REAL TIME OBJECT DETECTION

PROJECT SYNOPSIS

OF MINOR PROJECT

BACHELOR OF TECHNOLOGY

COMPUTER SCIENCE AND ENGINEERING

SUBMITTED BY

Akash Kumar	Aditya Kumar	Anmol Kumar
(2115008/2104062)	(2115003/2104057)	(2115016/2104070)

GURU NANAK DEV ENGINEERING COLLEGE

LUDHIANA

Table of content

INTRODUCTION

Most of the cases, Humans performs work by observing their environment and perform action accordingly, and the observation of the environment is done using their eyes, means that they take input through their eyes and this observed data is send to their mind for processing and process is done through their mind instantly in real time. The brain observes this data and detect different object and send command to our body part to do action.

In our project we are aims to make same system which can perform that work in real time just like the human brain. here our camera will work as an eye for our system and we will use neural network's different techniques to make a system which can detect different object in our scene in real time instant process.

There are several techniques which we use to build such project such as

- Convolutional Neural Networks (CNNs): These are deep learning models that consists of multiple layers of neural that learns features from image data.
- Region based Methods: These are methods that divide an image into multiple regions and apply CNNs to each region to detect objects.
- Single shot Methods: These are methods that perform object detection in one pass over image, using a single CNN. Some example of single shot methods are You Only Look Once (YOLO) and Single Shot Multi-Box Detector (SSD).

There are many fields where we need such kinds of systems which can be used to detect object like Video surveillance, Crowd counting, Anomaly detection, Self -Driving Cars and many more. And most importantly we can use these techniques in fields of robotics where robots can detect different object.

RATIONALE

Now a days most of the works are performed by machine which makes time consuming work easier and less time are consumed compared to humans. Such as in a car manufactures company, the body parts are combined using computer vision and join it faster than a human being. We can use computer vision in many fields which is mentioned above to make our work easy and fast.

In many fields where the presence of humans is dangerous, in that case we can use these machines to perform works and we need computer vision to detect surrounding and make further decision. That's why we need real time object detection project to use it in these fields.

OBJECTIVES

The objective of our real time object detection and identification project are as follow:

1. Develop a Real Time Object Detection System using YOLO.
2. To integrate Real Time object Visualization using deep learning.
3. Validation and Testing with Existing System.

.

LITERATURE REVIEW

1. **"Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks" (2015) by Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun:**
   This paper introduces the Faster R-CNN architecture, which employs a Region Proposal Network (RPN) to generate region proposals. It achieves state-of-the-art results in terms of both accuracy and speed, making it a landmark in real-time object detection.
2. **"YOLO9000: Better, Faster, Stronger" (2016) by Joseph Redmon and Santosh Divvala:**
   YOLO (You Only Look Once) is known for its speed. This paper, an extension of the original YOLO model, presents YOLO9000, capable of detecting over 9000 object categories in real-time. It introduces a unified detection model that can handle object detection and classification simultaneously.
3. **"SSD: Single Shot MultiBox Detector" (2016) by Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, and Scott Reed:**
   SSD is designed for real-time object detection by directly predicting object categories and bounding box offsets. The paper emphasizes the efficiency of SSD by using a single deep neural network to generate predictions at multiple scales, resulting in high accuracy and real-time performance.
4. **"Mask R-CNN" (2017) by Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick:**
   Mask R-CNN builds upon Faster R-CNN by extending it to also predict object masks in addition to bounding boxes and class labels. This is particularly useful for tasks that require pixel-level accuracy, such as instance segmentation. The paper provides a comprehensive solution for both object detection and segmentation.

5. **"EfficientDet: Scalable and Efficient Object Detection" (2020) by Mingxing Tan, Ruoming Pang, Quoc V. Le:**

   EfficientDet introduces a scalable and efficient object detection model by systematically balancing model accuracy and computational efficiency. It uses a compound scaling method that optimizes the model architecture to achieve better performance on various devices. This paper addresses the trade-off between accuracy and speed, making it valuable for real-time applications.

FEASIBILITY STUDY

- **Technical Feasibility:** Here are key aspects to consider when evaluating the technical feasibility of a real-time object detection system:
    - Algorithm Selection: Choose an object detection algorithm that aligns with the real-time processing requirements. Common algorithms include Faster R-CNN, SSD (Single Shot Multibox Detector), YOLO (You Only Look Once), and variants.
    - Hardware Requirement: Evaluate the computational resources required for real-time processing. Consider the capabilities of available hardware, such as GPUs (Graphics Processing Units), TPUs (Tensor Processing Units), or specialized AI accelerators.
    - Software Framework: Choose appropriate deep learning frameworks and libraries for developing and deploying the object detection model. Popular frameworks include TensorFlow, PyTorch, and Caffe.
- **Operational Feasibility:**
    - Privacy and Security Concerns: Address data privacy concerns of users and implement secure data handling practices.
    - Deployment and Integration: Consider how the system will be deployed (e.g., standalone application, integration with existing systems) and the necessary resources for maintenance and updates.
    - User Acceptance and Training: Ensure user friendliness and provide necessary training or support for users interacting with the system.

METHODOLOGY/PLANNING OF WORK

Below is the generalized methodology that can be adapted for developing this real time object detection system.

1. Define Objective and Scope: As mentioned above that the scope of our project is to detect, identify and locate an object from an given video or images.
2. Data Collection and Annotation: We need to gather a diverse and representative dataset that includes images or video frames relevance with the project.
3. Select Object Detection Algorithm: We need to choose an algorithm which is most feasible with our project. Common choices include faster R-CNN,SSD,YOLO or their variants.
4. Preprocess Data: Implement data preprocessing steps to prepare the input data for the object detection model. This may include resizing images, normalizing pixel values, and augmenting the dataset for increased diversity.
5. Model Training: Train the chosen object detection model using the annotated dataset. Consider using transfer learning with pre-trained models to expedite training and improve performance.
6. Optimize Model for inference: Optimize the trained model for real-time inference. This may involve model quantization, pruning, or other techniques to reduce the model size and computational requirements.
7. Choose Hardware and Software Stack: Select the appropriate hardware for running the object detection model in real-time. Consider GPUs, TPUs or specialized accelerators depending on the computational requirements. Choose a software stack, including deep learning frameworks (e.g., TensorFlow, PyTorch), for model deployment.
8. Testing and Validation: Conduct thorough testing and validation of the real-time object detection system.

FACILITIES REQUIRED FOR PROPOSED WORK

Implementing a real-time object detection system requires several facilities and resources to ensure its effective development, deployment, and maintenance. Such as we requires a high performance Hardware which is essential for real time object detection this may include GPUs(Graphic Processing Units), TPUs(Tensor Processing Units) or other specialized hardware accelerators to handle the computational load efficiency. We also requires different software such as python and its libraries such as TensorFlow, PyTorch, and object detection library such as OpenCV, Detectron2 etc. and most important we require data to train our model and then testing it to check how good our system is working.

EXPECTED OUTCOMES

After completion of our project we expect that our project will detect the object in image or video which is provided to our system. The system can be able to identify and can tell us what kind of object is shown in the image or video in voice form. We also expect that the project can perform in high accuracy and efficiency in real time environment. There should also be enhanced safety and security which lead to prevent any error in the system.

REFERENCES

1. Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks." In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015.
2. Joseph Redmon and Santosh Divvala. "YOLO9000: Better, Faster, Stronger." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
3. Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, and Scott Reed. "SSD: Single Shot MultiBox Detector." In Proceedings of the European Conference on Computer Vision (ECCV), 2016.
4. Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask R-CNN." In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017.
5. Mingxing Tan, Ruoming Pang, Quoc V. Le. "EfficientDet: Scalable and Efficient Object Detection." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.