Deteksi Kebakaran pada Video Berbasis Pengolahan Citra dengan Dukungan GPU

Adhi Prahara
Jurusan Ilmu Komputer dan Elektronika
Fakultas MIPA, Universitas Gadjah Mada
Sekip Utara Bulaksumur, Yogyakarta, Indonesia
adhi_prahara@yahoo.com

***Abstract*—Kebakaran adalah kejadian timbulnya api yang tidak terkendali yang dapat membahayakan jiwa maupun harta benda. Kebakaran sulit untuk diprediksi kapan dan dimana terjadinya. Misalnya kebakaran hutan, kebakaran pada perumahan atau bangunan bisa terjadi pada siang maupun malam hari. Kamera CCTV yang dapat memantau lokasi selama 24 jam sangat efektif untuk mencegah terjadinya kebakaran. Oleh karena itu, kami membangun sistem yang dapat mendeteksi kebakaran pada video berbasis pengolahan citra dengan dukungan GPU (*Graphic Processing Unit*).**

Sistem memproses data video dengan menerapkan deteksi gerakan menggunakan metode *Adaptive-Gaussian Mixture Model* (*Adaptive-GMM*) kemudian dikombinasikan dengan hasil segmentasi warna api untuk mendeteksi kebakaran. Sistem menggunakan pengolahan citra pada GPU untuk metode yang memerlukan komputasi tinggi agar bisa tercapai sistem deteksi kebakaran secara *real time*.

Hasil pengujian menghasilkan akurasi sebesar 97,96% dengan kesalahan sebesar 0,25% untuk deteksi kebakaran pada siang hari dan akurasi sebesar 98,65% dengan kesalahan sebesar 0,85% untuk deteksi kebakaran pada malam hari. Sistem mampu mendeteksi kebakaran secara *real time* yaitu sekitar 35,46 *fps* pada resolusi video 800x600 dan lebih cepat pada resolusi yang lebih kecil. Kesalahan deteksi terutama disebabkan adanya perubahan cahaya yang mendadak dari sumber api terhadap lingkungan sekitar dan pantulan cahaya kebakaran pada permukaan yang mengkilap.

Keywords— deteksi kebakaran; adaptive-GMM; statistik warna api; pengolahan citra GPU

I. PENDAHULUAN

Kebakaran adalah kejadian yang menimbulkan terjadinya api yang tidak terkendali yang dapat membahayakan jiwa maupun harta benda. Kebakaran dapat terjadi di hutan, bangunan di perkotaan, perumahan, tempat umum dan lain-lain baik siang maupun malam hari. Gambar 1 dan Gambar 2 secara berurutan menunjukkan kebakaran di hutan dan bangunan pada siang hari. Gambar 3 dan Gambar 4 secara berurutan menunjukkan kebakaran di hutan dan ruangan pada malam hari.

Karena sulit untuk untuk memperkirakan lokasi dan waktu terjadinya kebakaran maka fungsi kamera CCTV yang sudah banyak dipasang pada bangunan dan tempat-tempat umum sangat efektif untuk memantau lokasi setiap waktu. Dengan ini dampak dari kebakaran dapat diminimalisir dengan adanya deteksi secara dini. Oleh karena itu, kami membangun sistem yang dapat mendeteksi kebakaran pada video berbasis pengolahan citra dengan dukungan GPU. GPU digunakan agar metode yang digunakan pada sistem deteksi kebakaran dapat dilakukan secara *real time*.

Gambar 1. Kebakaran hutan pada siang hari

Gambar 2. Kebakaran bangunan pada siang hari

Gambar 3. Kebakaran hutan pada malam hari

Gambar 4. Kebakaran ruangan pada malam hari

II. METODE PENELITIAN

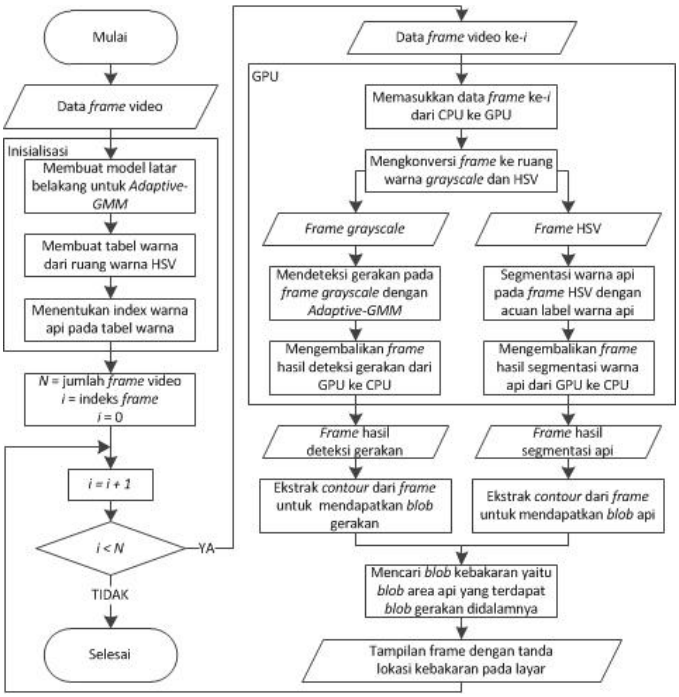
A. Analisis Sistem

Metode untuk mendeteksi kebakaran pada video dengan pengolahan citra sudah dikembangkan dan menunjukkan hasil yang baik diantaranya yang relevan dengan penelitian ini yaitu menggunakan deteksi gerakan yang digabungkan dengan statistik warna api untuk mendeteksi kebakaran. Celik dkk menggunakan *unimodal-Gaussian* untuk mendeteksi gerakan yang digabungkan dengan statistik warna api dalam ruang warna RGB untuk mendeteksi kebakaran [1]. Permana dkk menggunakan metode *frame different* untuk mendeteksi gerakan yang digabungkan dengan statistik warna api pada ruang warna RGB untuk mendeteksi kebakaran [2]. Pada penelitian ini, kami menggunakan kombinasi antara deteksi gerakan dengan *Adaptive-Gaussian Mixture Model* dengan statistik warna api pada ruang warna HSV untuk mendeteksi kebakaran. Untuk melakukan deteksi kebakaran secara *real*

time, kami menggunakan GPU dalam penerapan metodenya. Pengolahan citra pada GPU terutama yang menggunakan CUDA seperti penerapan filter dan konvolusi pada citra mengalami peningkatan kecepatan yang signifikan dibandingkan dengan proses pada CPU tanpa mengurangi presisi dari perhitungannya [3]. GPU mempunyai fitur komputasi paralel yang tinggi, memori yang besar dan mendukung pemrosesan dengan menggunakan banyak kartu grafis.

B. Perancangan Sistem

Secara umum, rancangan sistem deteksi kebakaran pada video berbasis pengolahan citra dengan dukungan GPU terdiri dari pembacaan data *frame* video, inisialisasi model untuk *Adaptive-GMM*, membuat tabel warna, praproses *frame* video, deteksi gerakan dengan *Adaptive-GMM*, segmentasi warna api, mendeteksi kebakaran dari kombinasi hasil deteksi gerakan dan segmentasi warna api, dan menampilkan hasil deteksi pada layar. GPU digunakan pada metode yang memerlukan komputasi yang tinggi seperti konversi warna ke ruang warna *grayscale* dan HSV, deteksi gerakan dan segmentasi warna api. Gambar 5 menunjukkan diagram alir rancangan sistem deteksi kebakaran pada video secara umum.



Gambar 5. Rancangan sistem deteksi kebakaran pada video

Adapun penjelasan tahapan sistemnya sebagai berikut :

1. Membaca data *frame* video.
2. Inisialisasi parameter dan model latar belakang untuk *Adaptive-Gaussian Mixture Model* (*Adaptive-GMM*).
3. Membuat tabel label warna berdasarkan nilai warna pada ruang warna HSV untuk keperluan segmentasi warna api.
4. Memasukkan *frame* hasil pembacaan data video ke GPU.
5. Mengkonversi *frame* dari RGB ke *grayscale* dan HSV.
6. Melakukan deteksi gerakan pada *frame grayscale* menggunakan *Adaptive-GMM*.
7. Melakukan segmentasi warna api pada *frame* HSV dengan mengacu pada tabel label warna api.
8. Mengembalikan *frame* hasil deteksi gerakan dan *frame* hasil segmentasi warna api dari GPU ke CPU.
9. Mengekstrak *contour* pada *frame* hasil deteksi gerakan untuk mendapatkan *blob* gerakan dan pada *frame* hasil segmentasi warna api untuk mendapatkan *blob* api.
10. *Blob* hasil segmentasi warna api yang didalamnya terdapat *blob* gerakan dari hasil deteksi gerakan merupakan *blob* kebakaran.
11. Menentukan lokasi piksel kebakaran berdasarkan posisi *blob* kebakaran pada *frame*.
12. Menampilkan *frame* yang sudah ditandai lokasi kebakarannya pada layar.
13. Mengulangi langkah 4-12 sampai semua *frame* pada video diproses.

Sistem dibuat menggunakan bahasa pemrograman C++ dengan tambahan *library* untuk pengolahan citra yaitu OpenCV 2.4.10 yang mendukung GPU dan CUDA *toolkit* 6.5 dari NVidia untuk operasi menggunakan GPU. Spesifikasi perangkat keras menggunakan *notebook* dengan prosesor Intel Core i5, NVidia GT540M 2GB CUDA dan RAM 4GB. Pembacaan *frame* video, konversi warna pada GPU, dan deteksi gerakan pada GPU dilakukan dengan dukungan *library* OpenCV sedangkan segmentasi warna api pada GPU menggunakan CUDA.

1) General Purpose-Graphic Processing Unit (GP-GPU)

Komputer grafik dan komputer vision sebenarnya saling berlawanan. Sekarang perangkat keras untuk grafik diperkuat dengan adanya prosesor yang disebut *Graphic Processing Unit* (GPU). GPU merupakan arsitektur paralel tingkat tinggi yang digunakan untuk melakukan operasi pada komputer grafik dengan cepat. Biasanya GPU digunakan untuk mengkonversi angka ke dalam citra (komputer grafik) dan bisa juga digunakan untuk mengkonversi citra ke angka (komputer vision). Proses tersebut sering disebut dengan GP-GPU (*General Purpose-GPU*) yang menggunakan kartu grafis untuk melakukan komputasi untuk tugas selain grafik [3].

Compute Unified Device Architecture (CUDA) dari NVidia mengijinkan pengguna untuk menggunakan API yang tidak memerlukan pengetahuan tentang grafik *pipeline* GPU. Kecepatan *sharing* memori diantara kelompok prosesor dapat digunakan sebagai *cache* program yang mempercepat operasi lokal. Dari pengenalan penggunaan GPU dengan CUDA dapat digambarkan bahwa GPU melakukan pendekatan pengolahan citra 2D sebagai bentuk dari pengolahan 3D. Secara sederhana, sebuah *quadrilateral* poligon dibariskan terhadap area kotak layar pada citra dan di-*render* ke dalam layar atau sebagai *buffer* pada memori. Pada setiap piksel dapat diterapkan pengolahan berbasis tekstur dan dapat diterapkan satu atau lebih program kecil pada piksel tersebut yang disebut dengan piksel *shader* untuk membuat warna keluaran pada setiap piksel. GPU mengeksekusi *shader* pada banyak piksel setiap waktu secara paralel. Piksel *shader* beroperasi pada lingkungan eksekusi *IEEE floating-point* walaupun masukan dan keluaran citranya dapat diekspresikan sebagai 8-*bit* atau 16-*bit integer* pada setiap *channel* atau sebagai 16-*bit* atau 32-*bit IEEE floating-point* pada setiap channel. Fleksibilitas dan presisi ini mengijinkan operasi piksel untuk dieksekusi pada GPU tanpa adanya kualitas citra yang hilang bila dibandingkan dengan metode intensif dengan CPU.

OpenCV merupakan *library* untuk pengolahan citra. Metode-metode penting dalam pengolahan citra kebanyakan sudah ada pada OpenCV. OpenCV juga mudah untuk digunakan dan selain itu OpenCV 2.4 ke atas sudah mengimplementasikan beberapa metodenya pada GPU [4]. OpenCV menggunakan NVidia CUDA dalam implementasi metode-metodenya pada GPU. Metode yang membutuhkan komputasi yang berat menjadi optimal pada GPU dan didapatkan peningkatan kecepatan beberapa kali dibanding dengan CPU tergantung dari kompleksitas metodenya.

2) Masukan sistem

Masukan sistem berupa data video beresolusi 320x240, 640x480, dan 800x600 yang masing-masing berjumlah 30 video dengan *fps* 30. Data video berupa hasil rekaman kebakaran dari kamera CCTV dan rekaman kamera amatir yang mempunyai kriteria berdurasi rata-rata 10 detik, rekaman bersifat statik atau relatif statik (sedikit gerakan kamera), *frame* hasil rekaman dalam ruang warna RGB, dan kebakaran dapat terlihat jelas. Data video terdiri dari kebakaran yang terjadi di hutan, bangunan, dan ruangan yang terjadi pada siang hari dan malam hari untuk variasi data.

3) Deteksi gerakan dengan Adaptive-Gaussian Mixture Model (Adaptive-GMM)

Deteksi gerakan menggunakan metode *Adaptive-Gaussian Mixture Model* [5]. Metode ini mendeteksi gerakan dengan memodelkan setiap piksel ke dalam *M* komponen distribusi *Gaussian* serta melakukan perbaikan bobot ($\hat{\pi}$), rata-rata ($\hat{\mu}$) dan standar deviasi ($\hat{\sigma}$) pada masing-masing komponen *Gaussian* tersebut. Kemungkinan sebuah piksel mempunyai nilai $\vec{x}$ pada waktu *T* ditunjukkan pada persamaan (1).

$$\hat{p}(\vec{x}|X_T, BG + FG) = \sum_{m=1}^{M} \hat{\pi}_m N(\vec{x}; \hat{\vec{\mu}}_m, \hat{\sigma}_m^2 I) \quad (1)$$

dimana $\hat{\pi}_m$ adalah parameter bobot, $\hat{\vec{\mu}}_1, \dots, \hat{\vec{\mu}}_m$ adalah perkiraan rata-rata dan $\hat{\sigma}_1, \dots, \hat{\sigma}_m$ adalah perkiraan varian pada komponen *Gaussian* ke-*m*. Bobot tidak negatif dan jumlah dari total bobot selalu satu sehingga diperlukan normalisasi setelah perbaikan bobot. Bila ada data baru $\vec{x}^{(t)}$ pada waktu *t* perbaikan model untuk bobot, rata-rata dan varian ditunjukkan pada persamaan (2), (3) dan (4) secara berurutan.

$$\hat{\pi}_m \leftarrow \hat{\pi}_m + \alpha\left(o_m^{(t)} - \hat{\pi}_m\right) \quad (2)$$

$$\hat{\vec{\mu}}_m \leftarrow \hat{\vec{\mu}}_m + o_m^{(t)}(\alpha/\hat{\pi}_m)\vec{\delta}_m \quad (3)$$

$$\hat{\sigma}_m^2 \leftarrow \hat{\sigma}_m^2 + o_m^{(t)}(\alpha/\hat{\pi}_m)\left(\vec{\delta}_m^T\vec{\delta}_m - \hat{\sigma}_m^2\right) \quad (4)$$

dimana $\vec{\delta}_m = \vec{x}^{(t)} - \hat{\vec{\mu}}_m$, dan kira-kira $\alpha = 1/T$. Untuk data baru dengan jarak komponen terdekat dan bobot terbesar, keanggotaan $o_m^{(t)}$ bernilai 1 selain itu bernilai 0. Jarak terdekat apabila jarak mahalanobisnya kurang dari tiga standar deviasi. Jarak kuadrat dari komponen ke-*m* yaitu $D_m^2\vec{x}^{(t)} = \vec{\delta}_m^T\vec{\delta}_m/\hat{\sigma}_m^2$. Bila tidak ada komponen dengan jarak terdekat maka dibuat komponen baru dengan $\hat{\pi}_{M+1} = \alpha, \hat{\vec{\mu}}_{M+1} = \vec{x}^{(t)}, \hat{\sigma}_{M+1}^2 = \sigma_0$, dimana σ_0 adalah inisial varian. Bila jumlah maksimum komponen tercapai, komponen dengan bobot terendah dapat dibuang. Komponen diurutkan berdasarkan bobotnya dari besar ke kecil sehingga latar belakang (*B*) dapat dihitung dengan persamaan (5).

$$B = arg\min_b\left(\sum_{m=1}^{b}\hat{\pi}_m > (1 - c_f)\right) \quad (5)$$

dimana c_f adalah maksimum bagian dari data yang boleh masuk dalam obyek bergerak tanpa mempengaruhi model latar belakang. Pemilihan jumlah komponen dilakukan secara adaptif tapi dengan batasan maksimum jumlah komponen yang diperbolehkan. Oleh karena itu, perbaikan bobot pada persamaan (2) perlu diubah menjadi persamaan (6).

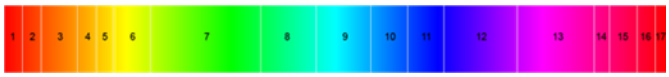
$$\hat{\pi}_m \leftarrow \hat{\pi}_m + \alpha\left(o_m^{(t)} - \hat{\pi}_m\right) - \alpha c_T \quad (6)$$

dimana $c_T = c/T$ adalah bias dengan *c* adalah koefisien dan *T* adalah jumlah sampel pada dataset. Kumulatif bobot dijaga agar selalu satu dan bila ada komponen dengan bobot negatif maka komponen tersebut dapat dibuang.

4) Segmentasi warna api

Segmentasi warna api dilakukan pada ruang warna HSV. Kriteria warna api antara lain memiliki warna merah-oranye-kuning dengan kecerahan yang tinggi. Dari kriteria tersebut, warna api dapat dengan mudah dikenali dari *Hue*. *Saturation* dan *Value* digunakan untuk mencari kecerahan warnanya. Kami menggolongkan warna pada ruang warna HSV untuk mempermudah dalam melakukan segmentasi warna api. Nilai *Hue* dari 0-360 dibagi menjadi 16 kelompok gradasi warna yang didapatkan dari percobaan. Gambar 6 menunjukkan pengelompokan label warna pada *Hue*. Nilai dari setiap label warna dari Gambar 6 dan batasan kecerahan pada *Saturation* dan *Value* ditunjukkan pada Tabel 1.

Dapat dilihat dari Tabel 1 ada satu kondisi yang belum terdaftar dalam label warna yaitu *Saturation* ≥ 12.5% dengan *Value* antara 12.5% - 87.5%. Label pada kondisi tersebut mengikuti label 1 – 16 untuk nilai *Hue*-nya akan tetapi dengan kecerahan yang lebih gelap sehingga labelnya menjadi merah gelap, merah-oranye gelap, dan seterusnya. Setiap piksel pada *frame* HSV dicocokkan dengan label warna pada Tabel 1 untuk mendapatkan piksel warna api.



Gambar 6. Kelompok label warna pada Hue

Tabel 1. Kelompok label warna pada HSV

Label	Nama Warna	Nilai H	Nilai S (%)	Nilai V (%)
1	Merah	0 – 10, 356 – 360	12.5 – 100.0	87.5 – 100.0
2	Merah-Oranye	11 – 20	12.5 – 100.0	87.5 – 100.0
3	Oranye	21 – 40	12.5 – 100.0	87.5 – 100.0
4	Oranye-Kuning	41 – 50	12.5 – 100.0	87.5 – 100.0
5	Kuning	51 – 65	12.5 – 100.0	87.5 – 100.0
6	Kuning-Hijau	66 – 80	12.5 – 100.0	87.5 – 100.0
7	Hijau	81 – 140	12.5 – 100.0	87.5 – 100.0
8	Hijau-Cyan	141 – 170	12.5 – 100.0	87.5 – 100.0
9	Cyan	171 – 200	12.5 – 100.0	87.5 – 100.0
10	Cyan-Biru	201 – 220	12.5 – 100.0	87.5 – 100.0
11	Biru	221 – 245	12.5 – 100.0	87.5 – 100.0
12	Biru-Magenta	246 – 280	12.5 – 100.0	87.5 – 100.0
13	Magenta	281 – 320	12.5 – 100.0	87.5 – 100.0
14	Magenta-Pink	321 – 330	12.5 – 100.0	87.5 – 100.0
15	Pink	331 – 345	12.5 – 100.0	87.5 – 100.0
16	Pink-Merah	346 – 355	12.5 – 100.0	87.5 – 100.0
17	Hitam	0 – 360	0.0 – 100.0	0.0 – 12.5
18	Abu-abu	0 – 360	0.0 – 12.5	12.5 – 87.5

19	Putih	0 – 360	0.0 – 12.5	87.5 – 100.0

III. HASIL DAN PEMBAHASAN

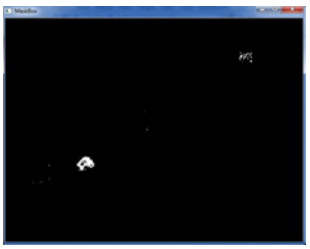
Pengujian dilakukan pada 30 data video kebakaran yang dibagi menjadi 2 bagian yaitu kebakaran pada siang hari dan malam hari yang masing-masing sebanyak 15 sampel. Dari 15 sampel dibagi menjadi 3 lokasi yang sering terjadi kebakaran yaitu di hutan, bangunan dan di dalam ruangan. Dengan variasi sampel data video, sistem diharapkan dapat mengatasi variasi kondisi kebakaran pada waktu dan lokasi kebakaran yang berbeda. Data video untuk pengujian juga menggunakan durasi yang pendek karena untuk mencapai deteksi secara dini secepat mungkin sistem harus dapat mendeteksi kebakaran terutama pada detik-detik awal kemunculan api.

Pengujian dilakukan dengan menghitung kecepatan penerapan metode pada setiap *frame* video yang diukur dalam milidetik dan *fps (frame rate per second)* serta menghitung akurasi dan kesalahan deteksi kebakaran dari sampel video. Akurasi dihitung dengan membagi jumlah *frame* yang terdeteksi sebagai kebakaran dengan jumlah *frame* kebakaran sebenarnya sedangkan kesalahan deteksi dihitung dengan membagi jumlah *frame* yang terdapat *false positif* dengan jumlah *frame* kebakaran sebenarnya. *False positif* adalah jumlah *frame* yang di dalamnya ada bagian yang bukan kebakaran tetapi terdeteksi sebagai kebakaran. *False positif* dapat dilihat dari hasil penanda lokasi kebakaran yang nampak pada layar hasil deteksi kebakaran.

C. Hasil deteksi gerakan dengan Adaptive-GMM

Deteksi gerakan digunakan dalam deteksi kebakaran karena kebakaran mempunyai kriteria seperti kibasan yang cepat dan senantiasa bergerak. Deteksi gerakan menggunakan metode *Adaptive-GMM* dengan menggunakan *library* OpenCV MOG2 GPU. Parameter yang digunakan antara lain jumlah riwayat sampel yaitu 500, *threshold* untuk jarak terdekat yaitu 25, c_f yaitu 16, dan tidak menggunakan deteksi bayangan. Gambar 7 menunjukkan video kebakaran dalam ruangan dengan lampu menyala pada siang hari sedangkan Gambar 8 menunjukkan hasil deteksi gerakan pada video kebakaran dalam ruangan dari Gambar 7.

Gambar 7. Video kebakaran dalam ruangan



Gambar 8. Hasil deteksi gerakan video kebakaran dalam ruangan

Pada Gambar 8 dapat dilihat bahwa gerakan api dapat terdeteksi dengan baik walaupun masih dalam waktu awal kemunculan api. *Learning rate* yang digunakan yaitu 0,005. Dengan *learning rate* yang relatif kecil, gerakan kibasan api yang cepat menjadi tidak terlalu cepat hilang dari layar dan tidak akan terlalu lama ada pada layar. Api yang kecil cenderung memiliki gerakan yang lambat sehingga diperlukan beberapa waktu agar dapat dideteksi sebagai gerakan sedangkan kebakaran memiliki gerakan api yang cepat sehingga akan cepat terdeteksi gerakannya. Jadi dengan deteksi gerakan, permasalahan seperti obyek yang mirip dengan warna api seperti cahaya lampu asalkan relatif statis maka tidak akan terdeteksi sebagai kebakaran karena tidak lolos dalam deteksi gerakan.

D. Hasil segmentasi api

Segmentasi warna api pada *frame* dengan melakukan pencocokan setiap piksel pada *frame* terhadap label warna pada Tabel 1 dengan menggunakan CUDA. Jumlah *thread* per blok yang digunakan yaitu 32x32 tergantung dari dukungan kartu grafis yang digunakan. Label warna api yang digunakan yaitu 1-5 yang mengacu pada Tabel 1. Gambar 9 adalah contoh kebakaran dalam ruangan pada malam hari. Hasil segmentasi warna api Gambar 9 ditunjukkan pada Gambar 10.

Pada Gambar 9, metode dapat melingkupi api dengan baik namun refleksi api pada dinding dan lampu juga ikut tersegmentasi. Warna api pada malam hari akan nampak lebih cerah, tampak silau pada kamera, serta mempengaruhi keadaan warna disekitarnya karena cahaya disekitarnya lebih redup. Walaupun hasil dari segmentasi warna api pada malam hari tidak sebaik pada siang hari, sebagian besar kesalahan dari deteksi ini akan dapat diatasi dengan metode deteksi gerakan. Misalnya untuk cahaya lampu yang relatif statis dan pantulannya pada dinding yang juga relatif statis tidak akan dideteksi. Segmentasi warna api menggunakan kriteria warna api sehingga metode ini hanya bisa diterapkan pada rekaman kamera yang menghasilkan *frame* RGB dan tidak bisa diterapkan pada kamera yang menghasilkan *frame grayscale* seperti pada mode malam.

Gambar 9. Video kebakaran ruangan pada malam hari

Gambar 10. Hasil segmentasi warna api kebakaran ruangan malam hari

E. Hasil pengujian akurasi sistem deteksi kebakaran

Pengujian ini dilakukan untuk menguji akurasi dan kesalahan sistem dalam mendeteksi kebakaran. Sampel video yang diujikan beresolusi 640x480 dengan 3 kelompok berdasarkan lokasi kebakaran yaitu kebakaran hutan, kebakaran di luar ruangan misalnya gedung, pom bensin, mobil, dan lain-lain serta kebakaran di dalam ruangan. Tabel 2 menunjukkan hasil perhitungan akurasi dan kesalahan deteksi kebakaran pada siang hari sedangkan Tabel 3 menunjukkan hasil perhitungan akurasi dan kesalahan deteksi kebakaran pada malam hari.

Berdasarkan Tabel 2, akurasi rata-rata yang didapatkan dari deteksi kebakaran pada siang hari yaitu 97,96% dengan kesalahan 0,25%. Kesalahan terutama disebabkan karena pantulan cahaya api pada dinding atau obyek yang lain yang terdeteksi sebagai kebakaran. Perubahan cahaya mendadak karena kebakaran yang membesar juga menyebabkan

terjadinya kesalahan deteksi. Gambar 11 menunjukkan hasil deteksi kebakaran pada data video Kebakaran kamar 1 yang ditandai dengan tanda kotak warna kuning. *False positif* yang terjadi ditandai dengan tanda kotak warna merah. Pada Gambar 11 terlihat bahwa layar televisi memantulkan cahaya kebakaran sehingga sistem mendeteksinya. Kesalahan seperti ini tidak bisa dihindari oleh sistem karena dari deteksi gerakan dan warna api, keduanya menganggap layar televisi pada saat itu sebagai kebakaran.

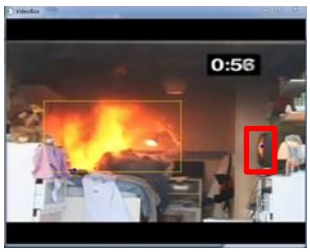
Berdasarkan Tabel 3, akurasi deteksi kebakaran pada malam hari meningkat menjadi 98,65% karena warna api yang dominan dari sekitarnya. Akan tetapi kesalahan deteksi juga semakin besar menjadi 0,85% dengan adanya sumber cahaya dari kebakaran menyebabkan lingkungan sekitarnya akan memantulkan cahaya tersebut. Gambar 12 menunjukkan hasil deteksi kebakaran pada data video Kebakaran ruang 3 yang ditandai dengan tanda kotak warna kuning. *False positif* yang terjadi ditandai dengan tanda kotak warna merah yaitu pada dinding ruangan yang memantulkan cahaya dari kebakaran. Cahaya lampu tidak terdeteksi sebagai kebakaran karena relatif statis. Karena perubahan cahaya yang mendadak dari sumber kebakaran di sekelilingnya, tanda lokasi kebakaran menjadi lebih besar. Masalah seperti ini juga tidak bisa dihindari oleh sistem karena deteksi gerakan dan warna api menganggap dinding pada saat itu sebagai piksel kebakaran. Deteksi gerakan mendeteksi gerakan dari perubahan cahaya yang mendadak karena kebakaran sedangkan segmentasi warna api mendeteksi pengaruh warna kebakaran yang memantul pada permukaan mengkilap disekelilingnya.

Tabel 2. Hasil deteksi kebakaran pada siang hari

Nama video	Total *frame*	*Frame* kebakaran	*Frame* terdeteksi	False positif	Akurasi (%)	Kesalahan (%)
Kebakaran hutan 1	265	265	262	0	98,86	0,00
Kebakaran hutan 2	611	611	564	0	92,30	0,00
Kebakaran hutan 3	178	178	176	0	98,87	0,00
Kebakaran hutan 4	247	247	247	0	100,00	0,00
Kebakaran hutan 5	220	220	220	0	100,00	0,00
Kebakaran mobil	542	542	542	0	100,00	0,00
Kebakaran truk	227	227	222	0	97,79	0,00
Kebakaran rumah	511	511	494	3	96,67	0,58
Kebakaran toko	369	369	357	0	96,74	0,00
Kebakaran gedung	203	203	197	4	97,04	1,97
Kebakaran ruang tamu	222	222	217	0	97,74	0,00
Kebakaran dapur 1	417	417	417	0	100,00	0,00
Kebakaran dapur 2	327	327	327	0	100,00	0,00
Kebakaran kamar 1	302	302	301	4	99,66	1,32
Kebakaran kamar 2	304	304	285	0	93,75	0,00
Rata-rata					97,96	0,25

Tabel 3. Hasil deteksi kebakaran pada malam hari

Nama video	Total *frame*	*Frame* kebakaran	*Frame* terdeteksi	False positif	Akurasi (%)	Kesalahan (%)
Kebakaran hutan 1	866	866	866	0	100,00	0,00
Kebakaran hutan 2	236	236	235	0	99,57	0,00
Kebakaran hutan 3	403	403	403	0	100,00	0,00
Kebakaran hutan 4	328	328	328	0	100,00	0,00
Kebakaran hutan 5	620	620	617	0	99,51	0,00
Kebakaran pom 1	302	302	301	10	99,66	3,31
Kebakaran pom 2	297	297	270	9	90,91	3,03
Kebakaran gedung 1	937	937	937	0	100,00	0,00
Kebakaran gedung 2	164	164	159	0	96,95	0,00
Kebakaran rumah	762	762	755	0	99,08	0,00
Kebakaran ruang 1	257	257	247	0	96,10	0,00
Kebakaran ruang 2	306	306	305	0	99,67	0,00
Kebakaran ruang 3	341	341	336	22	98,53	6,45
Kebakaran ruang 4	365	365	364	0	99,72	0,00
Kebakaran ruang 5	386	386	386	0	100,00	0,00
Rata-rata					98,65	0,85



Gambar 11. Hasil deteksi kebakaran kamar pada siang hari

Gambar 12. Hasil deteksi kebakaran ruang pada malam hari

F. Hasil pengujian kecepatan sistem deteksi kebakaran

Pengujian ini dilakukan untuk menguji performa metode yang menggunakan GPU dibandingkan dengan CPU. Kecepatan diukur dalam milidetik dan jumlah *frame* yang dieksekusi per detik (*fps*). Pada setiap kategori resolusi digunakan 3 video dengan *fps* 30 berdurasi rata-rata 10 detik untuk diukur kecepatan eksekusi metodenya selama 5 kali percobaan. Tabel 4 menunjukkan perbandingan kecepatan rata-rata eksekusi metode yang menggunakan CPU dan GPU pada sampel video dengan variasi resolusi video.

Dari Tabel 4 dapat dilihat bahwa pemrosesan metode deteksi kebakaran pada video pada GPU lebih cepat daripada CPU. Pada resolusi 800x600 pemrosesan video pada GPU masih mendekati *fps* video aslinya sedangkan pada CPU pemrosesan video sudah jauh lebih lambat dari *fps* video aslinya. Peningkatan kecepatan pemrosesan sistem dengan GPU sekitar 1,5 kali dari CPU. Pada metode deteksi gerakan

dengan GPU peningkatan kecepatan sekitar 3 kali dari CPU sedangkan untuk segmentasi warna api menggunakan CUDA peningkatan kecepatan sekitar 1,7 kali dari CPU. Metode yang membutuhkan pemrosesan setiap piksel pada *frame* lebih cepat apabila diproses pada GPU yang mendukung komputasi paralel. Semakin besar resolusi video maka semakin banyak piksel yang harus diproses untuk itu komputasi paralel sangat efektif. Dengan menggunakan GPU, sistem menjadi lebih fleksibel dan dapat memenuhi performa untuk deteksi secara *real time*.

Tabel 4. Kecepatan eksekusi metode dengan CPU dan GPU

Metode	Resolusi Video	Kecepatan rata-rata (milidetik)		*fps* rata-rata		Peningkatan kecepatan GPU vs CPU
		CPU	GPU	CPU	GPU	
Deteksi gerakan (*Adaptive-GMM)*	320x240	5,33	1,48	190,43	752,99	3,95
	640x480	13,83	3,90	78,25	258,74	3,30
	800x600	34,52	6,63	40,67	152,30	3,74
Segmentasi warna api	320x240	3,65	1,98	274,22	536,26	1,95
	640x480	10,47	5,73	103,75	176,68	1,70
	800x600	25,81	7,54	54,91	133,45	2,43
Total metode deteksi kebakaran	320x240	13,43	8,66	75,94	118,61	1,56
	640x480	36,23	21,32	29,85	47,46	1,59
	800x600	83,67	29,21	16,17	35,46	2,19

IV. KESIMPULAN

Berdasarkan hasil pengujian sistem deteksi kebakaran pada video berbasis pengolahan citra dengan dukungan GPU dapat diambil beberapa kesimpulan yaitu:

1. Hasil rata-rata akurasi sistem deteksi kebakaran pada siang hari yaitu 97,96% dengan kesalahan rata-rata sebesar 0,25% dari 15 sampel video.
2. Hasil rata-rata akurasi sistem deteksi kebakaran pada malam hari yaitu 98,65% dengan kesalahan rata-rata sebesar 0,85% dari 15 sampel video.
3. Kesalahan deteksi terutama disebabkan oleh pantulan cahaya kebakaran pada permukaan yang mengkilap misalnya pada sampel kebakaran ruang tamu yang dindingnya memantulkan cahaya dari kebakaran.
4. Sistem mampu mendeteksi kebakaran secara *real time* yaitu dengan rata-rata *fps* sebesar 118,61 pada resolusi 320x240, 47,46 pada resolusi 640x480, dan 35,46 pada resolusi video 800x600.
5. Peningkatan kecepatan sistem deteksi kebakaran dengan GPU sekitar 1,5 kali dari CPU.

V. SARAN

Penelitian sistem deteksi kebakaran pada video berbasis pengolahan citra dengan dukungan GPU masih banyak kekurangannya. Beberapa saran untuk pengembangan penelitian lebih lanjut sebagai berikut:

1. Sistem dapat dibuat lebih optimal dalam hal eksekusi metode dengan menggunakan GPU seperti cara penulisan kode dan penggunaan *thread* dan *block* pada GPU.
2. Sistem dapat ditingkatkan akurasinya dengan menambahkan metode untuk mengenali pantulan cahaya dari api pada permukaan mengkilap agar tidak tersegmentasi sebagai api.
3. Sistem dapat dikembangkan lebih lanjut untuk deteksi kebakaran secara *real time* pada video pemantauan lebih dari satu kamera secara bersamaan pada satu komputer.

DAFTAR PUSTAKA

[1] Celik, T., Demirel, H., dan Ozkaramanli, H., 2006, Automatic Fire Detection in Video Sequence, *14th European Signal Processing Conference (EUSIPCO)*, Florence, Italy, 4-8 September 2006.

[2] Permana, A. S., Usman, K., dan Murti, M. A., 2009, Deteksi Kebakaran Berbasis Webcam Secara Real Time dengan Pengolahan Citra Digital, *Konferensi Nasional Sistem dan Informatika*, Bali, Indonesia, 14 November 2009.

[3] Fung, J. dan Mann, S., 2008, *Using Graphics Devices in Reverse : GPU-Based Image Processing and Computer Vision*, NVidia Corporation, USA.

[4] OpenCV developer team, 2014, *GPU Module Introduction*, http://docs.opencv.org/modules/gpu/doc/introduction.html, diakses tanggal 5 Februari 2015.

[5] Zivkovic, Z., 2004, Improved Adaptive Gaussian Mixture Model for Background Subtraction, *Proceedings International Conference in Pattern Recognition (ICPR)*, UK, Agustus 2004.