IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⪬ Impact Factor 7.39 ⪬ Vol. 11, Issue 7, July 2022

DOI: 10.17148/IJARCCE.2022.11763

Methods to Accelerate the Automation of CCTV Surveillance

Hema Singaravelan[1], Dr. Roopa J.[2], Dr. Govinda Raju M. [3]

Post Graduate Student, Dept. of ECE, RV College of Engineering, Bengaluru, India[1]

Assistant Professor, Dept. of ECE, RV College of Engineering, Bengaluru, India [2]

Assistant Professor, Dept. of ECE, RV College of Engineering, Bengaluru, India [3]

Abstract: As we evolve in the field of technology, the applications of the same in our day-to-day life have improved our lives. But this ease comes at a cost since, as the number of appliances used exponentially increases in our society, so does our demand for energy. This energy is usually obtained after destructive emissions are released into the environment, and the wastage of said energy can also be seen in several areas. A survey shows that a plugged-in mobile charger, when not used, consumes 0.1 to 0.5 watts per hour, costing ~15 rupees per day. Such wastage can be expected from other devices as well. Therefore, there is a scope for efficient energy management and, in recent times, it can be done more effectively using automation systems that eliminate the need for human interactions with devices or tools as much as possible. These systems may require the additional fitting of hardware such as sensors that will incur supplementary energy consumption and overhead costs for maintenance. They may also offer low-coverage for detection. Collaterally, it is worthwhile to note that the need for closed-circuit television (CCTV) installations is a basic necessity in several areas of the society and cannot be compromised to conserve energy and yet, we can utilize them to make a region of interest (ROI) energy efficient. This can be achieved through rationing the power consumption of other devices by inferring from an intelligible detection of objects and activities observed by a camera. It is made possible through a technique called Computer Vision (CV). CV, though reducing the workload of building setups for recognition, is a computationally exhaustive technique that requires hardware support to function with an accelerated performance for object detection in real-time. Thus, this work details the different methods available to detect objects and the techniques that can be employed to accelerate the performance of a low power consuming detection system. Graphics Processing Unit (GPU) acceleration and Edge computing are also discussed as a way to offer additional support to CV computation. The advantages and specific drawbacks of each method are also elaborated.

Keywords: Closed Circuit Television (CCTV), Region of Interest (ROI), Computer Vision (CV), Region Based Convolutional Neural Network (R-CNN), You Only Look Once (YOLO), Graphics Processing Unit (GPU), Edge Computing.

I. INTRODUCTION

Several Automation designs have already been proposed with the rise of sensor usage in various system designs to conserve energy. Examples of such include, Passive Infrared (PIR) sensors, Light Dependant resistors (LDR), Heat sensors, etc. each of which can be utilized to monitor specific activities in a given region of interest (ROI) but these systems require additional installation and maintenance of these sensors. In recent times, CCTV cameras can also act as sensors with the support of an image processing algorithm that will detect the performance of most systems and offers more visibility in the detection of objects and activities. Thus, using the pre-installed camera as an automation tool to increase the energy efficiency of an area can be held as one of the most necessities in the future.

Detection automation is made possible through Computer Vision. CV turns the camera into the eyes of the computer, allowing it to not only collect video data but also to process it the way humans would do. Based on the observations made by CCTV assisted by CV, the corresponding reactions can be made with respect to necessary stimuli to control energy consumption.

Using CV to detect objects and activities in real-time will require additional hardware acceleration methods. This could include Graphics Processing Unit (GPU) based acceleration to provide the necessary boost in the speed of computation. Although, GPUs can be used for this purpose, installing them next to each camera for video processing would defeat the purpose of energy savings that need to be made as each GPU device will consume more power to perform with a wastage of the resource that can be cycled between several cameras. Thus, keeping just one processing system closer to most of the cameras is advantageous. This is where the edge computing platforms play a major role as they increase processing speeds without compromising the time required to act on the processed data since they are much closer and easily accessible to the cameras than a cloud platform would be.

DOI: 10.17148/IJARCCE.2022.11763

Keeping these techniques and paradigms in mind, a detailed literature review was done and the inferences from the same is presented after a few of the concepts required to understand the same are discussed.

II. Concepts Required to Understand Different Implementation Methods

A few of the basic concepts required to understand the literature are discussed below,

A. Image Classification

Image classification models requires images to have a unique label to identify the whole picture. The annotation process of image classification aims at identifying the presence of similar objects in the dataset images.

An image classification model detects different animals within input images. The annotator would be provided with a dataset of different animal images and made to classify each image with a label of the specific animal species. The animal species is the class, and the images are the inputs. Thus, the outcome of image classification is to simply identify the presence of a particular object and name its predefined class.

The unique visual characteristics of the animals are provided to the model during training as annotated images so new unannotated animal images can also now be classified.

B. Object Detection

Object detection or recognition models classify, locate, and count the number of objects in an image. The annotation process draws bounding boxes around each object, thus allowing us to locate the exact position and number of objects in an image. The main difference is that several classes are detected within an image rather than the entire image being classified as one class, which is the case in image classification.

The class location parameter must be considered in the definition of the class and its boundary, but in image classification, the entire image is a single class. Objects can be annotated within an image with the use of labels and bounding boxes.

C. YOLO Algorithm

You Only Look Once (YOLO) detects and recognizes various objects in a picture in through just a single propagation through the neural network and detect objects. This algorithm has several variants, to name a few; YOLOv3, YOLOv4, etc.

D. Region-Based Convolutional Neural Network (R-CNN)

The R-CNN is a multistage object detection algorithm. It provides high precision object recognition using region proposals, from which features are extracted, and SVMs (State Vector Machines) use the extracted features for image classification.

E. GPU Acceleration

GPU programming entails dividing multiple processes among multiple cores to accelerate the time needed for execution offering data processing parallelism in real-time through the use of hardware. Compute Unified Device Architecture (CUDA) is a parallel programming paradigm that was released in 2007 by NVIDIA Corporation.

F. Edge Computing

Edge computing, a distributed computing paradigm, brings the resources for computation and data storage devices closer to the sources of the said data. This computing structure is expected to improve response-times to stimuli and save bandwidth involved in the transfer of data to and from the edge device and its dependent data-yielding sources. It is a topology-sensitive and location-sensitive architecture designed to handle minimal computational requirements in real-time in comparison to a cloud platform.

Following this, a brief idea of the concepts is grasped to now understand the literature review done and presented in the next section.

III. LITERATURE REVIEW

In times before the development of CV, sensors were used to detect the presence of humans based on the usage of Passive Infrared (PIR) sensors. An array of HC SR501 PIR sensors were used to detect human presence below the sensor. In article [1], the control of resources is initiated by TI SimpleLink ultra-low-power remote MCU after obtaining input from the sensors. Human detection was possible, and the MCU turned the resources on and off. Conveniently, a delay was generated between human detections so that the system did not remain in the polling condition. The work carried out was limited to one laboratory with a load of 700 W under test. Fig. 1 depicts the setup used in this experiment.[1]

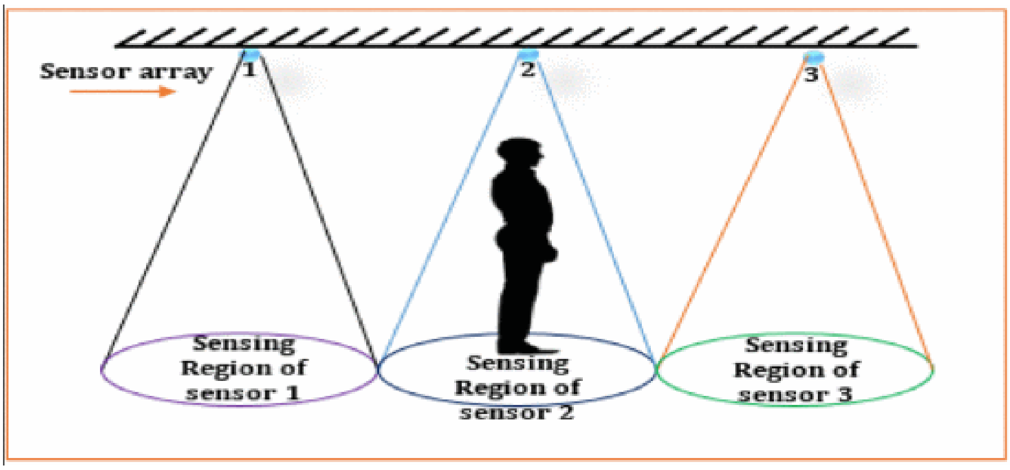

Sensor array
1
2
3
Sensing Region of sensor 1
Sensing Region of sensor 2
Sensing Region of sensor 3

Fig. 1 Human Presence Detection using PIR sensors [1]

Though PIR sensors and other methods [1-6] have been explored, these methods require additional hardware, energy consumption, implementation costs, and overhead with minimal fault insurance. It must also be highlighted that traditionally these sensors that are powered through a separate electrical system that would also require maintenance which creates an implementation overhead that constraints the development of stand-alone systems. Therefore, it can be inferred that the automation ideas must showcase minimal to no human intervention in terms of maintenance with least installation costs.

Additionally, to understand the full extent of the flexibility of a sensor-based, stand-alone hardware setup that would support automation and the expenses it would incur, the work described in paper [7] was viewed. It is a simulation of a hardware setup using the PROTEUS software that was realised as a real-time system. This system works based on the availability of light and the environment's temperature. The functioning depends entirely on the functioning and placement of sensors, but a fault detecting mechanism is designed. No recovery or by-pass mechanism in case sensors fail is described, and the PIC used is not indicated. But the main factor that hampers this solution is the power-grid placement for these sensors, which may soon become cumbersome if detection coverage needs to be extended. This system would also be redundant, yet power-consuming in an ROI observed by a CV enabled CCTV since human presence detection is possible through cameras. On a side note, with respect to the powering of cameras, the number of cables power and collect data from the cameras can be minimized if the ROI is set up with Internet Protocol (IP) based cameras that utilize Power over Ethernet (PoE) technology. Through PoE, a camera can obtain its necessary power supply through the ethernet interface to which it’s connected.

Moving on, as the first step to utilizing CV for the application of detecting humans in a given environment, a suitable dataset to train a model must be assembled, and an application-specific model must be designed to obtain the expected outcome. A new approach for the direct optimization of the Intersection over Union (IoU) measure in deep neural networks can be noted as it was applied to object category segmentation to address the issue of pre-processing the image data obtained for training a model. [8] Optimizing the IoU loss results in better performance than traditional softmax loss commonly used for learning DNNs. Paper [8] is focused on binary segmentation problems. Although this technique is resourceful, simple segmentation techniques and bounding-box-based image annotations can be utilized for a simple, small-scale project that does not require high precision that is costly. Thus, the bounding-box can be considered for the first level of optimization of the training data in a small-scale edge deployment.

In view of the models that can be optimized to detect humans and trained with such pre-processed data, various object detection algorithms implemented in Python, such as Haar-Like features, Template matching, Blob Detection, Gradient-Based Method, Local Binary Pattern, Bag-of -words Method, and Deep Face Method, were noted. [9] Although

DOI: 10.17148/IJARCCE.2022.11763

these techniques carry merit, the speed of execution of the YOLOv4 algorithm was unparalleled in these methods. But the YOLOv4 algorithm detects 91 objects which is redundant if the need of object detection is limited to a few objects such as humans or cars. It is also a relatively large model to be executed on an edge device with limited memory and processing capabilities. To obtain considerable speed in execution without compromising accuracy, a few more models need to be considered.

A new architecture for a Fully Convolutional Network (FCN) is proposed with a region-based detector that is fully convolutional, with almost all computation shared on the entire image in [10]. This architecture reduces the density of the neural network making the training, testing, and computation faster. Fig. 2 depicts the FCN. PASCAL VOC datasets were used with 83.6\% precision and with the 101-layer ResNet. It was also achieved at a test-time speed of 170ms per image (2.5-20 times faster) and was a more viable option compared to MobileNets [11] since MobileNets trade-offs a reasonable amount of accuracy to reduce size and latency.

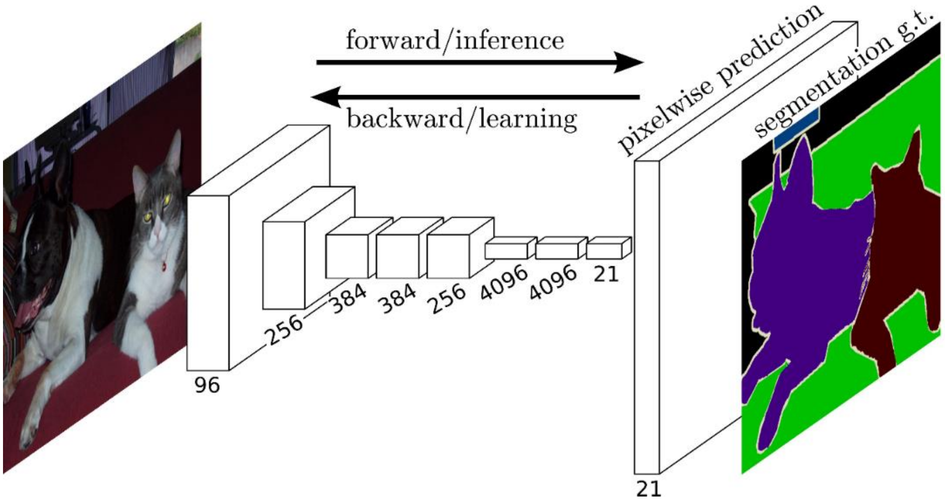


Fig 2. FCN architecture [23]

Comparable to the FCN in terms of accuracy and reduced computation, a Region Proposal Network (RPN) merged with the Fast R-CNN into a single network by sharing their convolutional features, suggested in [12], can be explored. The RPN component conveys to the unified network where it must look. The detection system has a frame rate (frames per second) of 5 fps on a GPU while achieving state-of-the-art object detection accuracy on PASCAL VOC 2007, 2012, and MS COCO datasets with a mere 300 proposals per image. To better understand a Fast R-CNN, a paper describing it was taken based on which a Fast R-CNN train the deep network 9× faster than R-CNN, is 213× faster at test-time, and reaches a higher mAP on PASCAL VOC 2012. Compared to SPPnet, Fast R-CNN trains 3× faster, tests 10× faster, and is more accurate.[13] Fast R-CNN is implemented in Python and C++ (using Caffe) and is available under the open-source MIT License. Fast R-CNN also occupies lesser memory with respect to the other models discussed, thus making it ideal for implementation into the edge computing paradigm.

Apart from the above, object tracking can be exercised to predict the movements of objects so that a targeted system may enter an idle state should the object behave in a certain way within the ROI. The implementation of Object Tracking in a video file is based on the colour and motion of single and multiple objects that are counted in numerous frames on MATLAB R2017b in [14]. The advantages and limitations of each aspect of the algorithms used were discussed in detail. Though a detailed analysis of the performance of each algorithm has not been mentioned, the usage of coloured images raises the question of redundant data supplied to the algorithm since, in recent times, grayscale image-based models are famous for their size while transmitted through a network. Thus, a few more techniques can be explored.

The traditional adaptive mixture Gaussian model was used in [15], and by dynamically adjusting the parameters and the number of Gaussian components, the computation cost was reduced significantly. It can quickly and accurately extract the foreground information of different types and eliminate noise. With both tracking and filtering in mind, a two-way matching method based on frame difference was designed to obtain the detected moving target, and a series of image filtering methods were combined in [16]. Bokeh plotting was used to detect and plot the time-frame in which the object was in front of the camera. This method showed high accuracy, but no detailed analysis of data was provided regarding the efficiency or performance of the project.

Thus, for simple detection tasks without the need for tracking, it can be concluded that the Fast R-CNN not only provides faster and accurate inference but also occupies less memory during execution done without polling as in the case of tracking. This can be considered as the model required for the edge computing host to detect objects.

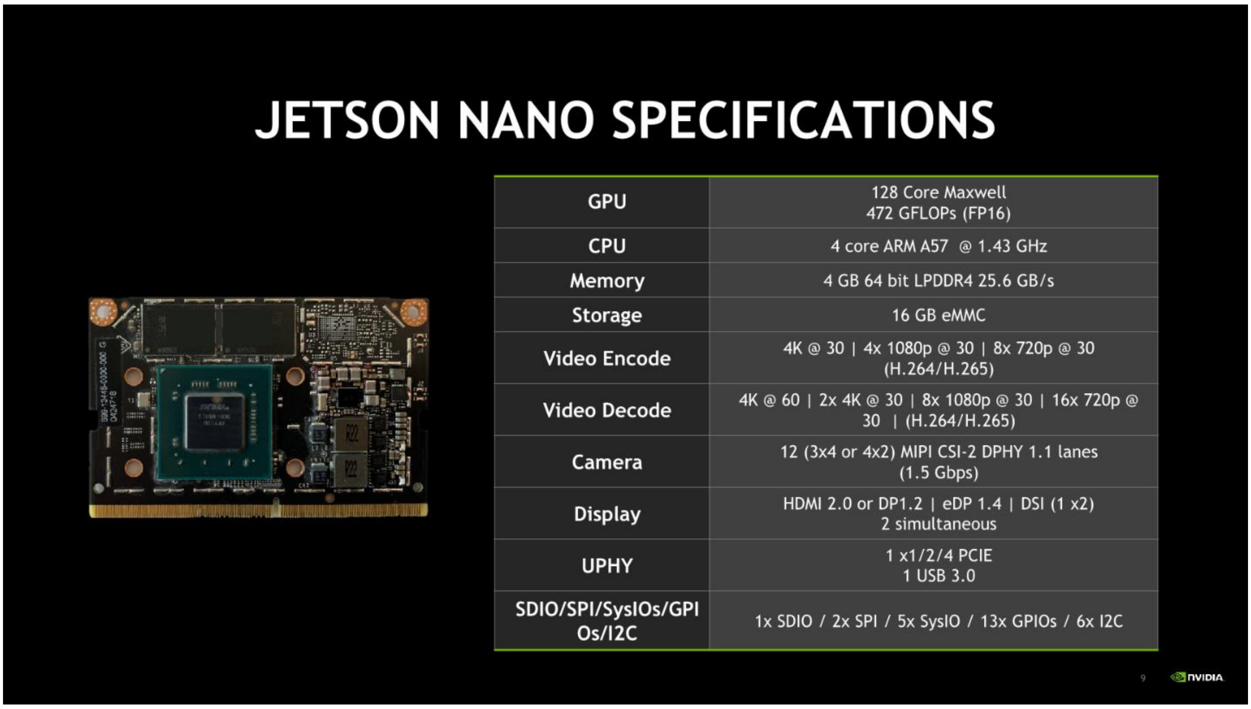
IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ∺ Impact Factor 7.39 ∺ Vol. 11, Issue 7, July 2022

DOI: 10.17148/IJARCCE.2022.11763

Then, to accelerate the model execution, a few hardware methods can be analysed. Object Detection Systems by Xilinx ZCU104 FPGA board and NVIDIA Mobile GPU boards with a USB camera was done in paper [19] provided some understanding. The image split method is adapted by YOLOv2 and FPGA. The performance of the FPGA and GPU were compared. FPGA achieves 547.0 FPS per image and is 3.9 times faster than Mobile GPU. When the image is split into 4 × 4 grids, the system realizes 34.2 FPS (the real-time requirement on the standard camera is 30FPS). Apart from this, a novel experimental software OpenVINO was considered to deploy models on Intel hardware to accelerate the model's performance by three times the execution speed. CUDA was pursued to perform a comparative analysis with the Nvidia Jetson Nano 2GB board. GPU computing [20] is explored through NVIDIA Compute Unified Device Architecture (CUDA), which is, at present, the most evolved application programming interface (API) for general-purpose computation on GPUs in [21]. A few case studies and fundamental concepts in GPU programming were introduced to readers, and insights were provided into CUDA-based programming to introduce acceleration in processing in this article. Fig. 3 shows the Nvidia Jetson Nano 2GB developer kit. CUDA C and CUDA python are used to program the Nvidia board. [22]



JETSON NANO SPECIFICATIONS

GPU	128 Core Maxwell 472 GFLOPs (FP16)
CPU	4 core ARM A57 @ 1.43 GHz
Memory	4 GB 64 bit LPDDR4 25.6 GB/s
Storage	16 GB eMMC
Video Encode	4K @ 30 | 4x 1080p @ 30 | 8x 720p @ 30 (H.264/H.265)
Video Decode	4K @ 60 | 2x 4K @ 30 | 8x 1080p @ 30 | 16x 720p @ 30 | (H.264/H.265)
Camera	12 (3x4 or 4x2) MIPI CSI-2 DPHY 1.1 lanes (1.5 Gbps)
Display	HDMI 2.0 or DP1.2 | eDP 1.4 | DSI (1 x2) 2 simultaneous
UPHY	1 x1/2/4 PCIE 1 USB 3.0
SDIO/SPI/SysIOs/GPIOs/I2C	1x SDIO / 2x SPI / 5x SysIO / 13x GPIOs / 6x I2C

Fig 3. Nvidia Jetson Nano 2GB specifications [24]

Lastly, to increase the range of operation and reduce the number of devices and resources employed, the video-streams must be communicated to a singular host machine that serves multiple cameras. Therefore, for the edge computing hosts running the detection algorithm to communicate with the cameras from a distance, a network must be established that transfers data to the edge computing host from the cameras, and delivers the respective feedback to the devices that can control the state of the appliances that may be connected to them to achieve the energy conservation target.

In order to set up the communication interface between the host machine and microcontroller, a host of communication methods such as Wi-Fi, GSM, Bluetooth, and ZigBee were considered. [17-18] Due to the wide range offered as well as the ease of setup and use of the Wireless Local Area Network (WLAN), it can be the preferred communication mechanism to transfer data over a network. Thus, Wi-Fi-based control [2] [17-18] has been suggested to not only increase the area of ROI but also offer easy human intervention schemes, if necessary, in cases where differently-abled are required to interact with the system or if the automation tool and authorized human personnel supervising the resources are remote by connecting to the setup through the internet.

Thus, from the literature, Fast R-CNN-based model can be considered for an edge object detecting application. The deployment platform can be chosen with respect to the performance of several CPUs and GPUs by conducting a study by deploying the Fast R-CNN on each of these platforms Furthermore, the design cost must also be kept in mind to ensure the cost of the implementation and the energy it consumes make it a feasible solution. Finally, the deployment platform, can use GPU programming to accelerate the detection model and the entire setup can on a Wireless Local Area Network (WLAN) for ease.

IV. Conclusion

In many areas, electrical appliances are most often continuously consuming power until humans manually switch them off; thus, energy savings must be extracted through the automation of state switching of these devices based on need. Conserving energy through the use of CCTV cameras by the selective enabling of the appliances of an ROI based can also be done using CV without the need for additional sensor or hardware implementation. From the literature, it can be understood that CV models can be computationally expensive leading to the exploitation of GPU acceleration and edge computing methods. With this in mind, Fast-RCNN and WLAN setups deployed on an edge platform such as the Nvidia Jetson Nano board, with limited processing ability and memory availability, were suggested to be implemented in a prototype design for the mentioned purpose. The prototype cost must also be factored in during the development of the system to make it a feasible real-time solution.

ACKNOWLEDGMENT

We extend our sincere gratitude to the staff of the RVCE Centre of Excellence for CCTV Research and all the staff of the departments of RV College of Engineering, all our friends who have patiently extended a helping hand for accomplishing this undertaking.

References

[1]. Harsha B K and Naveen Kumar G N. “Home Automated Power Saving System Using PIR Sensor”. In: 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA) (2020), pp. 1117–1121.

[2]. M. W. Y. C. Karunarthhna, W. A. S. Wijesinghe, and Hiroharu Kawanaka. “Development of a Home Automation Systems for Sri Lanka Using ARM Cortex M4”. In: 2019 IEEE 8th Global Conference on Consumer Electronics (GCCE) (2019), pp. 587–589.

[3]. Bhavkanwal Kaur, Pushpendra Kumar Pateriya, and Mritunjay Kumar Rai. “An Illustration of Making a Home Automation System Using Raspberry Pi and PIR Sensor”. In: 2018 International Conference on Intelligent Circuits and Systems (ICICS) (2018), pp. 439–444.

[4]. Alaa Alhamoud et al. “SMARTENERGY.KOM: An intelligent system for energy saving in smart home”. In: Proceedings -Conference on Local Computer Networks, LCN 2014 (Oct. 2014), pp. 685–692. doi: 10.1109/LCNW.2014.6927721.

[5]. Doan Mien et al. “A Scalable IoT Video Data Analytics for Smart Cities”. In: EAI Endorsed Transactions on Context-aware Systems and Applications 6 (July 2018), p. 163136. doi: 10.4108/eai.13-7- 2018.163136.

[6]. Sanjana Prasad et al. Smart Surveillance Monitoring System Using Raspberry PI and PIR Sensor.

[7]. Ranjit Singh Sarban Singh et al. “Designing Switching System for AC Powered Appliances Using Microcontroller”. In: 2010 Fourth Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation. 2010, pp. 46–50. doi: 10.1109/AMS.2010.22.

[8]. Md Atiqur Rahman and Yang Wang. “Optimizing Intersection-Over-Union in Deep Neural Networks for Image Segmentation”. In: Advances in Visual Computing. Cham: Springer International Publishing, 2016, pp. 234–244.

[9]. Bhumika Gupta et al. “Study on Object Detection using Open CV -Python”. In: International Journal of Computer Applications 162 (2017), pp. 17–21.

[10]. Jifeng Dai et al. R-FCN: Object Detection via Region-based Fully Convolutional Networks. 2016. doi: 10.48550/ARXIV.1605.06409. url:https://arxiv.org/ abs/1605.06409.

[11]. Andrew G. Howard et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. 2017. doi:10.48550/ARXIV.1704.04861. url: https: //arxiv.org/abs/1704.04861.

[12]. Shaoqing Ren et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: IEEE Transactions on Pattern Analysis and Machine Intelligence 39.6 (2017), pp. 1137–1149. doi: 10. 1109 / TPAMI . 2016 . 2577031.

[13]. Ross Girshick. “Fast R-CNN”. In: 2015 IEEE International Conference on Computer Vision (ICCV). 2015, pp. 1440–1448. doi:10.1109/ICCV.2015.169

[14]. Prasit Nangtin, Jitraphon Nangtin, and Sombat Vanichprapa. “Building automation system for energy saving using the simple PLC and VDO analytic”. In: 2018 International Workshop on Advanced Image Technology (IWAIT). 2018, pp. 1–4. doi: 10.1109/IWAIT.2018.8369797.

DOI: 10.17148/IJARCCE.2022.11763

[15]. Zhou Wei, Peng Li, and Huang Yue. “A Foreground-Background Segmentation Algorithm for Video Sequences”. In: 2015 14th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES). 2015, pp. 340–343. doi:10.1109/DCABES.2015.92.

[16]. Suraiya Parveen and Javeria Shah. “A Motion Detection System in Python and Opencv”. In: 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV). 2021, pp. 1378–1382. doi: 10.1109/ICICV50876.2021.9388404.

[17]. Pooja S. Chinchansure and Charudatta V. Kulkarni. “Home automation system based on FPGA and GSM”. In: 2014 International Conference on Computer Communication and Informatics (2014), pp. 1–5.

[18]. Vaishnavi S. Gunge and Pratibha S. Yalagi. “Article: Smart Home Automation: A Literature Review”. In: IJCA Proceedings on National Seminar on Recent Trends in Data Mining RTDM 2016.1 (Apr. 2016). Full text available, pp. 6–10.

[19]. Akira Jinguji, Youki Sada, and Hiroki Nakahara. “Real-Time Multi-Pedestrian Detection in Surveillance Camera using FPGA”. In: 2019 29th International Conference on Field Programmable Logic and Applications (FPL). 2019, pp. 424–425. doi: 10.1109/FPL.2019.00078.

[20]. John Nickolls and William J. Dally. “The GPU Computing Era”. In:IEEE Micro 30.2 (2010), pp. 56–69. doi: 10.1109/MM.2010.41.

[21]. Keyan Cao et al. “An Overview on Edge Computing Research”. In: IEEE Access PP (Jan. 2020), pp. 1–1. doi: 10.1109/ACCESS.2020.2991734.

[22]. USA. NVIDIA. Santa Clara CA. CUDA C Programming Guide (Version 6.5). https://docs.nvidia.com/cuda/cuda- c- programming-guide/index.html. 2014.

[23]. url: https://paperswithcode.com/method/fcn.

[24]. url: https://www.androidauthority.com/jetson-nano-review-969318/.