PERANCANGAN DAN PEMBUATAN APLIKASI TRACKING OBJECT PADA VIDEO DENGAN METODE KERNEL - BASED

Wilson[1], Liliana[2], Kartika Gunadi[3]
Jurusan Teknik Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) - 8417658
E-mail: wilson291901@hotmail.com[1], lilian@petra.ac.id[2], kgunadi@petra.ac.id[3]

ABSTRAK: Pemanfaatan kamera sebagai pengawasan (CCTV) masih memiliki banyak kelemahan. Kamera hanya merekam saja, sehingga masih dibutuhkan pengawasan dari penjaga secara terus menerus. Dengan keterbatasan tersebut maka dibutuhkan sebuah kemampuan tambahan untuk meningkatkan kerja kamera pengawas. Salah satu kemampuan tambahan yang dapat digunakan adalah *motion detection* yang membantu memberi tanda apabila terjadi pergerakan yang tidak umum yang tertangkap kamera.
Deteksi gerakan objek bisa digunakan untuk menangkap pergerakan yang terekam dari sebuah CCTV. Proses pendeteksian dimulai dengan membaca video setiap framenya, dari frame – frame tersebut akan dilakukan proses segmentasi sehingga didapatkan objek yang terdapat dari video tersebut. Setelah itu, objek – objek yang tersimpan akan diproses menggunakan metode kernel – based untuk mendeteksi gerakan yang terjadi dari objek – objek itu.
Hasil akhir dari perangkat lunak ini sebuah aplikasi yang dapat memberi tanda pada objek yang bergerak dari video tersebut. Perangkat lunak ini diujikan dengan beberapa video dengan kondisi yang berbeda – beda. Apabila objek pada video terlalu besar atau terlalu kecil proses *tracking* akan menjadi tidak akurat .

Kata kunci: Deteksi Gerakan, Pelacakan Gerakan, Kernel – Based, Bhattacharyya, Mean - shift

ABSTRACT: Utilization as a surveillance camera (CCTV) still has many weaknesses. The camera just record it, so it still takes control of guard constantly. With these limitations it needed an additional capability to boost surveillance cameras work. One of the additional capabilities that can be used is the motion detection which helps signal the event of unusual movements were caught on camera.
Detection of object motion can be used to capture movements recorded from a CCTV. Detection process starts by reading each video frame, from frame - frame segmentation process will be carried out so that objects that are obtained from the video. After that, the object - the object stored will be processed using the kernel method - based on detecting movement going from object - the object.
The end result is a software application that can signal the moving objects from the video. The software was tested with some video with different conditions - different. If the object in the video is too big or too small tracking process will be inaccurate.

Keywords: Motion Detection, Motion Tracking, Kernel – Based, Bhattacharyya, Mean – shift.

1. PENDAHULUAN

Saat ini kebutuhan sistem monitoring di berbagai sektor meningkat dengan pesat. Semakin banyak sistem monitoring diterapkan untuk tujuan peningkatan aspek keamanan dan produktivitas. Penerapan monitoring selalu berdasarkan pada kebutuhan pengawasan secara berkala dan merekam segala aktivitas yang berlangsung di lokasi tersebut dengan harapan ketika terjadi suatu hal kritis/ penting, maka dapat segera diketahui dan ditangani.
Sistem monitoring biasanya diterapkan untuk aspek keamanan sebagai contoh pada perbankan, pergudangan, perkantoran, berbagai fasilitas publik seperti bandara, stasiun, hingga digunakan pada rumah tinggal. Sedangkan penerapan sistem monitoring untuk aspek produktivitas sebaga contoh diterapkan pada sektor manufaktur atau industri dimana manajemen dapat memonitor atau memantau aktivitas produksi para pekerja / buruh, mengkontrol instrumentasi proses, instalasi permesinan, dan lain – lain. Dan tentunya masih banyak tujuan – tujuan lain yang mendasari penerapan sistem monitoring tersebut.
Oleh karena itu penggunaan kamera pada sistem pengawasan sangat dibutuhkan. Akan tetapi yang menjadi permasalahan adalah kamera yang dipasang hanya merekam saja meskipun tidak ada gerakan atau kejadian yang terjadi, akibatnya waktu dari penjaga pengawas kamera tersebut terbuang sia – sia dan salah satu alternatif untuk mengatasi permasalahan ini adalah dengan merancang suatu perangkat lunak yang dapat meningkatkan efisiensi kamera, sehingga kamera akan mendeteksi dan memberi tanda apabila ada gerak atau benda yang bergerak.
Dengan menggunakan algoritma kernel – based gerakan yang terekam dalam kamera dapat dideteksi. Jika terdefinisikan beberapa jenis pergerakan, maka problem pengenalan gerakan suatu objek bisa digunakan untuk memisahkan objek yang bergerak tersebut dengan latar belakangnya (Background Substraction). Karena berhubungan dengan gerakan, maka yang menjadi input dalam sistem deteksi adalah klip video atau kumpulan frame image berukuran n x m yang merekam dan mempresentasikan gerakan objek.
Besarnya dimensi image frame tentunya memperlambat proses komputasi, oleh karenanya biasanya video frame yang digunakan adalah video yang beresolusi rendah dan telah dikonversi terlebih dahulu ke Gray Scale. Untuk melakukan pengurangan dimesi dan ekstraksi fitur terdapat beberapa metode yang dibedakan antara lain metode linear seperti Thresholding,Principal Component

Analysis, Linear Discrimant Analysis atau metode non-linear seperti Isomap, Locally Linear Embedding.
Pada penelitian sebelumnya telah banyak digunakan metode lain seperti template matching, Lucas Kanade, Contour Tracking, dan sebagainya. Pada metode – metode tersebut terdapat kekurangan dan kelebihannya masing – masing dibandingkan dengan metode kernel – based. Metode ini lebih cepat dalam pemrosesannya akan tetapi bila dibandingkan dengan metode yang lain metode ini kurang akurat dalam mendeteksi objek.

2. KERNEL – BASED OBJECT TRACKING

2.1 Target Model

Sebuah target diwakili oleh daerah ellipsoidal dalam gambar [3], [5]. Untuk menghilangkan pengaruh dari perbedaan target dimensi, semua target dinormalkan menjadi satuan lingkaran. Ini didapatkan dengan cara rescaling secara independen pada dimensi baris dan kolom dengan hx dan hy.
$\{X_i^*\}$ i = 1...n menjadi lokasi pixel yang telah di normalisasi pada daerah yang menjadi target model. Daerah target model ini berpusat pada 0. Sebuah kernel isotropik, dengan penurunan kernel cembung dan monoton k(x)1, memberikan bobot yang lebih kecil pada pixel yang jauh dari pusat. Menggunakan kenaikan bobot ini kekokohan dari estimasi kerapatan pixel paling dapat diandalkan, yang sering dipengarauhi oleh oklusi atau gangguan dari background.
Fungsi b : R2→{1...m} menunjuk pada pixel yang berlokasi pada $\{X_i^*\}$ index b(X_i^*) dari array yang telah terkuantisasi. Probabilitas dari u = 1...m pada target model kemudian dapat dihitung menjadi seperti pada persamaan 1.

$$\hat{q}_u = C \sum_{i=1}^{n} k(\|x_i^*\|^2)\delta[b(\mathrm{x}_i^*) - u] \qquad (1)$$

Dimana δ adalah fungsi selisih Kronicker. Normalisasi dari konstan C diturukan dengan memaksakan kondisi q̂u = 1, dari persamaan 2.

$$C = \frac{1}{\sum_{i=1}^{n} k\left(\left\|\mathrm{x}_i^*\right\|^2\right)} \qquad (2)$$

Jumlah total dari fungsi selisih untuk u = 1...m adalah 1.

2.2 Target Kandidat

$\{X_i^*\}$i = 1...n menjadi lokasi pixel yang telah di normalisasi dari target kandidat, berpusat di y pada frame saat ini [1], [2], [6]. Nilai dari normalisasi merupakan turunan dari frame yang memuat target model. Menggunakan profil kernel yang sama k(x), tetapi dengan luas bidang h, probabilitas dari u = 1...m pada target kandidat didapat dari persamaan 3.

$$\hat{p}_u(\mathrm{y}) = C_h \sum_{i=1}^{n_h} k\left(\left\|\frac{\mathrm{y}-\mathrm{x}_i}{h}\right\|^2\right)\delta[b(\mathrm{x}_i) - u] \qquad (3)$$

Dimana

$$C_h = \frac{1}{\sum_{i=1}^{n} k\left(\left\|\frac{\mathrm{y}-\mathrm{x}_i}{h}\right\|^2\right)} \qquad (4)$$

Adalah normalisasi konstan. Ch tidak tergantung pada y, karena lokasi pixel xi disusun pada kisi regular dan y adalah salah satu dari kisi node [2]. Oleh sebab itu, Ch dapat dihiting menggunakan kernel dan nilai h yang berbeda. Luas bidang h menunjukkan skala dari target kandidat, misalnya, jumlah pixel yang terdapat pada proses lokalisasi.

2.3 Ellipsoidal Region

Suatu target di representasikan dengan menggunakan ellipsoidal region pada gambar [1]. Cara ini dapat dilakukan dengan cara mengambil dua minimum dan maximum (x, y) koordinat pixel dari bounding box target, dimana diinisialisasi pada frame pertama dengan warna merah. Dari koordinat tersebut nilai center koordinat, tinggi dari target, lebar dari target, panjang dari target, panjang semimajor dan minoraxis dari ellipsoidalregion dihitung. Dengan bantuan dari nilai – nilai yang didapat tersebut, semua koordinat pixel dari ellipsoidal region dari ruang target dapat diperoleh dengan menggunakan rumus pada persamaan 5.

$$\left(1 - \left(\left(\frac{x_i - c_o}{b}\right)^2 + \left(\frac{y_i - d_0}{a}\right)^2\right) \geq 0\right) \qquad (5)$$

Dimana Xi dan Yi adalah baris dan kolom dari koordinat pixel pada ruang target, dan center koordinat dari ellipse, dan a adalah panjang dari semimajoraxis, dan b adalah panjang dari semi minor axis.

2.4 Color Histogram

Probabilty density function (PDF) dari warna u di representasikan dengan menggunakan m – bin *color histogram* dan ruang warna (R, G, B) di bagi menjadi 16*16*16 bin [1]. Setiap bin menunjukkan jarak dari nilai *pixel*, misalnya bin0 adalah (0 – 15), jarak dari bin1 adalah (16 – 31) dan seterusnya hingga bin15 adalah (240 – 255). Nilai dari m adalah 4095 warna dan jarak dari warna adalah u yang bernilai 0 sampai 4095 (m).

2.5 Bhattacharyya Coefficient

Fungsi bhattacharyya digunakan untuk menentukan jarak antara target model dan target kandidat [4]. Untuk mengakomodasi perbandingan antara beberapa target, jarak ini harus memiliki struktur matriks [4], [5], [6]. Kita mendefinisikan jarak antara dua distribusi diskrit seperti pada persamaan 6.

$$d(\mathrm{y}) = \sqrt{1 - \rho[\hat{\mathrm{p}}(y), \hat{\mathrm{q}}]} \qquad (6)$$

Dimana kita memilih

$$\hat{\rho}(\mathrm{y}) \equiv \rho[\hat{\mathrm{p}}(\mathrm{y}), \hat{\mathrm{q}}] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(\mathrm{y})\hat{q}_u} \qquad (7)$$

Koefisien Bhattacharya adalah ukuran perbedaan jenis yang memiliki interpretasi geometrik. Itu adalah cosinus dari sudut antara m-dimensi unit vektor $(\sqrt{p_1}.....\sqrt{p_m})$T dan $(\sqrt{q_1}.....\sqrt{q_m})$T. Pada kenyataannya p dan q adalah distribusi yang secara tereksplisit diperhitungan dengan mewakili mereka pada unit hypersphere. Pada waktu yang sama kita dapat mengintepretasikan sebagai korelasi antara vector $(\sqrt{p_1}.....\sqrt{p_m})$T dan $(\sqrt{q_1}.....\sqrt{q_m})$T. Properti dari koefisien Bhattacharyya seperti relasinya pada informasi ukuran Fisher, kualitas dari estimasi sampel, dan bentuk eksplisit untuk berbagai distribusi yang diberikan.

Pengukuran berbasis divergen telah digunakan pada computer vision. Hubungan antara Chernoff dan Bhattacharyya telah digunakan untuk menentukan efektifitas dari pendeteksian ujung.

2.6 Lokalisasi Target

Untuk mendapatkan lokasi yang berhubungan dengan target pada frame saat ini, jarak target harus diminimalisasi dengan cara memaksimalkan nilai dari bhattacharyya [4]. Prosedur lokalisasi dimulai dari posisi target pada frame sebelumnya (target model)

dan dicari pada daerah sekitarnya menggunakan informasi gradien yang di hasilkan oleh vektor mean – shift [3].

2.7 Minimalisasi Jarak

Meminimalisasikan jarak adalah sama dengan memaksimalkan nilai dari koefisien Bhattacharyya $\hat{p}(y)$ [1] – [6]. Pencarian lokasi target yang baru pada frame saat ini dimulai dari lokasi ŷ0 target pada frame yang sebelumnya. Oleh karena itu, probabilitas $\{\hat{p}_u(\hat{y}_0)\}$ u = 1...m dari target kandidat pada lokasi $\hat{y}_0$ di frame saat ini harus di hitung terlebih dahulu. Menggunakan Taylor expansion disekitar nilai – nilai $\hat{p}_u(\hat{y}_0)$, pendekatan linear dari koefisien Bhattacharyya diperoleh setelah beberapa manipulasi seperti pada persamaan 8.

$$\rho\,[\hat{p}(y), \hat{q}] \approx \frac{1}{2}\sum_{u=1}^{m}\sqrt{\hat{p}_u(\hat{y}_0)\hat{q}_u} + \frac{1}{2}\sum_{u=1}^{m}\hat{p}_u(y)\sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{y}_0)}} \qquad (8)$$

Pendekatan yang diinginkan adalah ketika target kandidat {p̂u(y)}u = 1...m tidak mengalami perubahan yang drastis dari inisialisasi {p̂u(ŷ0)}u = 1...m, yang merupakan asumsi yang paling sering valid antara frame secara berturut – turut. Kondisi p̂u(ŷ0) > 0 untuk semua u = 1...m, selalu dapat dipaksakan dengan tidak menggunakan nilai yang melanggar. Dengan beberapa manipulasi yang dilakukan, fungsi dari target kandidat berubah menjadi seperti pada persamaan 9.

$$\rho[\hat{p}(y), \hat{q}] \approx \frac{1}{2}\sum_{u=1}^{m}\sqrt{\hat{p}_u(\hat{y}_0)\hat{q}_u} + \frac{C_h}{2}\sum_{i=1}^{n_h} w_i k\left(\left\|\frac{y-x_i}{h}\right\|^2\right) \qquad (9)$$

Dimana

$$w_i = \sum_{u=1}^{m}\sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{y}_0)}}\,\delta\,[b(x_i) - u] \qquad (10)$$

Dengan demikian, untuk meminimalisasikan jarak, nilai dari koefisien Bhattacharyya harus dimaksimalkan. Estimasi kepadatan dihitung dengan profil kernel k(x) pada y di frame saat ini, dengan data yang dihitung oleh wi. Pada prosedur ini, kernel secara berulang di pindahkan dari lokasi saat ini ŷ0 menuju lokasi bari ŷ1 sesuai dengan persamaan 11.

$$\hat{y}_1 = \frac{\sum_{i=1}^{n_h} x_i w_i g\left(\left\|\frac{\hat{y}_0 - x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\|\frac{\hat{y}_0 - x_i}{h}\right\|^2\right)} \qquad (11)$$

Dimana g(x) = - k'(x), dengan asumsi bahwa turunan dari k(x) ada untuk semua x $\in$[0, ∞], kecuali untuk nilai yang terbatas.

2.8 Implementasi Algoritma

Langkah – langkah yang dilakukan dalam penggunakan algoritma *kernel – based adalah* sebagai berikut :

- Inisialisasi lokasi dari target pada frame saat ini dengan ŷ0, hitung {p̂u(ŷ0)}u = 1...m, dan evaluasi
$$\rho[\hat{p}(y), \hat{q}] = \sum_{u=1}^{m}\sqrt{\hat{p}_u(y)\hat{q}_u} \qquad (7)$$
- Turunkan weight $\{w_i\}$ i = 1...nh sesuai dengan fungsi
$$w_i = \sum_{u=1}^{m}\sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{y}_0)}}\,\delta\,[b(x_i) - u] \qquad (10)$$
- Temukan lokasi berikutnya dari target kandidat menggunakan fungsi
$$\hat{y}_1 = \frac{\sum_{i=1}^{n_h} x_i w_i g\left(\left\|\frac{\hat{y}_0 - x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\|\frac{\hat{y}_0 - x_i}{h}\right\|^2\right)} \qquad (11)$$
- While $\rho[\hat{p}(\hat{y}_1, \hat{q})] < \rho[\hat{p}(\hat{y}_0, \hat{q})]$

  Do $\hat{y}_1 \leftarrow \frac{1}{2}(\hat{y}_0 + \hat{y}_1)$

  Evaluasi $\rho[\hat{p}(\hat{y}_1, \hat{q})]$
- If $\|\hat{y}_1 - \hat{y}_0\| < \varepsilon$ Stop.

  Else Set $\hat{y}_0 \leftarrow \hat{y}_1$ kembali ke langkah ke 2.

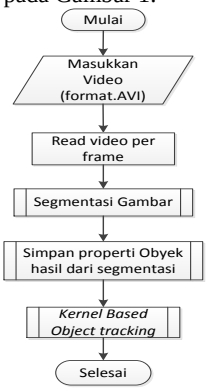
Pada pengimplementasian algoritma ini dapat dilakukan dengan cara yang lebih singkat daripada langkah – langkah yang telah dijelaskan [2]. Tujuan dari step 5 hanya untuk menghindari masalah numerik yang mungkin muncul pada proses maksimalisasi mean – shift. Masalah tersebut dapat muncul karena pendekatan linear koefisien Bhattacharyya. Akan tetapi, dari setiap penelitian yang dilakukan sejak dahulu telah menunjukkan bahwa koefisien Bhattacharyya yang dihitung pada lokasi baru y1 gagal mengalami peningkatan hanya pada 0,1% kasus. Oleh karena itu, step 5 tidak diperlukan dalam proses perhitungan, dan karena itu pula proses pengevaluasian koefisien Bhattacharyya pada step 1 dan step 1 tidak diperlukan.

Dalam algoritma yang telah dipersingkat ini kita hanya menghitung weight pada step 2, menurunkan lokasi baru pada step 3, dan menguji besar pergeseran kernel pada step 6. Koefisien Bhattacharyya hanya dihitung setelah algoritma selesai dijalankan untuk menguji kesamaan antara target model dan target kandidat.

3. DESAIN SISTEM

3.1 Desain Sistem Kerja Aplikasi

Sistem perangkat lunak yang dikembangkan untuk *object tracking* ini, memiliki beberapa tahapan penting yang harus dilakukan. Rancangan sistem kerja perangkat lunak secara garis besar dapat dilihat pada Gambar 1.



Gambar 1

Pada saat perangkat lunak pertama di jalankan user diminta untuk memasukkan video yang berformat .avi. Format video ini digunakan karena pada umumnya video menggunakan format tersebut.

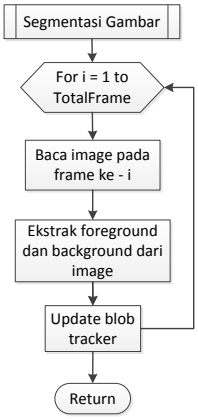
Setelah video dimasukkan, menggunakan library A Forge Video FFMPEG video tersebut akan dibaca setiap frame nya sehingga proses segmentasi dapat dilakukan. Proses segmentasi gambar bertujuan untuk menyimpan setiap obyek yang ada pada frame video yang telah dibaca tersebut.

Setelah proses segmentasi dijalankan dan obyek telah di dapatkan maka properti dari obyek tersebut akan di simpan. Properti obyek yang akan di simpan adalah titik tengah obyek, batas kiri, batas kanan, batas atas, batas bawah, dan ukuran obyek.

Proses kernel – based merupakan proses terakhir yang dijalankan ketika obyek dari setiap frame telah di dapatkan untuk dapat mengetahui obyek mana yang bergerak.

3.2 Desain Sistem Segmentasi Gambar

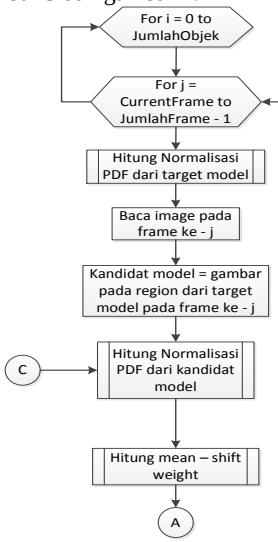
Pada proses ini, sistem akan melakukan segmentasi gambar pada setiap frame yang terdapat pada video tersebut untuk mendapatkan obyek apa saja yang ada pada frame tersebut.Proses tersebut dilakukan dengan membaca image pada setiap frame. Dari setiap frame yang dibaca akan diekstrak foreground dan background agar dapat diproses. Gambar pada frame setelah proses ekstraksi akan di proses menggunakan algoritma blob tracker untuk segmentasi sehingga bisa didapatkan obyek yang terdapat pada frame tersebut. Desain sistem segmentasi gambar dapat dilihat pada gambar 2.



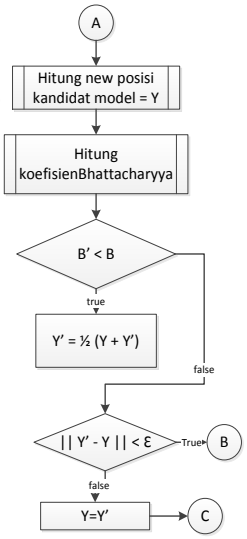
Gambar 2

3.3 Desain Sistem Metode Kernel – Based

Proses ini merupakan proses yang dilakukan setelah obyek pada setiap frame telah tersegmentasi dan di simpan pada class Objects. Obyek – obyek yang ada menggunakan algoritma kernel based akan di proses sehingga setiap obyek yang bergerak dari gabungan seluruh frame pada video dapat di deteksi dan di beri tanda. Desain sistem dari metode kernel – based dapat dilihat pada gambar 3 dan gambar 4.



Gambar 3

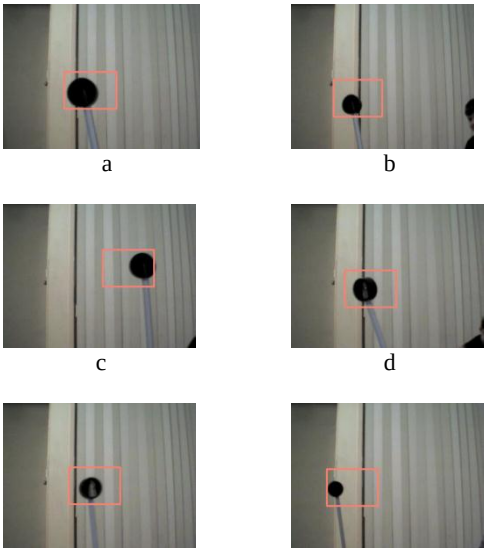


Gambar 4

Proses ini dijalankan pada setiap obyek yang ada oleh karena itu dilakukan *looping* sebanyak jumlah obyek dan juga dilakukan pada setiap *frame* sehingga dapat diketahui ada atau tidaknya perpindahan obyek tersebut pada *frame* berikutnya. Nilai normalisasi dari *Probability Density Function* (PDF) target model akan dihitung terlebih dahulu. Untuk inisialisasi posisi dari kandidat model akan dianggap sama dengan target model. Setelah itu nilai normalisasi PDF dari kandidat model akan dihitung. Perhitungan *mean – shift weight* juga dilakukan agar dapat mengetahui kemungkinan posisi dari kandidat model. Setelah didapatkan kemungkinan posisi dari kandidat model, dilakukan perhitungan koefisien Bhattacharyya dari target model maupun kandidat model. Semakin besar koefisien Bhattacharyya dari kandidat model maka kemungkinan posisi dari target model dan kandidat model semakin dekat. Update posisi dari kandidat model. Bila koefisien Bhattacharyya dari kandidat model tidak lebih besar dari koefisien Bhattacharyya dari target model maka dilakukan pengecekan apakah selisih nilai dari posisi kandidat model dengan target model sudah lebih kecil dari bilangan natural, bila benar maka proses dilanjutkan pada *frame* berikutnya. Bila hasil dari pengecekan yang dilakukan tidak sesuai dengan yang diharapkan yaitu selisih dari posisi kandidat model dengan target model lebih kecil dari bilangan natural, maka proses akan diulang dari penghitungan nilai normalisasi PDF kandidat model. Setelah semua dijalankan proses yang sama akan dilakukan pada obyek berikutnya.

4. HASIL

Aplikasi di uji cobakan pada komputer dengan spesifikasi prosesor Intel® i7™ Q740 4 core @ 1.73 GHz dengan RAM 8 GB dan graphic card NVIDIA GeForce GTX 460M dengan VRAM 1.5GB dan menggunakan program C#. Rata – rata waktu yang diperlukan untuk pendeteksian objek adalah 13.477 detik. Nilai koefisien Bhattacharyya yang digunakan adalah 0,8 dan nilai numFrameToRead yang digunakan adalah 10. Gambar 5, 6, dan 7 menunjukkan hasil dari objek yang di *track*



a b c d e f

Gambar 5 : Bola hitam – frame 51 (a), 100 (b), 242 (c), 323 (d), 400 (e), 546 (f)

a b

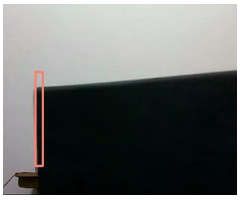
c

d

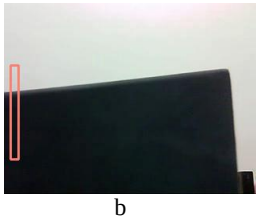
e

f

Gambar 6 : Orang Berjalan – frame 11 (a), 21 (b), 31 (c), 41 (d), 51 (e), 61 (f)



a



b

d

e

f

Gambar 7 : Kotak Hitam – frame 12 (a), 35 (b), 55 (c), 75 (d), 81 (e)

5. KESIMPULAN

Berdasarkan hasil pengujian yang dilakukan, dapat disimpulkan beberapa hal sebagai berikut:

- Pengenalan objek pertama kali ketika video di load terpengaruh dari jumlah frame yang dibaca dan nilai koefisien Bhattacharyya yang merupakan parameter yang diinput oleh user.
- Objek yang berukuran kurang dari 1% dan lebih dari 7% total resolusi video membuat proses pendeteksian objek menjadi kacau

6. DAFTAR PUSTAKA

[1] Baviskar, S.P., Ujgare, N,S. (2010). *Kernel Based Object Tracking Using Means Shift Method*, Prentice Hall Horngren, Charles T., Foster, George. (1994).

[2] Comaniciu, D., Ramesh, V., Meer, P. (2003). *Kernel – Based Object Tracking*. IEEE Trans. on Pattern Analysis and Machine Intelligence Volume 25, no.5, May 2003

[3] Dedeoglu, Y. (2004). Moving Object Detection, Tracking and Classification for Smart Video Surveillance, Dept. Of Computer Engineering, Bilkent University

[4] Hudson, F., Thacker, N., Rockett, P. (1998).The Bhattacharyya Metric as an Absolute Similiarity Measure for Frequency Coded Data, Versailles, France

[5] Kadarla, S.K. (2009). Object Tracking in Video Image Based On Image Segmentation and Pattern Matching, Dept. Of Electrical Engineering,Rourkela

[6] Liu R., Jing Z. (2011). *Robust kernel – based tracking algortihm with background contrasting*, School of Aeronautics and Astronautics, Jiao Tong University, Shanghai