BAB V
UJI COBA DAN EVALUASI

Pada bab ini akan dijelaskan hasil uji coba dan evaluasi program yang telah selesai diimplementasi.

5.1 Lingkungan Uji Coba

Lingkungan uji coba yang akan digunakan adalah,

1. Perangkat Keras
   Prosesor Intel® Core™ i3-2350M CPU @ 2.30GHz 2.30GHz RAM 4 GB.
   Sistem Operasi 64-bit .
2. Perangkat Lunak
   Sistem Operasi Microsoft Windows 8 64-bit Pro.
   Perangkat Pengembang PyCharm.

5.2 Data Uji Coba

Data yang digunakan untuk uji coba implementasi deteksi api berbasis sensor visual menggunakan metode *support vector machines* adalah potongan video yang didapatkan dari berbagai sumber. Kualitas video yang digunakan adalah video dengan *size* 240x320 piksel dan memiliki *channel* R,G,B. Data video yang digunakan diambil dari beberapa kejadian. Data video yang digunakan meliputi dua buah jenis video. Video dengan objek api dan video dengan objek bukan api. Jumlah video yang diuji berjumlah enam puluh tujuh video dengan jumlah video api sejumlah tiga puluh empat dan video bukan api berjumlah tiga puluh tiga. Contoh video kejadian dapat dilihat pada Gambar 5.1.

5.3 Alur Uji Coba

Pada sub bab ini akan dijelaskan mengenai alur kerja dari sistem deteksi api. Dimulai dari *preprocessing* hingga verifikasi.

Movie 1 Movie 2 Movie 3 Movie 4

Movie 5 Movie 6 Movie 7 Movie 8

Movie 9 Movie 10 Movie 11 Movie 12

Movie 13 Movie 14 Movie 15 Movie 16

Gambar 5.1 Contoh Video Kejadian

5.3.1 Preprocessing

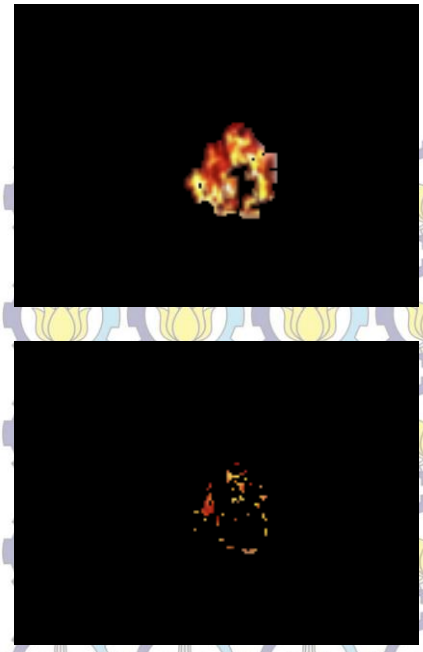
Tahap *preprocessing* akan dijelaskan bagaimana alur setiap *frame* masuk hingga menghasilkan kandidat api yang selanjutnya akan di proses pada tahap verifikasi. Ilustrasi tahap *proprocessing* dapat dilihat pada Gambar 5.2.

Frame
Masukan

Reduksi *Size*
Frame

Deteksi Gerak

Deteksi Warna
Piksel

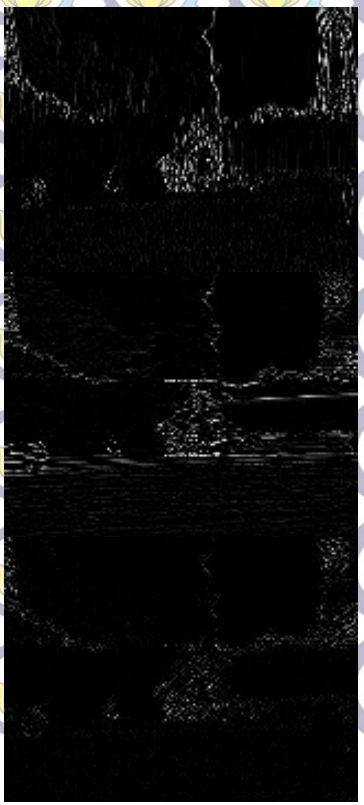


Gambar 5.2 Tahap *Preprocessing*

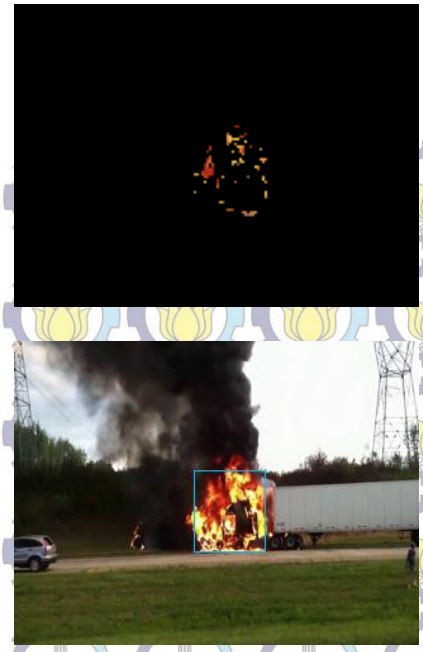
5.3.2 Verifikasi

Tahap verifikasi akan dijelaskan bagaimana alur verifikasi dilakukan. Masukan dari tahap ini adalah hasil akhir dari tahap *preprocessing*. Hasil akhir dari proses verifikasi adalah *region* yang masuk kedalam objek api. Ilustrasi proses verifikasi dapat dilihat pada Gambar 5.3.

Frame Masukan dari proses *preprocessing*



Ekstraksi Fitur, detail gambar vertikal, horizontal, digonal



Gambar 5.3 Tahap Verifikasi

5.4 Skenario Uji Coba

Pada sub bab ini akan dijelaskan mengenai skenario uji coba yang telah dilakukan. Telah dilakukan beberapa skenario uji coba, diantaranya yaitu:

1. Perbandingan hasil *true positif*, *false positif*, dan *missing rate* berdasarkan variasi nilai *threshold* pada deteksi warna api. *Threshold* yang akan diuji yaitu 10^{-7}, 10^{-8}, 5 x 10^{-9}, 10^{-9}.
2. Perbandingan hasil *true positif*, *false positif*, dan *missing rate* berdasarkan variasi nilai C (*penalty error term*) pada klasifikasi dengan kernel tetap yaitu RBF. Nilai C yang akan diuji yaitu 1, 3.5, 5,dan 7.
3. Perbandingan hasi *true positif*, *false positif*, dan *missing rate* berdasarkan variasi kernel yang digunakan pada

klasifikasi. Kernel yang akan diuji yaitu *polynomial 2*, *polynomial 3*, dan RBF.

4. Perbandingan hasil *true positif*, *false positif*, dan *missing rate* berdasarkan variasi besarnya *region* objek. Variasi yang digunakan adalah 1%, 5%, dan 10%.
5. Perbandingan kecepatan deteksi dengan variasi *size frame* yang telah direduksi. *Size frame* yang diuji adalah 240 x 320, 120 x 160, 60 x 80.
6. Perbandingan hasil *true positif*, *false positif*, dan *missing rate* dengan menghilangkan tahap *region growing* dan perhitungan luasan *region*.

5.4.1 Skenario Uji Coba 1

Skenario uji coba 1 adalah perhitungan *true positif*, *false positif*, dan *missing rate*. Dimana *true positif* adalah kondisi suatu *frame* mengandung gambar api dan terdeteksi api atau *frame* tidak mengandung api dan tidak terdeteksi api. *False positif* adalah kondisi dimana *frame* tidak mengandung gambar api, namun terdeteksi api dan *missing rate* adalah keadaan dimana suatu *frame* yang mempunyai gambar api namun tidak terdeteksi api. Pada skenario uji coba 1 dilakukan uji coba pada tahap probabilitas warna api dengan mengubah nilai *threshold* probabilitas piksel api. Nilai *threshold* yang diuji yaitu 10^{-7}, 10^{-8}, 5 x 10^{-9}, 10^{-9}. Untuk parameter nilai C pada uji coba 1 diberikan nilai 5 menggunakan kernel RBF.

Tabel 5.1 Hasil Uji Coba 1

Threshold	*True Positif (%)*	*False Positif (%)*	*Missing Rate (%)*
10^{-7}	77.58	0.37	22.05
10^{-8}	94.58	1.22	4.19
$5x10^{-9}$	96.32	1.46	2.23
10^{-9}	91.95	7.85	0.20

Dari hasil uji yang dilakukan, semakin kecil nilai *threshold* yang digunakan, hasil dari *true positif* akan semakin besar. Begitu juga untuk *false positif*, dimana makin kecil nilai *threshold* makin besar nilai *false positif*. Hal ini dikarenakan piksel yang dianggap piksel api sudah melewati batas warna kuning hingga merah. Dari uji coba tersebut didapatkan nilai *threshold* $5x10^{-9}$ sebagai nilai terbaik, karena *False positif* yang dihasilkan tidak terlalu besar dan *True Positif* bernilai besar. Hasil uji coba 1 lebih lengkap terdapat pada lampiran.

5.4.2 Skenario Uji Coba 2

Skenario uji coba 2 dilakukan dengan menghitung nilai *true positif, false positif*, dan *missing rate*. Pada skenario uji 2 dilakukan uji coba variasi nilai C pada klasifikasi, dimana nilai C adalah nilai *penalty error term*. Nilai C yang diuji yaitu 1, 3.5, 5, dan 7. Untuk parameter *threshold* warna piksel diberikan nilai 5×10^{-9} menggunakan kernel RBF sebagai klasifikasi. Hasil uji coba 2 lebih lengkap terdapat pada lampiran.

Tabel 5.2 Hasil Uji Coba 2

C	*True Positif (%)*	*False Positif (%)*	*Missing Rate (%)*
1	96.30	1.43	2.27
3.5	96.32	1.46	2.23
5	96.32	1.46	2.23
7	96.32	1.46	2.23

Hasil uji coba tahap 2 hasil terbaik didapatkan ketika nilai $C = 7$, dimana memiliki nilai *true positif* yang lebih tinggi dari yang lainnya.

5.4.3 Skenario Uji Coba 3

Pada skenario uji 3 dilakukan uji coba variasi kernel klasifikasi. Variasi kernel yang digunakan yaitu *polynomial 2*,

polynomial 3, dan RBF. Untuk parameter *threshold* warna piksel diberikan nilai 5 x 10^{-9} dan nilai C diberikan nilai 5.

Tabel 5.3 Hasil Uji Coba 3

Kernel	*True Positif (%)*	*False Positif (%)*	*Missing Rate (%)*
Polynomial 2	87.93	0.95	11.12
Polynomial 3	65.78	0.05	34.17
RBF	96.32	1.46	2.23

Hasil uji coba tahap 3 didapatkan nilai *true positif* terbaik didapatkan dengan menggunakan kernel RBF. Hasil uji coba 3 lebih lengkap terdapat pada lampiran.

5.4.4 Skenario Uji Coba 4

Pada skenario uji 4 dilakukan uji coba variasi besarnya *region*. Variasi *region* yang digunakan yaitu 1%, 5%, 10%. Untuk parameter *threshold* warna piksel diberikan nilai 5 x 10^{-9}, nilai C diberikan nilai 5 dan menggunakan kernel RBF.

Tabel 5.4 Hasil Uji Coba 4

Konstanta *Region*	*True Positif (%)*	*False Positif (%)*	*Missing Rate (%)*
1%	96.32	1.46	2.23
5%	62.92	0.47	36.61
10%	53.99	0.41	45.60

5.4.5 Skenario Uji Coba 5

Pada skenario uji 5 dilakukan uji coba variasi *size frame*. Variasi *size frame* yang digunakan yaitu 240 x 320, 120 x 160, 60 x 80. Untuk parameter *threshold* warna piksel

diberikan nilai 5 x 10^{-9} , nilai C diberikan nilai 5 dan menggunakan kernel RBF.

Tabel 5.5 Hasil Uji Coba 5

Piksel	*True Positif (%)*	*False Positif (%)*	*Missing Rate (%)*	*Execution Time (s)*
240 x 320	92.96	0.79	6.25	123.03
120 x 160	96.32	1.46	2.23	35.89
60 x 80	96.40	3.22	0.37	11.75

5.4.6 Skenario Uji Coba 6

Pada skenario uji coba 6, dilakukan Perbandingan hasil *true positif*, *false positif*, dan *missing rate* dengan menghilangkan tahap *region growing* dan perhitungan luasan *region*.

Tabel 5.6 Hasil Uji Coba 6

	True Positif (%)	*False Positif (%)*	*Missing Rate (%)*
Tanpa Menggunakan *region growing* dan perhitungan luasan *region*	88.62	11.38	0.00
Menggunakan *region growing* dan perhitungan luasan *region*	96.32	1.46	2.23

5.5 Analisis Hasil Uji Coba

Dari hasil skenario uji coba yang telah dilakukan, beberapa parameter memberikan pengaruh terhadap hasil

deteksi. Parameter yang digunakan antara lain nilai *threshold* pada deteksi warna api dan nilai C pada klasifikasi. Uji coba dilakukan dengan membandingkan nilai *true positif, false pisitif*, dan *missing rate*.

Dari uji coba 1, parameter yang di uji adalah *threshold* pada deteksi warna piksel. Hasil percobaan menunjukkan semakin tinggi nilai *threshold* yang digunakan, semakin besar *missing rate* yang dihasilkan. Hal ini dikarenakan semakin kecil kombinasi warna piksel yang dianggap sebagai warna api. Sebaliknya jika nilai *threshold* yang digunakan terlalu kecil, maka banyak piksel yang tidak termasuk warna api lolos sebagai kandidat piksel berwarna api. Nilai terbaik yang didapatkan dari hasil percobaan adalah nilai *threshold* 5×10^{-9}.

Uji coba 2, dapat diambil kesimpulan bahwa nilai C tidak berpengaruh besar pada klasifikasi. Pada uji coba 3, variasi kernel yang digunakan adalah *polynomial 3*, dan RBF. Hasil uji coba menunjukkan kernel terbaik dari variasi kernel yang digunakan adalah kernel RBF. Uji coba 4, dapat diambil kesimpulan jika konstanta yang digunakan terlalu besar banyak *region* yang dianggap *noise*. Pada uji coba 5, semakin kecil *size frame* yang diproses, waktu eksekusi yang diperlukan semakin kecil. Pada uji coba 6, penghilangan proses *region growing* dan perhitungan luasan *region* menurunkan hasil yang dikeluarkan. Hal ini disebabkan banyaknya piksel-piksel *noise* yang masuk kedalam piksel api.

Dari keseluruhan uji coba yang dilakukan, parameter-parameter tersebut menghasilkan presentase terbaik ketika *threshold* yang digunakan 5×10^{-9} dan C yang digunakan sebesar 7 dan menggunakan kernel RBF.