Pendeteksian Api pada Video Menggunakan Wavelet Dan LBP

1st Naufal Luthfi Saputra
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
naufalluthfis@student.telkomuniversity.ac.id

2nd Febryanti Sthevanie
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
sthevanie@telkomuniversity.ac.id

3rd Kurniawan Nur Ramadhani
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
kurniawannurr@telkomuniversity.ac.id

Abstrak—Keakuratan deteksi api selalu menjadi tujuan utama dalam penelitian deteksi api. Pemilihan metode yang digunakan dalam melakukan pendeteksian api merupakan hal yang paling mempengaruhi nilai akurasi dalam deteksi api. Penulis menggunakan metode rule based untuk memfilter nilai pixel api, lalu digunakan metode spatial analisis untuk mendapatkan nilai energi dari objek api dan menggunakan *Local Binary Pattern sebagai texture analysis* dan *Support Vector Machine* sebagai *classifier* dan didapatkan akurasi 89,35%.

Kata kunci—deteksi api, *rule based, spatial analisis , wavelet transform, Local Binary Pattern, Support Vector Machine*

Abstract—The accuracy of fire detection has always been the main goal in fire detection research. The selection of the method used in detecting fire is the thing that most affects the accuracy value in detecting fire. The author uses a rule based method to filter the fire pixel values, then uses a spatial analysis method to get the energy value of the fire object and uses a Local Binary Pattern as texture analysis and a Support Vector Machine as a classifier and gets an accuracy of 89.35%.

Keywords—fire detection, rule based, spatial analysis , wavelet transform, Local Binary Pattern, Support Vector Machine

I. PENDAHULUAN

A. Latar Belakang

Api dihasilkan dari sebuah reaksi kimia yang memiliki sifat panas dan membakar. Sifat dari api dapat menyebabkan proses terbakarnya suatu benda yang dapat terjadi secara sengaja maupun tidak disengaja dan didalam ruangan maupun diluar ruangan. Proses pembakaran yang terjadi secara tidak sengaja dapat menimbulkan berbagai dampak negatif. Oleh sebab itu, penulis merancang sistem untuk mendeteksi keberadaan api sejak dini, dengan memanfaatkan kamera CCTV sebagai media pembantu proses pendeteksian api melalui *video processing*. Sistem yang dibuat diharapkan dapat mendeteksi api sejak dini dan menghindari dampak negatif yang dapat ditimbulkan oleh api.

Berbagai penelitian telah banyak dilakukan oleh peneliti dalam mengembangkan sistem pendeteksian api. Pada tahun 2010 dilakukan penelitian untuk mendeteksi api pada ruang lingkup yang besar dengan menggabungkan metode thresholding dan edge detection. Pada penelitian ini didapatkan sebuah hasil apabila metode edge detection sangat rentan terhadap noise yang ada pada sebuah frame video, sehingga menghancurkan garis dari *nonuniform illumination* dan menghasilkan diskontinuitas intensitas palsu. Namun dengan metode ini didapatkan hasil apabila objek api kurang terpengaruh oleh berbagai faktor interferensi dan lebih efektif dalam mendeteksi tepi dari objek api[1]. Pada tahun 2015 dilakukan penelitian dengan menggunakan metode RGB dan YCbCr *color space*. Metode ini difokuskan untuk meningkatkan dan menguatkan gambar, sehingga dapat menghilangkan semua *noise* yang ada pada gambar. Pada metode ini dilakukan proses pendeteksian api dengan meneliti *pixel* api dalam format RGB dan mengkonversi *pixel* menjadi format YCbCr agar dapat dilakukan analisis. *Region* dari *pixel* api didapatkan dari hasil *thresholding* yang kemudian dilakukan proses ekstraksi fitur menggunakan tujuh buah rules untuk mengenali objek api. Dari penelitian yang dilakukan dengan cara mengaplikasikan metode ini ke 100 buah gambar, didapatkan hasil akurasi sebesar 90% pada metode RGB color space dan 100% untuk metode YCbCr. Kedua metode tersebut dapat digunakan untuk mendeteksi api, namun YCbCr mendapatkan hasil yang lebih besar dikarenakan YCbCr *color space* memisahkan *luminance* dari *chrominance* lebih efektif daripada RGB color space. Pada metode ini dapat membedakan objek sekitar api dan objek mirip api yang memiliki *pixel value* mirip dengan api sesungguhnya[2]. Pada tahun 2018 dilakukan penelitian menggunakan metode fitur warna dan analisis wavelet. Fitur warna digunakan untuk menghilangkan efek iluminasi dari gambar dengan melakukan normalisasi RGB. Hasil dari fitur warna digunakan untuk mengekstraksi area dari api dengan memanfaatkan nilai *threshold* yang didapatkan dari menghitung rata rata dari standar deviasi dari area api. Area api yang dihasilkan kemudian dilakukan *eroding* untuk menghilangkan *noise* dari gambar. Setelah *noise* sudah berkurang kemudian dilakukan metode *filtering* dengan memanfaatkan *wavelet transform*. Wavelet *transform* yang digunakan yaitu discrete wavelet transform (DWT) untuk mendapatkan tiga buah hasil frekuensi tinggi yang terdiri dari HH, LH, HL dan LL yang merepresentasikan sebagai *high-high, low-high, high-low* dan *low-low* secara berurutan. Frekuensi yang didapatkan lalu dilakukan perhitungan untuk mendapatkan nilai energy. Dari

penelitian ini didapatkan apabila metode fitur warna dan wavelet *transform* dapat meningkatkan reliabilitas dari pendeteksian api[3]. Pada tahun 2019 dilakukan penelitian pada video menggunakan LBP dan *spread ascending of smoke*. Pada penelitian ini didapatkan hasil yang menunjukkan *boundary* dari region objek bisa didapatkan dengan mencari karakteristik, ekspansi, dilusi, transfluenci dan tekstur[4]. Pada tahun 2013 telah dilakukan penelitian pendeteksian api menggunakan SVM *classification* yang menunjukkan hasil deteksi api bergantung terhadap data *training* dan data *test*. Pada penelitian ini digunakan dua metode yaitu *pixel-based algorithm* dan SVM *classifier*. Dalam fase *pixel-base*, lima vektor fitur berdasarkan ruang warna RGB digunakan untuk mengklasifikasikan *pixel* dengan menggunakan pengklasifikasi Bayes untuk membangun *mask* api dari gambar. Kemudian dilanjutkan dengan SVM *classifier* untuk membedakan potensi dari objek api atau bukan. Berdasarkan penelitian tersebut didapatkan hasil yang menunjukkan apabila penelitian tersebut mendapatkan hasil yang efektif dan juga memiliki kualitas yang baik[5].

Berdasarkan hasil dari beberapa penelitian diatas, penulis mengembangkan sebuah sistem yang memanfaatkan kelebihan dari setiap metode dan mengurangi kekurangan dari setiap metode dengan cara menggabungkan setiap metode untuk meningkatkan keakuratan dalam pendeteksian api. Sistem pendeteksi api akan diawali dengan melakukan *moving object detection* untuk mendapatkan perbedaan dari setiap frame yang didapatkan dari video dikarenakan api pada dasarnya selalu bergerak dan tidak memiliki *pattern* pergerakan yang sama. Kemudian RGB dan HSV *color space*, dikarenakan pada hasil peneltian sebelumnya YCbCr telah menunjukkan nilai akurasi 100%. Sehingga sistem yang dirancang menggunakan RGB dan HSV color space. Setelah dilakukan metode RGB dan HSV color space terlihat jika *pixel* api banyak yang tereliminasi sehingga pada metode ini ditambahkan dengan menggunakan metode region growing untuk memperbesar region dari objek api. Kemudian dilakukan *spatial analysis* berbasis pada wavelet analysis untuk mengkalkulasikan *spatial energy* agar dapat membedakan antara api dengan objek yang mirip api. Kalkulasi *spatial energy* akan dilakukan dengan menggunakan metode wavelet *analysis*. Apabila terdapat nilai *spatial energy* antara api dengan objek mirip api yang tidak berbeda jauh maka akan dilakukan deteksi menggunakan *texture analysis* menggunakan *Local Binary Pattern* (LBP). *Classifier* yang akan digunakan yaitu SVM.

B. Perumusan Masalah

Adapun perumusan masalah dari pengerjaan tugas akhir ini sebagai berikut :

1. Bagaimana cara mengimplementasikan sistem pendeteksian api pada sebuah video dengan menggunakan metode *rule based fire detection*, *spatial analysis* (wavelet), *Local Binary Pattern* (LBP) dan *Support Vector Machine* (SVM) ?
2. Bagaimana performa dari sistem pendeteksian api yang didapat dengan menggunakan metode *rule based fire detection*, *spatial analysis* (wavelet), *Local Binary Pattern* (LBP) dan *Support Vector Machine* (SVM) ?

C. Batasan Masalah

Adapun ruang lingkup batasan masalah dari pengerjaan Tugas Akhir ini sebagai berikut:

1. Video yang dipakai memiliki resolusi yang berbeda-beda.
2. Video yang dipakai memiliki sudut pengambilan video yang berbeda-beda dan juga memiliki jarak jauh dekat yang berbeda.

D. Tujuan

Berdasarkan perumusan masalah di atas, maka tujuan dari pengerjaan tugas akhir ini adalah:

1. Membangun sistem pendeteksian api dengan menerapkan metode *rule based fire detection*, *spatial analysis* (wavelet), *Local Binary Pattern* (LBP) dan *Support Vector Machine* (SVM)
2. Mengetahui performa dari sistem pendeteksian api dengan menggunakan *rule based fire detection*, *spatial analysis* (wavelet), *Local Binary Pattern* (LBP) dan *Support Vector Machine* (SVM).

II. KAJIAN TEORI

A. *Moving Object Detection*

Moving object detection merupakan sebuah tindakan segmentasi objek *non-stasioner* berdasar kepentingan hubungan antar daerah sekitarnya atau wilayah dari *frame* yang diberikan[6]. Penentuan target bergerak merupakan langkah dasar untuk proses klasifikasi dan pelacakan objek yang bergerak. Tujuan utama dari *moving object detection* adalah untuk menemukan target bergerak pada *foreground*[7] baik di setiap frame video atau pada kemunculan pertama target bergerak dalam video [8]. Setiap ditemukan objek baru yang bergerak, objek tersebut tidak memiliki pattern pergerakan yang sama setiap terjadi pergerakan dapat mempengaruhi nilai *pixel* dari setiap frame pada pergerakan objek tersebut. *Moving object detection* dilakukan dengan cara mengurangi nilai biner dari frame(i) dengan frame(i-(i-10)) yang didapat dari video. Pengurangan dilakukan dengan menggunakan absdiff dari opencv dikarenakan dengan menggunakan metode tersebut hasil yang didapatkan akan selalu bernilai *absolute*. Pengurangan dilakukan dengan diberi batasan setiap 30 frame agar dapat terlihat perbedaan pergerakan yang terjadi. Apabila jarak antara frame yang diproses terlalu dekat, pergerakan yang terjadi tidak terlalu terlihat.

B. *Rule Based Algorithm*

Rule Based algorithm terdiri dari dua bentuk klasifikasi warna yang terdiri dari RGB *color space* dan HSV *color space*. RGB *color space* mendeskripsikan warna menggunakan tiga komponen: merah (R), hijau (G), dan biru (B). RGB *color space* merupakan skema warna aditif yang menggunakan prinsip fungsi mata manusia yang berdasarkan sensitivitas tiga jenis kerucut di retina terhadap spektrum cahaya tertentu [9]. Dengan demikian warna yang terlihat dapat direproduksi dengan menambahkan berbagai intensitas lampu merah, hijau, dan biru. Konsep pencampuran RGB banyak digunakan dalam perangkat layar dan kamera yang membuat model warna ini penting

untuk sebagian besar grafik komputer. HSV *color space* mendefinisikan warna dalam tiga komponen: hue (H), saturation (S), dan value (V). value dari HSV color space memiliki persepsi yang mendekati seragam dan berkorelasi baik dengan sensasi warna manusia[10]. *Rule based algorithm* dilakukan untuk mendeteksi nilai *pixel* yang termasuk kedalam objek api. Dalam implementasinya digunakan 5 buah rule[11]:

$$\textbf{Rule 1}: \quad \textbf{if } \mathrm{I} > (\mathrm{p,q},1) > \mathrm{I}(\mathrm{p,q},2) > \mathrm{I}(\mathrm{p,q},3);\ R[1](p,q) = 1; \quad \textbf{else } \mathrm{R}[1](\mathrm{p,q}) = 0; \tag{1}$$

Dimana I(p,q) merepresentasikan sebagai *pixel value* dari setiap *channel* dalan RGB image, 1 untuk merah, 2 untuk hijau dan 3 untuk biru. R[1](p,q)=1 apabila kondisi terpenuhi dan R[1](p,q)=0 apabila kondisi tidak terpenuhi.

$$\textbf{Rule 2}: \quad \textbf{if } \mathrm{I}(\mathrm{p,q},1) > 140 \ \&\&\ \mathrm{I}(\mathrm{p,q},2) > 80 \ \&\&\ \mathrm{I}(\mathrm{p,q},3) < 150;\ R[2](p,q) = 1; \quad \textbf{else } \mathrm{R}[2](\mathrm{p,q}) = 0; \tag{2}$$

Dimana R[2](p,q) = 1 apabila kondisi terpenuhi dan R[2](p,q) = 0 apabila kondisi tidak terpenuhi. Berdasarkan kedua *rule* RGB *color space* tersebut dapat diambil hasil dengan fungsi:

$$\textbf{if } \mathrm{R}[2](p,q) = 1 \ \&\&\ R[1](p,q) = 1;\ R(p,q) = 1; \quad \textbf{else } 0; \tag{3}$$

Berdasarkan fungsi tersebut didapatkan sebuah hasil apabila *rule-1* adalah 1 dan *rule-2* adalah 1. Maka *pixel* R(p,q) diprediksi sebagai objek api. Berdasarkan pengamatan yang didapat apabila objek api hanya dideteksi dengan kedua *rule* tersebut, masih terdapat banyak objek yang memiliki range *pixel* yang mirip dengan objek api. untuk meningkatkan deteksi objek api sensungguhnya maka dilanjutkan dengan mengaplikasian HSV *color space* menggunakan fungsi:

$$\textbf{Rule 3}: \quad R[3](p,q) = 1; \quad \textbf{if } \mathrm{Im}(\mathrm{p,q},2) > ((255 - \mathrm{I}(\mathrm{p,q},1)) * \mathrm{avg_s} / \mathrm{avr_r}; \quad \textbf{else } \mathrm{R}[3](\mathrm{p,q}) = 0; \tag{4}$$

Dimana avg_s merepresentasikan sebagai nilai rata-rata dari *saturation* pada HSV, avr_r merepresentasikan sebagai rata-rata dari pixel merah yang didapatkan dari hasil RGB color space, Im(p,q,2) merepresentasikan nilai *pixel* dari S *plane* setelah hasil dari RGB *color space* yang telah dirubah format dari RGB menjadi HSV dan R[3](p,q)=1 apabila kondisi terpenuhi dan R[3](p,q)= 0 apabila kondisi tidak terpenuhi.

$$\textbf{Rule 4}: \quad R[4](p,q) = 1; \quad \textbf{if } \mathrm{Im}(\mathrm{p,q},2) < (255 * 0.65); \quad \textbf{else } \mathrm{R}[3](\mathrm{p,q}) = 0; \tag{5}$$

Dimana Im(p,q,2) merepresentasikan nilai *pixel* dari S *plane* pada HSV color space, R[4](p,q) = 1 apabila kondisi terpenuhi dan R[4](p,q) = 0 apabila kondisi tidak terpenuhi.

$$\textbf{Rule } 5: \quad \mathrm{R}[5](\mathrm{p,q}) = 1; \quad \textbf{if } \mathrm{Im}(\mathrm{p,q},3) > (255 * .97): \quad \textbf{else } \mathrm{R}[5](\mathrm{p,q}) = 0; \tag{6}$$

Dimana Im(p,q,3) merepresentasikan nilai *pixel* dari V *plane* pada HSV *color space*. R[5](p,q) = 1 apabila kondisi terpenuhi dan R[5](p,q) = 0 apabila kondisi tidak terpenuhi. Berdasarkan 3 rule HSV *color space*, dapat diambil hasil akhir dengan menggunakan :

$$\textbf{if } (\mathbf{R}[\mathbf{1}](\boldsymbol{p},\boldsymbol{q}) \ \&\&\ \boldsymbol{R}[\mathbf{2}](\boldsymbol{p},\boldsymbol{q}) \ \&\&\ \boldsymbol{R}[\mathbf{3}](\boldsymbol{p},\boldsymbol{q}) \&\& R[4](p,q) \ \&\&\ \boldsymbol{R}[\mathbf{5}](\boldsymbol{p},\boldsymbol{q}) = \mathbf{1} \quad \textbf{else } \mathrm{R}(p,q) = 0; \tag{7}$$

Dimana R[1](p,q) merepresentasikan *rule*-1, R[2](p,q) merepresentasikan *rule*-2, R[3](p,q) merepresentasikan *rule*-3, R[4](p,q) merepresentasikan *rule*-4, R[5](p,q) merepresentasikan *rule*-5 dan semua kondisi dari ke-5 *rule* terpenuhi.

C. Region Growing

Region growing merupakan sebuah metode yang didasarkan pada *homongeity* prinsip yang merujuk pada *pixel* dengan sifat-sifat yang sebanding, yang mengelompok secara kolektif membentuk suatu wilayah yang homogen [12]. *Region growing* dilakukan untuk melakukan rekonstruksi *pixel* dari hasil proses *rule based algorithm*, dikarenakan terlalu banyaknya nilai *pixel* yang tereliminasi. *Region growing* termasuk kedalam *region based image segmentation*. Hal pertama yang dilakukan pada metode *region growing* yaitu dengan menentukan *seed point* yang berupa kriteria intensitas *pixel*, *grayscale texture* atau warna. *Seed point* digunakan sebagai titik awal proses *region growing* dan sebagai acuan dari nilai *pixel* ketetanggaanya, apabila intensitas *pixel* ketetanggaan mirip dengan intensitas *pixel* dari seed point maka diklasifikasikan kedalam nilai titik seed point. Terdapat 2 macam proses region growing yaitu 4 dimensi dan 8 dimensi.

D. Spatial Analysis

Spatial analysis merupakan sebuah metode untuk mendeteksi karakteristik unik dari api yang selalu bergerak. Sehingga spatial karakteristik dari api juga akan berbeda

setiap dari pergerakan yang dihasilkan. Karakter unik yang dipakai pada penelitian ini berupa nilai energi yang didapatkan dari proses *spatial analysis* menggunakan *open source library* PyWavelets. Perhitungan energy dilakukan dengan membagi sebuah frame menjadi *grid* berukuran sebesar 32 x 32 pixel. Dari setiap *grid* yang didapat, kemudian dilakukan 4 buah perhitungan untuk mendapatkan nilai *approximation*, *horizontal detail*, *vertical detail* dan *diagonal detail*. Berdasarkan 4 perhitungan tersebut didapatkan 4 nilai *coefficient*: LL(*Low Low*), LH(*Low High*), HL(*High Low*) dan HH(*High High*). Fungsi perhitungan *coefficient* dapat dilihat pada fungsi 8,9,10,11 [13].

$$f^{LL(g-1)}(i,j) = \sum_{k_1=0}^{L-1}(\sum_{k_2=0}^{L-1} f^{LL(g)}(2i + k_1, 2j + k_2) * l_{k_2})l_{k_1} \quad (8)$$

$$f^{HL(g-1)}(i,j) = \sum_{k_1=0}^{L-1}(\sum_{k_2=0}^{L-1} f^{LL(g)}(2i + k_1, 2j + k_2) * h_{k_2})l_{k_1} \quad (9)$$

$$f^{LH(g-1)}(i,j) = \sum_{k_1=0}^{L-1}(\sum_{k_2=0}^{L-1} f^{LL(g)}(2i + k_1, 2j + k_2) * l_{k_2})h_{k_1} \quad (10)$$

$$f^{HH(g-1)}(i,j) = \sum_{k_1=0}^{L-1}(\sum_{k_2=0}^{L-1} f^{LL(g)}(2i + k_1, 2j + k_2) * h_{k_2})h_{k_1} \quad (11)$$

Dimana $f^{LL(g)}(i,j)$ merepresentasikan nilai *approximation* dari *wavelet coefficient,* g – merepresentasikan resolusi dari gambar, $f^{LL(g)}(i,j) = f(i,j)$ merepresentasikan nilai *coefficient* tertinggi pada gambar, l_{k_1}, l_{k_2}, h_{k_1}, h_{k_2} merepresentasikan *coefficient* dari lowpass dan highpass filter.

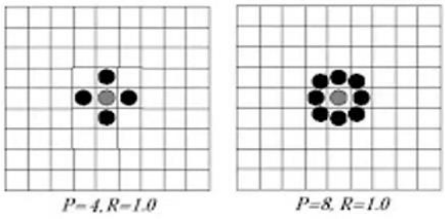
Berdasar 4 nilai coefficient LL,LH,HL,HH dapat dilakukan perhitungan nilai energi dengan menerapkan fungsi 12.

$$\mathrm{E(i,j)} = \mathrm{HH(i,j)}^2 + \mathrm{HL(i,j)}^2 + \mathrm{LH(i,j)}^2 \quad (12)$$

Dimana E(i,j) merepresentasikan nilai *energy* dari *pixel* i,j, HH(*High High*), HL(*Low High*) dan LH(*Low High*).

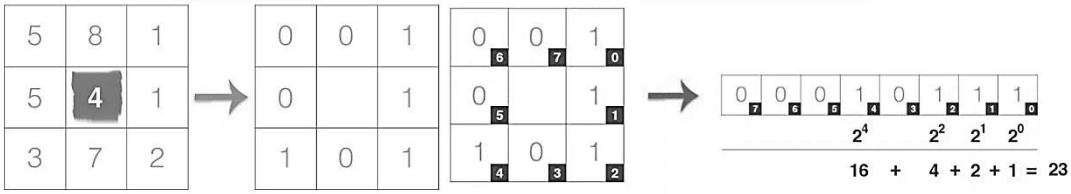
E. Texture Analysis

Texture analysis merupakan sebuah metode untuk meningkatkan keakuratan dari sistem pendeteksian objek. Metode yang akan digunakan dalam melakukan *texture analysis* yaitu dengan metode *Local Binary Pattern* (LBP). *Local Binary Pattern* adalah metode analisis tekstur yang menggunakan model statistika dan struktur[14]. Tujuan utama dari Local Binary Pattern yaitu untuk melakukan klasifikasi tekstur [15].

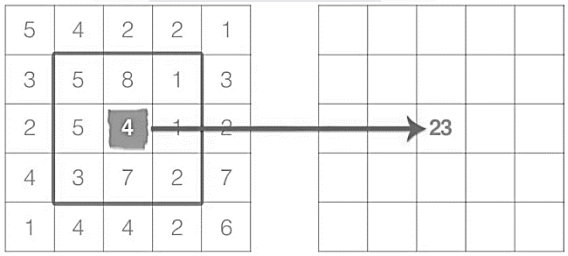


GAMBAR 1
RADIUS DAN POINT

Gambar 1: memperlihatkan nilai P sebagai point yang menentukan jumlah ketetanggaan dan R sebagai radius sebagai nilai jarak dari titik tengah. Titik tengah didapatkan dengan membandingkan intensitas sebuah nilai *pixel* dengan intensitas *pixel* ketetanggannya.



(a) (b)



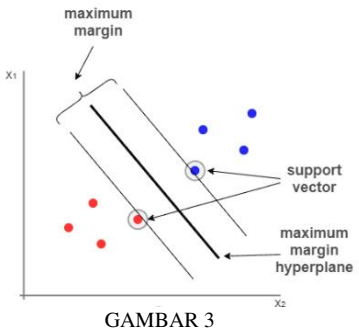
(c)

GAMBAR 2
PROSES LOCAL BINARY PATTERN

Gambar 2: (a) memperlihatkan sebuah blok dengan nilai P=8 dan R=1 atau blok dengan ukuran 3x3. Nilai 4 sebagai titik tengah atau dianggap sebagai *threshold* dan 8 pixel ke-tetanggaanya dianggap sebagai *pixel value* yang dioperasikan oleh nilai *threshold*/titik tengah. Pada proses ini dilakukan perbandingan *pixel value* ke-tetanggaan dengan nilai *threshold*/titik tengah, apabila pixel value ke-tetanggaan lebih besar dari nilai *threshold*/titik tengah maka diset sebagai 0 dan apabila pixel value ke-tetanggaan lebih kecil dari nilai *threshold*/titik tengah maka diset sebagai Gambar 2: (b) memperlihatkan hasil perbandingan nilai pixel value yang dimulai dari ujung kanan atas dan berurutan searah jarum jam. Berdasarkan urutan tersebut didapatkan sebuah nilai biner. Gambar 2: (c) memperlihatkan hasil nilai biner yang didapat kemudian diassign ke dalam blok gambar original dengan ukuran yang sama yaitu blok dengan ukuran 3x3.

F. Classification

Metode *classification* yang digunakan yaitu dengan menggunakan *Support Vector Machine* (SVM).*Support Vector Machine* (SVM) merupakan sebuah proses untuk melakukan klasifikasi dengan cara menemukan *hyperplane* yang akan memaksimalkan *margin* antara dua kelas. *Vektor* (kasus) yang menentukan *hyperplane* adalah *support vector*[16]. *Hyperplane* merupakan sebuah fungsi yang digunakan sebagai batas dalam proses klasifikasi data. Data pada *Support Vector Machine* berbentuk titik data yang memiliki dua kelas nilai yaitu positif dan negatif. *Support vector* adalah titik data yang memiliki posisi paling dekat dengan batas *hyperplane* yang mempengaruhi letak posisi dan orientasi dari *hyperplane*. Terdapat sebuah area yang berada di antara *hyperplane* dan juga *support vector* yang disebut dengan *margin*. Tujuan utama dari *Support Vector Machine* (SVM) adalah untuk mencari nilai *margin* terbesar dari *support vector* dan *hyperplane* berdasarkan letak posisi titik data.

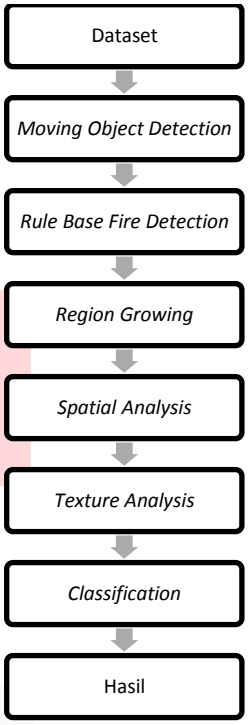


GAMBAR 3
SUPPORT VECTOR MACHINE

III. METODE

A. Perancangan Sistem

Perancangan sistem yang dibangun dapat dilihat pada gambar 1.



GAMBAR 4
PERANCANGAN SISTEM

B. Pengumpulan Dataset

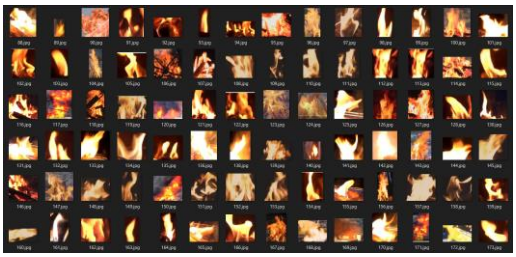
1. Pengumpulan Dataset Video

   Pengumpulan dataset dilakukan dengan cara mengumpulkan video yang terdapat objek api, objek menyerupai api dan objek yang bukan api.

GAMBAR 5
DATASET

2. Pengumpulan Dataset Image

   Pengumpulan dataset dilakukan dengan cara mengumpulkan berbagai image yang terdapat objek api dan objek bukan api yang akan digunakan pada proses *training* data *Local Binary Pattern*.



GAMBAR 6
DATA TRAIN FIRE

GAMBAR 7
DATA TRAIN NONFIRE

C. *Moving Object Detection*

Moving object detection merupakan proses untuk mendeteksi objek bergerak pada suatu video. Untuk mendapatkan objek bergerak pada video dilakukan:

1. Melakukan ekstraksi keseluruhan frame dari dataset video.

GAMBAR 8
FRAME

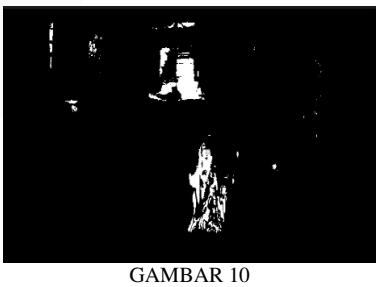
2. Frame yang didapatkan kemudian dicari perbedaan antara *frame* satu dengan lainya menggunakan absdiff dari opencv. Perbedaan antar *frame* tidak terlalu terlihat apabila dilakukan perbandingan frame secara berurutan. Sehingga *frame* yang dipakai untuk mencari perbedaan pergerakan digunakan: (i-(i-10)) dimana i merepresentasikan urutan dari frame yang dikurangi dengan nilai i-10, sehingga i pertama berapa pada nilai 11 agar hasil rumusan tersebut memiliki perbedaan i *frame* yang akan diproses.

(a) (b) (c)

GAMBAR 9
PROSES FRAME DIFFERENT

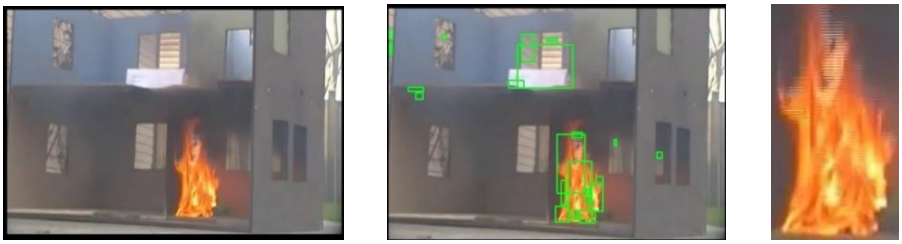
Gambar 9: (a) menunjukkan *frame*(i), (b) menunjukkan *frame*(i(i-10)), (c) menunjukkan hasil dari proses *frame different*.

3. *Frame subraction* dilakukan dengan memberi jarak i+30 agar perbedaan *frame* lebih terlihat.
4. *Frame different* yang didapatkan kemudian dilakukan *otsu's thresholding*.



GAMBAR 10
OTSU THRESHOLDING

5. Hasil dari *thresholding* ditandai dengan *bounding box* dan objek didalamnya menjadi *region of interest.*



(a) (b) (c)

GAMBAR 11
BOUNDING BOX

Gambar 11: (a) memperlihatkan frame video, (b) hasil *bounding box*, (c) hasil *region of interest*.

D. *Rule Base Algorithm*

Rule base algorithm dilakukan 2 proses *filtering pixel value* menggunakan RGB *color space* dan HSV *color space*. RGB *color space* diaplikasikan dengan *Rule*-1 dan *Rule*-2, sehingga didapatkan hasil pada gambar 12.

(a) (b)

(c)

(d)

(e)

(f)

(g)

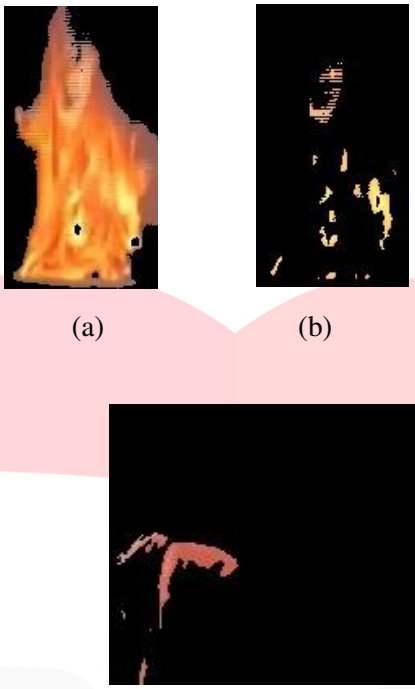
(h)

GAMBAR 12
RGB COLOR SPACE

Gambar 12:(a) (c) memperlihatkan hasil *region of interest* dari video 1, (b)(d) memperlihatkan hasil dari pengaplikasian *rule*-1 dan *rule*-2. Gambar 12:(e) memperlihatkan objek dengan *pixel value* mirip api,

Gambar 12:(g) memperlihatkan objek bukan api (f) (h) memperlihatkan hasil pengaplikasian *rule*-1 dan *rule*-2..

Berdasarkan *rule*-1 dan *rule*-2 dapat diamati apabila masih terdapat banyak objek yang memiliki *range pixel* yang mirip dengan api. Sehingga dilanjutkan dengan pengaplikasian HSV *color space* dengan menggunakan *rule*-3, *rule*-4 dan *rule*-5.



(a) (b)

(c) (d)

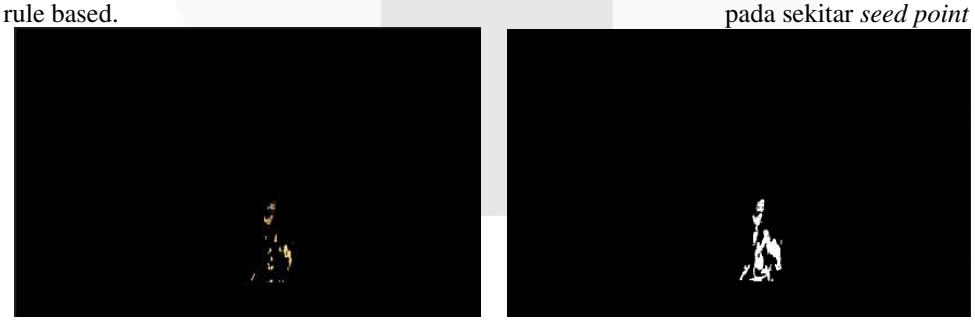
GAMBAR 13
RGB COLOR SPACE KE HSV COLOR SPACE

Gambar 13:(a) (c) memperlihatkan hasil dari metode RGB *color space*, (b) (d) memperlihatkan hasil setelah dilakukan HSV *color space*

E. Region Growing

Region growing dilakukan untuk menindak lanjuti hasil yang didapatkan dari pengaplikasian metode RGB dan HSV *color space*. Terdapat banyak *pixel* area api yang tereliminasi karena ke 5 rule pada metode rule based.

1. Hasil dari metode *rule based* digunakan sebagai *seed point* untuk melakukan *region growing*.
2. *Region growing* yang dilakukan menggunakan metode *region growing* 8-dimensi.
3. Melakukan pemrosesan *region growing* dengan melakukan perhitungan nilai ketetanggaan berdasarkan intensitas *pixel* pada sekitar *seed point* yang digunakan.



(a) (b)

GAMBAR 14
REGION GROWING

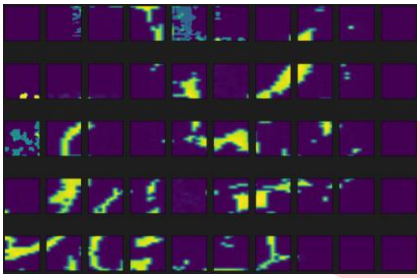
Gambar 14:(a) memperlihatkan hasil metode *rule based* yang digunakan sebagai *seed point*, (b) memperlihatkan hasil dari *region growing*.

F. Spatial Analysis

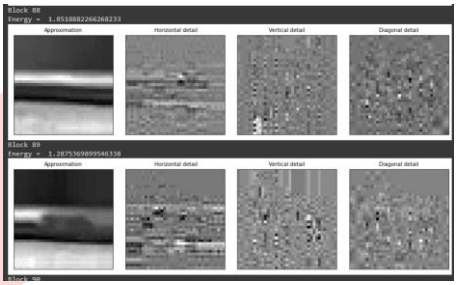
Metode *spatial analysis* dilakukan secara dua kali yang pertama yaitu menggunakan frame video biasa dan langsung dilakukan proses wavelet transform dan cara ke dua dengan menggunakan

frame hasil dari metode *rule based* dan *region growing*. Proses yang dilakukan yaitu:

1. Membagi *frame* hasil dari *rule based* dan *region growing* menjadi grid dengan berukuran 32 x 32 pixel.
2. Mengaplikasikan *wavelet transform* kepada setiap *grid* yang didapatkan.
3. Menghitung nilai energy per-grid berdasarkan nilai coefficient dari wavelet transform.
4. Penghitungan nilai energi menggunakan perhitungan $E(i,j) = HH(i,j)^2 + HL(i,j)^2 + LH(i,j)^2$ dimana E(i,j) merepresentasikan sebagai nilai energy, HH,HL,LH merepresentasikan *High-High, High-Low, Low-High* secara berurutan.
5. Membentuk histogram berdasarkan nilai energi dari setiap grid, pada x berisikan nomor dari grid dan y berisikan nilai energy dari grid tersebut.
6. Membentuk histogram keseluruhan grid yang didapat dari keseluruhan frame yang dari video yang dilakukan pemrosesan.
7. Meng-outputkan grid frame yang memiliki nilai energi lebih besar dari 500.



(a)

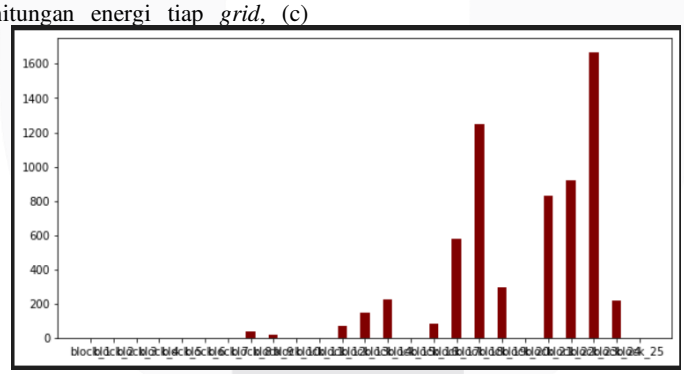


(b)

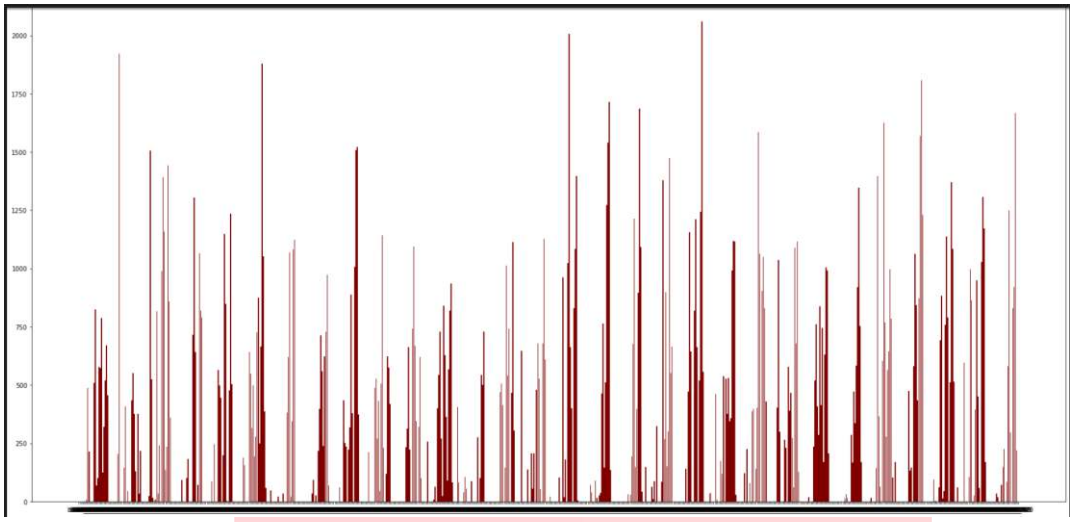
(c)

GAMBAR 15
HASIL WAVELET DARI CITRA REGION GROWING

Gambar 15:(a) memperlihatkan hasil dari proses *rule based* yang dilakukan pembagian *grid* berukuran 32x32 pixel, (b) memperlihatkan hasil dari perhitungan energi tiap *grid*, (c) memperlihatkan hasil dari *grid* yang memiliki nilai energi lebih dari 500.



GAMBAR 16
NILAI ENERGI DARI SEBUAH OBJEK



GAMBAR
17 NILAI ENERGY DARI KESELURUHAN OBJEK PADA VIDEO

G. *Local Binary Pattern*

Perancangan *Local Binary Pattern* dilakukan dengan memproses *local data images* yang telah dikumpulkan dan dianggaps sebagai objek api.

1. Data *testing* yang digunakan berasal dari proses perhitungan nilai energi yang didapatkan dari proses *spatial analysis* yang memenuhi kondisi.
2. Ekstraksi *Local Binary Pattern* menggunakan parameter ketetanggaan 4,8 dan radius 3,4,5 untuk mencari nilai hasil ekstraksi fitur terbaik.
3. Hasil yang didapat setelah melakukan proses perbandingan nilai tengah *pixel* dengan nilai ketetanggaan *pixel* akan menghasilkan vektor ciri yang digunakan untuk membedakan vector ciri objek api.

```
[[0.05093967 0.04037977 0.02676649 ... 0.04054036 0.08025499 0.5153635 ]
 [0.05470633 0.03916921 0.01656223 ... 0.04049563 0.10071943 0.55705568]
 [0.03047009 0.02248935 0.01671775 ... 0.02183258 0.04571508 0.34422641]
 ...
 [0.04887778 0.03364444 0.01795556 ... 0.03234444 0.05282222 0.46918889]
 [0.03471173 0.01837249 0.01672391 ... 0.02004854 0.09719284 0.36357558]
 [0.02607778 0.02064444 0.01247778 ... 0.02262222 0.03668889 0.33901111]]
```

GAMBAR 18
VECTOR CIRI

H. *Support Vector Machine*

Support Vector Machine digunakan untuk melakukan klasifikasi data untuk data training dan sebagai dasar untuk melakukan testing berdasarkan hasil training yang didapat.

1. *Support Vector Machine* menggunakan kernel linier sebagai metode klasifikasi.
2. Mencari nilai *hyperplane* yang digunakan untuk memaksimalkan *margin* antara kelas data api dan kelas data non api.
3. Data training yang digunakan berupa 500 data api dan 500 data non-api.
4. Masing-masing citra yang didapat, digunakan sebagai inputan dan dilakukan prediksi untuk mencari objek api sesungguhnya, dengan menggunakan nilai 1 sebagai objek api dan nilai 0 sebagai objek bukan api.

IV. HASIL DAN PEMBAHASAN

A. Skenario Pengujian

Skenario pengujian dilakukan untuk menguji dan menganalisis keakuratan dari sistem pendeteksian api menggunakan setiap metode sebagai berikut:

1. Melakukan percobaan terhadap pengaruh nilai ketetanggaan dan radius pada metode Local Binary Pattern dengan menggunakan nilai ketetanggaan 8,16 dan radius 3,4,5.
2. Melakukan perhitungan akurasi, *precision*, *recall*,dan *f1-score* pada setiap citra inputan.

B. Evaluasi Hasil Pengujian

Pengujian dilakukan dengan cara mencari nilai akurasi, *precision*, *recall*,dan *f1-score* dari setiap citra yang dilakukan pengujian. Dalam melakukan perhitungan akurasi digunakan fungsi 13.

$$\frac{Jumlah\ hasil\ data\ yang\ diprediksi}{Jumlah\ data\ berhasil\ diprediksi + Data\ salah\ predisi} \quad (13)$$

Dalam menentukan nilai *precision* digunakan fungsi 14.

$$\frac{tp}{(tp+fp)} * 100 \quad (14)$$

Dalam menentukan nilai *recall* digunakan fungsi 15.

$$\frac{tp}{(tp+fn)} * 100 \quad (15)$$

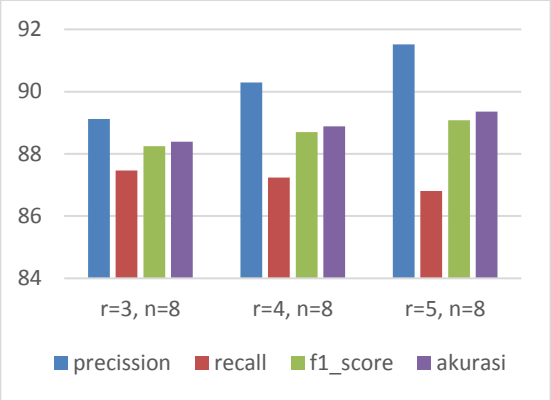
Dalam menentukan nilai *f1-score* digunakan fungsi 16.

$$\frac{(2*Presisi*Recall)}{Presisi+Recall} \quad (16)$$

Nilai tp, tn, fp dan fn merupakan true *positive, true negative, false positive* dan *false negative*.

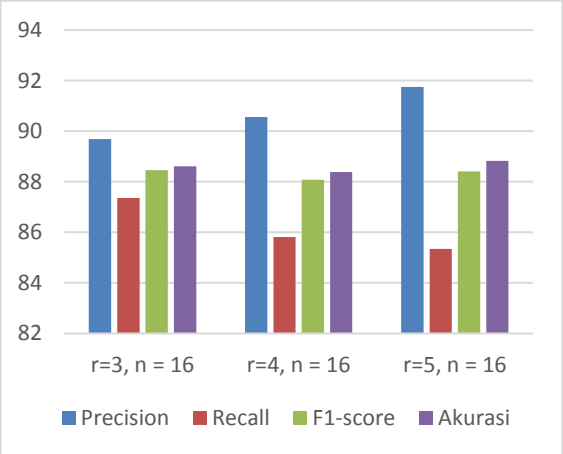
Berikut merupakan hasil dari pengujian yang didapatkan :

1. Pengaruh Nilai Radius dan Numpoint



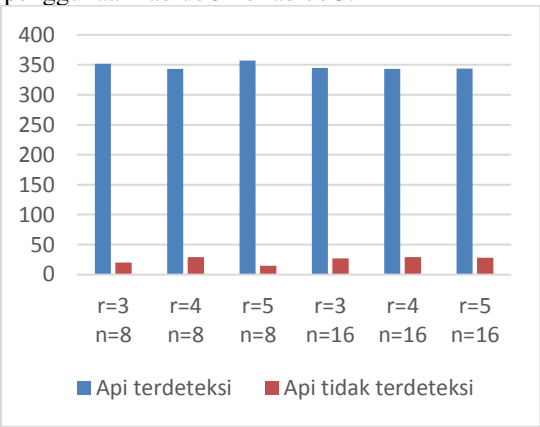
GAMBAR 19
PENGGUNAAN NUM-POINT 8

Pada akurasi yang didapatkan terjadi peningkatan 0.494623660000002 sampai 0.9731182799999942. Pada *precision* terjadi peningkatan 1.1783366299999898 sampai 2.4053027799999995. Pada *recall* terjadi penurunan 0.23003715000000113 sampai 0.6545478200000048. Pada *f1_score* terjadi peningkatan 0.37825879000000384 sampai 0.8317203699999993. Sehingga dapat disimpulkan bahwa hasil terbaik didapatkan pada penggunaan nilai radius= 5 dan num-point=8.



GAMBAR 20
PENGGUNAAN NUM-POINT 16.

Hasil dari gambar diatas menunjukkan nilai akurasi, *precision*, *recall* dan *f1-score* terpengaruh oleh semakin besarnya nilai radius. Pada nilai *precision* terjadi peningkatan sebesar 0.8804032599999942 sampai dengan 2.0623566900000014. Pada akurasi terjadi penurunan sebesar 0.22526881000000287 pada penggunaan radius 3 dengan radius 4, terjadi peningkatan sebesar 0.4467741899999993 pada penggunaan radius 4 dengan radius 5 dan terjadi peningkatan sebesar 0.22150537999999642 pada penggunaan radius 3 dengan radius 5. Pada nilai *recall* terjadi penurunan sebesar 0.4619947300000007 sampai dengan 2.0146067600000066. Pada nilai f1-score terjadi penurunan sebesar 0.37969512000000805 dari penggunaan nilai radius 3 ke radius 4, terjadi peningkatan sebesar 0.319016579999996 dari penggunaan radius 4 ke radius 5 dan terjadi penurunan sebesar 0.06067854000001205 dari penggunaan radius 3 ke radius 5.



GAMBAR 21
DETEKSI API

Berdasarkan hasil yang didapat, dapat disimpulkan bahwa nilai radius dan num-point mempengaruhi hasil dari nilai akurasi dikarenakan perbedaan nilai radius dan numpoint mempengaruhi nilai *feature vector* yang didapatkan. Radius mempengaruhi jarak antar nilai ketetanggaan ke titik tengah sehingga dapat menghasilkan nilai yang berbeda pada tiap perhitungan. Sedangkan num-point berpengaruh terhadap jumlah nilai ketetanggaan. Dari keseluruhan data yang didapatkan, bisa disimpulkan bahwa nilai akurasi terbaik terdapat pada penggunaan nilai radius 5 dan nilai num-point 8 dengan nilai akurasi tertinggi sebesar 89,35967741935485 dan deteksi objek api terbanyak dengan 357 terdeteksi dan 15 tidak terdeteksi.

2. Analisa Data Bermasalah

Proses deteksi objek api mengalami kekurangan apabila *grid* yang dihasilkan dari proses *spatial analysis* memiliki nilai energi diatas 500 namun objek didalam *grid* terlalu kecil atau terlalu besar, sehingga tidak dikenali sebagai objek api. Berdasarkan hasil *spatial analysis* didapatkan citra sebanyak 372 yang diproses dengan *Local Binary Pattern* dan diklasifikasikan dengan *Support Vector Machine*. Dari 372 citra didapatkan hasil deteksi objek api terbanyak pada penggunaan radius 5 dan numpoint 8 dengan 357 terdeteksi dan 15 tidak terdeteksi. .

3. Hasil Deteksi Api

Deteksi api diklasifikasikan menggunakan metode *Support Vector Machine,* dengan memberikan tanda pada citra menggunakan 1 apabila dianggap sebagai objek api dan 0 apabila dianggap sebagai objek bukan api.

GAMBAR 22
HASIL PREDIKSI

4. Kesalahan deteksi

Pada proses kesalahan deteksi terdapat beberapa citra api setelah dilakukan proses *region growing* dan dibagi menjadi *grid* berukuran 32x32 *pixel*. Menghasilkan sebuah citra yang memiliki objek api terlalu besar ataupun terlalu kecil sehingga tidak dikenali sebagai objek api. *Grid* yang tidak dikenali sebagai objek api dapat dilihat pada gambar 23.

GAMBAR 23
FALSE DETECTION

V. KESIMPULAN

A. Kesimpulan

Setelah dilakukan beberapa skenario pengujian maka dapat disimpulkan bahwa:

1. Penggunaan metode *rule based, region growing, spatial analysis, Local Binary Pattern* dan *Support Vector Machine* dapat mendeteksi api dengan baik.
2. Grid yang memiliki objek api terlalu besar atau terlalu kecil tidak dapat dikenali dengan baik.
3. Parameter radius 3,4,5 dan num-point 8,16 pada *Local Binary Pattern* dapat melakukan pengenalan objek api dengan baik, hasil terbaik didapatkan pada radius 5 dan num-point 8.
4. Parameter radius dan numpoint pada *Local Binary Pattern* sangat mempengaruhi hasil ekstraksi fitur, *precision, recall, f1 score* dan akurasi yang didapat.

B. Saran

Untuk mengembangkan sistem di masa mendatang, saran yang dapat peneliti berikan adalah:

1. Menambahkan metode lain agar dapat memperbaiki nilai akurasi.
2. Mencoba metode baru untuk dapat menanggulangi objek api yang tidak terdeteksi api dikarenakan terlalu besar atau terlalu kecilnya objek api pada citra yang diuji.
3. Mencoba mengubah parameter pada Local Binary Pattern.
4. Menambahkan atau merubah data training.
5. Mencari atau menciptakan data uji yang lebih baik untuk dilakukan klasifikasi api.

REFERENCE

[1] Hu, G. L., & Jiang, X. Early Fire Detection of Large Space Combining Thresholding with Edge Detection Techniques. Applied Mechanics and Materials, 44-47, 2060–2064, (2010).

[2] Norsyahirah Izzati binti Zaidi, Nor Anis Aneza binti Lokman, Mohd Razali bin Daud, Hendriyawan Achmad and Khor Ai Chia. FIRE RECOGNITION USING RGB AND YCBCR COLOR SPACE VOL. 10, NO. 21, ISSN 1819-6608, (2015).

[3] Jiao, Z., Zhang, Y., Xin, J., Yi, Y., Liu, D., & Liu, H. Forest Fire Detection with Color Features and Wavelet Analysis Based on Aerial Imagery. Chinese Automation Congress (CAC), (2018).

[4] Olivares-Mercado, J., Toscano-Medina, K., Sánchez-Perez, G., Hernandez-Suarez, A., Perez-Meana, H., Sandoval Orozco, A. L., & García Villalba, L. J. Early Fire Detection on Video Using LBP and Spread Ascending of Smoke. Sustainability, 11(12), 3261, (2019).

[5] Duong, H. D., & Tinh, D. T. An efficient method for vision-based fire detection using SVM classification. 2013 International Conference on Soft Computing and Pattern Recognition (SoCPaR), (2013).

[6] Hu, W., Tan, T., Wang, L., & Maybank, S. A Survey on Visual Surveillance of Object Motion and Behaviors. IEEE Transactions on Systems, Man and Cybernetics,

Part C (Applications and Reviews), 34(3), 334–352, (2004).

[7] Bahadir Karasulu and Serdar Korukoglu. Performance Evaluation Software: Moving Object Detection and Tracking in Videos. [Online]. Available: http://link.springer.com/book/10.1007%2F978-1- 4614-6534-8 , (2013).

[8] Dipali Shahare and Ranjana Shende, "Moving Object Detection with Fixed Camera and Moving Camera for Automated Video Analysis," International Journal of Computer Applications Technology and Research, vol. 3, issue 5, pp. 277-283, (2014).

[9] Poynton, C. A. A Guided Tour of Colour Space. New Foundation for Video Technology: The SMPTE Advanced Television and Electronic Imaging Conference, (1995).

[10] Paschos, G. Perceptually uniform color spaces for color texture analysis: an empirical evaluation. IEEE Transactions on Image Processing, 10(6), 932–937, (2001).

[11] Gupta, A., Bokde, N., Marathe, D., … Kishore. A Novel approach for Video based Fire Detection system using Spatial and Texture analysis. Indian Journal of Science and Technology, 11(19), 1–17, (2018).

[12] Nyma, A., Kang, M., Kwon, Y.-K., Kim, C.-H., & Kim, J.-M. A Hybrid Technique for Medical Image Segmentation. Journal of Biomedicine and Biotechnology, 2012, 1–7, (2012).

[13] R. Bogush, S. Maltsev, A. Aniskovich, Object Detection Using Wavelet Transform, 2005.

[14] Kurniawardhani, A., Suciati, N., & Arieshanti, I. Klasifikasi Citra Batik Menggunakan Metode Ekstraksi Ciri Yang Invariant Terhadap Rotasi. JUTI: Jurnal Ilmiah Teknologi Informasi, 12(2), 48-60, (2014).

[15] T. Ojala, M. Pietikainen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," in Pattern Recognition, vol. 29, no. 1, pp. 51 - 59, (1996).

[16] Bahal, Bhiwani, Haryana, "Data Classification Using Support VectorMachine," Journal of Theoretical and Applied Information Technology, (2009).