BoWFire: detection of fire in still images by integrating pixel color and texture analysis

BoWFire: Detection of Fire in Still Images by Integrating Pixel Color and Texture Analysis

Daniel Y. T. Chino, Letricia P. S. Avalhais, Jose F. Rodrigues Jr., Agma J. M. Traina
Institute of Mathematics and Computer Science, University of Sao Paulo
Sao Carlos, Brazil
{chinodyt,letricia,junio,agma}@icmc.usp.br

***Abstract*—Emergency events involving fire are potentially harmful, demanding a fast and precise decision making. The use of crowdsourcing image and videos on crisis management systems can aid in these situations by providing more information than verbal/textual descriptions. Due to the usual high volume of data, automatic solutions need to discard non-relevant content without losing relevant information. There are several methods for fire detection on video using color-based models. However, they are not adequate for still image processing, because they can suffer on high false-positive results. These methods also suffer from parameters with little physical meaning, which makes fine tuning a difficult task. In this context, we propose a novel fire detection method for still images that uses classification based on color features combined with texture classification on superpixel regions. Our method uses a reduced number of parameters if compared to previous works, easing the process of fine tuning the method. Results show the effectiveness of our method of reducing false-positives while its precision remains compatible with the state-of-the-art methods.**

***Keywords*-fire detection; still images; pixel-color classification; texture feature**

I. Introduction

Emergency situations can cause economic losses, environmental disasters or serious damage to human life. In particular, accidents involving fire and explosion, have attracted interest to the development of automatic fire detection systems. Existing solutions are based on ultraviolet and infrared sensors, and usually explore the chemical properties of fire and smoke in particle samplings [1]. However, the main constraint of these solutions is that sensors must be set near to the fire source, which brings complexity and cost of installation and maintenance, especially in large open areas.

Alternative to sensors, cameras can provide visual information of wider spaces, and have been increasingly embedded in a variety of portable devices such as smartphones. To take advantage of this, the *RESCUER*[1] Project is developing an emergency system to support Crisis Control Committees (CCC) during a crisis situation. The system under development in the *RESCUER* Project allows witnesses, victims or rescue staff, present at the emergency location, to send images and videos of the incident to a crowdsourcing mobile framework. However, this approach might lead to a volume of information flowing at a rate higher than what human specialists are able to analyze. For this reason, as part of the *RESCUER* Project, an automated data analysis solution is under investigation aimed at identifying the most relevant pieces of information, filtering out irrelevant data. Relevance here refers to images that actually contain fire elements, and that can effectively assist in the decision making of a given crisis.

Several methods regarding to fire detection on videos have been proposed in the last years. These methods use two steps to detect fire. First, they explore the visual features extracted from the video frames (images); second, they take advantage of the motion and other temporal features of the videos [3]. In the first step, the general approach is to create a mathematical/rule-based model, defining a sub-space on the color space that represents all the fire-colored pixels in the image. There are several empirical models using different color spaces as RGB [1], YCbCr [4], CIE Lab [5] and HSV [6]. In these cases, the limitation is the lack of correspondence of these models to fire properties beyond color. The problem is that high illumination value or reddish-yellowish objects lead to a higher false-positive rate. These false-positives are usually eliminated on the second step through temporal analysis.

In contrast to such methods, our proposal is to detect fire in still images, without any further (temporal) information, using only visual features extracted from the images. To overcome the problems aforementioned, we propose a new method to detect fire in still images that is based on the combination of two approaches: pixel-color classification and texture classification. The use of color is a traditional approach to the problem; whilst, the use of texture is promising, because fire traces present particular textures that permit to distinguish between actual fire and fire-like regions. We show that, even with just the information present in the images, it is possible to achieve a high accuracy level in such detection.

The main contribution of this research is the proposal of *BoWFire* (*Best of both Worlds Fire detection*), a novel method to detect fire in still images. By merging color and texture information, our method showed to be effective in detecting true-positive regions of fire in real-scenario images, while discarding a considerable quantity of false-positives. Our method uses fewer parameters than former works, what leads to a more intuitive process of fine tuning the automated detection. Regarding these claims, in the experiments, we systematically compare *BoWFire* with four works that currently define the state-of-the-art, that is, the works of Celik *et al.* [4], Chen *et*

[1]Project FP7-ICT-2013-EU-Brazil - "RESCUER - Reliable and Smart Crowdsourcing Solution for Emergency and Crisis Management"

al. [1], Rossi *et al.* [7], and Rudz *et al.* [8]. The remaining of this manuscript is organized as follows: Section II briefly surveys previous approaches related to fire detection methods; Section IV presents the proposed method using color and texture segmentation; Section V describes the experimentation; Section VI presents the discussion about the results and, finally, Section VII concludes this study.

II. Related Works

A fire detection method based on rules was proposed in the work of Chen *et al.* [1]. They define a set of three rules using a combination of the RGB and the HSI color spaces; the user, in turn, must set two threshold parameters to detect fire pixels. Another method based on color was proposed by Celik *et al.* [4], who conducted a wide-ranging study regarding the color of fire pixels to define a model. This method defines a set of five mathematical rules to compare the intensity of the channels in the YCbCr color space; this was because the YCbCr has a better discrimination regarding fire [4][8]. Also in this work, the user must set a threshold for one of the rules.

Rossi *et al.* [7] proposed a method to extract geometric fire characteristics using stereoscope videos. One of the steps is a segmentation based on a clustering algorithm, in which the image is divided into two clusters based on the channel V of the YUV color space. The cluster with the highest value of V corresponds to fire. Thereafter, Rossi *et al.* used a 3D-Gaussian model to classify pixels as fire. In this method, the accuracy of the classification depends on a parameter provided by the user. This method presents limitations, since the authors assume that the fire is registered in a controlled environment.

Rudz *et al.* [8] proposed another method based on clustering. Instead of using the YUV color space, Rudz *et al.* computes four clusters using the channel Cb of the YCbCr color space. The cluster with the lowest value of Cb refers to a fire region. A second step eliminates false-positive pixels using a reference dataset. The method treats small and large regions with different approaches; small regions are compared with the mean value of a reference region, while large regions are compared to the reference histogram. This comparison is made for each RGB color channel. The user must set three constants for the small regions, and three thresholds for the large regions, resulting in a total of six parameters.

In comparison to the previous works, our method improves the state-of-the-art by using texture, beyond color, to reduce false-positives; and by using a smaller set of parameters, an important characteristic to fine tune the detection process. Besides, the parameters of former methods do not carry physical significance, therefore, they are less intuitive to adjust.

III. Basic Concepts and Notations

In this section we present important concepts related to the problem under analysis. An image can be defined as a set of pixels $I = \{P_i | 0 \leq i < n\}$, where n is the total number of pixels in the image. Each pixel is a tuple $P_i = (R_i, G_i, B_i)$, where the values of R_i, G_i and B_i represents the intensity of each channel of the RGB color space. Global or local information can be extracted from images. A feature extraction is a function F that for a given image I, generates a feature vector $V \in \mathbb{R}^d$, of size d.

Let a set $\mathcal{T} \subset \mathbb{R}^d$ of tuples of size d and a set of possible labels $\mathcal{L}$. Given a training set in which every $t_i \in \mathcal{T}$ has a label c_i assigned by an expert, where $c_i \in \mathcal{L}$; a supervised classifier C must build a model capable to predict the label of a new data item. Given a tuple $x \in \mathbb{R}^d$, a classifier can be defined as the function $C(x) = c$, where $c \in \mathcal{L}$. Among the most used classifiers are the NaÃ¯ve-Bayes [9] and K-Nearest Neighbors (*KNN*) [10]. We refer to the above definitions in the rest of this work.

A. Feature Extraction

Images are processed by means of extracted features. The features extracted from a given image correspond to numerical measurements that describe visual properties. Such properties are able to discover and represent connections between pixels of the whole image (global) [11], or of small regions of the image (local) [12]. Low-level descriptors [13], as those base on color, shape and texture, are frequently used.

Usually, color-feature extraction methods have a low computing cost. Color Layout [14] is an example of color extraction, it describes the space distribution of the colors. The color histogram also allows to compute the color moments [15] to describe the probability distribution of the image colors. Shape information, in turn, is considered the closest approximation to the human perception of an object's image [16]. Feature extractors of this depend on a pre-processing step that segments and detects the border of the objects. There are various methods to extract shape features, as the Zernike moments [17] and the Fourier descriptors [18]. Texture is also a common feature in image processing. It is important because, together with color, it describes the surface of naturally-occurring phenomena, as fire, smoke, and water. Classical feature extractors of texture are LBP [19] and Haralick [20].

The Local Binary Pattern (LBP) is a texture feature extractor that considers the neighborhood of a pixel [19]. The LBP can be used in several applications, as helmet detection on motorcyclists [21], MR image retrieval systems [22], and image segmentation [23]. The LPB can be performed in gray scale, in a region of 3×3 pixels. For each pixel P_i from the neighborhood of the central pixel P_c, a binary code is created by assigning the value 1 if $P_i > P_c$, or 0, otherwise. Then, the histogram of codes is used as a feature vector.

In order to make the LBP rotation invariant, a variation of the original algorithm shifts the code until it reaches its minimum value [24]. Another variation is defined as *uniform patterns* [25]. A code is called uniform if the binary pattern contains at most two bitwise transitions from 0 to 1 (or vice versa). The histogram is obtained by taking one bin for each uniform pattern and a single bin for the non-uniform patterns.

B. Superpixel Generation

Superpixel algorithms group pixels into atomic regions with similar homogeneity. Doing so, superpixels can capture the image redundancy and reduce the complexity of subsequent image processing tasks. Superpixels can be used as building blocks of many computer vision algorithms [26], in this paper, we will focus on image segmentation [27][28].

A superpixel Sp is defined as a subset of the image I that contains pixels from a continuous region of the image. A superpixel generation algorithm can be described as the process $S(I, K_{sp}) = \{Sp_j | 0 \leq j < K_{sp}\}$ that takes the image I and returns K_{sp} partitions. A superpixel generation algorithm must have a good adherence to image boundaries and should improve the performance of the segmentation algorithm. An empirical comparison between the state-of-the art superpixel algorithms was made by Achanta *et al.* [26]. Their results showed that the algorithm with the best overall performance was the Simple Linear Iterative Clustering (SLIC) technique. The SLIC technique is an adaptation of K-Means for superpixel generation using a distance function based on the values of pixels using the Lab color space and their geometric position. The user gives as parameter the number of superpixels K_{sp} and their compactness m. The algorithm then positions the centroids on a regular grid, avoiding seeding a superpixel on an edge pixel.

IV. Our Proposal

We propose *BoWFire* (Best of Both Worlds Fire detection), a novel method for fire detection in emergency-situation images. We explore the fact that color combined with texture can improve the detection of fire, reducing the number of false-positives as compared to related works from the literature. We show that such combination can distinguish actual fire from fire-like regions (reddish/yellowish) of a given image. The goal is to provide a more effective automated detection of fire scenes in the context of the crisis situations, as those of the *RESCUER* Project. Figure 1 shows the basic architecture of our proposal. The *BoWFire* method consists of three basic steps: *Color Classification*, *Texture Classification*, and *Region Merge*. As shown in Figure 1, the two first steps occur in parallel to produce images in which fire-classified pixels are marked. Then, the output from both classifications is merged into a single output image by the *Region Merge* step.

Different from other methods, usually based on mathematical models, the use of a *Color Classification* step avoids the need of a great number of parameters. Any machine learning classification algorithm could be used, specifically, in this work, we use NaÃ¯ve-Bayes and *KNN*, as detailed in Section V. By doing so, we also avoid the use of the global information of the image to classify only one pixel as opposed to other approaches; this is a desired feature because the semantics of the image may vary according to the emergency situation (small/large fire regions or day/night time). Figure 2 presents more details of the color-based classification. Given an image I with n pixels P_i, $0 \leq i < n$. Each pixel $P_i = (R_i, G_i, B_i)$ of the image is converted to $P'_i = (Y_i, Cb_i, Cr_i)$ in the YCbCr color space, since this color space provides a better discrimination of fire regions. Then P'_i goes through a *Pixel-Color Classification* ($pixelClass$), which

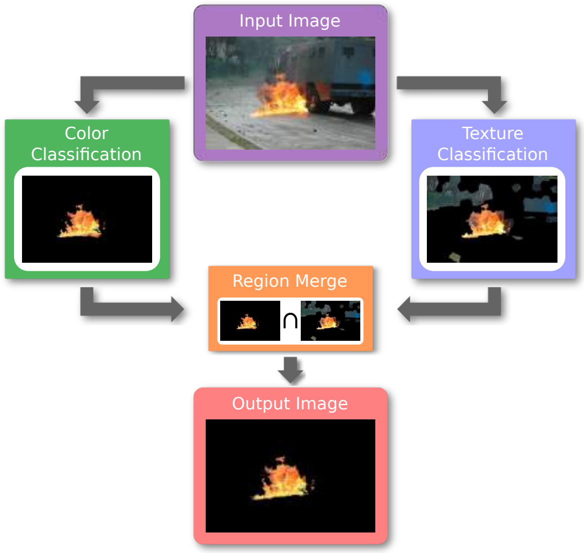


Fig. 1: Architecture of the *BoWFire* method.

consists of a *Color Training Set* and a *Color Classifier*. Then, if $pixelClass(P'_i) = \langle \text{fire} \rangle$, P_i is used to build the output image I_{color}, otherwise P_i is discarded.

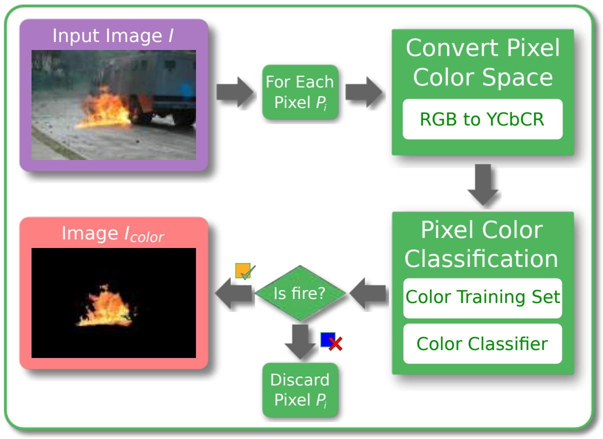


Fig. 2: Color-based classification step.

As mentioned earlier, the *Texture Classification* step allows for a more accurate detection; however, it brings a challenge. Since there may be a variety of fire images according to the emergency situation, it is not possible to extract global features of the image because the small fire regions would vanish in the global context. Therefore, we extract only local features from regular shaped regions with similar patterns automatically detected by superpixel methods. Figure 3 presents details of the *Texture Classification* step. Given the same image I, we use a superpixel method $S(I, K_{sp})$ to generate a set of K_{sp} superpixels Sp_j, where $0 \leq j < K_{sp}$. Next, each superpixel Sp_j passes through a local *Feature Extraction* process, resulting in a feature vector $V_j = (v_{j0}, \ldots, v_{j(d-1)})$ of size d. Then, V_j is classified using a *Feature Classification* ($featClass$), which consists of a *Feature Training Set* and a *Feature Classifier*.

If $featClass(V_j) = \langle\text{fire}\rangle$, all pixels $P_i \in Sp_j$ are used to build the output image $I_{texture}$, otherwise they are discarded. After this, the superpixel region is no longer necessary since the method is performed in pixel-level only.

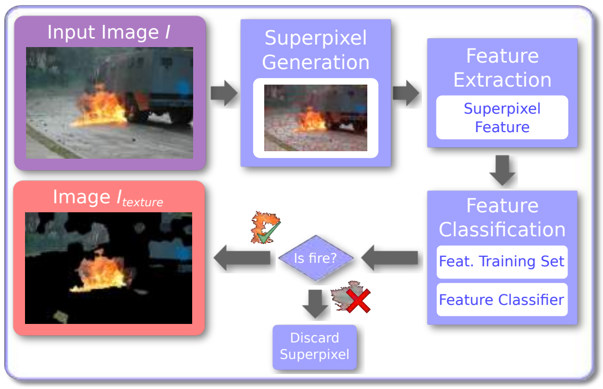


Fig. 3: Texture-based classification step.

With the outputs from the *Color Classification* (image I_{color}), and from the *Texture Classification* (image $I_{texture}$), it is still necessary to join the results in an output image $I_{classified}$. We perform this task in the *Region Merge* step. According to our hypothesis, if a pixel is simultaneously classified as fire following color and texture classification, then there is a higher chance that this pixel is actual fire. Therefore, given an image I and its color and texture classifications I_{color} and $I_{texture}$, the final classified image $I_{classified}$ is defined as $I_{classified} = \{P_i | P_i \in I_{color} \text{ and } P_i \in I_{texture}\}$. That is, a given pixel is added to the final output only if it was detected in both color and texture classifications, otherwise it is discarded. Consequently, we dismiss false-positives from both approaches, taking advantage of the best of both worlds.

The *BoWFire* method was developed in a modularized scheme, allowing an easy way to add and set different feature extraction algorithms, as well as different classifiers. We note that, since the *BoWFire* method is fully customizable, the number of parameters is dependent only on the algorithms used in the intermediate steps.

V. Experiments

A. Configuration

Following, we describe the configuration of our experiments.

Datasets: We performed experiments using a dataset of fire images. It consists of 226 images with various resolutions[2]. Also, it was divided in two categories: 119 images containing fire, and 107 images without fire. The fire images consist of emergency situations with different fire incidents, as buildings on fire, industrial fire, car accidents, and riots. These images were manually cropped by human experts. The remaining images consist of emergency situations with no visible fire and also images with fire-like regions, such as sunsets, and red or

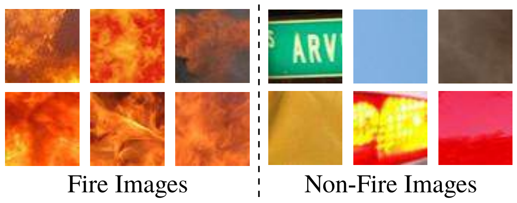


Fig. 4: Sample images of the training dataset.

yellow objects. Figures 9, 10 and 11 show some samples of this dataset. Since we are using supervised machine learning, we also used a training dataset[2]. This second dataset consists of 240 images of 50×50 pixels resolution; 80 images classified as fire, and 160 as non-fire. Figure 4 shows some samples of this dataset. It is important to note that the non-fire images also contain red or yellow objects.

Intermediate Algorithms: Since the goal of our method is to have a fewer number of parameters, for the *BoWFire* intermediate steps, we used the following algorithms. The *Pixel-Color Classification* is done by a NaÃ¯ve-Bayes classifier, using an automatic discretization method; the superpixel algorithm was the SLIC method with a modification. Instead of using the Lab color space, we used the YCbCr space due to its discriminative property. Since we wanted to add texture information, our implementation uses the *uniform patterns* LBP. Other feature extraction methods were evaluated, but were omitted due to space limitation. The features were classified using the *KNN* classification with the Manhattan Distance.

Parameters: Considering the configuration given by the choice of intermediate algorithms, the *BoWFire* method needs only 3 parameters: K_{sp}, m and K. For all experiments we empirically evaluated the best values for parameters m and K; for parameter K, we used the value $K = 11$. Regarding to parameter m, we observed that a more compact superpixel generates a more regular region, which leads to a better representation of the texture feature. In this case, the best value was $m = 40$. With these parameters, each method was executed on three different datasets: only fire images, only non-fire images, and a complete dataset with both fire and non-fire. For each execution, we computed the confusion matrix for the classification of all pixels and calculated four measures: Precision, Recall, F1-Score, and False-Positive Rate (FPR).

B. Description of the experiments

In this section, we describe the experiments (i) on the impact of parameter K_{sp}, (ii) on the *Color Classification* Evaluation, and (iii) on the *BoWFire* Evaluation. Next, in Section VI, we report on the execution and results of these experiments.

Impact of K_{sp}: The first experiment evaluates the impact of the number of superpixels on the *BoWFire* performance. We vary the number of superpixels K_{sp} with the following values: 50, 100, 150, 200, 250 and 300.

***Color Classification* Evaluation:** In this experiment, we aim at evaluating the capability of the *Color Classification* step proposed in this paper. Since *BoWFire* is based on

[2]Available at http://www.gbdi.icmc.usp.br

a combination of two different approaches, it is important that the color-based method recovers as many fire pixels as possible. So, Recall is the measure that closely meets this need. Also, on this step FPR is not so important, since it will be handled on the *Texture Classification* step. We evaluated the behavior of our proposed *Color Classification* in comparison with the state-of-the-art, as in the works of Celik [4], Chen [1], Rossi [7] and Rudz [8].

***BoWFire* Evaluation:** After evaluating only color, we evaluate the impact of considering texture together with color, as defined in our proposal. The most important aspect of this step is to reduce false-positives without affecting the overall performance. In this context, we analyze the *BoWFire* performance, which is the combination of the *Color Classification* step with the *Texture Classification* step. We also evaluated the performance of the state-of-the-art methods combined with the *Texture Classification*. We used the best value of K_{sp} as obtained in the experimentation.

VI. Results and Discussion

Impact of K_{sp}: Figure 5 shows the results obtained while varying the number of superpixels. Figure 5a shows the results for the fire dataset. In this case, there was a slight increase of all measure until $K_{sp} = 150$, then for greater values they had a similar behavior. The Precision obtained was around 0.8, Recall around 0.65 and F1-Score around 0.72. Figure 5b shows the results for the non-fire dataset. For this dataset we computed only the False-Positive Rate. There is a slight increasing of FPR as the number of superpixels increases, except for $K_{sp} = 300$. It is important to notice that although FPR increases, the values remain around 0.045, that is, less than 5% of false-positives. And Figure 5c shows the results combining both datasets. Again, there is a similar behavior regarding K_{sp}, except for $K_{sp} = 50$. The Precision obtained was around 0.5, Recall around 0.65 and F1-Score around 0.57. The FPR values were not shown on both Figures 5a and 5c due to their low values for all K_{sp}. There is also a slight increasing of FPR as K_{sp} increases, but with lower values. On the fire dataset, FPR went from 0.0169 to 0.0175, and on the complete dataset it varied from 0.0305 to 0.0323.

The main goal of the *BoWFire* is to decrease the FPR while maintaining a good performance. With that in mind, we evaluated that the best result is achieved when the number of superpixels $K_{sp} = 150$. This number presented better results while dealing with just the fire and complete dataset (fire and non-fire), as showed by F1-score. Also, the value of FPR for this K_{sp} is close to the lowest FPR value.

***Color Classification* Evaluation:** Figures 6a, 6b and 6c, show the results for the *Color Classification* step. Considering only color-based approaches, *Color Classification*, Celik and Rudz presented the best overall performance. Although Chen obtained the highest value of Precision, its Recall got the lowest value. As seen in Figures 9 and 10, Chen missed too many true-positive pixels and Rossi has the lowest overall performance. We observed that in outdoor emergency situations, fire was not in the cluster with the higher values of V, as shown in Figures 9 and 10. From now on, we will focus our analysis only on the methods with the best overall performance.

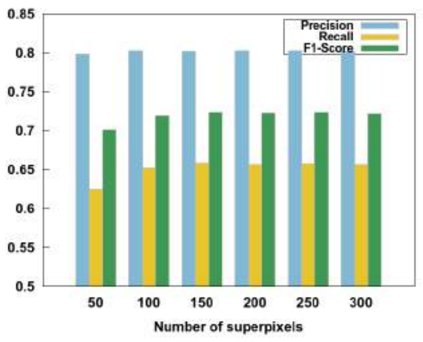
Regarding to Precision, Rudz achieved the best value, 0.84 on the fire dataset and 0.31 on the complete dataset, while *Color Classification* and Celik had similar behavior with values around 0.62 on the fire dataset and 0.24 on the complete. On the other hand, *Color Classification* achieved the highest value of Recall, 0.77 on both fire and complete dataset, followed by Celik, 0.63 on both fire and complete, and Rudz, 0.41 on both fire and complete. Analyzing the F1-Score, *Color Classification* and Celik methods outperformed Rudz by at most 23.6% on the fire dataset with values of 0.68 to *Color Classification*, 0.63 to Celik and 0.55 to Rudz. On the complete dataset, all methods achieved similar F1-Score with the value of 0.35.

On the fire dataset, *Color Classification* and Celik achieved similar values of FPR (0.05 and 0.04) and Rudz method achieved 0.01 FPR. On the non-fire dataset, *Color Classification*, Celik and Rudz achieved respectively 0.21, 0.15 and 0.08. And on the complete dataset, *Color Classification*, Celik and Rudz methods achieved respectively 0.13, 0.10 and 0.05. On all datasets, Rudz achieved the best FRP value, less than 9% of the pixels was incorrectly classified. However, while discarding more false-positives, Rudz also discarded true-positives, reducing its Recall capability. Except for FPR, *Color Classification* had a similar behavior of Celik, but had better values of Recall and F1-Score. Therefore, the *Color Classification* outperformed the other methods.

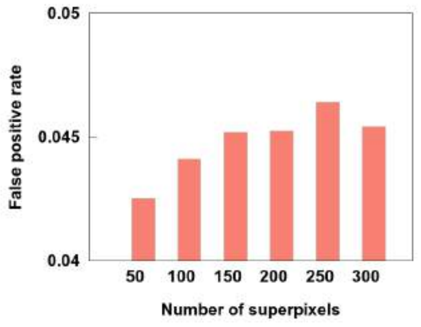
***BoWFire* Evaluation:** Figures 6d, 6e and 6f show the results when added texture information. It is possible to note that there was an overall improvement for all methods. Regarding Precision, with the exception of Rudz and Chen, all methods had a considerable improvement. *Color Classification* and Celik had a Precision improvement of up to 1.30× on the fire dataset and 2.28× on the complete dataset. Rossi had the greatest improvement, 4.43× on fire dataset and 5.65× on the complete dataset. This high improvement was due to the fact that Rossi, on outdoor images, detected other regions than fire, as shown on Figures 9 and 10. When adding texture information, these false-positive regions were discarded. For both Chen and Rudz, there was a slightly improvement on the fire dataset, but it is due to the fact that they already had low FPR. On the other hand, on the complete dataset, there was an improvement of 1.64× to Chen and 2.06× to Rudz.

There was a decreasing on the Recall value of up to 15% less for all methods, except Rudz. This is due to the fact that the combination of both approaches discarded a few true-positives. However, the considerable gain on the precision can justify this drawback. Analyzing the F1-Score, there was a slight increase of up to 7% for all methods, except Rossi, on the fire dataset, which had an improvement of 69%. On the complete dataset, all methods had a considerable improvement, up to 65%.

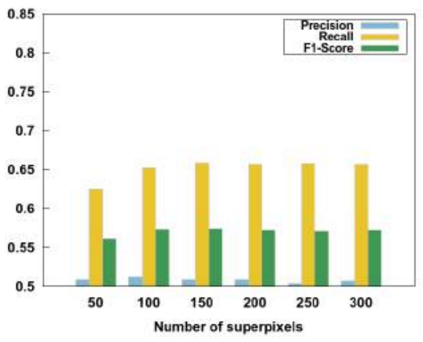
As one of the goals of *BoWFire* is to reduce the number of false-positives, it is important to analyze FPR. On the fire dataset, there was a reduction of up to 68% of FPR for *Color*



(a) Fire dataset

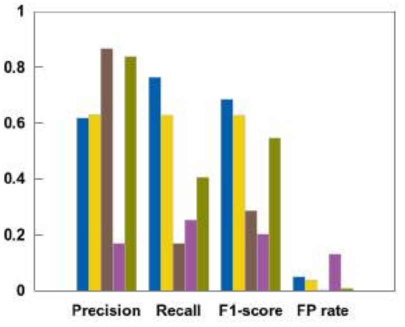


(b) Non-fire dataset

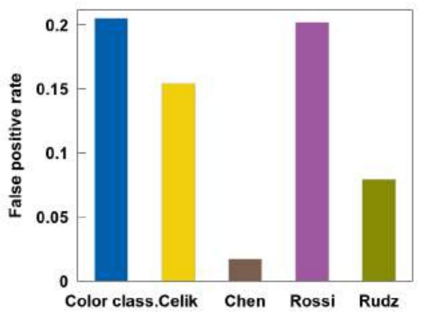


(c) Complete dataset

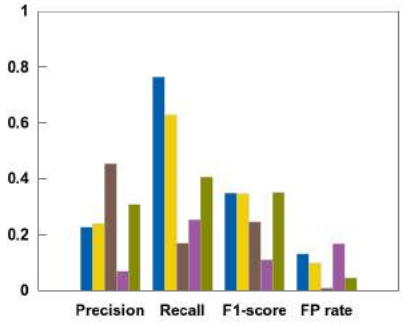
Fig. 5: Impact evaluation of the number of superPixel K_{sp} in three different datasets.



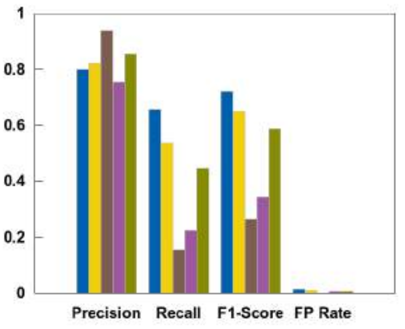
(a) Fire dataset (only color)



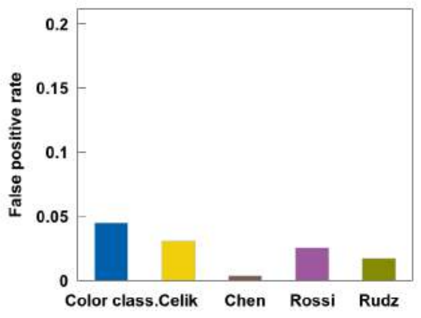
(b) Non-fire dataset (only color)



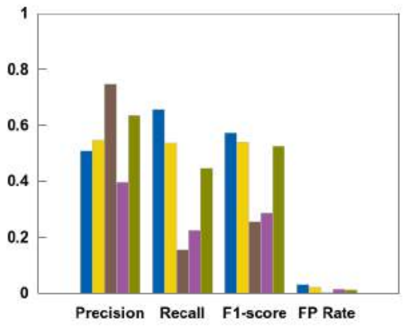
(c) Complete dataset (only color)



(d) Fire dataset (adding texture)



(e) Non-fire dataset (adding texture)



(f) Complete dataset (adding texture)

Fig. 6: Evaluation of the *BoWFire* method with the state-of-the-art methods.

Classification, using Celik and Chen methods. Rossi had 94% less false-positives. Rudz was the least affected by this step, reducing 5% false-positives, since they had already dismissed false-positives on a post processing step. On both the non-fire and complete dataset, all methods reduced FPR by up to 80%. This result confirms that the *Texture Classification* step is capable of discarding false-positives without compromising the overall performance.

We can now use the Receiver Operating Characteristic (ROC) space to analyze the performance behavior between all methods. The ROC Space shows the relation between FPR and the true-positive rate (Recall). Figures 7 and 8 show the ROC Space on, respectively, the fire and the complete datasets for all methods. On both ROC Spaces, it is possible to note that all methods move to the left, i.e., achieve less FPR when texture information is added. The *Color Classification* and the *BoWFire* method presented the best classification results among the other methods, followed by Celik. Also, the *BoWFire* achieved a similar Recall value as Celik without texture information, but with a smaller FPR.

Figures 9, 10 and 11 show visual samples of output images from three different situations. Figure 9 shows an emergency situation with fire and low percentage of possible false-positives. On this input image it is possible to note that *Color Classification*, Celik and Rudz methods had similar outputs. The *BoWFire* method was able to detect the same fire region

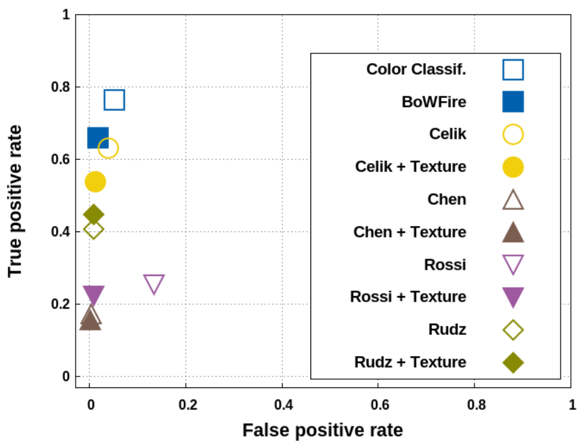


Fig. 7: ROC Space using the fire dataset.

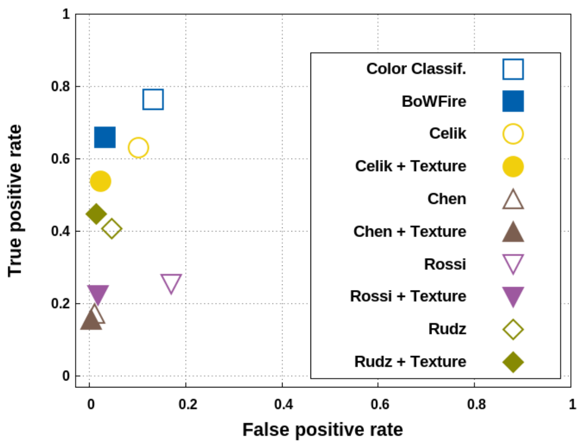


Fig. 8: ROC Space using the complete dataset.

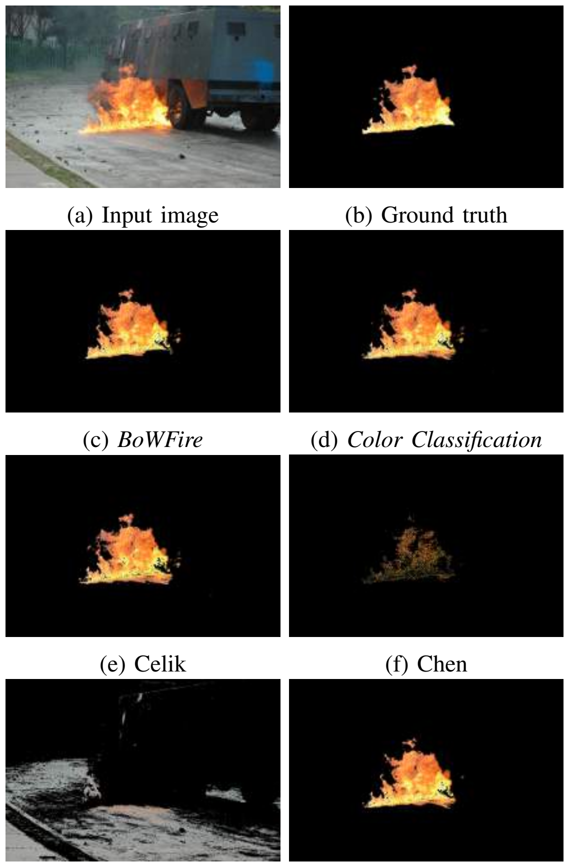


Fig. 9: Output from the methods with an input image with fire.

as these methods but discarded the fire reflection on the ground. Rossi was not able to correctly detect fire regions, while Chen discarded more than half of the true-positives. Figure 10 also shows an emergency situation with fire with a higher percentage of false-positives. In this case, all methods detected false-positives, with the exception of *BoWFire*. It is also possible to note that in some cases, Rudz discards more fire pixels than necessary. This image also shows the problem with the Rossi method, since no fire region was detected as fire. Once again, Chen discarded almost every true-positive. Figure 11 shows a sunset skyline image. For this input image, both *Color Classification* and Rudz detected false-positives. Meanwhile, when adding texture information to the *Color Classification*, *BoWFire* was capable of discarding all false-positives for this image.

VII. Conclusions

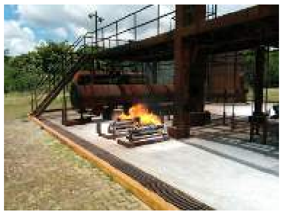
In this paper, we presented the *BoWFire* method, a novel approach for fire detection on images in emergency context. Our results showed that *BoWFire* was capable of detecting fire with a perfomance similar to what is observed in the works of the state-of-the-art, but with less false-positives. We systematically compared our work with four former works, demonstrating that we achieved consistent improvements.

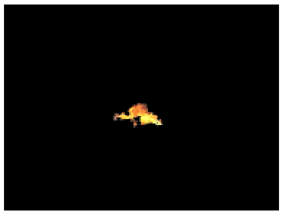
The course of action of *BoWFire* was that, by simultaneously using color and texture information, it was able to dismiss false-positives relying solely on the information present in the images; as opposed to former methods that use temporal information. Furthermore, since *BoWFire* is based on classification methods, rather than on mathematical modeling, it was able to solve the problem with only three parameters. In addition, these parameters were more intuitive for tuning, unlike those of previous works, which are based on thresholds and color-based values. Given its performance, we conclude that BoWFire is suitable to integrate a crisis management system as the one that motivates this work.

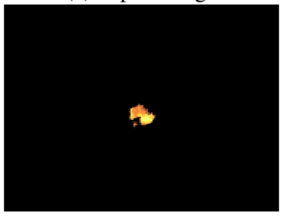
As future work, we plan to investigate the combination of different features extraction methods. We also envision the extension of *BoWFire* to detect other types of incident, such as smoke and explosions.

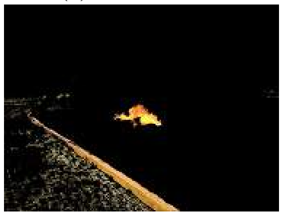
Acknowledgment

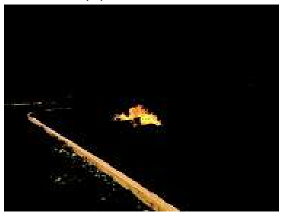
This research is partially funded by FAPESP, CNPq, CAPES, STIC-AmSud and *RESCUER* Project, funded by the European Commission (Grant: 614154) and by the CNPq/MCTI (Grant: 490084/2013-3).

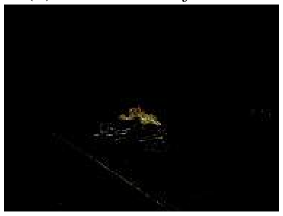

(a) Input image

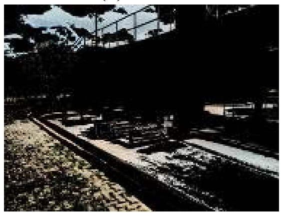

(b) Ground truth


(c) *BoWFire*


(d) *Color Classification*


(e) Celik


(f) Chen


(g) Rossi

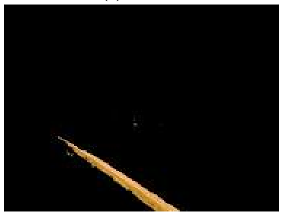

(h) Rudz

Fig. 10: Output from the methods with an input image with fire and possible false-positives pixels.

References

[1] T.-H. Chen, P.-H. Wu, and Y.-C. Chiou, “An early fire-detection method based on image processing,” in *ICIP*, vol. 3, 2004, pp. 1707–1710.

[2] T. Qiu, Y. Yan, and G. Lu, “An autoadaptive edge-detection algorithm for flame and fire image processing,” *IEEE TIM*, vol. 61, no. 5, pp. 1486–1493, 2012.

[3] A. K. Yoon-Ho Kim and H.-Y. Jeong, “Rgb color model based the fire detection algorithm in video sequences on wireless sensor network,” *International Journal of Distributed Sensor Networks*, p. 10, 2014.

[4] T. Çelik and H. Demirel, “Fire detection in video sequences using a generic color model,” *Fire Saf. J.*, vol. 44, no. 2, pp. 147–158, 2009.

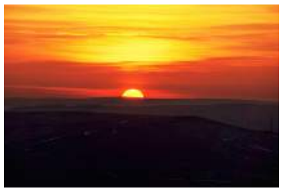
[5] C. Ha, U. Hwang, G. Jeon, J. Cho, and J. Jeong, “Vision-based fire detection algorithm using optical flow,” in *CISIS*, 2012, pp. 526–530.

[6] J. Zhao, Z. Zhang, S. Han, C. Qu, Z. Yuan, and D. Zhang, “Svm based forest fire detection using static and dynamic features,” *Computer Science and Information Systems*, vol. 8, no. 3, pp. 821–841, 2011.

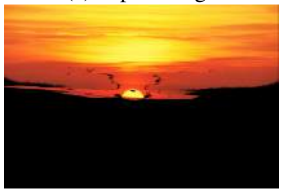
[7] L. Rossi, M. Akhloufi, and Y. Tison, “On the use of stereovision to develop a novel instrumentation system to extract geometric fire fronts characteristics,” *Fire Saf. J.*, vol. 46, no. 1–2, pp. 9–20, 2011.

[8] S. Rudz, K. Chetehouna, A. Hafiane, H. Laurent, and O. Séro-Guillaume, “Investigation of a novel image segmentation method dedicated to forest fire applications,” *IET Meas. Sci. Technol.*, vol. 24, no. 7, p. 75403, 2013.

[9] S. Garcia, J. Luengo, J. Sáez, V. López, and F. Herrera, “A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning,” *IEEE TKDE*, vol. 25, no. 4, pp. 734–750, 2013.


(a) Input image

(b) *BoWFire*


(c) *Color Classification*

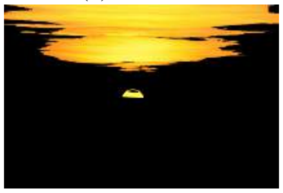

(d) Rudz

Fig. 11: Output of a non-fire image.

[10] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2Nd Edition)*. Wiley-Interscience, 2000.

[11] R. D. S. Torres and A. X. Falcão, “Content-based image retrieval: Theory and applications,” *Rev. de Inf. Teórica e Ap.*, vol. 13, pp. 161–185, 2006.

[12] A. Shaban, H. Rabiee, M. Farajtabar, and M. Ghazvininejad, “From local similarity to global coding: An application to image classification,” in *CVPR*, 2013, pp. 2794–2801.

[13] T. Deselaers, D. Keysers, and H. Ney, “Features for image retrieval: an experimental comparison,” *Information Retrieval*, vol. 11, no. 2, pp. 77–107, 2008.

[14] H. Jalab, “Image retrieval system based on color layout descriptor and gabor filters,” in *ICOS*, 2011, pp. 32–36.

[15] R. Datta, D. Joshi, J. Li, and J. Z. Wang, “Image retrieval: Ideas, influences, and trends of the new age,” *ACM CSUR*, vol. 40, no. 2, p. 5, 2008.

[16] M. Yang, K. Kpalma, J. Ronsin *et al.*, “A survey of shape feature extraction techniques,” *Pattern recognition*, pp. 43–90, 2008.

[17] K. M. Hosny, “Fast computation of accurate zernike moments,” *J. Real-Time Image Processing*, vol. 3, no. 1-2, pp. 97–107, 2008.

[18] C.-S. Chen, C.-W. Yeh, and P.-Y. Yin, “A novel fourier descriptor based image alignment algorithm for automatic optical inspection,” *J. of Vis. Commun. and Image Repres.*, vol. 20, no. 3, pp. 178–189, 2009.

[19] T. Ojala, M. Pietikainen, and D. Harwood, “Performance evaluation of texture measures with classification based on kullback discrimination of distributions,” in *Pattern Recognition*, vol. 1, 1994, pp. 582–585 vol.1.

[20] R. Haralick, K. Shanmugam, and I. Dinstein, “Textural features for image classification,” *IEEE SMC*, vol. 3, no. 6, pp. 610–621, 1973.

[21] R. Rodrigues Veloso e Silva, K. Teixeira Aires, and R. De Melo Souza Veras, “Helmet detection on motorcyclists using image descriptors and classifiers,” in *SIBGRAPI*, 2014, pp. 141–148.

[22] A. Varghese, K. Balakrishnan, R. R. Varghese, and J. S. Paul, “Content-Based Image Retrieval of Axial Brain Slices Using a Novel LBP with a Ternary Encoding,” *The Computer Journal*, vol. 57, no. 9, pp. 1383 – 1394, 2014.

[23] J. Vargas, P. Saito, A. Falcao, P. De Rezende, and J. dos Santos, “Superpixel-based interactive classification of very high resolution images,” in *SIBGRAPI*, 2014, pp. 173–179.

[24] T. Mäenpää, “The local binary pattern approach to texture analysis - extensions and applications.” Ph.D. dissertation, Univ. of Oulu, 2003.

[25] T. Ojala, M. Pietikainen, and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *IEEE TPAMI*, vol. 24, no. 7, pp. 971–987, 2002.

[26] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, “Slic superpixels compared to state-of-the-art superpixel methods,” *IEEE TPAMI*, vol. 34, no. 11, pp. 2274–2282, 2012.

[27] P. Rauber, A. Falcao, T. Spina, and P. De Rezende, “Interactive segmentation by image foresting transform on superpixel graphs,” in *SIBGRAPI*, 2013, pp. 131–138.

[28] I. Gallo, A. Zamberletti, and L. Noce, “Interactive object class segmentation for mobile devices,” in *SIBGRAPI*, 2014, pp. 73–79.