Video-based Real-time Intrusion Detection System using Deep-Learning for Smart City Applications

Abstract—**There is a huge demand of video surveillance based intelligent security systems which can automatically detect the unauthorized entry or mal-intentional intrusion to the unattended sensitive areas and notify to the concerned authorities in real-time. A novel video-based Intrusion Detection System (IDS) using deep learning is proposed. Here, You Only Look Once (YOLO) algorithm is used for object detection and intrusion is decided using our proposed algorithm based on the shifted center of mass of the detected object. Further, Simple Online and Real-time Tracking (SORT) algorithm is used for the tracking of the intruder in real-time. The developed system is also implemented and tested for live video stream using NVIDIA Jetson TX2 development platform with an accuracy of 97% and average fps of 30. Here, the proposed IDS is a generic one where the user can select the region of interest (the area to be intrusion free) of any size and shape from the reference (starting) frame and potential intruders such as a person, vehicle, etc. from the list of trained object classes. Hence, it can have a wide range of smart city applications such as person intrusion free zone, no vehicle entry zone, no parking zone, smart home security, etc.**

Index Terms—**Deep learning, intrusion detection system, transfer learning, smart city, SORT algorithm, YOLO algorithm**

I. Introduction

Video surveillance based intelligent security systems are widely used to detect potential crimes in the early stage. This enables sending of intimations in real-time and allows the users to take preventive measures to reduce the damages [1]. Mal-intentional intrusion into sensitive places such as prohibited zone, unattended locations and high-risk security zones is a crime. In the case of the intrusion detection, 'seeing is believing', i.e., simultaneously intrusion detection and visual observation are preferred [2]. Further, given the increasing crime rate and the availability of efficient general purpose processors and video cameras in affordable prices, provides an edge to the video-based intrusion detection system over other systems such as microwave Doppler detector, infrared detector, vibration detector, magnetic wall, and so on [3]. The video surveillance based intelligent security systems are economical as they can be used as a multi-purpose security system, i.e., the video-based Intrusion Detection System (IDS) can utilize the same platform used for the other security applications. Different variants of the IDS for various applications have been proposed [1], [2], [4]–[7]. However, these systems lack of desired online performances such as high processing speed, less execution memory, and high accuracy. The video based intrusion detection mainly involves three steps such as object detection, target tracking, and implementation of intrusion rules.

Object detection in the complex environment using the real-time video streams is the most challenging and important task for the development of intrusion detection system. Hence, the main objective is to extract the changing area from the video frames by measuring the subsequent changes in the consecutive frames. Different approaches such as Gaussian Mixture Model (GMM) [8], a morphological operation based object detection [9], background subtraction and frame difference method [10], etc. have been used for the object detection for developing the intrusion detection systems. However, these methods are not robust enough for providing desired processing speed and accuracy for real-time applications. Further, deep learning based object detection techniques can be used for intrusion detection to achieve better accuracy and performance.

The detected object can be tracked by exploring the relationship among continuous frames based on the shape, position, velocity, and time occupancy of the object of interest. Generally, target tracking can be performed using Kalman filter, particle filter and mean shift filter [8]. The major challenges in tracking include the changes in illumination, occlusion, object deformation, and complexity of the environment. However, the tracking methods should be fast enough with high accuracy so that the intruding object will not be missed out.

Finally, the intrusion is said to occur only when the detected object enters inside of the area supposed to be intrusion free, restricted zone or Region of Interest (ROI). The ROI is selected as the boundary from the reference frame of the video streams. When centroid of the object being monitored enters inside the boundary of the restricted zone, then intrusion is detected and alarm signal is generated [8]. An intrusion decision making rule based on the concept of area of triangle using Helen's formula [10] suffers from computational complexity when number of vertices of the ROI increases. Hence, in practice, there is a need of generic intrusion decision making rule which can be applied for any shape of the ROI. Though a number of research works have been attempted to improve the performance of the video-based intrusion detection system to make it more suitable for real-time applications, still, there are ample scopes to improve it with respect to the accuracy, processing speed and time latency.

In this regard, we have proposed and implemented a video-based low-cost IDS using deep learning approach for real-time applications with desired online performances such as high processing speed, low-latency and high detection accuracy. The target is detected using You Only Look Once (YOLO) algorithm and decision about the occurrence of the intrusion is made when the shifted center of mass of the detected object crosses the selected boundary. Subsequently, the intruded object is tracked using Simple Online and Real-time Tracking (SORT) algorithm in real time. Finally, real-time notification about the intrusion is sent to the concerned users.

The rest of the paper is organized as follows. Section II describes the proposed IDS. The development and implementation are discussed in Section III. The results are discussed in Section IV followed by conclusions in Section V.

II. Proposed Intrusion Detection System

The proposed IDS can detect various potential intruders such as the persons, car, etc. in the ROI and intimate about the same to the concerned users in real-time.

A. *The Proposed System Model*

The system model of the proposed IDS is shown in Fig. 1. The complete working of the system can be explained in three steps such as edge computing, cloud computing and user interfacing.

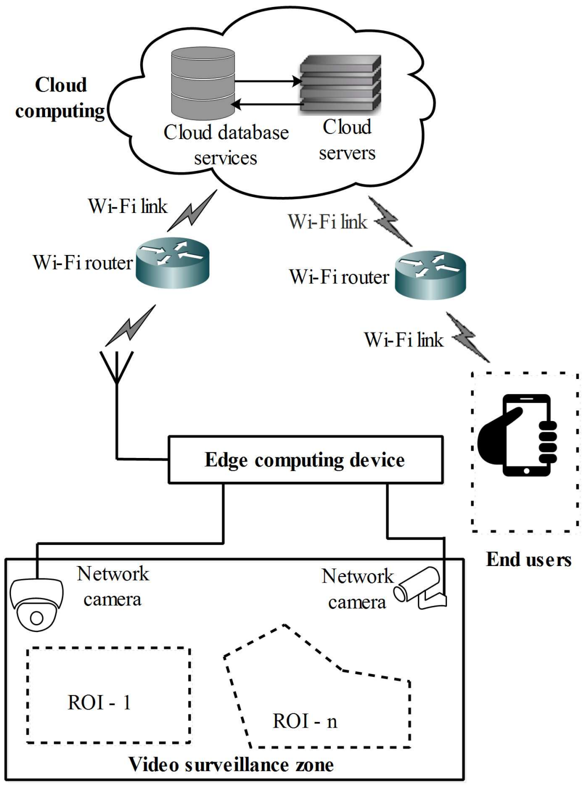


Fig. 1: Complete system model of the proposed IDS.

1) Edge Computing: The live video streams are captured by the network cameras (Hikvision 4 MP) and then processed at the edge computing device (NVIDIA Jetson TX2 development board or platform). The NVIDIA Jetson TX2 development kit is preferred as compared to other hardware platforms such as Raspberry Pi 3 Model B, Asus Tinker board, and similar development boards due to its NVIDIA CUDA-enabled graphical processing architecture which offers high processing speed for real-time applications. Further, the information about the occurrence of the intrusion is sent to a cloud database unit in real-time through the Wi-Fi.

2) Cloud Computing: The processed information sent by the edge computing device are stored in the cloud database and then the information is published or intimated through the cloud services to the subscribers or users. Here, we have used Twilio which is a cloud communication platform limitedly free for research and development purposes.

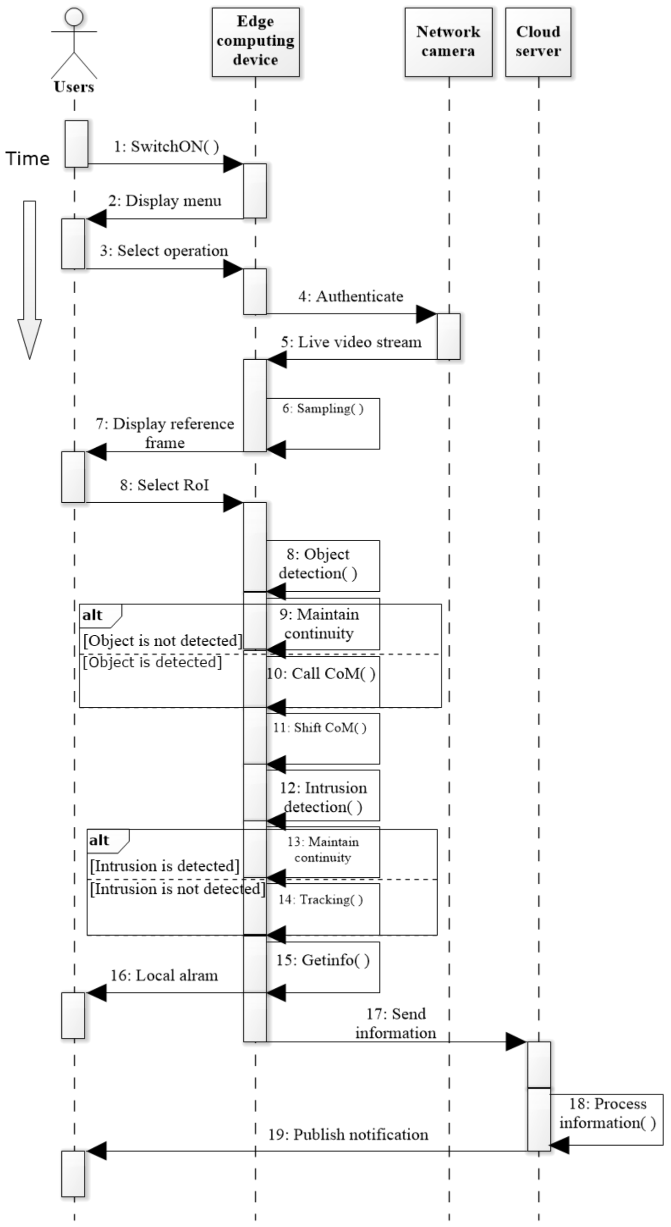


Fig. 2: Sequence diagram of the proposed system.

3) User Interfacing: The user draws or selects the area to be intrusion free or region of interest (ROI) or restricted zone of any shape and size by using the reference frame displayed in the graphical user interface (GUI) of the developed application software at the starting. Specifically, the ROI is a polygon having n number of vertices which is stored in a vector named $poly[n]$. The elements of $poly[n]$ are represented as $(p_i x, p_i y)$ for $0 \leqslant i \leqslant n$. The user receives a comprehensive information about the intrusion such as photos of the intruder, location of the intrusion, time and duration of the intrusion at the registered WhatsApp from the Twilio and email by using Simple Mail Transfer Protocol (SMTP) in real-time.

B. Sequence Diagram

The Unified Modeling Language (UML) sequence diagram [11] used for the software development as shown in the Fig. 2 to explain the real-time collaborative relationship among the various entities of the IDS. Here, all the operations are time dependent and are represented in the order of interactions. The processing time for all the executions is constant except the data processing time (t_d). Total execution time (t) should be minimized to make the algorithm suitable for the real-time applications with desired on-line performances and hence the execution time of the data processing should be minimized. This is achieved by implementing the IDS with the help of CUDA enabled NVIDIA Jetson TX2 development platform which provides hardware acceleration.

C. Object Detection

A deep-learning based algorithm, i.e., YOLO [12] is used for the object detection. The unified architecture of the YOLO ensures fast detection. The YOLO is preferred over other deep learning based object detectors such as Single-Shot Detector (SSD) and Region-based Convolutional Neural Networks(R-CNN) due to its high accuracy with low false alarm for the real-time applications [12]. We have used the same network architecture of the YOLOv2 [12] and pertained weight file for the detecting objects on the Pascal VOC 2012 and coco datasets in the darkflow platform (an open source neural network framework) [13].

D. Feature Extraction

During the object detection, a rectangular bounding box of width w and height h are created around the box having starting point at $P_0\left(x_0, y_0\right)$. Instead of tracking the whole box, a unique point corresponding to the object can be extracted and used for the tracking. This also helps in the increase in computational efficiency. Here, the Center of the Mass (CoM) $P_c\left(x_c, y_c\right)$ is calculated by using the Eq. 1.

$$x_c = x_0 + (w/2)\,, \qquad y_c = y_0 + (h/2) \tag{1}$$

Generally, the video cameras are placed at a height with an inclined angle θ far from the detection area for better coverage and the restricted zone is relatively large as compared to the invasion target. The calculated CoM, $P_c\left(x_c, y_c\right)$ of the object and the selected RoI lie at different heights. Hence, we have calculated the shifted CoM, $P_{sc}\left(x_{sc}, y_{sc}\right)$ by shifting the $P_c\left(x_c, y_c\right)$ towards the ground by multiplying a factor f with the y coordinate value as expressed in Eq. 2.

$$x_{sc} = x_c, \qquad y_{sc} = y_c/f \tag{2}$$

Few experimentally found typical values of the f are 2, 1.1, 1.5 for corresponding θ of the values 0^0, 30^0, 45^0 respectively. We have experimentally observed that the accuracy of intrusion detection increases when shifted CoM is used to decide the intrusion. The representation for the feature extraction from the detected object is shown in Fig. 3.

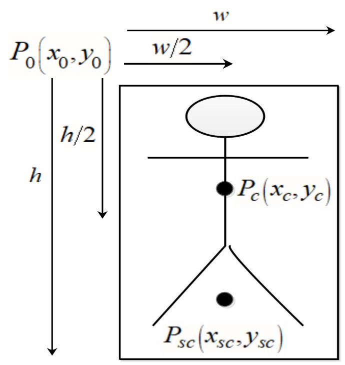

w
$P_0\left(x_0, y_0\right)$
$w/2$
$h/2$
h
$P_c\left(x_c, y_c\right)$
$P_{sc}\left(x_{sc}, y_{sc}\right)$

Fig. 3: Representation for feature extraction process from the detected object.

E. Intrusion Detection

When the shifted CoM, $P_{sc}\left(x_{sc}, y_{sc}\right)$ enters into the ROI $poly\left[n\right]$, it will be treated as intrusion. Then the number of frames are counted for the duration for which the object is present in side the ROI. The total duration of the intrusion is calculated from the total number of frames using time library of the python. The pseudo code of the proposed intrusion detection technique is presented in Algorithm 1.

Algorithm 1 Pseudo code of Intrusion Detection

```
Input: RoI: $poly[n]$ and shifted CoM: $P_{sc}\left(x_{sc}, y_{sc}\right)$.
Output: $Intrusion$.
    Initialisation: $Intrusion = False$,
                    $n = length\left(poly\right)$,
                    $[p_1x, p_2y] = poly\left[0\right]$.
 1: for $i = 0$ to $n$ do
 2:   $[p_2x, p_2y] = poly\left[i\%n\right]$
 3:   if $y_{sc} > \min\left(p_1y, p_2y\right)$ then
 4:     if $\left(y_{sc} <= \max\left(p_1y, p_2y\right)\right)$ then
 5:       if $\left(x_{sc} <= \max\left(p_1x, p_2x\right)\right)$ then
 6:         if $\left(p_1y \neq p_2y\right)$ then
 7:           $m = \left(p_2x - p_1x\right)/\left(p_2y - p_1y\right)$
 8:           $x_{\text{inters}} = \left(y_{sc} - p_1y\right)m + p_1x$
 9:         end if
10:         if $\left(p_1x == p_2x\right)$ or $x_{sc} <= x_{\text{int}\,ers}$ then
11:           $Intrusion$=$Not\ Intrusion$
12:         end if
13:       end if
14:     end if
15:   end if
16:   $[p_1x, p_2y] = [p_2x, p_2y]$
17: end for
18: return $Intrusion$
```

F. Target Tracking

Finally, the object is tracked using simple on-line and real-time tracking (SORT) algorithm [14]. The SORT algorithm is used as it is simple, yet provides accuracy comparable to the state-of-art online trackers for multiple object tracking for real-time applications. Here, the detections only from the previous and current frame are presented to the tracker to make it suitable for online tracking. The tracker uses the bounding box generated by the YOLO based object detection, propagates the object states into the future frames, associates the current detection with the existing objects and manages the lifespan of the tracked object.

III. Development and Implementation

The proposed IDS is developed and implemented using the experimental setup as mentioned in the Table I.

TABLE I: Experimental Hardware and Software Setup

Hardware platform		
Configurations	Workstation (Dell Precision 7920)	Edge device (NVIDIA JetsonTx2 Dev. Platform)
CPU	Intel Xeon 6140 (Dual processor, 64-bit, 18 core)	2 Denver 64-bit CPUs +Quad-Core A57 Complex
GPU	NVIDIA Quadro P6000 (24 GB)	NVIDIA Pascal Architecture GPU
RAM	DDR4 (256GB)	8 GB L128 DDR4 Memory
Software platform		
Operating System		Ubuntu 16.04 (64-bit)
Deep learning framework		Darknet
Programming language		Python 3.5
CUDA Compatibility		CUDA 9.0, CuDNN 7.1

A. Hardware Implementation

The proposed IDS is practically implemented using NVIDIA Jetson TX2 development platform and tested in real-time successfully. The experimental set-up of the IDS is shown in Fig. 4.

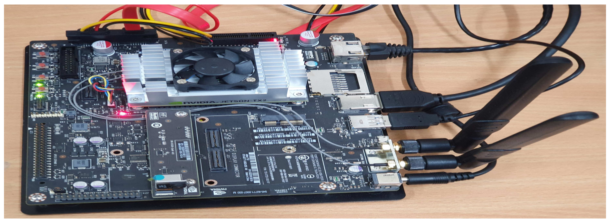
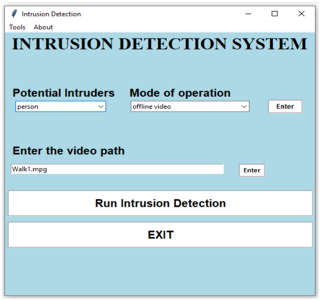
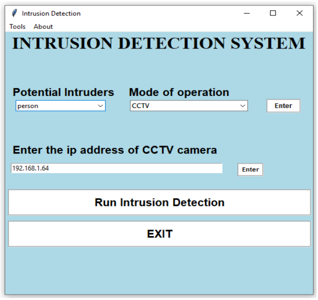


Fig. 4: Implementation using NVIDIA Jetson TX2 development platform.



Potential Intruders: person
Mode of operation: offline video
Enter the video path: Walk1.mpg

(a) Offline mode



Potential Intruders: person
Mode of operation: CCTV
Enter the ip address of CCTV camera: 192.168.1.64

(b) Online mode

Fig. 5: Developed GUI for the proposed IDS.

B. Software Development

The application software for the proposed IDS as shown in Fig. 5 is developed using Python programming. There is facilities for selecting mode of operations such as offline mode of operation (post-processing of video data) and online mode of operation (real-time processing of live video stream). Further, the IDS is developed as a generic one where the user can select the potential intruders, i.e., the object meant to be treated as the intruder from the trained classes in starting. Few examples of the potential smart city applications of the IDS with corresponding trained object classes are given in the Table II.

TABLE II: Potential Applications of the Proposed IDS

Applications	Object Classes
Complete intrusion free zone	All the defined and trained classes
No person entry zone	Person
No vehicle entry zone	Bicylce, bus, car, motorbike
No fly zone at low altitudes	bird, aeroplane

IV. Results and Discussions

The implemented YOLOv2 based object detection provides the mAP of 76.8% and recall of 81.24% which are similar to the reported results of the YOLO algorithm [12], [13].

A. Results of the IDS for various test scenarios

The results of the IDS for various test scenarios are shown in Table III. Here, the detected object is bounded with the yellow box and the ROI is green color when there is no intrusion. When an intrusion occurs, the color of the ROI changes to red and a text mentioning intrusion appears at the top-right most corner of the bounding box. Simultaneously, a beep sound is auto-generated locally and subsequently notification is pushed to the cloud database service. Only the objects which have entered into the ROI are tracked by the SORT algorithm instead of tracking all objects. This helps in the reduction of computational complexity. We have plotted the intrusion profile (i.e. plot of intrusion versus time) of the processed video stream shown in Fig. 6 for explaining the real-time status of the intrusion for the corresponding ROI.

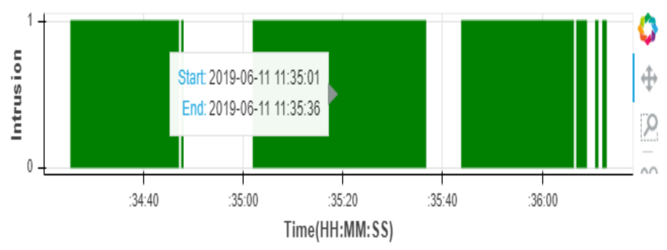
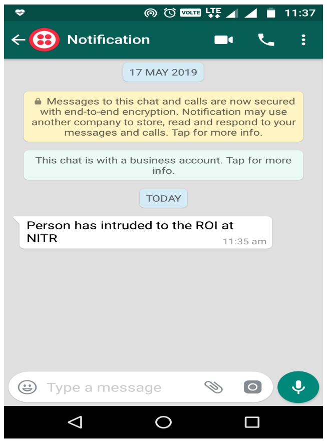


Fig. 6: Intrusion profile corresponding to the ROI as shown in the 3^{rd} row of the Table III.

B. Real-time notifications

Once the intrusion is detected, a real-time notification comprising of the class of intruder (i.e., person, car, etc.), location and time of the intrusion are sent to the user's registered WhatsApp number using the cloud communication platform Twilio. Further, the intimation about the intrusion along with the photos of the intruder is sent to the registered email using SMTP. The real-time notifications corresponding to the online processing of live video for group of intruders in day condition as mentioned in the Table III are presented in Fig. 7.



17 MAY 2019

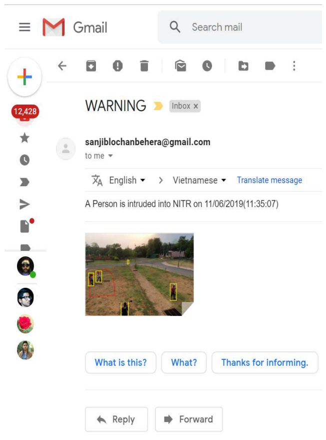
Messages to this chat and calls are now secured with end-to-end encryption. Notification may use another company to store, read and respond to your messages and calls. Tap for more info.

This chat is with a business account. Tap for more info.

TODAY

Person has intruded to the ROI at NITR 11:35 am

(a) Notification in WhatsApp



WARNING

sanjiblochanbehera@gmail.com
to me

English > Vietnamese Translate message

A Person is intruded into NITR on 11/06/2019(11:35:07)

(b) Notification in Email

Fig. 7: Real-time notification received in user's registered (a) WhatsApp and (b) email.

C. Comparative Analysis

A comparative analysis of the proposed IDS with the state-of-art is presented in Table IV. Our system is found to be more efficient with respect to speed and accuracy as compared to the existing techniques used for video-based intrusion detection.

V. Conclusion

A generic, yet efficient video based IDS for real-time applications is proposed where the user can select the potential intruders from the list of classes for which the model is being trained. Further, the user can draw the ROI of any size and shape using the reference frame within the camera view. A low-computational novel algorithm for the detection of the intrusion is proposed. The practical system is implemented using the NVIDIA Jetson TX2 development platform at the edge with average processing speed of 30 fps. The real-time notification regarding the occurrence of the intrusion are sent to the user's registered email ID and WhatsApp number. We are exploring further to integrate the developed model with the models of anomalous activity detection.

References

[1] S. M. Tahir, O. P. Shen, L. C. Yang, and E. K. Karuppiah, "Implementation of intrusion detection system in cuda for real-time multi-node streaming," in *Proc. IEEE Conf. on Systems, Process & Control (ICSPC)*, Dec. 2013, pp. 97–102.

[2] R. Hariprakash, S. Ananthi, and K. Padmanabhan, "An economical wireless network monitored scheme for camera based intrusion detection at unattended sites," in *Proc. IEEE Int. Conf. on Computer Applications and Industrial Electronics (ICCAIE)*, Dec. 2011, pp. 150–155.

[3] Y.-L. Zhang, Z.-Q. Zhang, G. Xiao, R.-D. Wang, and X. He, "Perimeter intrusion detection based on intelligent video analysis," in *Proc. IEEE 15th Int. Conf. on Control, Automation and Systems (ICCAS)*, Oct. 2015, pp. 1199–1204.

[4] J. Schutte and S. Scholz, "Guideway intrusion detection," *IEEE Veh. Technol. Mag.*, vol. 4, no. 3, pp. 76–81, Sep. 2009.

[5] K. Kaneda, E. Nakamae, E. Takahashi, and K. Yazawa, "An unmanned watching system using video cameras," *IEEE Comput. Appl. in Power*, vol. 3, no. 2, pp. 20–24, Apr. 1990.

[6] D. ALshukri, E. Sumesh, P. Krishnan *et al.*, "Intelligent border security intrusion detection using iot and embedded systems," in *Proc. IEEE 4th MEC Int. Conf. on Big Data and Smart City (ICBDSC)*, Jan. 2019, pp. 1–3.

[7] R. Deshmukh, S. Kamdi, M. Pingle, S. Rajebhosale, and A. Bhosale, "Intelligent surveillance system using energy efficient intrusion detection and tracking techniques," in *Proc. IEEE 2nd Int. Conf. on Electronics, Communication and Aerospace Technology (ICECA)*, Mar. 2018, pp. 1214–1218.

[8] H. Chen, D. Chen, and X. Wang, "Intrusion detection of specific area based on video," in *Proc. IEEE 9th Int. Congr. on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, Oct. 2016, pp. 23–29.

[9] N. Kongurgsa, N. Chumuang, and M. Ketcham, "Real-time intrusion detecting and alert system by image processing techniques," in *Proc. IEEE 10th Int. Conf. on Ubi-media Computing and Workshops (Ubi-Media)*, Aug. 2017, pp. 1–6.

[10] J.-x. Wang, "Research and implementation of intrusion detection algorithm in video surveillance," in *Proc. IEEE Int. Conf. on Audio, Language and Image Processing (ICALIP)*, Jul. 2016, pp. 345–348.

[11] B. P. Douglass, "Real-time uml," in *Proc. Springer 7th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT)*, vol. 2469, Sept. 2003, pp. 53–70.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE conf. on computer vision and pattern recognition*, Jun. 2016, pp. 779–788.

[13] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proc. IEEE conf. on computer vision and pattern recognition*, Jul. 2017, pp. 7263–7271.

[14] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *Proc. IEEE Int.Conf. on Image Processing (ICIP)*, Sep. 2016, pp. 3464–3468.

[15] R. Fisher, J. Santos-Victor, and J. Crowley. (2004, Jan 20) Caviar: Context aware vision using image-based active recognition. (EC Funded CAVIAR project/IST 2001 37540). [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CAVIAR/

[16] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah, "Part-based multiple-person tracking with partial occlusion handling," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, June 2012, pp. 1815–1821.

[17] A. Dehghan, S. Modiri Assari, and M. Shah, "Gmmcp tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, June 2015, pp. 4091–4099.

TABLE III: Results of IDS for different test scenarios

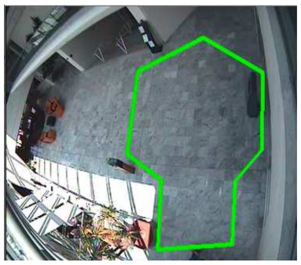
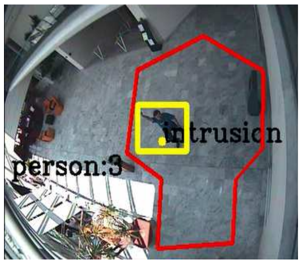
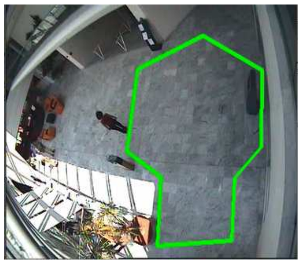
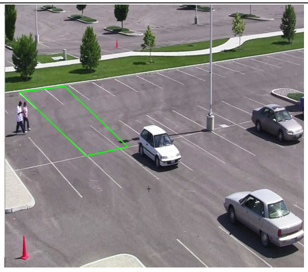
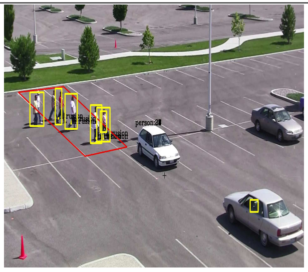
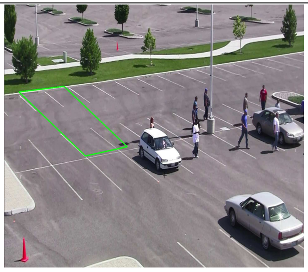
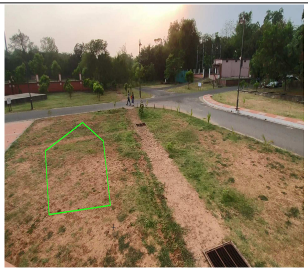
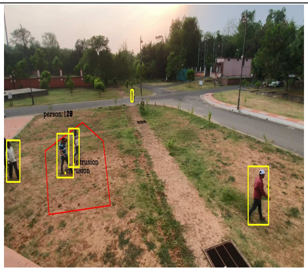
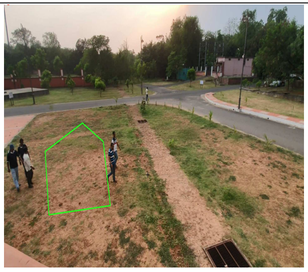
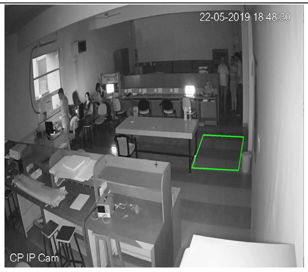
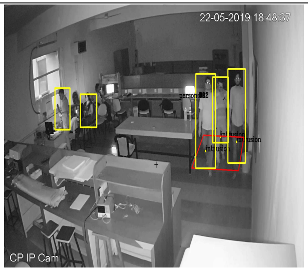
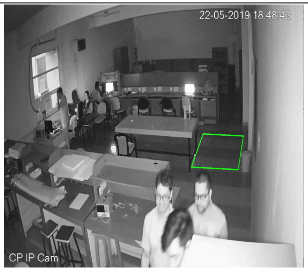
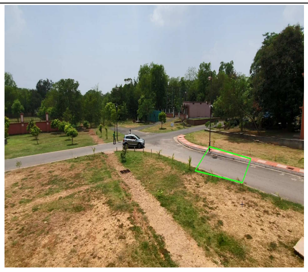
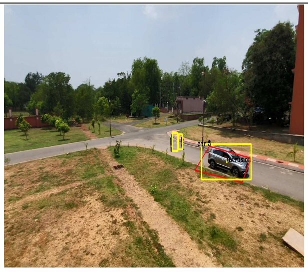
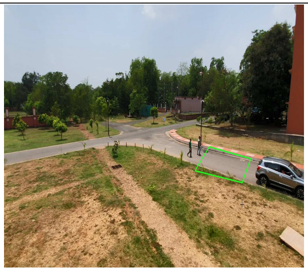
Dataset	Environment	ROI	Intrusion	No Intrusion
CAVIAR [15]	Single person, indoor environment, offline processing			
PNNL Parking [16], [17]	Multiple persons, outdoor environment, offline processing			
Live video stream (inside NITR campus)	Multiple persons, outdoor environment, online processing			
Live video stream (inside ISDR Lab, NITR campus)	Multiple persons, indoor environment, night condition, online processing			
Live video stream (inside NITR campus)	Car, outdoor environment, day condition, online processing			

TABLE IV: Comparison of Performance Parameters of the Proposed IDS with the State-of-art

Technique	**Platform**	**Video Dataset**	**Object Detection**	**Object Tracking**	**Intrusion**	**fps**	**Accuracy (%)**
[8]	CPU	PETS, Own video	Modified GMM	Kalman filter	Centroid based	Not available	88.96
		Their own video	Modified GMM	Kaman filter	Centroid based	Not available	83.50
[9]	CPU	Their own dataset	Morphological operations and background subtraction	Kalman filter	ROI based	Not available	Not available
[10]	CPU	Their own datset	Adaptive background subtraction	No Tracking	Centroid based	Not available	Not available
Proposed IDS	GPU	CAVIAR	YOLO based	SORT algorithm	Shifted center of mass	32	**91**
		Parking				33	**92**
		Our Live Video				29	**89**