Sistem Deteksi Api Berbasis Visual menggunakan Metode Local Binary Patterns-Three Orthogonal Planes dan Grey-level Co-occurrence Matrix

Vision-based Fire Detection System using Local Binary Patterns-Three Orthogonal Planes and Grey-level Co-occurrence Matrix

[1]Gugy Lucky Khamdani, [2]Tjokorda Agung Budi Wirayuda ST., MT., [3]Febryanti Sthevanie ST., MT.

[123]Program Studi Sarjana Teknik Informatika, Fakultas Informatika, Universitas Telkom

[1]gugylucky@gmail.com, [2]cokagung@telkomuniversity.ac.id, [3]febryantisthevanie@gmail.com

Abstrak

Alat pendeteksi api tradisional yang sudah terpasang di kebanyakan gedung saat ini biasanya berbasis sensor, seperti sensor inframerah, sensor asap, dan lain-lain. Akan tetapi deteksi berbasis sensor tersebut penempatannya harus pas dan tidak cocok jika digunakan di luar ruangan atau di tempat terbuka. Dengan adanya kamera CCTV, sangat membantu dalam mendeteksi api secara efisien dalam waktu yang singkat jika kamera tersebut dilengkapi dengan perangkat lunak khusus.

Pada tugas akhir ini dirancang dan diimplementasikan sistem pendeteksi api berbasis visual. Tahapan pada sistem ini diantaranya adalah Three-frame differencing untuk mendeteksi gerakan pada video, fire color image segmentation, ekstraksi ciri menggunakan Grey-level Co-occurrence Matrix (GLCM) dan Local Binary Patterns-Three Orthogonal Planes (LBP-TOP), dan pengklasifikasian menggunakan k-Nearest Neighbors (kNN). Keluaran dari tugas akhir ini adalah sistem dapat mendeteksi dan menunjukkan lokasi objek api pada gambar video.

Kata Kunci: ***fire detection, three-frame differencing*****, LBP-TOP, GLCM.**

Abstract

Traditional fire detectors that are already been installed in most of the buildings these days are usually the sensor-based ones, infrared-based sensor, smoke-based sensor, etc. But these sensor-based detectors need to be placed in the right position and not suitable for outdoor environtment. With a well-positioned CCTV camera, it can be a lot helpful to detect wildfire earlier if equipped with the right software.

This final project explains the design and implementation of a visual-based early fire detection system. Stages in this system includes moving pixel detection using Three-frame differencing, feature extraction using Grey-level Co-occurrence Matrix (GLCM) and Local Binary Patterns-Three Orthogonal Planes, and classification using k-Nearest Neighbors (kNN).The output of this final project is a system that can detect and show the location of the fire in a video image.

Keywords: fire detection, three-frame differencing, LBP-TOP, GLCM.

1. Pendahuluan

Kebakaran merupakan bencana yang sering terjadi. Kebakaran sering terjadi disaat musim panas, terutama di tahun-tahun kemarau [4]. Kebakaran jenis ini bisa ditanggulangi jika bisa dideteksi lebih dini. Akan tetapi, alat pendeteksi api tradisional yang terpasang di kebanyakan tempat saat ini biasanya berbasis sensor, seperti sensor inframerah, sensor asap, dan lain-lain, yang penempatannya harus tepat dan tidak cocok jika digunakan di luar ruangan atau di tempat terbuka.

Dengan semakin banyaknya penggunaan CCTV video systems untuk tujuan keamanan dalam memonitor lingkungan industri, lingkungan publik dan umum, semakin banyak pula yang mempertimbangkan sistem tersebut sebagai sumber sekunder dalam pendeteksian api secara dini jika kamera tersebut dilengkapi dengan perangkat lunak khusus [2].

Dalam beberapa tahun terakhir ada banyak metode yang diusulkan oleh para peneliti [1] [2] [5] [6]. Salah satu penelitian melakukan pendekatan non-temporal dengan menggunakan Gray-level Co-occurence Matrix (GLCM) untuk mendeskripsikan tekstur api, dan hasilnya sangat efektif [2]. Lalu penelitian selanjutnya meningkatkan performanya dengan melakukan perhitungan terlebih dahulu dengan menggunakan operator Local Binary Patterns (LBP) sebelum dilanjutkan dengan GLCM [1]. Pada penelitian tersebut juga disebutkan penggunaan metode Optical Flow, akan tetapi metode tersebut sangat sensitif terhadap noise dan komputasinya sangat tinggi. Penelitian berikutnya menggunakan Hidden Markov Model dan wavelet transform untuk mendeteksi flickering pixel¸ akan tetapi metode tersebut akan memberikan alarm palsu jika terdapat objek yang pergerakan flickernya menyerupai pergerakan api [5]. Lalu paper lainnya menggunakan metode Volume Local Binary Patterns (VLBP) yang merupakan ekstensi dari LBP, menggabungkan tampilan dan gerakan, sehingga bisa digunakan untuk analisis

spatio-temporal [6]. Akan tetapi, VLBP memiliki komputasi yang kompleks dan susah untuk diextend, sehingga dikembangkan metode Local Binary Patterns-Three Orthogonal Planes untuk mengatasi masalah tersebut [7].

Metode yang digunakan pada tugas akhir ini adalah Grey Level Co-occurrence Matrix (GLCM) yang sudah dilakukan perhitungan menggunakan Local Binary Patterns-Three Orthogonal Planes (LBP-TOP) terlebih dahulu untuk mendapatkan deskripsi tekstur. Klasifikasi menggunakan k-Nearest Neighbors. Studi kasus dalam pembuatan tugas akhir ini yaitu pendeteksian api di dalam ruangan dan luar ruangan.

2. Perancangan dan Implementasi

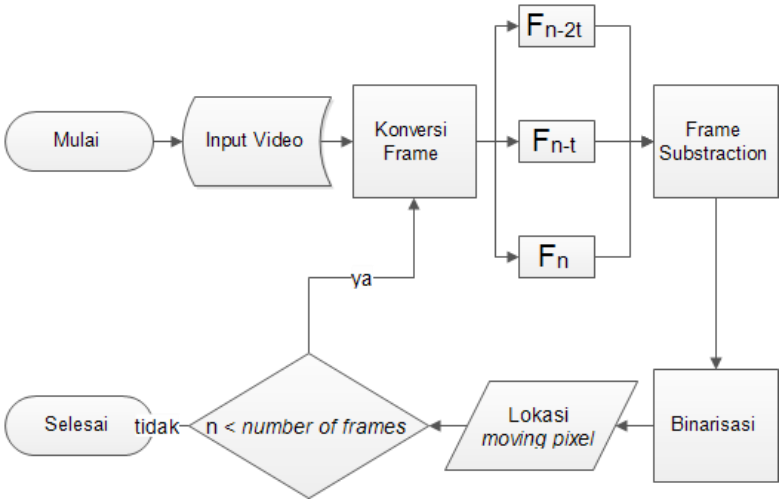
Secara umum, sistem deteksi api terbagi menjadi empat bagian diantaranya adalah *moving pixel detection* menggunakan *Three-frame differencing*, *fire color detection* menggunakan HSI *flame color rules*, ekstraksi ciri menggunakan LBP-TOP dan GLCM, dan klasifikasi menggunakan k-*Nearest Neighbors*. Blok diagram sistem deteksi api dapat dilihat pada Gambar 1.

Gambar 1 Blok diagram sistem

Berikut ini akan dijelaskan empat bagian dari sistem ini.

2.1. Three-frame differencing

Untuk tahapan pertama kali dilakukan deteksi *moving pixel* untuk menemukan pixel-pixel mana saja yang bergerak dan memisahkannya dari pixel-pixel yang tidak bergerak. Metode yang digunakan adalah *three-frame differencing*, dengan alur prosesnya seperti pada gambar Gambar 2.



Gambar 2 Flow diagram dari three-frame differencing

2.2. HSI Flame Color Rules

Pada tahap ini dideteksi pixel yang memiliki warna yang menyerupai api. Dari lokasi *moving pixel* yang didapatkan dari tahap sebelumnya, lokasi tersebut akan difilter lagi, pixel mana sajakah yang terdapat pada lokasi tersebut yang memiliki warna yang menyerupai api. Karena video yang digunakan menggunakan RGB *color model*, maka proses pertama yang dilakukan adalah mengkonversi frame menjadi *HSI color model*. Untuk mengkonversi dari RGB ke HSI didefinisikan pada persamaan berikut:

$$I = \frac{R+G+B}{3}$$

$$H = \begin{cases} \theta, & if\ B \le G \\ 360° - \theta, & if\ B > G \end{cases}, \quad with\ \theta = cos^{-1}\left\{\frac{\frac{1}{2}[(R-G)+(R-B)]}{\left[(R-G)^2+(R-B)(G-B)^{1/2}\right]}\right\}$$

$$S = 1 - \frac{3}{R+G+B}\min(R,G,B)$$

(1) [3]

Setelah dikonversi, frame lalu dicek satu persatu pixelnya, berdasarkan lokasi yang didapatkan dari tahap sebelumnya. Hasil eksperimen [15] menunjukkan setiap pixel dari objek api harus memenuhi kondisi-kondisi berikut :

(i) $0 \leq H \leq 60°$

(ii) $0 \leq S \leq 0.2$

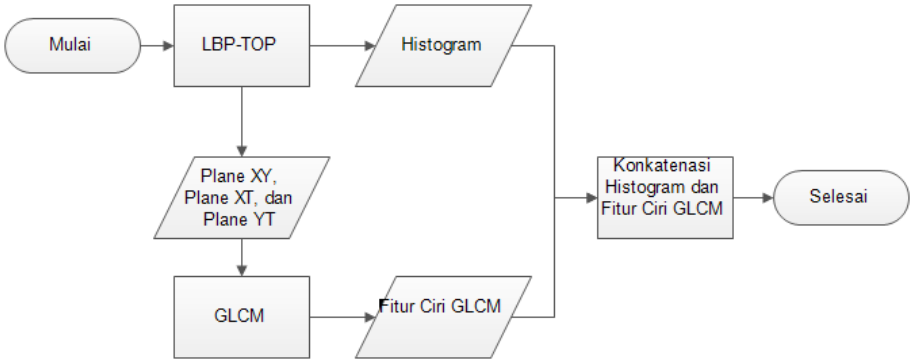
(iii) $127 \leq I \leq 255$

(2)

Pixel yang memenuhi kondisi diatas maka akan disebut sebagai *Candidate Flame Regions* (CFR). [15]

2.3. Ekstraksi Ciri

Setelah CFR didapatkan dari tahap sebelumnya, diperlukan identifikasi dan klasifikasi apakah objek yang dikenali tersebut merupakan objek api atau bukan. Proses ekstraksi ciri pada sistem dapat dilihat pada gambar:



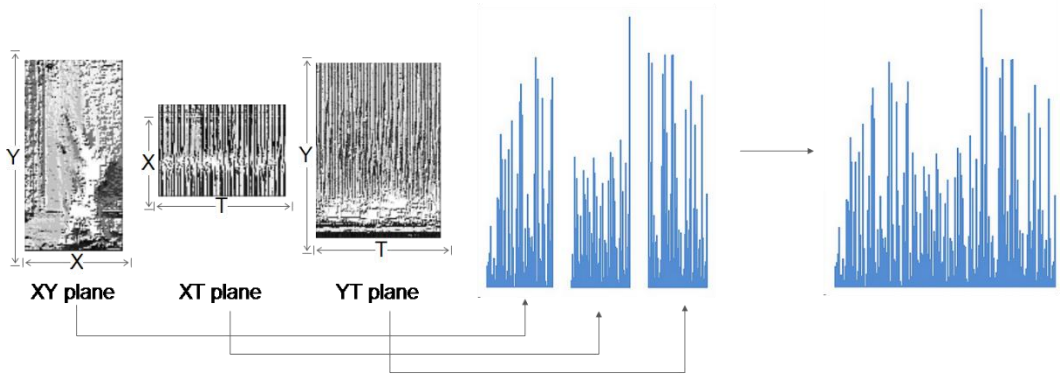
Gambar 3 Flow diagram dari ekstraksi ciri

a. LBP-TOP

Dari CFR yang didapat selanjutnya akan melalui proses ekstraksi ciri yang diawali dengan LBP-TOP. LBP-TOP memerlukan masukan berupa data volume. Oleh sebab itu, maka diambil tumpukan frame mulai dari *previous frame* sampai *next frame* (yang terdapat pada tahap *Three-frame difference* diatas) dan dibuat menjadi *volume data*.

Dari *volume data* yang didapatkan, berfokus pada area CFRnya, diambil 3 *plane*, yaitu *plane* XY, *plane* XT, dan *plane* YT. *Plane* XY merupakan data 2D yang diambil dari sumbu x dan sumbu y pada nilai tengah sumbu t yang merepresentasikan informasi spatial, sedangkan XT dan YT, yang masing masing merupakan data 2D yang diambil dari sumbu x-t dengan nilai tengah sumbu y dan sumbu y-t dengan nilai tengah sumbu x, yang merepresentasikan informasi temporal pada baris (untuk XT) dan kolomnya (untuk YT). Masing-masing plane dikonversikan menjadi *grayscale* karena LBP menerima input data berupa *grayscale*.

Selanjutnya *plane-plane* tersebut dilakukan operasi LBP sehingga didapatkan hasil seperti gambar berikut:

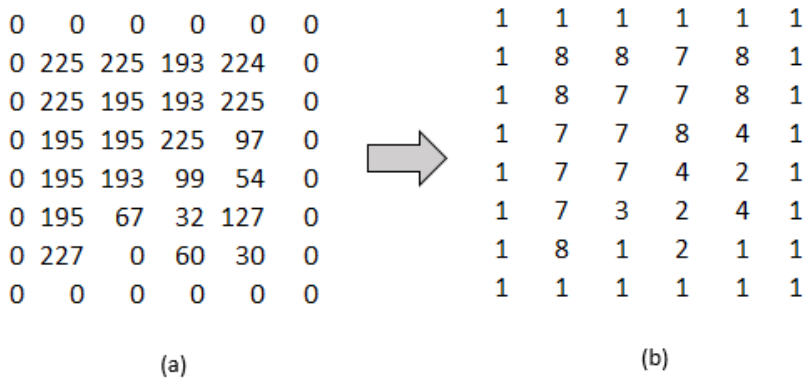


Gambar 4 (a) plane XY, (b) plane XT, dan (c) plane YT setelah operasi LBP.

Ketiga gambar tersebut lalu akan dilanjutkan ke tahap GLCM. Dari ketiga *plane* diatas, masing-masing dibuat histogram lalu digabungkan menjadi 1 buah vektor ciri sepanjang 256×3 dimensi.

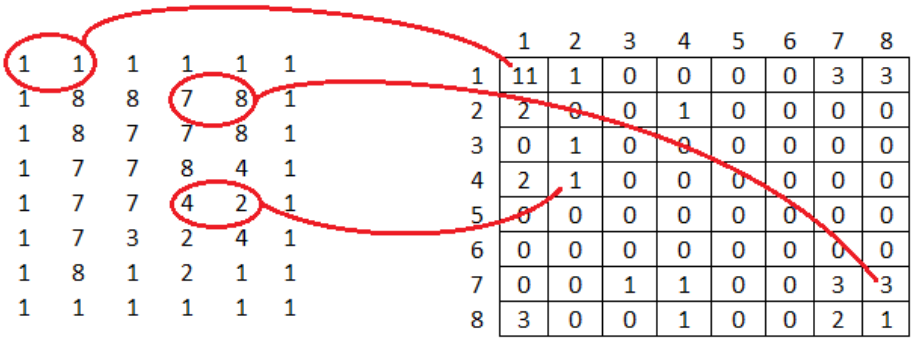
b. GLCM

Pada tahap ini, 3 gambar yang didapat dari tahap sebelumnya akan di hitung *co-occurrence matrix*nya berdasarkan nilai *offset* yang telah ditentukan. Untuk menghitung *co-occurrence matrix*, gambar terlebih dahulu diubah skala intensitas nilai *gray-level*nya menjadi 8 level, sehingga nilainya menjadi integer antara 1 dan 8.



Gambar 5 (a) contoh matriks dari *plane* XY, dan (b) matriks dari *plane* XY setelah diubah skala intensitasnya.

Setelah itu, dari matriks yang didapat baru dihitung *co-occurence matrix*nya. Visualisasinya terlihat pada gambar 5, dengan contoh menggunakan nilai offset [0 1].



Gambar 6 Visualisasi perhitungan *co-occurrence matrix*.

c. Ekstraksi Fitur Tekstur Haralick

Pada tahap ini, masing-masing *plane* yang telah dihitung *co-occurrence matrix*nya lalu diekstraksi fitur tekstur Haralicknya. Pengukuran statistik yang digunakan sebagai fitur adalah *contrast*, *correlation*, *energy*, dan *homogenity*. Perhitungan dari *contrast*, *correlation*, *energy*, dan *homogenity* tersebut dapat dilihat pada persamaan berikut:

Contrast, mengukur variasi lokal di dalam GLCM, dimana i dan j adalah iterasi baris dan kolom dari matriks, K merupakan dimensi baris dan kolom dari matriks, $G(i,j)$ adalah matriks grey-level yang sudah dinormalisasi, dari pasangan pixel (i,j) yang memenuhi offset. $G(i,j)$ itu sendiri bisa diartikan sebagai matriks probabilitas yang berisi dari pasangan pixel (i,j).

$$Contrast = \sum_{i=0}^{K}\sum_{j=0}^{K}(i-j)^2 G(i,j) \quad (3)$$

Correlation texture, mengukur ketergantungan linier dari tingkat keabuan pada pixel-pixel sekitarnya. Fungsi ini mengembalikan nilai berupa pengukuran seberapa berkorelasinya sebuah pixel dengan pixel-pixel sekitarnya pada keseluruhan gambar.

$$Correlation = \sum_{i}^{K}\sum_{j}^{K} G(i,j)\left[\frac{(i-\mu_i)(j-\mu_j)}{\sqrt{\sigma_i^2\sigma_j^2}}\right] \quad (4)$$

dengan $\mu_i$dan μ_j merupakan mean atau rata-rata dari suatu sebaran nilai intensitas *i* dan *j* dari citra keabuan, σ_i dan σ_j adalah standar deviasi, yang menunjukkan sebaran nilai pixel dari bidang citra. μ_i, μ_j, σ_i dan σ_j didefinisikan sebagai berikut:

$$\mu_i = \sum_{i}^{K} iP(i); \qquad \mu_j = \sum_{j}^{K} jP(j) \tag{5}$$

$$\sigma_i = \sum_{i}^{K} (i-\mu_i)^2 P(i); \qquad \sigma_j = \sum_{j}^{K} \left(j-\mu_j\right)^2 P(j) \tag{6}$$

dimana *P*(*i*) dan *P*(*j*) merupakan nilai probabilitas dari nilai intensitas *i* dan *j*.

Energy, mengukur tingkat homogenitas dari sebuah gambar, dan menyediakan jumlah dari elemen-element kuadrat di dalam GLCM.

$$Energy = \sum_{i}\sum_{j} G(i,j)^2 \tag{7}$$

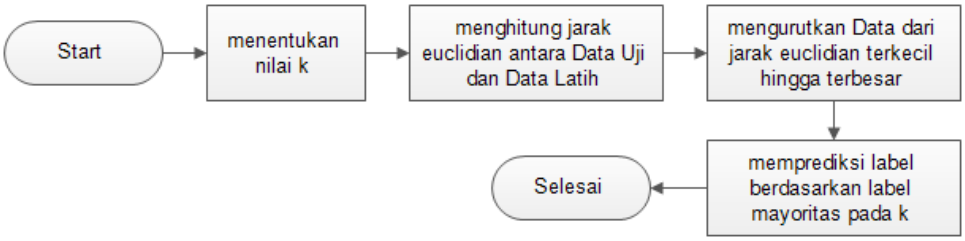
Fungsi *Homogenity* mengembalikan nilai yang mengukur kedekatan antara distribusi elemen-elemen di dalam GLCM dan elemen-elemen diagonal GLCM.

$$Homogenity = \sum_{i}\sum_{j} \frac{G(i,j)}{1+|i-j|} \tag{8}$$

Masing-masing *plane* (XY, XT, dan YT) dihitung fitur tekstur Haralicknya, lalu digabungkan menjadi 1 vektor ciri 12 dimensi. Hasil vektor ciri tersebut lalu digabungkan dengan hasil histogram dari tahap ekstraksi ciri LBP-TOP, sehingga menghasilkan vektor ciri sepanjang (256×3)+12 dimensi. Vektor ciri tersebut yang menjadi keluaran dari tahap ekstraksi ciri.

d. Klasifkasi k-Nearest Neighbors

Klasifikasi ciri menggunakan vektor ciri sepanjang (256×3)+12 dimensi yang didapat dari proses Ekstraksi Ciri (nomor 3.1.4) dari data latih positif dan negatif lalu diklasifikasikan dengan menggunakan kNN. Data latih yang digunakan dibagi menjadi 2 jenis yang berbeda yaitu terdapat data latih positif dimana berisi kumpulan video yang terfokus ke objek api saja dan data latih negatif berisi kumpulan video objek non-api.

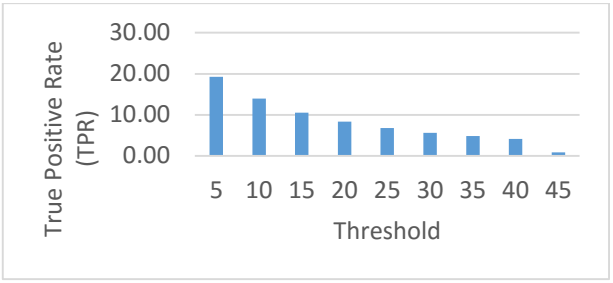


Gambar 7 Alur klasifikasi kNN

3. Pengujian

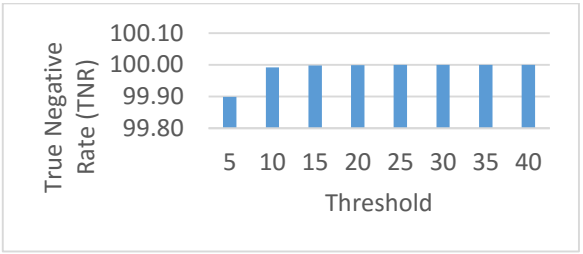
Pada penelitian ini terdapat beberapa tahapan skenario pengujian. Berikut ini adalah skenarionya:

a. Skenario I, pengujian three-frame differencing menggunakan 6 data yang berisi pergerakan dan 6 data yang tidak berisi pergerakan. Parameter yang diuji adalah parameter *threshold*.



Gambar 8 Pengujian three-frame differencing (moving video)

Dari Gambar 8 dapat dilihat bahwa nilai *threshold* dengan TPR terbaik adalah 5 untuk video yang berisi pergerakan. Nilai *threshold* tersebut berlaku untuk setiap jenis video, dimana nilai TPR terbaik untuk setiap video yang memiliki kasus berbeda-beda didapatkan dengan nilai *threshold* 5. Semakin besar nilai *threshold* yang digunakan, nilai TPR semakin kecil, itu berarti area objek yang terdeteksi semakin kecil. Dengan nilai *threshold* yang kecil, area hasil deteksi objek menjadi besar akan tetapi *noise*nya pun menjadi banyak. Hasil yang diharapkan dari tahap three-frame detection ini adalah area deteksi yang besar karena masih akan difilter oleh tahap berikutnya, oleh sebab itu, semakin besar area objek yang terdeteksi, semakin baik.



Gambar 9 Pengujian three-frame differencing (non-moving)

Hasil pengujian diatas menunjukkan tidak perubahan yang signifikan pada perubahan nilai *threshold* untuk video yang tidak terdapat pergerakan sama sekali. Maka dari itu pengujian keseluruhan sistem akan digunakan nilai *threshold* sebesar 5 berdasarkan hasil pengujian sebelumnya.

b. Skenario II, pengujian ekstraksi ciri LBP-TOP, menggunakan data sebagai data latih berjumlah 666 volume data positif dan 276 volume data negatif, dan data yang digunakan sebagai data uji berjumlah 225 volume data positif dan 92 volume data negatif. Parameter yang diuji adalah radius, T (rentang frame yang diambil), dan neighbor points. Masing-masing parameter LBP-TOP pada skenario ini bersifat dependen terhadap parameter GLCM.

Tabel 1 Tabel Pengujian LBP-TOP

Radius	T	Neighbor Points	Confusion Matrix				Akurasi
			TP	TN	FP	FN	
1	5	8	181	86	44	6	84.23
		4	176	86	49	6	82.65
	10	8	183	86	42	6	84.86
		4	178	86	47	6	83.28
2	5	8	177	85	48	7	82.65
		4	169	85	56	7	80.13
	10	8	183	85	42	7	84.54
		4	172	85	53	7	81.07
3	5	8	179	85	46	7	83.28
		4	173	85	52	7	81.39
	10	8	187	85	38	7	85.80
		4	173	85	52	7	81.39
4	5	8	179	85	46	7	83.28
		4	175	85	50	7	82.02
	10	8	187	85	38	7	85.80
		4	174	85	51	7	81.70

Dari hasil pengujian yang dilakukan, terdapat dua hasil terbaik yang diperoleh, dengan nilai akurasi 85.80%. Pertama dengan parameter Radius=3, *t*=10, Neighbor Points=8, dan kedua dengan parameter Radius=4, *t*=10, Neighbor Points=8. Hasil dengan parameter Neighbor Points=8 selalu menunjukkan akurasi yang lebih besar dibanding hasil dengan parameter Neighbor Points=4. Hal tersebut dikarenakan vektor ciri dari hasil Neighbor Points=8 lebih panjang sehingga memiliki dimensi yang lebih banyak yang

mengakibatkan pengklasifikasian kNN menjadi lebih akurat. Untuk parameter Radius=3 dan 4, dikarenakan jangkauannya yang lebih besar sehingga memberikan hasil yang lebih baik, akan tetapi *computational cost* untuk Radius=4 lebih besar dibandingkan Radius=3, sehingga digunakan parameter LBP dengan konfigurasi Radius=3 untuk pengujian selanjutnya.

c. Skenario III, pengujian ekstraksi ciri GLCM, menggunakan data seperti pada skenario sebelumnya. Parameter yang diuji adalah arah offset dan jarak offset. Masing-masing parameter GLCM pada skenario ini bersifat dependen terhadap parameter LBP-TOP.

Tabel 2 Tabel Pengujian GLCM

Jarak Offset	Offset		Confusion Matrix				Akurasi
			TP	TN	FP	FN	
1	0	1	187	85	38	7	85.80
	-1	1	191	77	34	15	84.54
	-1	0	191	69	34	23	82.02
	-1	-1	197	83	28	9	88.33
2	0	1	218	86	7	6	95.90
	-1	1	197	47	28	45	76.97
	-1	0	195	69	30	23	83.28
	-1	-1	199	53	26	39	79.50
3	0	1	224	86	1	6	97.79
	-1	1	223	59	2	33	88.96
	-1	0	224	69	1	23	92.43
	-1	-1	216	62	9	30	87.70
4	0	1	225	86	0	6	98.11
	-1	1	223	46	2	46	84.86
	-1	0	221	63	4	29	89.59
	-1	-1	223	61	2	31	89.59

Dari hasil pengujian diatas, dapat dilihat bahwa hasil terbaik selalu pada nilai offset [0 1] dan [0 -1], yang sebenarnya memiliki arah yang sama, yaitu nilai tetangga yang diambil adalah tetangga horizontalnya, hanya saja yang berbeda adalah untuk [0 1] tetangga yang dipilih adalah sebelah kanan dan [0 -1] adalah tetangga sebelah kiri. Untuk jarak ofset dengan hasil akurasi terbaik adalah 4, dan akan digunakan untuk pengujian selanjutnya.

d. Skenario IV, pengujian klasifikasi kNN, menggunakan data seperti pada skenario sebelumnya. Parameter yang diuji adalah nilai k.

Tabel 3 Tabel Pengujian kNN

k	TP	TN	FP	FN	Akurasi
10	225	86	0	6	98.107
50	225	87	0	5	98.423
100	225	87	0	5	98.423
150	225	86	0	6	98.107
200	225	86	0	6	98.107

Hasil pengujian menunjukkan tidak ada perubahan akurasi untuk nilai k mulai dari 1 sampai 40, sehingga dilakukan pengujian nilai k dengan kelipatan 50. Dari Tabel 4.6 diatas terlihat untuk nilai k 50 dan 100 menunjukkan perubahan nilai akurasi yang tidak signifikan, sehingga bisa disimpulkan bahwa nilai k tidak mempengaruhi hasil klasifikasi pada pengujian ini, dan nilai k=10 akan digunakan untuk pengujian selanjutnya.

e. Skenario V, pengujian keseluruhan sistem, menggunakan parameter-parameter terbaik dari hasil pengujian-pengujian sebelumnya. Data yang digunakan adalah 6 video berisi objek api dan 6 video berisi objek non-api.

Tabel 4 Tabel pengujian pada video berisi objek api

Nama Video	Moving					Fire				
	TP	TN	FP	FN	Akurasi (%)	TP	TN	FP	FN	Akurasi (%)
fire4_2	419	0	0	1	99.762	323	12	63	22	79.76
fire5_2	509	0	0	2	99.609	496	0	0	15	97.07
fire9_1	558	0	0	1	99.821	335	79	1	144	74.06
fireIndoor2_1	430	0	0	22	95.133	283	27	0	142	68.58
fireIndoor2_3	407	0	0	7	98.309	372	0	0	42	89.86
fireIndoor4_4	323	0	0	84	79.361	207	0	0	200	50.86
Rata-Rata										76.70

Hasil pengujian keseluruhan sistem diatas dapat diambil rata-rata akurasi untuk video berisi objek api sebesar 76.70% dan untuk video berisi objek non-api sebesar 65.18%. Untuk video berisi objek api memiliki akurasi terbesar, yaitu 97.07% pada video fireOutdoor5_2 yang merupakan video api ideal dengan lokasi di luar ruangan dan pengambilan video dilakukan saat siang hari. Lalu untuk akurasi terkecilnya, yaitu 50.86% pada video fireIndoor4_4 yang merupakan video api dengan lokasi dalam ruangan, dengan kondisi api sudah mulai padam. Penurunan akurasi terjadi karena sistem tidak mampu mendeteksi api yang sudah mulai padam tersebut meskipun masih ada terlihat lidah api pada video, meskipun kecil.

Tabel 5 Tabel pengujian pada video berisi objek non-api

Nama Video	Moving					Fire				
	TP	TN	FP	FN	Akurasi (%)	TP	TN	FP	FN	Akurasi (%)
nonfire10_1	300	0	0	0	100	0	234	66	0	78.00
nonfire10_2	302	0	0	0	100	0	112	190	0	37.09
nonfire11_1	334	0	0	2	99.405	0	176	160	0	52.38
nonfire11_2	256	0	0	0	100	0	138	118	0	53.91
nonfire2_1	46	58	1	1	98.113	0	85	21	0	80.19
nonfire2_5	247	12	34	22	82.222	0	282	33	0	89.52
Rata-Rata										65.18

Untuk video berisi objek non-api memiliki akurasi terbesar, yaitu 89.52% pada video nonfire2_5 yang merupakan video sepeda motor lewat dengan warna lampu merah. Hasilnya cukup baik. Untuk akurasi terkecil, yaitu 37.09% pada video nonfire10_2, berupa video orang lewat dengan warna pakaian menyerupai warna api.

4. Kesimpulan

Berdasarkan percobaan yang telah dilakukan, maka dapat ditarik beberapa kesimpulan bahwa sistem dapat mengenali objek api dan non-api dengan cukup baik menggunakan metode LBP-TOP dan GLCM dengan tingkat akurasi sebesar 76.70% untuk objek api dan 65.18% untuk objek non-api. Parameter yang digunakan untuk mencapai akurasi tersebut dalam proses ekstraksi ciri adalah t=10, radius=3, dan Neighbor Points=8 untuk parameter LBP-TOP, lalu arah offset = [0 1] dan jarak offset = 4 untuk parameter GLCM. Nilai *True Positive Rate* (TPR) dari hasil pengujian untuk metode *three-frame differencing* adalah 19.265%. Hal ini dikarenakan metode tidak sensitif terhadap gerak dan tidak dapat mendeteksi bentuk objek secara keseluruhan. Sistem membutuhkan hasil deteksi yang baik karena proses ekstraksi ciri menggunakan data berupa objek penuh. Nilai k pada klasifikasi kNN tidak berpengaruh terhadap akurasi sistem ini karena setelah dilakukan pengujian pada nilai k perubahan nilai akurasinya sangat kecil.

Daftar Pustaka

[1] Bohush, Rykhard, and Broukak Nadzeay , "Smoke and flame detection in video sequences based on static and dynamic features.," *Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*, pp. 20-25, 2013.

[2] Chenebert, Audrey, Toby P. Breckon, and Anna Gaszczak, "A non-temporal texture driven approach to real-time fire detection.," *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pp. 1741-1744, 2011.

[3] Chien, Chun-Liang, and Din-Chang Tseng, "Color image enhancement with exact HSI color model.," *international journal of innovative computing, information and control*, vol. 7, no. 12, pp. 6691-6710, 2011.

[4] Pereira et al., "Wildfire effects on extractable elements in ash from a Pinus pinaster forest in Portugal.," *Hydrological Processes*, vol. 28, no. 11, pp. 3681-3690, 2014.

[5] Töreyin, B. Uğur, and Yiğithan Dedeoğlu, "Flame detection in video using hidden markov models.," *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, vol. 2, pp. II-1230, 2005.

[6] Zhao, Guoying, and Matti Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions.," *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 29, no. 6, pp. 915-928, 2007.

[7] Ojala, Timo, Matti Pietikäinen, and David Harwood, "A comparative study of texture measures with classification based on featured distributions.," *Pattern recognition*, vol. 29, no. 1, pp. 51-59, 1996.