AUTONOMOUS MONITORING SYSTEM FOR THE DETECTION OF DISRUPTING POWER CABLES ON DISTRIBUTION NETWORKS USING COMPUTER VISION TECHNIQUES

José Francisco Bianchi FILHO
LACTEC – Brazil
jose.bianchi@lactec.org.br

Ignacio Leonardo Del HOYO
UTFPR – Brazil
hoyo@alunos.utfpr.edu.br

Gerson Alcantara ANDRADE
COPEL - Brazil
gerson.andrade@copel.com

Guilherme Cordeiro VOGT
UFPR – Brazil
vogt@ufpr.br

Yan Victor MURMEL
UTFPR – Brazil
yan@alunos.utfpr.edu.br

Sebastião Ribeiro JÚNIOR
UFPR - Brazil
sebastiao@ufpr.br

Alan Naoto TABATA
LACTEC – Brazil
alan.tabata@lactec.org.br

Gabriel dos Santos HAVEROTH
LACTEC – Brazil
gabriel.haveroth@lactec.org.br

ABSTRACT

This paper aims on evaluating classical computer vision techniques in order to detect whether electric cables are falling to the ground or not. A proprietary dataset containing several simulation cases was built with the Unreal Engine 4 framework and simulations were performed on MATLAB with an algorithm that performs frame subtraction and applies filters in order to detect movement on images. Evaluated metrics were precision, recall, F1 score and Frames Per Second (FPS) processed. The algorithm performed satisfactorily on most cases, being able to achieve a precision of up to 100.00%, 26.67% recall, 0.4058 F1 score and up to11 FPS.

Index Terms – high impedance fault, computer vision, distribution network, matlab, video analysis.

I INTRODUCTION

Faults on the distribution network are recurrent problems, which can lead to shortage of energy to consumers and provide hazardous sites to the population. One particular type of fault is when cables fall to the ground, producing high impedance faults, which presents little current circulation, consequently being difficult to locate by the electric utility [1]. In order to assess this problem, the utility depends on consumer notification, which in turn leads to slowing down the whole process of acknowledging and fixing the problem.

There are a plethora of ways to solve this problem using electric protection equipment, such as current transformers adjusted specifically for this problem [2]. However, these tend not to be usually widespread installed around the distribution grid. Therefore, usage of a more generic sensor, such as visible cameras, seem an attractive alternative [3] since they are capable of detecting a wide range of visible faults and are commonly found on urban environments.

Analysis of visible cameras image can be either manual, semi-automatic or automatic. When manual, an operator must be constantly visualizing and analyzing the images. This, however, turns out to be costly, time consuming and boring, thus being more prone to human error. When semi-automatic, there is an algorithm that analyzes images and can detect or set a flag up before handing them to a human operator to set the final decision. When automatic, then the algorithm autonomously analyzes the images and takes the decision in place of the human operator. However, since an automatic operation must be ideally flawless, then this paper will focus on the semi-automatic analysis.

Analysis of the image can be performed with classical computer vision techniques, artificial intelligence-based techniques or a mixture of both. As a first step of a bigger project, on this paper classical computer vision techniques were deployed and evaluated, so that an algorithm is able to detect falling cables by observing a sequence of images.

On section II an overview on computer vision techniques is introduced. Afterwards, on section III the workflow on this paper's project is presented, detailing the methodology, dataset and evaluation metrics. On section IV simulation results and the discussion around it is presented. Lastly, on section V this paper is concluded.

II COMPUTER VISION

Computer vision is a field of study that covers a group of techniques and for extraction of features from images or sequence of images and it is analysis inside of a computer environment. This science uses a sensor-derived image to perform the description or modelling of the real physical environment [4].

In this article, standard computational techniques used in digital video and image processing are applied. These are the binarization thresholding, morphological operations, frames subtraction, and temporal filtering.

The thresholding binarization consists in a classical

technique that is currently used in the process of segmentation. This process transforms a gray-level image into a binary image. Basically, the output of the thresholding is a binary image in which objects appear in a black and the background in white color or vice versa. [5]. The process of thresholding is considered computationally inexpensive and fast. It is the oldest segmentation method applied in simple applications that can be implemented using specific hardware devices or computers [6].

Morphological operations apply algebraic nonlinear operations in images in order to perform tasks such as image pre-processing, structure enhancement, segmentation and quantitative description of objects. There are four basic morphological operations: dilation, erosion, closing and opening [6]. The dilation expands objects and is able to removes "salt" noise that is present on the image, this operation can also be used to remove cracks in the objects of the image. Differently from the dilation, erosion removes "pepper" noise and suppresses thin objects in the image [5].

Other morphological operations are derived from a combination of erosion and dilation operations. Opening is the process of applying dilation on an image that is eroded, making the effect of opening gaps in the image. The opposite process, the closing operation, has the effect of closing gaps within objects [6] [7]. On the context of power lines monitoring and surveillance through computer vision techniques, [8] applied an erosion operation as a step of an algorithm to detect power line problems.

The frame subtraction is a video processing method in which frames on different timestamps are subtracted to perform motion tracking on the video, where the differential motion analysis method assumes a stationary camera position [6]. [9] introduces this method as a highly adaptive one to an application where the environment is dynamic, allowing for less computational cost. [10] also addresses this method to detect the motion of targets on a sequence of video frames.

The temporal filter application consists in using interrelation in frames from the video to suppress noise on a sequence of frames. To apply this method, noise must not be derived from correlation interframes, but must be from other sources, whereas consecutive frames must have a strong correlation. The simplest temporal filter is the Temporal Arithmetic Mean Filter, which averages individual pixels between successive video frames [11]. Another employed technique is a temporal filter that uses a median operation between frames. [12] proposed the use of the median operation technique for noise reduction on a gray-scale image sequence.

III WORKFLOW

On this section, an overview of the dataset, methodology and evaluation metrics are presented.

A. Dataset

The dataset was built on top of Unreal Engine's simulation engine [13], and a general structure of the data is presented on Table 1. Each scenario of the dataset is compounded by a combination of one element from each column presented on Table 1, where ambient indicates how is light affecting the ambient, Camera angle the camera's viewing angle in relation to the fault and Phase indicates which cable is at fault. For each case, there is a total of 80 frames, with the first 30 presenting normal conditioning and the subsequently 50 the cable's fall. Sample images from each ambient are presented on Figure 1.

Ambient	Phase	Camera angle
Sun on zenith	A	Top
Afternoon sun	B	Mid
Late afternoon sun	C	
Mist		
Rain		
Night		

Table 1 - Ambient types

Figure 1 - Sample images from the dataset

B. Methodology and algorithm

The methodology's flowchart is presented on Figure 2. First, a dataset must be arranged. This step can be performed by either collecting samples from open databases or by creating them. Afterwards, a labeled dataset is obtained and validated. Then, an object detection algorithm is applied on the dataset, generating bounding boxes on the detected objects. Finally, the bounding boxes are evaluated against the ground truth.

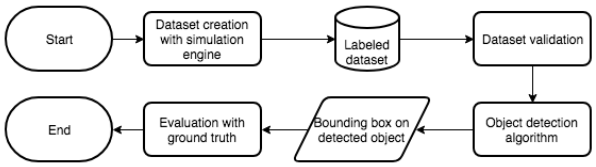

Figure 2 - Methodology flowchart

The object detection algorithm architecture is presented on Figure 3 and Figure 4. The algorithm's main idea is to collect video and apply subtraction between two frames in order to detect movement on the scene. Thus, two separate modules that gather a fixed number of frames, with a frame offset interval between them, are built. On each module, a moving average technique is applied between the frames collected so that small noises are attenuated. This way, there is a potential tradeoff when changing buffer size (B). When it is increased, more noise is attenuated, however more information is lost. The same way, when it's smaller, then less noise is attenuated, however more information is kept.

Afterwards, opening morphological operation is performed on the subtracted image to eliminate noise and recover image information. Next, a bounding box containing every relevant pixel is proposed. Finally, the proposed bounding box is evaluated against the ground truth and performance metrics are calculated.

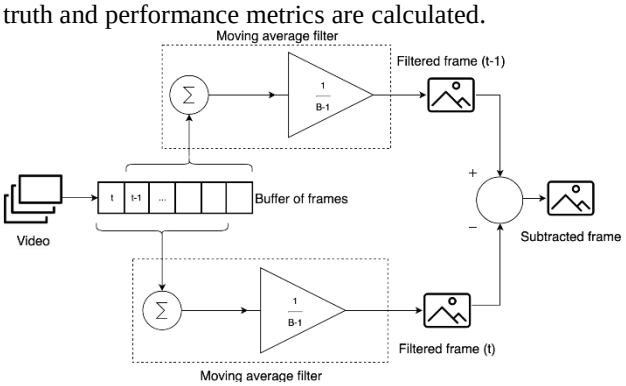

Figure 3 - Object detection algorithm architecture – part 1

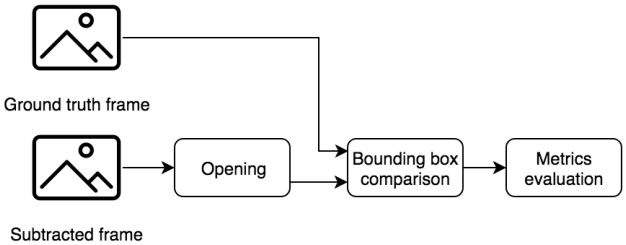

Figure 4 - Object detection algorithm architecture – part 2

C. Evaluation metrics

In order to evaluate algorithm performance, evaluation metrics are needed. The common metrics on object detection are precision, recall, F1 score, Intersection over Union (IoU).Precision and Recall are computed by Eqs. (1) and (2) where they measure, respectively, how many of the detected occurrences were correct and how many of the total occurrences were detected, where:

- TP are the true positives;
- FP are the false positives and
- FN are the false negatives.

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

F1 score is an index capable of aggregating both precision and recall into a single equation, and is presented on Eq. (3).

$$F1\ score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (3)$$

IoU is an index that measures how well does a proposed bounding box matches with the ground truth bounding box, and is presented on Eq. (4). The closer its value to 1, then the more both boxes fit; the closer to 0, then the less they fit.

$$IoU = \frac{Area\ of\ intersection}{Area\ of\ union} \quad (4)$$

Lastly, algorithm speed performance is going to be evaluated by calculating how many frames per second (FPS) are able to be processed.

IV RESULTS AND DISCUSSION

It is relevant to note that the bounding box object detection method was chosen in place of pixel segmentation due to simulation restrictions. When generating the dataset for a specific case and its ground truth counterpart, noise was generated due to randomness associated with physical effects such as wind force. This way, the ground truth turned to be slightly different from the correspondent scenario. Hence, choosing a more generous detection method makes it fairer on evaluation of the algorithm. Table 2 presents the algorithm's hyperparameters.

Buffer size	9
Filter type	Mean
Binary threshold	7
Morphological opening element	5x5 size with every pixel being active

Table 2 - Algorithm hyperparameters

Also, in order to simplify simulation results presentation, simulation cases were grouped with each case containing every possible combination of phase and camera angle, thus being:

- Case 1 – Sun on zenith;
- Case 2 – Afternoon sun;
- Case 3 – Late afternoon sun;
- Case 4 – Mist;
- Case 5 – Rain;
- Case 6 – Night.

On Figure 5 the result of the subtraction and opening operations when a fault is occurring is presented for the Case 1. The frame evaluated, the ground truth and the algorithm result are presented respectively. Black pixels represent the background and white pixels are the pixels with relevant information.

Figure 5 - Subtraction algorithm result

On Table 3 the algorithm results are presented for the proposed cases. Notably, highest precision values occur on cases 4 and 6. However, on case 6 the algorithm is more reluctant to predict bounding boxes, and thus presents the lowest recall value. This difficulty is understood when one

observes the ambient lighting, being predominantly back and with the cable being hidden by it. Case 2 presented the most balanced results, with a F1 score of 0.4058. On Figure 6 a few simulation results are shown, with the green bounding box being the ground truth and the red bounding box the one predicted by the algorithm.

	Precision (%)	Recall (%)	F1 Score
Case 1	93.67	23.49	0.3756
Case 2	84.85	**26.67**	**0.4058**
Case 3	97.50	13.00	0.2294
Case 4	**100.00**	24.44	0.3929
Case 5	34.25	19.68	0.2500
Case 6	**100.00**	1.59	0.0312

Table 3 – Algorithm results for each case

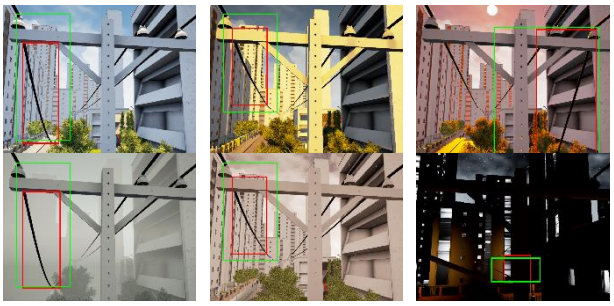

Figure 6 - Simulation results

In order to evaluate the impact of each hyperparameter of the algorithm, a grid search-like method on top of the fine-tuned model was employed, with the results being presented next. Algorithm results when performing adjustment on binarization threshold is presented on Table 4 and Table 5. It its evident that higher levels of thresholding lead to a higher precision but lower recall levels, whereas lower levels incur in lower precision but higher recall levels. This occurs because the binary threshold level acts as a pixel filter, where high levels eliminate more noise with the cost of also losing relevant information and low levels eliminate less noise, but retains more relevant information.

	Precision (%)	Recall (%)	F1 Score
Case 1	**100.00**	16.19	0.3110
Case 2	93.51	**22.86**	**0.3674**
Case 3	**100.00**	16.83	0.2880
Case 4	35.07	16.19	0.2247
Case 5	0.00	0.00	0.0000
Case 6			

Table 4 – Algorithm results for each case when binary threshold is set to 9

	Precision (%)	Recall (%)	F1 Score
Case 1	81.68	33.97	0.4798
Case 2	72.41	40.00	0.5153
Case 3	50.00	16.00	0.2424
Case 4	**100.00**	**40.64**	**0.5383**
Case 5	33.79	23.49	0.2772
Case 6	55.56	3.17	0.0601

Table 5 – Algorithm results for each case when binary threshold is set to 5

On Table 6 a performance evaluation when changing the filter type to a median one is presented. In order to achieve best results, buffer size was changed to 3 and the binary threshold to 9. When comparing the mean and median filters, it's perceptible that the mean filter presents better performance in almost every metric; on the downside, this algorithm presents low recall values in every case.

	Precision (%)	Recall (%)	F1 Score
Case 1	76.0	6.03	0.1118
Case 2	71.4	4.76	0.0893
Case 3	58.82	3.17	0.0602
Case 4	**86.36**	6.03	0.1128
Case 5	68.57	**7.62**	**0.1371**
Case 6	2.50	0.63	0.1058

Table 6 – Algorithm results for each case when filter type is set to median

Regarding FPS processed, even though the algorithm was built on top of MATLAB language, a notably high-level language, running the code on an Intel Core i7-7700K with 16 GB RAM up to 11 FPS, with image pixel resolution of 1920x1080.

V CONCLUSION

An object detection algorithm and a simulation environment capable of generating dataset were successfully developed, with the algorithm being able to accurately detect cable faults on a simulated environment. The tested model was capable of achieving up to a 100.00% precision, 26.67% recall and 0.4058 F1 score at 11 FPS. It's important to note that the even tough recall were low the algorithm's parameters were fine-tuned with the intent on achieving the highest precision so that false alarm occurrence is minimized.

In addition to that, it is notable that changing the binarization threshold changes how rigorous is the filter to noise, meaning there is an apparent trade-off between noise reduction and information loss.

Also, the algorithm doesn't work ideally when the ambient is night or rain, suggesting that, if deployed, an auxiliary system should be ideally used in order to take over the device and avoid false alarms.

For future works, the authors propose developing a similar code in a lower level language in order to speed up FPS and applying deep convolutional neural networks together with classical computer vision techniques in order to improve performance metrics.

ACKNOWLEDGEMENTS

The authors would like to thank the power utility COPEL that through the Brazilian Electricity Regulatory Agency (ANEEL) R&D program, Project PD 02866-0408/2014, funded this project.

REFERENCES

[1] Neto, P. and de Arruda, R., 2005. Sistemas para detecção de falta de alta impedancia e de rompimento de condutores em redes de distribuição de energia elétrica (portuguese).

[2] RUSH, P., 2011. *Proteção e Automação de Redes-Conceito e Aplicação*, São Paulo, Brazil (portuguese).

[3] Thananunsophon, K., Mangalabruks, B., Fujii, Y. and Yupapin, P.P., 2011, "Community monitoring and security using an intelligent camera in EAT smart grids", Procedia Engineering, vol.8, 332-336.

[4] Fischler, M.A. and Firschein, O. eds., 2014, principles, and paradigms, *Readings in computer vision: issues, problem*, Elsevier.

[5] Davies, E.R., 2004. *Machine vision: theory, algorithms, practicalities*. Elsevier, 31-125.

[6] Sonka, M., Hlavac, V. and Boyle, R., 2014, *Image processing, analysis, and machine vision*, Cengage Learning, 176-230.

[7] Soille, P., 2013, *Morphological image analysis: principles and applications*, Springer Science & Business Media, 105-108.

[8] Sharma, H., Bhujade, R., Adithya, V. and Balamuralidhar, P., 2014, "Vision-based detection of power distribution lines in complex remote surroundings", Communications (NCC) 2014 Twentieth National Conference, 1-6.

[9] Wei, S., Chen, Z., Li, M. and Zhuo, L., 2009, "An improved method of motion detection based on temporal difference", Intelligent Systems and Applications, 2009. ISA 2009, International Workshop IEE, 1-4.

[10] Xia, Y., Ning, S. and Shen, H., 2010, "Moving targets detection algorithm based on background subtraction and frames subtraction", Industrial Mechatronics and Automation (ICIMA), 2010 2nd International Conference IEEE, vol.1, 122-125.

[11] Szczepanski, M., 2011, *Computer Recognition Systems*, Springer, Berlin, 421-430.

[12] Viero, T. and Neuvo, Y., 1991, "Non-moving regions preserving median filters for image sequence filtering", IEEE International Conference on Systems Engineering, 245-248.

[13] Games, E., 2007. *Unreal engine*. Online: https://www. unrealengine. com.