IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Real-Time Object Detection Using The YOLO Algorithm With The Opencv Framework

Ms. Shaik Ishrath Anjum[*1], Ms.Syed Roohi[*2] , Ms.Vibudi Divya Priya[*3], Ms.Venicherla Bhargavi[*4] , Mr.B.Avinash[*5]

[1,2,3,4]B. Tech Students, Department of Information Technology,
Vasireddy Venkatadri Institute of Technology, Pedakakani Mandal, Nambur,
Guntur-522508 Andhra Pradesh, India.
[5]Assistant Professor, Department of Information Technology,
Vasireddy Venkatadri Institute of Technology, Pedakakani Mandal, Nambur,
Guntur-522508 Andhra Pradesh, India.
vvitguntur.com

Abstract:

In the realm of computer vision, object detection poses a significant challenge, especially in accurately localizing and categorizing objects within images. This paper introduces a novel method that utilizes a pre-trained deep learning model to detect objects in real time by processing webcam-captured images and video streams. Leveraging the renowned speed and accuracy of the YOLO algorithms, the system efficiently counts and monitors specific objects in each video frame, making it especially apt for visualization-focused applications. By integrating the YOLO model with the COCO dataset, the approach outperforms existing techniques in terms of precision and real-time responsiveness.

Keywords: Computer Vision, Object Detection, YOLO algorithm, Deep Learning, Open-CV, COCO dataset.

1. INTRODUCTION

Object detection stands at the forefront of technological advancements, utilizing AI to identify and locate objects within images, videos, or real-time webcam feeds. This technique finds its roots in a plethora of applications, spanning from self-driving vehicles and security systems to retail and healthcare innovations. As deep learning algorithms become more refined, the power of object detection in interpreting the visual world around us has exponentially grown, paving the way for unprecedented applications across various industries.

Among the myriad of deep learning algorithms available for object detection, a few have emerged as industry standards due to their efficiency and accuracy. YOLO, Faster R-CNN, and SSD are some of the most prominent algorithms in this domain. To harness these algorithms for object detection, one commences by gathering a labeled dataset of images or videos containing the objects of interest. Post-training, the model is primed to detect these objects in any new visual content introduced to it.

The growing popularity and efficacy of object detection via deep learning cannot be understated. Machines, equipped with this capability, can delve deeper into understanding and interacting with visual stimuli. In light of this, we present a Python script designed to leverage the nuances of computer vision, specifically focusing on real-time object detection and counting. This script exemplifies the potential and versatility of integrating deep learning into practical applications.

We are diving deep into the realm of computer vision, focusing primarily on object detection. Harnessing the power of sophisticated algorithms like YOLO, we are developing solutions capable of real-time identification and localization of objects within diverse visual mediums, from static images to dynamic video streams. Recognizing the wide applications of such technology, from security to retail and beyond, we've ensured that our approach is both robust and adaptable. Our project is rooted in extensive research, iterative testing, and a commitment to precision, reflecting our ongoing dedication to pushing the boundaries of what's possible in the ever-evolving world of AI-driven solutions.

2. LITERATURE SURVEY

The most important step in the software development process is the literature review. This will describe some preliminary research that was carried out by several authors on this appropriate work and we are going to take some important articles into consideration and further extend our work.

Real Time Object Detection System with YOLO and CNN Models - Vishwanatha, Chandana RK, Ramachandra A.C (2022): The paper focuses on reviewing object detection techniques, particularly the YOLO (You Only Look Once) algorithm and its evolved versions. The problem statement is to evaluate and understand the differences and similarities between various versions of YOLO and between YOLO and convolutional neural networks (CNNs).

YOLO Objection detection using Open-cv - Akshara Guptha, Aditya Verma, Aditya Yadav (2021): This paper introduces object detection and algorithm to recognize objects where the object is located inside the given input image on a variety of vehicles and other means of transportation and footing.

Identification and Detection of Automotive Door Panel Solder Joints based on YOLO - Zhimin Mol, Liding Chen, Wen-jing You (2019): This domain is related to automotive manufacturing and quality control. The problem was to identify and detect solder joints in automotive door panels in real-time, improving the efficiency and flexibility of the production process.

Obstacle Detection based on YOLO and Light Field Camera - Rumin Zhang, Yifeng Yang (2018): This work likely addressed the need for obstacle detection, especially in indoor environments, by combining the YOLO object detection algorithm with a light field camera. It aimed to classify and mark objects within the image.

Pedestrian Detection Based Yangon YOLO Network Model - Wenbo Lan, Jianwu Dang, Ping Wang, Song Wang (2018): The domain appears to be pedestrian detection, with a focus on improving this aspect using the YOLO network within the Yangon model. Specific problem details are not provided in the text you shared.

Object Detection Based on YOLO Network - Yufan Tao (2018): A generalized object detection network was developed by applying complex degradation processes on training sets like noise, blurring, rotating, and cropping of images. The model was trained with the degraded training sets, which resulted in better generalizing ability.

3. EXISTING SYSTEM

The existing object detection system uses traditional computer vision methods to recognize things in pictures or video frames through a systematic procedure. It goes through pre-processing for quality improvement, such as re-sizing and noise reduction, after sourcing input from cameras or video files. Then, traditional techniques are employed for detection, clearly designating each object found with a bounding box. But the system is just designed for object recognition; it cannot count or analyse detected things quantitatively; therefore, it is more appropriate for applications that are more concerned with simple identification than with tracking or in-depth analysis.

1. **Limited Functionality:** Without object counting, the system can only detect and draw bounding boxes around objects. It lacks the ability to provide insights about the quantity or distribution of objects in the scene, which can be important in various applications.

2. **Inability to Track Objects Over Time:** The SSD, by itself, doesn't provide tracking capabilities. It doesn't maintain object identities across frames, which can be crucial in scenarios where objects move or interact.

3. **No Object Identification:** The system can detect objects but doesn't identify them. You won't get information about the specific classes or labels of detected objects. This is important for understanding what the objects are.

4. **Accuracy and False Positives:** SSD, like any object detection model, can have false positives and false negatives. Without counting and object identification, it's challenging to assess and mitigate these errors effectively.

5. **Lack of Insights for Decision-Making:** Without object counting, the system won't provide insights into trends, patterns, or anomalies in the data, limiting its utility for decision-making.

6. **Resource Demands:** SSD can be computationally intensive, and this remains true even without object counting. The system may still require a powerful CPU/GPU to run in real-time.

7. **Difficulty in Object Management:** Managing and analyzing multiple objects in real-time without counting can be complex. It doesn't provide a structured way to keep track of object interactions or behaviors.

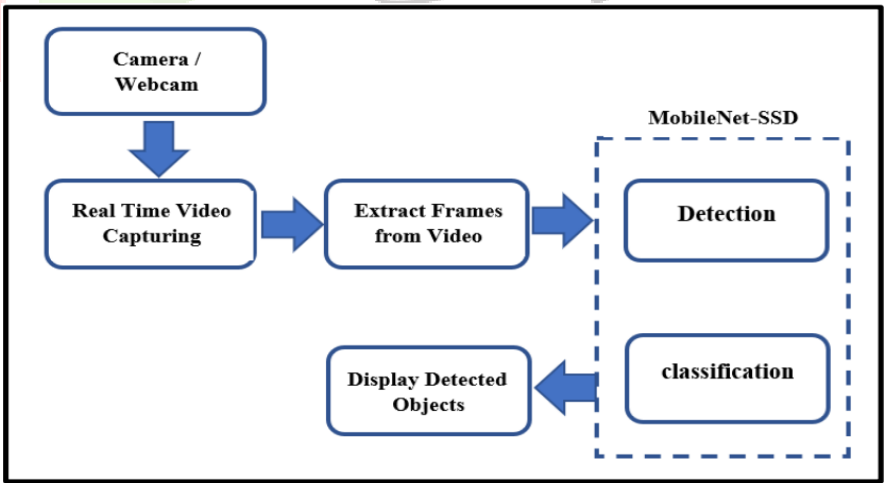


Figure 1: Existing System with Mobile Net SSD

4. PROPOSED SYSTEM

The proposed system leverages Vid Gear, OpenCV, and cv lib to implement real-time object detection in video streams. Vid Gear is utilized for efficient video input handling, while cv lib facilitates object detection in individual frames. As objects are identified, the system outlines them with bounding boxes, offering a clear visual representation of their locations. Additionally, the system continuously tallies specific objects based on their type, and this count is overlaid onto the video frame. For an enhanced user experience, the processed video stream is presented in full screen, ensuring optimal visualization of detected objects.

Principle features of the proposed work could include:

1. **Real-time Detection and Enhanced Accuracy:** The system offers instant object identification from various video sources, ensuring real-time responsiveness. Using cv lib's object detection, the accuracy is significantly bolstered, which ensures reliable results. Additionally, visual clarity is achieved by emphasizing identified objects with bounding boxes.

2. **Object Counting and Efficient Processing:** The system is capable of precise quantification of specific detected objects, providing a clear count for analytical or monitoring purposes. This counting is facilitated by the efficient real-time processing made possible by vid gear's Cam Gear, guaranteeing minimal latency in video stream analysis.

3. **Adaptability and User-friendly Display:** Crafted with flexibility in mind, the system can easily adjust to different sources and environmental scenarios. Moreover, its user-friendly display ensures that the processed video frames are not only informative but also visually engaging, enhancing user comprehension.

4. **User Interaction and Graceful Exit:** The system encourages an immersive experience via a full-screen window display, ensuring users are fully engaged with the content. In addition to this, it's equipped with a built-in mechanism that allows users to safely and smoothly terminate the application whenever desired, ensuring system integrity and user control.

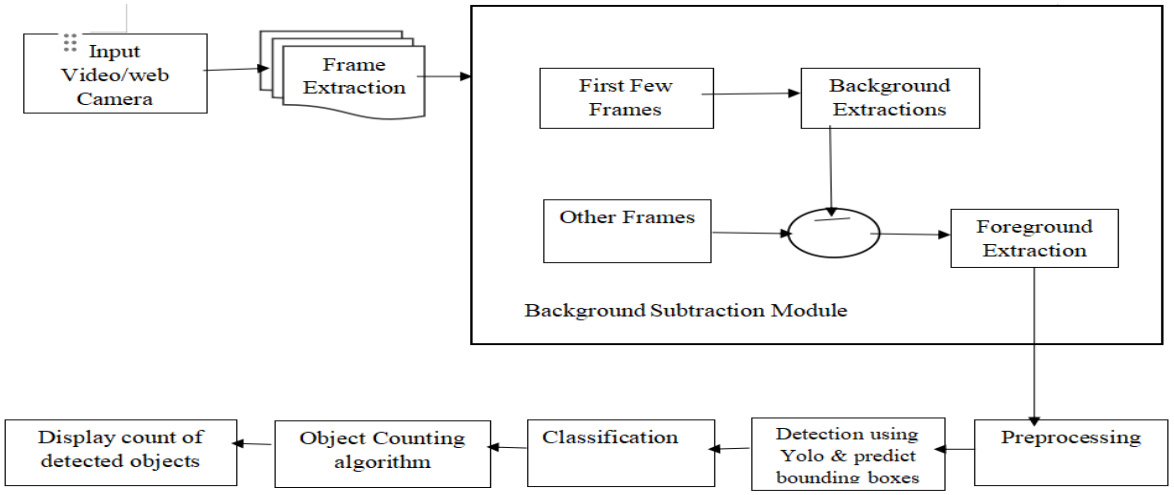


Figure 2: Proposed system with YOLO implementation

4.1 PROPOSED DATA SET:

The COCO (Common Objects in Context) dataset is a widely used computer vision dataset that contains images with object annotations, segmentation masks, and captions. Certainly, here is a list of all 80 classes in the COCO dataset:

COCO dataset:

person	fire hydrant	elephant	skis	wine glass	broccoli	dining table	toaster
bicycle	stop sign	bear	snowboard	cup	carrot	toilet	sink
car	parking meter	zebra	sports ball	fork	hot dog	tv	refrigerator
motorcycle	bench	giraffe	kite	knife	pizza	laptop	book
airplane	bird	backpack	baseball bat	spoon	donut	mouse	clock
bus	cat	umbrella	baseball glove	bowl	cake	remote	vase
train	dog	handbag	skateboard	banana	chair	keyboard	scissors
truck	horse	tie	surfboard	apple	couch	cell phone	teddy bear
boat	sheep	suitcase	tennis racket	sandwich	potted plant	microwave	hair drier
traffic light	cow	frisbee	bottle	orange	bed	oven	toothbrush

Figure 3: COCO dataset

5. IMPLEMENTATION

The implementation of real-time object detection follows these key steps:

1. **Input Video/web Camera:** This component represents the source of the video stream, which could be a physical camera capturing real-time footage or a web camera capturing video from a web source.

2. **Frame Extraction:** Frames are individual images extracted from the video stream. This includes both the initial frames for system initialization (First Few Frames) and the subsequent frames (Other Frames) processed for various tasks.

3. **Background Subtraction Module (Including Background Extraction and Foreground Extraction):** The Background Subtraction Module is a critical component of video processing systems. It encompasses two key processes:
   3.1 Background Extraction: Identifying and storing the stationary background from the video.
   3.2 Foreground Extraction: Isolating the dynamic objects in the video, effectively separating them from the background.

4. **Preprocessing**: The foreground frames undergo preprocessing, which can include tasks like noise reduction, image enhancement, or other techniques to improve object detection accuracy.

5. **using YOLO & Predict Bounding Boxes:** YOLO, a deep learning model, is used for object detection within pre-processed frames. Bounding boxes are drawn around the detected objects to locate them.

```
bbox, label, conf = cv. detect_common_objects(frame)
frame = draw_bbox (frame, bbox, label, conf)
```

6. **Classification:** This step involves categorizing the detected objects into predefined classes or categories. For example, distinguishing between cars, pedestrians, or other objects.

7. **Object Counting Algorithm:** An algorithm is employed to count the number of specific objects of interest within the frames. This could include counting the number of cars in a traffic scene or people in a crowd.

```
object_counts = {}
for obj in label:
    object_counts[obj] = object_counts.get (obj, 0) + 1
```

8. **Display Count of Specific Objects:** The final output of the system is the count of specific objects, which can be displayed and used for various applications such as traffic monitoring, crowd control, or management.

```
text = ', '.join([f"{obj}={count}" for obj, count in object_counts. items()])
cv2.putText(frame, text, (50, 60), cv2.FONT_HERSHEY_PLAIN, 3, (255, 0, 0), 3)
```

```
cv2.namedWindow("FRAME", cv2.WND_PROP_FULLSCREEN)

cv2.setWindowProperty("FRAME",cv2.WND_PROP_FULLSCREENcv2.WINDOW_FULLSCREEN)
cv2.imshow("FRAME", frame)
```

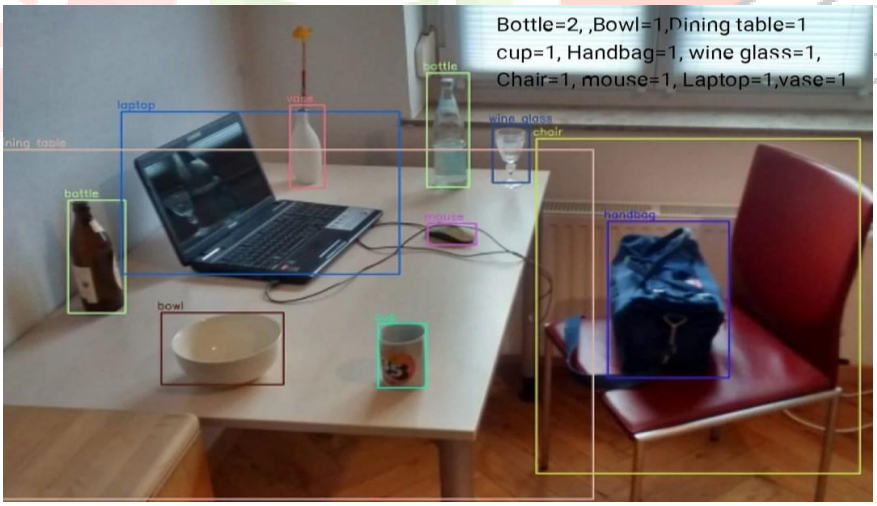
6. EXPERIMENTAL RESULTS

From the below two figures, it can be seen that the proposed model is more accurate in order to prove our proposed system.

Object Detection in Video Feed

Explanation: From the above window we can see that the objects are detected using boundary boxes and
are categorized. The count of specific objects is displayed on the output window.

Object Detection in Web Camera



Explanation: From the above window we can see that the objects are detected in the live web camera and the count of objects is specified.

7. CONCLUSION

In conclusion, Utilizing the YOLO (You Only Look Once) algorithm, object detection in video streams and web cameras is expedited, outperforming other models in terms of speed. This capability, combined with the feature to count and display objects, finds utility across diverse domains such as security, surveillance, retail, marketing, and traffic management. YOLO showcases immense promise for real-time object detection, furnishing pivotal insights and fostering smart decision-making. In essence, leveraging the YOLO model for object detection in videos promises significant advancements, with its potential set to further amplify through ongoing research and refinement.

8. FUTURE ENHANCEMENT

As autonomous machines, including mobile robots, quadcopters, drones, and upcoming service robots, become prevalent, the demand for object detection systems rises notably. This need extends to specialized applications such as nano-robots. For surveillance, incorporating a night vision mode in CCTV cameras becomes essential for effective nighttime visual tracking. When environmental conditions are favourable, these systems can also be optimized for underground mining operations. Furthermore, advanced detection of intricate objects like guns and other weapons is paramount for ensuring safety and security.

References

[1] Alvarez-Jimenez, Charkes Daikin. (2013). Why does human vision be superior to any other mammal, 143(1), 143–149.

[2] Anwas Jiwabi, Stip, E., & Kara, N. (2017) Learning Computer Vision through Magic and Python, 12(1),70-76.

[3] Georgia Razdana, Mukhtar Abbas, Amit Sharma (2018). Classifying everyday objects through computer vision, (13)(1).

[4] Grav-Dadhich, Shruti Sambit. Classifying objects- based on their outline and greyscale image (2018) (7).

[5] D Erhan, C Szegedy, A. Toshev, and D. Angelov. Scalable object detection using deep neural networks. In Computer Vision and Pattern Recognition (CVPR), 2014 IEBE Conference on, pages 2155-2162. IEEE. 2014.5.6.

[6] M-Everingham, S. M. A. Eslami, L. Van Gool, C: K. L. Williams, J. Winn, and A. Zisserman The Pascal visual object classes challenge A retrospective. International Journal of Computer Vision. 111(1):98-136, Jun. 2015. 2.

[7] PF Felzen szwalb, R. B Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models, transactions on Pattern Analysis and Machine Intelligence, 32(9): 1627-1645, 2010, 1,4.

[8] S. Ginosar, D. Haas, T. Brown, and J. Malik. Detecting people in cubist art. In Computer Vision-ECCV 2014 Workshops, pages 101-116. Springer, 2014.

[9] S. Gidaris-and N. Komodakis. Object detection- via a multi-region & semantic segmentation-aware CNN model. CORR, abs/1505.01749, 2015. 7.

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik Rich feature-hierarchies for accurate object detection and semantic segmentation. In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, pages 580 587, IEEE, 2014. 1, 4.