Journal of Multimedia Information System VOL. 6, NO. 4, December 2019 (pp. 203-208): ISSN 2383-7632 (Online)
http://doi.org/10.33851/JMIS.2019.6.4.203

Fast ROI Detection for Speed up in a CNN based Object Detection

Jin-Sung Kim[1], Youhak Lee[2], Kyujoong Lee[1], Hyuk-Jae Lee[3*]

Abstract

Fast operation of a CNN based object detection is important in many application areas. It is an efficient approach to reduce the size of an input image. However, it is difficult to find an area that includes a target object with minimal computation. This paper proposes a ROI detection method that is fast and robust to noise. The proposed method is not affected by a flicker line noise that is a kind of aliasing between camera and LED light. Fast operation is achieved by using down-sampling efficiently. The accuracy of the proposed ROI detection method is 92.5% and the operation time for a frame with a resolution of 640 x 360 is 0.388msec.

Key Words: ROI detection, Noise, Image difference, Object detection.

I. INTRODUCTION

Object detection is widely used for various applications. Recently, convolutional neural network (CNN) based object detection methods give high performance, which increases the possibility of utilization of object detection in many application areas such as autonomous vehicles, unmanned stores, and smart homes. In these applications, object detections in multiple sensor inputs are required. In order to operate in real-time, fast operation is needed because multiple inputs should be processed in time. However, the operation time of CNN based object detection methods is not enough to process multiple inputs in real-time.

There have been many region based CNNs for object detection and localization. Faster R-CNN employs a region based CNN [1], and there are many variations of Faster R-CNN [2, 3]. There are single stage CNNs such as SSD [4] and YOLO [5].

In object detection, there are many noise and distortion in input image, which results in accuracy degradation. When LED illumination is used for a system, there is invisible flicker in LED light. However, the flicker rate of LED affects images captured by a camera. When the flicker rate is similar to the frequency of line scan in camera, aliasing effect degrades the captured image, and the accuracy of image analysis deteriorates because of the flicker line noise.

In order to enhance the operation speed, region of interest (ROI) which is a candidate area including a target object is detected, and only ROI is used for an input for a detection network. ROI is defined by analyzing the difference between a current frame and a reference frame. In this paper, the noise effect caused by the LED flicker is removed by using minimum and maximum filtering. The proposed ROI detection method processes down-sampled image which reduces operation time of ROI detection.

This paper is organized as follows. In Section II, the problem that is addressed in this paper is shown, and the proposed method is presented. Section III concludes this paper.

II. FAST ROI DETECTION FOR CNN-BASED OBJECT DETECTION

In order to enhance the operation speed of a CNN based object detection method, this paper proposes a method for detecting ROI. When a CNN only processes ROI instead of

Manuscript received December 10, 2019; Revised December 20, 2019; Accepted December 22, 2019. (ID No. JMIS-19M-12-056)

Corresponding Author (*): Hyuk-Jae Lee, 82-2-880-1825, hjlee@capp.snu.ac.kr

[1]Department of Electronic Engineering, Sun Moon University, Asan, Korea, jinsungk@sunmoon.ac.kr

[2]Computer vision team, Chips&Media Inc., Seoul, South Korea, jeremy.lee@chipsnmedia.com

[3]Department of Electrical and Computer Engineering, Seoul National University, Seoul, Korea, hjlee@capp.snu.ac.kr

a whole input image, the processing time will decrease because the resolution of an input image is reduced. To detect ROI, image difference between a current frame and a reference frame is used. In this paper, input image noise caused by the LED flicker is considered. This noise is a kind of aliasing and the position of the noise in an image is not fixed. Thus, the LED flicker noise is an obstacle to obtaining ROI by using image difference. This paper proposes a method for removing the noise effect. Since the objective of the proposed method is speed up, the operation speed of the proposed ROI detection method is reduced.

The target system of this paper is the object detection in an unmanned store with LED light and multiple cameras. YOLO v3 is used for the object detection network.

2.1. Difference Image and Noise Effect

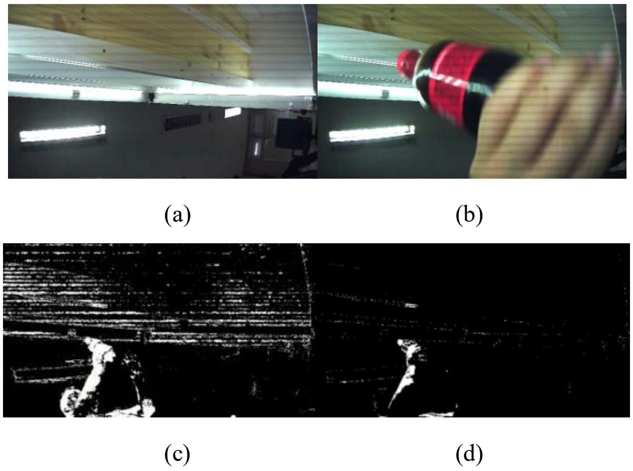
Target object in an unmanned store tends to be a moving object because of purchasing action. Therefore, this object can be included in an image area with variation of pixel value. In this paper, ROI that includes a target object is obtained by using a difference image between a current frame and a reference frame. The reference frame is obtained when there is no purchase action and no human. When pixel value difference between co-located pixels of a reference frame and a current frame is higher than a predefined threshold, the pixel is determined as a pixel of ROI.

Most of lights including LED flickers with high frequency. The flicker cannot be perceived by a human, but it has an effect on a captured image. Figs.1 (a) and (b) show images with the LED flicker noise, which are horizontal dark lines. The position of a noise line moves frame by frame, so the dark line affects the difference image. Fig. 1 (c) shows a difference image, in which horizontal noise lines can be included in ROI. Fig. 1 (d) shows a difference image with a higher threshold, in which most of pixels on a target object are removed but horizontal noise lines are not removed completely. Therefore, the LED flicker noise deteriorates the performance of ROI detection method by using difference images.

2.2. Proposed ROI Detection Method

2.2.1. Min-Max Filtering

In order to reduce the computation time of the ROI detection, an input image is down-sampled. The vertical distance between the flicker noise lines is 7 or 8 pixels. In this paper, 10x10 pixels in an input image is down-sampled into a single pixel, so a 10x10 block includes at least one flicker noise line in an input image. There are two down-sampled images, *fmax* and *fmin*. A pixel value in *fmax* is set to maximum value in a 10x10 block while that of *fmin* is set to minimum one. When the size of a block is smaller than the distance between noise lines, *fmin* and *fmax* images are affected by the flicker noise line. Recall that the flicker noise line moves because it is a kind of aliasing. Consider a case when a block includes the noise at some time and it does not have the noise at the other time. In this case, *fmin* and/or *fmax* of this block changes due to the position of the flicker noise line even when the image in the block does not change.


(a) (b)

(c) (d)

Fig. 1. Enlarged images with the LED flicker noise and difference images with low and high thresholds; (a) a reference frame, (b) a current frame, (c) a difference image with a low threshold, (d) a difference image with a high threshold.

2.2.2. Fast Generation of a Difference Image

The difference image is obtained by comparing *fmax* and *fmin* of a reference frame and a current frame. The horizontal and vertical resolutions of the difference image is reduced by 10, respectively. The difference image value of a position (x, y) is given by (1).

$$\mathrm{D(x,y)} = \begin{cases} 1, f_{max}(x,y) > Th \; or \; f_{min}(x,y) > Th, \\ 0, f_{max}(x,y) < Th \; and \; f_{min}(x,y) < Th. \end{cases} \qquad (1)$$

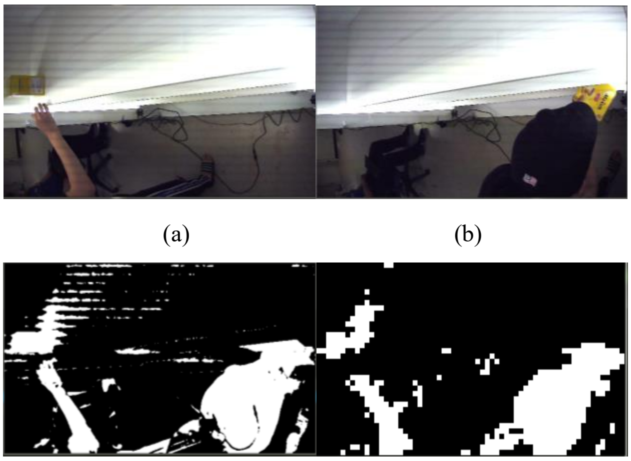
Using *fmax* and *fmin* is less affected by the flicker noise compared with using average or median value. Average value is affected by the number of the noise line included in a 10x10 block. Furthermore, average and median filters cannot find the difference caused by the appearance of a target object because the pixel value of a target object can be filtered out. The pixel value included in the noise line is reduced as shown in Fig. 1. Thus, the chance is very low that *fmax* is affected by the noise. D(x, y) in (1) can be determined falsely only when minimum pixel value in an input image is included in the noise line, but the portion of this case is not large.

Fig. 2 shows an example which presents difference images obtained by using median filter and the proposed filter. Figs. 2 (a) and (b) show a reference frame and a current frame, respectively. The yellow object in Fig. 2 (b) is a target object. In this case, the yellow object and hand in Fig. 2 (a) are included in a difference image. Fig. 2 (c) is a difference image when a reference image and a current image are filtered by median filter with a 5x5 kernel. In this figure, the flicker line noise remains, and an unnecessary area will be included in ROI. Fig. 2 (d) shows a difference image obtained by using the proposed *fmax* and *fmin*. This figure shows that the flicker line noise is removed. The computation time of the median filter is 7msec while that of the proposed method is 5msec. Note that the resolution of the proposed method after *fmax* and *fmin* filtering is reduced by 10 in horizontal and vertical directions.

After obtaining difference image value by using (1), an isolated ‘1’ that has no neighbor pixel with ‘1’ is removed because this small object cannot be a target object. After removing isolated pixels, the difference image is down-sampled by four in horizontal and vertical directions. Fig. 3 shows a final image in which small areas are removed.

2.2.3. ROI Generation

ROI is a candidate area that may include a target object. In this paper, a difference image filtered by the proposed method is used for generating ROI. For example, a pixel with ‘1’ in Fig. 3 (b) is determined to be included ROI. A single pixel in Fig. 3 (b) corresponds to a 40 x 40 block in an input image. Thus, ROI in this paper is a group of 40 x 40 blocks.


(a) (b)

(c) (d)

Fig. 2. Comparison of difference images obtained by using a median filtering and the proposed filtering; (a) a reference frame, (b) a current frame, (c) a difference image when median filter is used (flicker line noise affects a difference image), (d) a difference image when the proposed filter is used (the effect of flicker line noise is removed).

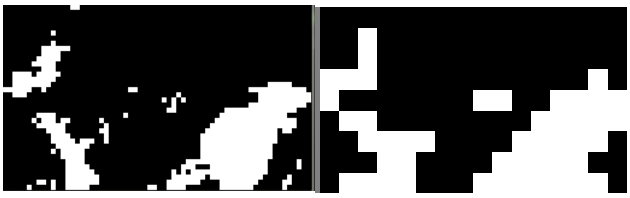

(a) (b)

Fig. 3. Final detected ROI after removing small areas in a difference image; (a) a difference image filtered by the proposed method, (b) a final image of ROI detection after a small area is removed. (A pixel corresponds to a 40x40 block in an original image.)

p(x-1, y-1)	*p*(x, y-1)	*p*(x+1, y-1)
p(x-1, y)	*p*(x, y)	*p*(x+1, y)
p(x-1, y+1)	*p*(x, y+1)	*p*(x+1, y+1)

(a)

c(x-1, y-1)	*c*(x, y-1)	*c*(x+1, y-1)
c(x-1, y)	*c*(x, y)	

(b)

Fig. 4. Pixels which have an effect on the determination of a threshold for c(x, y); (a) temporally neighboring pixels for c(x, y), (b) spatially neighboring pixels for c(x, y).

2.2.4. Adaptive Threshold

The accuracy of a ROI detection is affected by illumination, appearance of a target object, and background. In order to improve the accuracy, the threshold in (1) is defined adaptively. In this paper, the threshold is lowered when a spatial and temporal neighbor pixel is determined as ‘1’ in (1). The reason is that pixels of a target object are located contiguously and those appears in a similar location in consecutive frames.

Consider a pixel of which the position is (x, y). Let c(x, y) denote D(x, y) in a current frame. p(x, y) represents the value of D(x, y) in a previous frame. Fig. 4 (a) shows difference values in a previous frame, which affect the threshold for c(x, y) in a current frame. Among these values, p(x, y) that is a co-located pixel for c(x, y) has the largest effect on the threshold for c(x, y). Fig. 4 (b) shows neighboring pixels which have an influence on the threshold for c(x, y) when pixels are processed in a raster scan order.

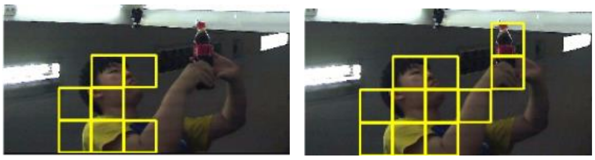
The effect of neighboring pixels on a threshold of c(x, y) is estimated by using the difference image value in (1). val_T represents the effect of temporally neighboring pixels in Fig. 4 (a), and val_S represents that of spatially neighboring pixels in Fig. 4 (b). Then, val_{TH} is defined as the sum of val_T and val_S.

$$val_T = \left(\sum_{m,n=-1,0,1} p(x+m, y+n) + \left(11 \cdot p(x,y)\right) \right) / 4,$$

$$val_S = \sum_{k=-1,0,1} c(x+k, y-1) + c(x-1, y).$$

The threshold in (1) is determined by using the following pseudo-code:

```
if ($val_{TH}$ >= 4), $Th = 15$
else if ($val_{TH}$ >= 1), $Th = 30$
else $Th = 50$.
```



(a) (b)

Fig. 5. Accuracy improvement by using an adaptive threshold; (a) ROI obtained by using a fixed threshold, (b) ROI obtained by using a spatiotemporally adaptive threshold.

Fig. 5 show an example of the effect of the adaptive threshold. In this figure, a yellow square represents ROI. Fig. 5 (a) shows that a target object is not included in ROI. In Fig. 5 (b), the threshold is lowered when a spatial and temporal neighbor block is determined as a ROI. In this figure, the target object is included in ROI.

2.3. Flow of the Proposed Method

Fig. 6 shows the operation flow of the proposed method. *fmax* and *fmin* of a reference frame are calculated and stored. Then, *fmax* and *fmin* of an input frame are calculated. A 10 x 10 block in an input image corresponds to a pixel in *fmax* and *fmin*. In order to enhance the operation speed, only four pixels in a 10 x 10 block are investigated for determining the maximum and minimum values for *fmax* and *fmin*. Then, *fmax* and *fmin* of an input frame are compared with *fmax* and *fmin* of a reference frame. The difference is compared with an adaptive threshold. Then, a small area is removed by the proposed filter, and this image is used for determining ROI.

Fig. 7 shows an example of the proposed method. Figs. 7 (a) and (b) are a reference frame and an input frame, respectively. Fig. 7 (c) shows a difference image after adaptive filtering. Fig. 7 (d) represents the final image in which a pixel corresponds to a 40 x 40 block of ROI in an input image.

III. PERFORMANCE EVALUATION

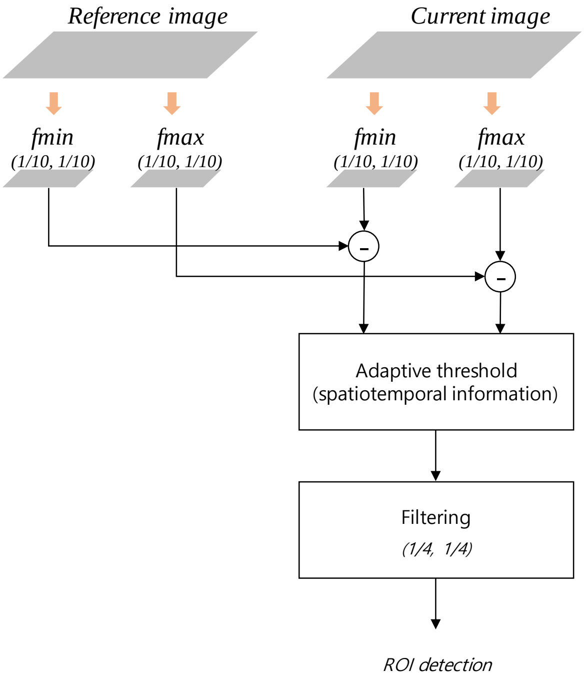
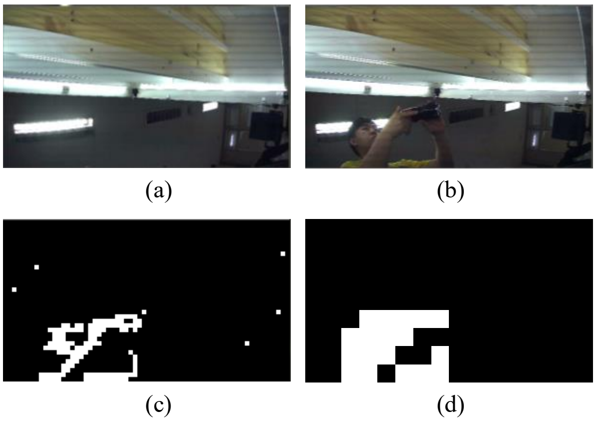


Fig. 6. Operation flow of the proposed method.



(a) (b) (c) (d)

Fig. 7. An example of the proposed ROI detection method; (a) a reference image (640 x 360), (b) a current image, (c) a difference image with an adaptive threshold (64 x 36), (d) a final image (16 x 9)

Table 1. Accuracy of the proposed ROI detection.

The number of frames	The number of target objects	The number of true detections
1,989	1,261	1,167

The accuracy and operation speed of the proposed method are evaluated. In the experiments, CPU is i7-3930K and 32G RAM is used. A video sequence with 1,989 frames is generated. When a target object is included in ROI, the detection is determined as true detection. Table 1 shows the experimental results. In the test sequence, a target object appears in 1,261 frames, and the number of true detections is 1,167. In this paper, true detection means that a target object is included in the detected ROI. The detection accuracy is defined as the ratio of the number of true detections to that of target objects in the test sequence. The detection accuracy of the proposed method is 92.5%.

The operation speed is evaluated with 640 x 360 images. When a difference image is obtained by subtracting each pixel value simply and thresholding is performed, it takes 3.153msec for a frame. The operation time for a frame in the proposed method is 0.388msec. In the proposed method, only 1/16 pixels in an input image are used for *fmax* and *fmin* filtering. The accuracy of the proposed method is 92.5% even when there is the flicker line noise. Recall that the flicker noise cannot be removed by a median filter as shown in Fig. 2.

Table 2. Comparison of the operation speed.

Method	Operation time (msec/frame)
Simple difference image	3.153
The proposed method	0.388

The proposed method is used for generating ROI input for YOLO v3 [6]. When the performance of this work is compared with that of the original YOLO v3 without ROI detection, the operation speed is improved by 3.29 times while the accuracy degradation is 2.81%.

III. CONCLUSION

The operation speed of a CNN based object detection method is important in many application area. This paper proposes a method for detecting ROI that includes a target object. The size of an input image can be reduced by using ROI, which can improve the operation speed of a CNN. In this paper, a flicker line noise that is a kind of aliasing between camera and LED light is considered. A simple method that uses maximum and minimum values addresses the flicker line noise. The operation speed of the proposed method is very fast by using data sampling. Experimental results show that the ROI detection accuracy is 92.5% while the operation time for a frame with a resolution of 640 x 360 is 0.388msec.

REFERENCES

[1] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, issue 6, pp. 1137-1149, June 2016.

[2] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proceeding of International Conference on Neural Information Processing Systems*, 2016.

[3] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S.Belongie, "Feature pyramid networks for object detection," in *Proceeding of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

[4] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed, "SSD: Single shot multibox detector," in *Proceeding of European Conference on Computer Vision*, 2016.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceeding of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016.

[6] Youhak Lee, Chulhee Lee, Jin-Sung Kim, and Hyuk-Jae Lee, “Fast Detection of Objects Using a YOLOv3 Network for a Vending Machine,” in *IEEE International Conference on Artificial Intelligence Circuits and Systems(AICAS)*, Mar. 2019.

Authors

Jin-Sung Kim received B.S., M.S., and Ph.D. degrees in electrical engineering and computer science from Seoul National University, Seoul, South Korea, in 1996, 1998, and 2009, respectively. From 1998 to 2004 and from 2009 to 2010, he was with Samsung SDI Ltd., Cheonan, South Korea, as a Senior Researcher, where he was involved in driver circuit and discharge waveform research. From 2010 to 2011, he was a Post-Doctoral Researcher with Seoul National University. In 2011, he joined the Department of Electronic Engineering, Sun Moon University, Asan, South Korea, where he is currently an Associate Professor. His current research interests include algorithm and architecture for video compression and computer vision.

Youhak Lee received the B.S. degree in electrical engineering from Korea University, Seoul, South Korea, in 2017 and the M.S. degree in electrical and computer engineering from Seoul National University, Seoul, South Korea, in 2019. He is currently working for computer vision team in Chips&Media Inc., Seoul, South Korea. His current research interests include object detection, super-resolution and deep-learning hardware design.

Kyujoong Lee received the B.S. degree in Electrical Engineering from Seoul National University, Seoul, Korea, in 2002 and the M.S. degree in Electrical Engineering from University of Southern California, Los Angeles, USA, in 2008. He got Ph.D. degree in Electrical Engineering and Computer Science at Seoul National University Seoul, Korea, in 2013. From 2002 to 2005, he was with Com2us corporation, Seoul, Korea, as a developer. From 2013 to 2017, he worked for S.LSI division of Samsung Electronics corporation. In 2017, he was appointed as an Assistant Professor in the department of Electronic Engineering at Sun Moon University, Asan, Korea. His major research interests include the algorithms and architectures of deep learning and image/video compression.

Hyuk-Jae Lee received the B.S. and M.S. degrees in electronics engineering from Seoul National University, South Korea, in 1987 and 1989, respectively, and the Ph.D. degree in Electrical and Computer Engineering from Purdue University, West Lafayette, IN, in 1996. From 1998 to 2001, he was with the Server and Workstation Chipset Division, Intel Corporation, Hillsboro, OR, as a Senior Component Design Engineer. From 1996 to 1998, he was with the Faculty of the Department of Computer Science, Louisiana Tech University, Ruston, LS. In 2001, he joined the School of Electrical Engineering and Computer Science, Seoul National University, South Korea, where he is currently a Professor. He is a Founder of Mamurian Design, Inc., a fabless SoC design house for multimedia applications. His research interests are in the areas of computer architecture and SoC design for multimedia applications.