasi yang lebih baik didapat dari hasil pengujian unta dengan *range* sebesar 3[illegible]% – 3[illegible]% dibandingkan dari hasil pengujian banteng dengan *range* sebesar [illegible]% – [illegible]%. Dalam pengujian terhadap data *training* banteng dan unta, aplikasi membutuhkan total waktu deteksi lebih banyak yaitu **1795981 – 1840859 *milliseconds*** untuk data *training* unta dibanding total waktu yang dibutuhkan dalam mendeteksi data *training* banteng, yaitu **738222 – 875591 *milliseconds***.



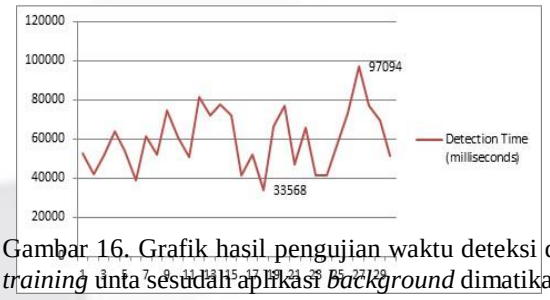
Gambar 13. Grafik hasil pengujian waktu deteksi data *training* banteng sebelum aplikasi *background* dimatikan.



Gambar 14. Grafik hasil pengujian waktu deteksi data *training* banteng sesudah aplikasi *background* dimatikan.



Gambar 15. Grafik hasil pengujian waktu deteksi data *training* unta sebelum aplikasi *background* dimatikan.



Gambar 16. Grafik hasil pengujian waktu deteksi data *training* unta sesudah aplikasi *background* dimatikan.

5. Kesimpulan

Berdasarkan hasil pengujian dan implementasi yang telah dilakukan, dapat diambil kesimpulan sebagai berikut :

1. Fitur HOG+SVM dapat diimplemetasikan ke perangkat *mobile* berbasis *android* menggunakan *Octa-Core Processor* dengan metode *motion tracking* secara *real-time*, namun masih belum fleksibel. Hal ini dikarenakan, perangkat *mobile* membutuhkan minimal waktu 7534 *milliseconds* atau 7.534 *seconds* (pengujian waktu deteksi data *training* banteng sesudah aplikasi *background* dimatikan) untuk melakukan proses terhadap data training banteng, sedangkan untuk data *training* unta, perangkat *mobile* membutuhkan minimal waktu 25027 *milliseconds* atau 25.027 *seconds* (pengujian waktu deteksi data *training* unta sebelum aplikasi *background* dimatikan).
2. Banyaknya data *training* sangat berpengaruh untuk nilai akurasi deteksi dua data *training*. Hal ini dikarenakan data *training* unta lebih banyak dibanding data *training* banteng. Data *training* unta yang

digunakan sebanyak 556 citra positif dan 300 citra negatif sedangkan data *training* banteng sebanyak 310 citra positif dan 200 citra negatif.

3. Aplikasi *background* yang sedang berjalan (*running*) sangat mempengaruhi lamanya waktu yang diperlukan aplikasi dalam melakukan deteksi. Hal ini dikarenakan dalam kondisi aplikasi *background* yang sudah dimatikan, total waktu maksimal yang diperlukan untuk banteng sebesar 738222 *milliseconds* dan unta sebesar 1795981 *milliseconds.* Dalam kondisi aplikasi *background* sebelum dimatikan, total waktu yang diperlukan untuk banteng sebesar 875591 *milliseconds* dan unta sebesar 1840859 *milliseconds.*
4. Berdasarkan hasil pengujian dan analisis, metode HOG+SVM dapat mendeteksi adanya gerakan objek binatang walaupun dengan nilai perolehan akurasi kurang dari 50%, yaitu dengan *range* sebesar 13.33% - 16.67% untuk data *training* banteng dan *range* sebesar 30% - 36.67% untuk data *training* unta.

6. Saran

Saran yang dapat diajukan untuk penelitian selanjutnya adalah :

1. Perolehan nilai akurasi dapat ditingkatkan lagi dengan menambahkan data *training* citra positif dan negatif.
2. Menggunakan *machine learning* yang lain, seperti *AdaBoost* dengan mengkombinasikan fitur lain, seperti kombinasi fitur Haar+AdaBoost, LBP+AdaBoost, dan HOG+AdaBoost untuk melakukan deteksi bintang.
3. Untuk kedepannya, manajemen *resources* diperangkat *mobile* perlu diperhatikan demi menambah fleksibelitas perangkat dalam menjalankan aplikasi.
4. Menggunakan perangkat *mobile* yang memiliki spesifikasi yang lebih besar dibandingkan perangkat yang telah digunakan untuk tugas akhir ini.Hal ini dilakukan untuk mengetahui tingkat fleksibelitas dari segi total waktu yang diperlukan

Daftar Pustaka

[1] Dalal, Navneet & Triggs, Bill.2005. *Histogram Of Oriented Gradients For Human Detection.*

[2] Effendi, Ilham. *Pengertian Augmented Reality(AR)*. (http://www.it-jurnal.com/2014/05/Pengrtian-Augmented-Reality-AR.html).

[3] Hardi, Vialli.,2012. *Sistem Antar Muka Senapan Pada Game First Person Shooter Menggunakan Deteksi Gerak Obyek Berbasis Computer Vision.*

[4] Lee, S., & Wook Jeon, J. (2010). Evaluating Performance of Android Platform Using Native C for Embedded Systems. International Conference on Control, Automation and Systems.

[5] LennyDoank.*3 Hal Tentang Tour Guide*. (http://jadiberita.com/9773/3-hal-tentang-tour-guide.html

[6] P*ramuwisata*, (http://kbbi.web.id/pramuwisata).

[7] Santosa, Budi.2010. *Tutorial Support Vector Machine*.Teknik Industri, ITS.

[8] Sari Mekar Sekar, Safitri Diah, dkk. "Klasifikasi Wilayah Desa-Perdesaan dan Desa-Perkotaan Wilayah Kabupaten Semarang dengan Support Vector Machine (SVM)," Jurnal Gaussian, Volume 3, Nomor 4, Tahun 2014, Halaman 751-760..

[9] Zhou, Depu.2014. *Real-Time Animal Detection System for Intelligence Vehicles*. Thesis.Ottawa : Faculty of Engineering, University of Ottawa.