A Flood Detection and Warning System Based on Video Content Analysis

Martin Joshua P. San Miguel and Conrado R. Ruiz Jr.(✉)

College of Computer Studies, De La Salle University, Manila, Philippines
conrado.ruiz@dlsu.edu.ph

Abstract. Floods are becoming more frequent and extreme due to climate change. Early detection is critical in providing a timely response to prevent damage to property and life. Previous methods for flood detection make use of specialized sensors or satellite imagery. In this paper, we propose a method for event detection based on video content analysis of feeds from surveillance cameras, which have become more common and readily available. Since these cameras are static, we can use image masks to identify regions of interest in the video where the flood would likely occur. We then perform background subtraction and then use image segmentation on the foreground region. The main features of the segment that we use to identify if it is a flooded region are: color, size and edge density. We use a probabilistic model of the color of the flood based on our set of collected flood images. We determine the size of the segment relative to the frame size as another indicator that it is flood since flooded regions tend to occupy a huge region of the frame. Finally, we perform a form of ripple detection by performing edge detection and using the edge density as a possible indicator for ripples and consequently flood. We then broadcast an SMS message after detecting a flood event consistently across multiple frames for a specified time period. Our results show that this simple technique can adequately detect floods in real-time.

1 Introduction

Flooding is a perennial problem in typhoon prone and coastal cities. Exacerbated by global climate change, weather has become more extreme and unpredictable making flash floods more frequent. Countries use a variety of weather forecasting systems to assist disaster prevention, relief and evacuation in order to drastically reduce the number of casualties and the amount of economic loss caused by disastrous weather conditions. These forecast systems however are normally based on predictions for a widespread region and require a long lead-time. At present, it is still not easy to achieve reliable accuracy for precise regional flood forecasting.

It is therefore important to detect disasters where they happen and in a timely manner. There have been numerous works on automatic disaster monitoring. However, very few focuses on specifically flood detection. In addition, most of the work that focus on flood use remote sensors or satellites, which are costly and require complicated decision systems.

G. Bebis et al. (Eds.): ISVC 2016, Part II, LNCS 10073, pp. 65–74, 2016.
DOI: 10.1007/978-3-319-50832-0_7

The study proposes a flood detection technique using video content analysis of surveillance camera feeds. This has several advantages. First, it would reduce cost since it does not necessitate purchasing expensive sensors or using satellite imagery. Second, it will also reduce labor cost because we instead utilize automatic event detection. Lastly, it is also easier to expand and modify the system to suit future requirements.

In our work, since we use static cameras, we already know where the flood would likely occur in a video frame. We use an image mask to focus on this region of interest. We then perform normalization and background subtraction using a background model specific to a particular camera and time of day. Consequently, we use image segmentation on the foreground region.

The main features of the image segment that we use to determine if it is a flooded region are: color, size and edge density. We use a probabilistic model of the color of the flood based on collected flood images from a specific camera. We determine the size of the segment relative to the frame size as another indicator that it is flood since flooded regions relatively larger that other objects in the frame. Finally, we obtain the edge density of the segment as a form of ripple detection. Our results show that these techniques can effectively detect floods in real-time.

2 Related Work

There have been several papers on automatic disaster monitoring. Most of these focus on snow, ice or fire detection [1–3]. Very few researches have focused specifically on flood detection. In addition, most of these works use remote sensors [4] and only a few use video processing analysis.

Most of the video or image-based methods for flood detection make use of satellite imagery [5–9]. These aerial images however have specific characteristics, which make them very different from images captured from CCTV cameras.

One work that detects flood in video sequences is the work of Borges et al. [10]. They proposed a method for retrieving flood content from newscast content. The features that they used were texture, the relation among color channels and saturation characteristics. Their approach analyzes the frame-to-frame differences of these features and used a Bayes classifier to determine the presence of flood. Their method can also be used for surveillance systems.

One paper that is specifically for surveillance systems is the work by Lai et al. [11]. They used real-time video processing to detect both fire and flood. For flood, the first feature they used is the color information and changes in the background. This is represented by histograms in HSV. The second feature they used was the spectral energy change or specific patterns of ripples due to the movement of water.

Our work also tries to identify ripples, however we use edge densities as a determining feature. We combine this with other features like size and color to provide better detection rates.

3 Flood Detection Algorithm

This section discusses the details of our algorithm for flood detection. Figure 1 shows the process flow of the entire system. We start by pre-processing every frame of the video. We use image masks on predetermined areas of the frame that the flood would mostly likely occur. We normalize the images before performing background subtraction. We then segment the image. After, we use a scoring system based on a set of features to distinguish between flood and non-flood objects. If a flood has consistently been detected in a series of frames, we proceed to the information dissemination module that sends out warning SMS messages. The subsections explain each process in more detail.

Fig. 1. Flood detection process flow.

3.1 Image Masking

Since the location and orientation of a CCTV camera is usually fixed, we can identify beforehand the regions in the camera's view where the flood would likely occur. This will minimize detecting other objects, i.e. people in the sidewalks, plants, trash cans and other objects. We set a binary image mask to the region of interest (ROI) and apply it to every frame in the video sequence captured by the camera (Fig. 2). This is preset for every camera based on the location of the camera and observed area. Alternatively, automatic image segmentation can be done on the ground plane to set the ROI.

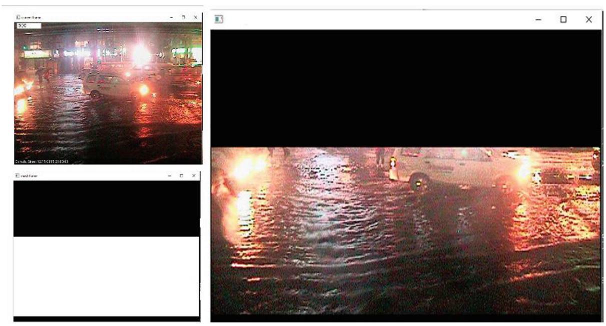

Fig. 2. Image mask for the region of interest.

3.2 Background Subtraction and Image Segmentation

Similarly, since we are using static cameras, we can come up with a good background model of the area under surveillance. After which, we can use background subtraction (using MOG2) to focus on the foreground region that is more likely to be the flood region (Fig. 3).

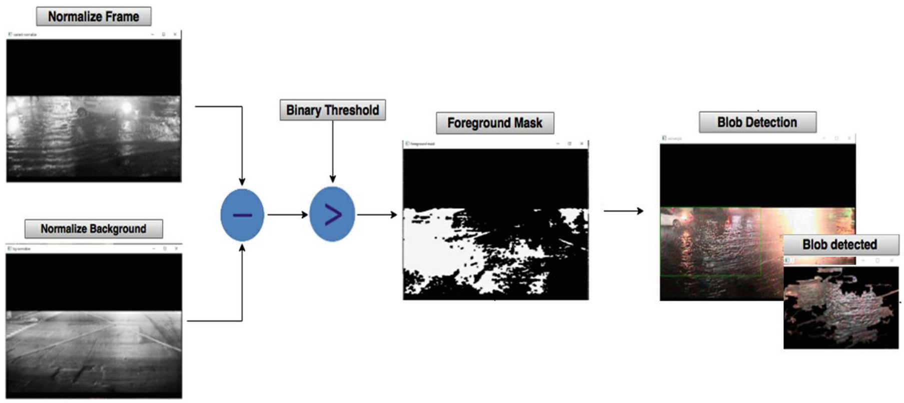

Fig. 3. Background subtraction and image segmentation.

We begin with normalizing the brightness of an image frame from the video captured. The system converts the frame into YUV color space, perform histogram equalization on the Y channel, and then convert back to RGB. We use a different background model for daytime feeds and another for night time videos based on the system clock. We then segment the image by using blob detection in OpenCV.

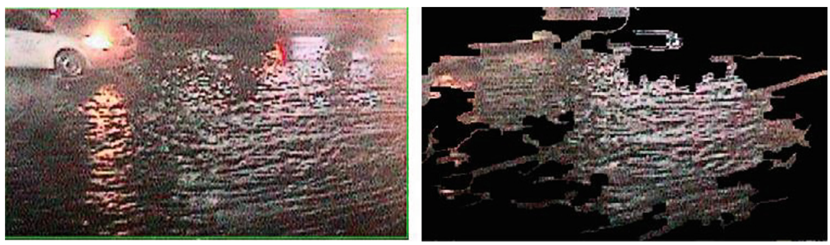

Fig. 4. (Left) Foreground region, (Right) Image segment of a potential flood.

3.3 Feature Extraction

We then extract features from the images segment. These features are as follows:

Size. In order to distinguish the image segments of floods with other objects, we consider the size of the segment, refer to Fig. 4. It can be observed that most flood image segments occupy a significant portion of the region of interest. While other objects like vehicles and people are relatively smaller in size. As such, we also do not consider small regions that do not meet a minimum segment size.

We compute the size based on the number of pixels in the image segment, P_{seg} over the number of pixels in the foreground region, P_{region}. It is given by

$$f_{size} = \frac{P_{seg}}{P_{region}}. \tag{1}$$

Color. We use a probabilistic model to identify the color features of flood images, similar to the work of Borges et al. [10]. We collect a set of flood training images for every camera. We have however observed that there is a significant discrepancy between night and day time frame captures. As such, we utilize the training set depending on the time of day, which can be easily obtained from CCTV cameras systems' historical recorded floods. Figure 5 show some training images used by our system.

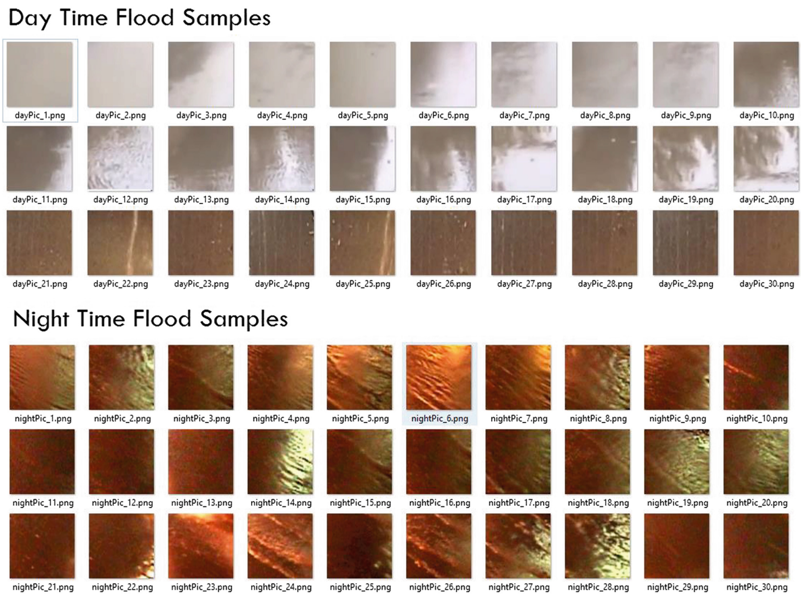

Fig. 5. Flood images training set.

Flood pixel $f(m, n)$ in an image f where f_B, f_G and f_R are the blue, green and red channels representation of f, respectively. Where f represents the images from the database. Let $\overline{f_B}$, $\overline{f_G}$ and $\overline{f_R}$ represent the blob average of the pixels in a flooded image region, for the blue, green and red channels, respectively.

Interpreting $\overline{f_B}$, $\overline{f_G}$ and $\overline{f_R}$ as variables and making use of the central limit theorem, the system employ a Gaussian model for these variables using the formula:

$$F(x|\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2)}{2\sigma^2}}, \tag{2}$$

such that $\overline{f_R} \sim N(\mu_{fR}, \sigma^2, f_R)$, $\overline{f_G} \sim N(\mu_{fG}, \sigma^2, f_G)$ and $\overline{f_B} \sim N(\mu_{fB}, \sigma^2, f_B)$. Based on these assumptions, a color based detection metric, f_{color} is given by:

$$f_{color} = \frac{DC_R + DC_G + DC_B}{3}, \tag{3}$$

where

$$\begin{aligned} DC_R &= \overline{f_R}(\overline{f_{Robs}})/\overline{f_R}(\mu f_R) \\ DC_G &= \overline{f_G}(\overline{f_{Gobs}})/\overline{f_G}(\mu f_G) \\ DC_B &= \overline{f_B}(\overline{f_{Bobs}})/\overline{f_B}(\mu f_B). \end{aligned} \tag{4}$$

In this case $\overline{f_{Robs}}$, $\overline{f_{Gobs}}$ and $\overline{f_{Bobs}}$ represents the average value in the red, green and blue channel of an observed region. If $\overline{f_R}$, $\overline{f_G}$ and $\overline{f_R}$ can be assumed independent, DC can be interpreted as the degree of confidence (represented by a probability) that a set of pixels represent a flood region (based only on color analysis).

Edge Density. Ripples can be visual indicators of the presence of flood. This is especially the case in urban areas, where cars, people or debris can cause ripples on the water. We however need a relative fast method for detecting ripples in real-time. In this paper, we employ canny edge detection and use the density of the edges as a possible characteristic for ripples, refer to Fig. 6.

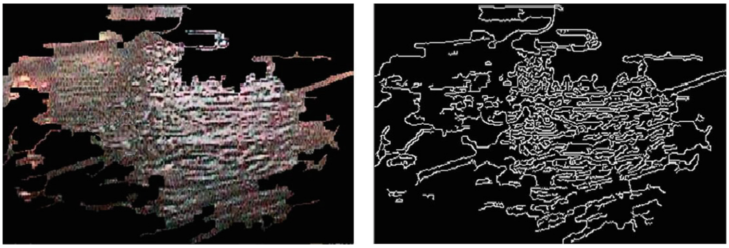

Fig. 6. (Left) Image segment of a potential flood, (Right) Edges detected.

We obtain the edge density by counting the edge pixels in the image segment, P_{edge} over the number of pixels in segment, P_{seg}. It is given by

$$f_{density} = \frac{P_{edge}}{P_{seg}}. \tag{5}$$

Feature Combination. The individual features alone are not sufficient to detect floods. The combination however provides us with a better discriminative descriptor. We combine these features using different weights into one confidence score and using thresholding to determine if it is a potential flood region. This is given by the following equation,

$$f_{comb} = \sum f_i * w_i, \tag{6}$$

where $i = 1, 2$ and 3 correspond to the features *size, color* and *edge density*. Based on our experiments, we have determined the best values for the weights. We set $w_1 = 0.20, w_2 = 0.20$ and $w_3 = 0.60$. Generally, edge density (our indicator for ripples) has the best discriminating power among the three features. We usually set the threshold, τ from 0.55 to 0.65.

3.4 Warning and Information Dissemination System

Detecting a flood in one frame will not automatically trigger a flood warning. It is only after the system consistently detects a flood for a certain period or number of frames that the system goes into the information dissemination phase (Fig. 7). We also have a tolerance of a few frames (5–6 frames) where some flood regions might not be detected. On the other hand, it is also possible that a proper low-pass filter might be more robust. We maintain a database of names, and contact information of those who will likely be affected by the flood, and send out a SMS message to all the recipients based on their location.

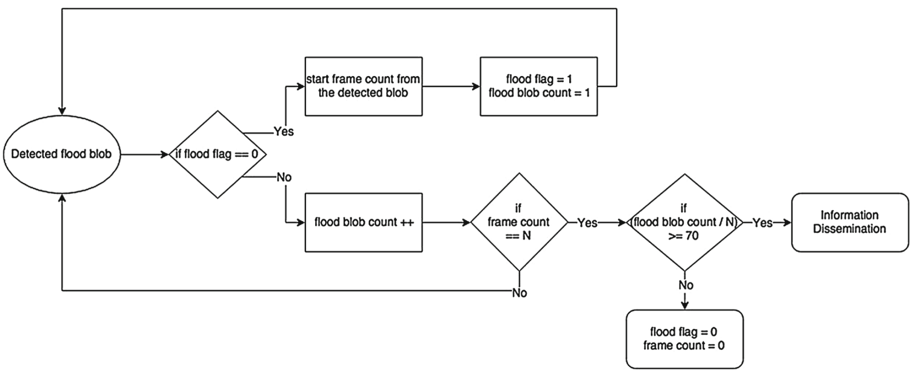

Fig. 7. Information dissemination flowchart.

4 Results

The system utilizes Visual Studio 2013 as an integrated development environment for C++ and the open source library, OpenCV. The system also uses

XAMPP, an open source cross-platform web server solution, to act as a local server for a MySQL relational database. We run our tests on a system with an Intel i7 processor and 8 GB of RAM. We are able to process the video that are 24–30 fps in real-time.

The videos we use are: the flooded visitor center of the Eden Project-UK [12], CCTV feed from Archer's Eye in Manila [13] and the CCTV feed of the flood at Madrid Metro [14]. Figure 8 shows some sample frames from these videos. These are public videos or accessible feeds available on the internet. Resolutions include 176 × 144, 320 × 240 up to 858 × 480.

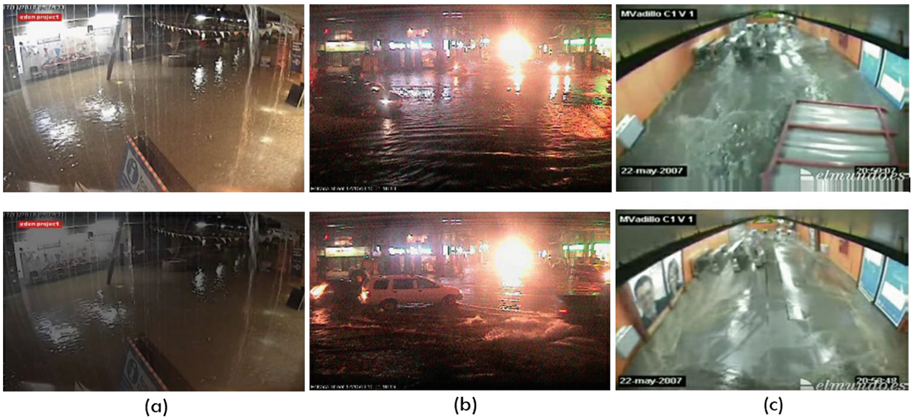

Fig. 8. Sample flood frames detected by the system: (a) Eden's Visitor Centre - UK [12], (b) Taft Ave. - Manila [13] and (c) Madrid Metro [14].

Table 1. Error rate table for individual feature.

	Size	Color	Edge density
True positive	79.36%	20.79%	83.10%
True negative	34.35%	84.91%	44.12%
False positive	65.65%	15.09%	55.88%
False negative	20.64%	79.21%	16.90%

First we try each feature individually to detect floods, Table 1 shows the individual error rates. Based on these results it is clear that edge density should be given the highest weight among all the features because it has the highest true positive rate among all the features while also having the lowest false negative rate. It also has an acceptable true negative rate and false positive rate. While size has a high true positive rate, it also has the highest false positive rate and even though the color feature has the lowest true positive rate but compared to

Table 2. Error rate table for frame flood detection.

	Eden's Visitor Centre -UK	Taft Ave. Manila	Madrid Metro
True positive	70.74%	65.91%	98.96%
True negative	30.64%	84.78%	24.60%
False positive	69.36%	15.22%	75.40%
False negative	29.26%	34.09%	1.04%

the other features, it has the lowest false positive rate which can also be used to minimize the error rate of the system in general.

We have systematically tried different combinations for the weights of the features. Based on our experiments, we have observed that the best values for the weights for size, color and edge density are 0.20, 0.20 and 0.80 respectively. Table 2 shows the error rates based on these weights and the threshold, τ to 0.60. Here we see that we achieve good true positive rates.

Although, there are still cases when there is a significant false negative value. This usually happens when the video is taken at night and there is a considerable amount of reflections on the surface of the water. Moreover, there are also some false positives, such as a frame with a big gray truck that occupies a significant portion of the frame. The segmented region would then be large in size, have similar color to flooded regions and contain a significant number of edges.

Nonetheless, although we encounter some errors in detection, these are in a frame-per-frame basis. The system in general, considers a set of consecutive frames for a given period of time and is usually still able to correctly detect floods.

5 Conclusion

In this paper, we have presented a straight forward approach for detecting flood using video analysis from static cameras. We employ standard computer vision techniques that are effective and efficient enough to be run in real-time for usual CCTV resolutions and frame rates. It generally has good flood detection capabilities, although there are some missed flood frames. This is usually caused by reflections on the flood that is currently not being modelled by the system.

It is therefore recommended to also consider the reflections on the water in future flood detection methods. This is especially a problem in urban areas. Currently, we also only use simple thresholding to distinguish between flood and non-flood segments. It is recommended to use more advanced methods on the features extracted, such as a Bayes classifier or neural networks.

We also consider frames independently of each other, it may also be useful to incorporate more video processing and consider frame-to-frame differences. It may also be interesting to use optic flow to model the movement of water or try to identify the level of the water. Incorporating other input, such as weather reports, into the warning system can also be explored in future work.

References

1. Rios-Gutirrez, F., Hasan, M.A.: Survey and evaluation of ice/snow detection technologies. Technical report, University of Minnesota, ITS Institute (2003)
2. Narasimhan, S.G., Nayar, S.K.: Interactive (de)weathering of an image using physical models. In: IEEE Workshop on Color and Photometric Methods in Computer Vision, France, vol. 6, no. 6.4, p. 1 (2003)
3. Liu, C.B., Ahuja, N.: Vision based fire detection. In: Proceedings of the 17th International Conference on Pattern Recognition, ICPR 2004, vol. 4. 134–137 (2004)
4. Chien, S., Cichy, B., Davies, A., Tran, D., Rabideau, G., Castano, R., Sherwood, R., Nghiem, S., Greeley, R., Doggett, T., Baker, V., Dohm, J., Ip, F., Mandl, D., Frye, S., Shuman, S., Ungar, S., Brakke, T., Ong, L., Descloitres, J., Jones, J., Grosvenor, S., Wright, R., Flynn, L., Harris, A., Brakenridge, R., Cacquard, S.: An autonomous earth observing sensorweb. In: IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC 2006), vol. 1, p. 8 (2006)
5. Martino, G.D., Iodice, A., Riccio, D., Ruello, G.: A novel approach for disaster monitoring: fractal models and tools. IEEE Trans. Geosci. Remote Sens. **45**, 1559–1570 (2007)
6. Yuhaniz, S., Vladimirova, T., Gleason, S.: An intelligent decision-making system for flood monitoring from space. In: ECSIS Symposium on Bio-inspired, Learning, and Intelligent Systems for Security, BLISS 2007, pp. 65–71 (2007)
7. Mason, D.C., Davenport, I.J., Neal, J.C., Schumann, G.J.P., Bates, P.D.: Near real-time flood detection in urban and rural areas using high-resolution synthetic aperture radar images. IEEE Trans. Geosci. Remote Sens. **50**, 3041–3052 (2012)
8. Zhou, Z., Tang, P., Zhang, Z.: A method for monitoring land-cover disturbance using satellite time series images. In: SPIE Asia Pacific Remote Sensing, International Society for Optics and Photonics, p. 926038 (2014)
9. Zhou, Z.G., Tang, P.: Continuous anomaly detection in satellite image time series based on Z-scores of season-trend model residuals. In: IEEE International Geoscience and Remote Sensing Symposium (2016)
10. Borges, P.V.K., Mayer, J., Izquierdo, E.: A probabilistic model for flood detection in video sequences. In: 2008 15th IEEE International Conference on Image Processing, pp. 13–16 (2008)
11. Lai, C.L., Yang, J.C., Chen, Y.H.: A real time video processing based surveillance system for early fire and flood detection. In: 2007 IEEE Instrumentation Measurement Technology Conference IMTC 2007, pp. 1–6 (2007)
12. YouTube: Live footage of Eden project flooding (2016). https://www.youtube.com/watch?v=L66tZN051IU
13. DLSU: Archer's eye (2016). http://archers-eye.dlsu.edu.ph
14. YouTube: Flooding on madrid metro. CCTV footage (2016). https://www.youtube.com/watch?v=zpvEmk_XvIc