Navigating the Point: Analyzing Pointing Gestures for Object Detection

Adarsh Saripalli* Shyam Ganatra* Somisetti Kasinath* SriSatya JayaKrishna Vaddi*
Arizona State University

1 Introduction

Object detection is a critical component of computer vision, utilized across diverse sectors from autonomous navigation to interactive robotic systems. Traditional methods, however, suffer from inefficiencies as they scan entire images to locate objects, often processing vast irrelevant regions[19]. This not only consumes excessive computational resources but also hampers real-time applications where fast, efficient decision-making is paramount. For example, in a retail environment, conventional object detection might continuously analyze an entire store layout to track inventory or monitor customer movements. This method wastes computational power on empty shelves or areas without significant customer activity, leading to slower response times and increased operational costs.

To overcome these limitations, our project proposes an approach by integrating human pointing gestures into the detection system. This method focuses the system's attention on specific areas indicated by a human operator, significantly improving both the speed and accuracy of object detection. By honing in on relevant parts of the image, our approach interprets the semantic context of scenes, which is crucial for decision-making on resource-limited edge devices like drones. This targeted detection method promises to revolutionize applications by reducing unnecessary processing and prioritizing critical data, ultimately leading to more efficient and context-aware computing in real-time environments.

2 Related Work

Object detection technology has seen remarkable progress from the initial Viola-Jones Detector[1], a pioneer in real-time face detection, to advanced systems like AlexNet[2], which transformed image classification with deep convolutional layers. The shift from basic to intricate methods continued with the introduction of R-CNN, which significantly improved object localization by integrating region proposals with convolutional neural networks (CNNs). This evolution saw a groundbreaking shift with YOLOv1[3], which increased processing speeds by treating detection as a single regression problem across the entire image, a departure from traditional patch-based methods. Further advancements brought about techniques like CornerNet[4], which uses paired corner points for object detection, eliminating the need for predefined anchor boxes and simplifying the detection framework. In the field of dynamic interaction, early endeavors utilized multiple cameras to detect pointing actions in real-time[9]. For instance, research focused on glove-free interfaces[5], which laid the foundation for subsequent studies exploring uncalibrated stereo vision[7]. These advancements paved the way for further exploration into real-time detection and estimation of omnidirectional pointing gestures, providing valuable insights into the challenges and opportunities in this field. The PKU-MMD model became a significant development for gesture recognition. Deepoint[12] focuses on extracting pointing gestures and training on RGBD datasets, enhancing the dynamics of human-machine interaction.

Building on these technologies, our project advances the Faster R-CNN framework with "Deepoint," a system designed to concentrate detection on areas specifically indicated by human gestures. This integration not only simplifies the detection process but also significantly enhances efficiency in scenarios involving human-robot interaction, where quick and precise detection is essential. This targeted approach optimizes resource usage and improves interaction quality, marking a step forward in object detection technology for efficient, context-aware processing in various applications.

3 Baseline results

Our project's baseline relies on DeePoint, a deep neural network designed for pointing gesture recognition. Implementing DeePoint involved capturing 1200 video frames on iPhone 14 Pro videos with a depth sensor and test them on the DeePoint model, for statistical accuracy, we also used the roughly 1,60,000 frames from DP dataset, combining all the data. Challenges arose with OpenGL on Mac, resolved by switching to SoL computing. GPU consumption fluctuations (80% peak to 10%) were observed from frame to frame, aided by 12 CPU cores at 70% utilization. Successful video generation demonstrated DeePoint's potential, with results available for analysis on our Google Drive: `https://drive.google.com/drive/folders/1FfVAFbd2YZucIGEEUn6W89iuFjXGNKx_?usp=sharing`. DeePoint's baseline performance highlights challenges and optimizations for streamlining implementation.

Table 1: Pointing direction estimation errors

Set split	Value
Temporal split	22.83°
Scene split	28.47°
Person split	22.50°

Table 2: Recall and precision for the pointing action detection

Set split	Accuracy	Precision
Temporal split	0.612	0.837
Scene split	0.642	0.683
Person split	0.479	0.816

For evaluation, our dataset with around 161,200 frames from 4 people in three environments: Indoor environment without noise[Fig 1.1], Indoor environment with noise[Fig 1.3] and Outdoor environment[Fig 1.2]. We split the data into three parts: Temporal Split (to check how well the model performs for each person across different sessions), Scene Split (to see if the model can adapt to different environments) and Person Split (to understand if the model works differently for different people). We have trained the models using these splits and measured their performance [Table 2] in terms of pointing direction accuracy and pointing detection recall/precision and errors [Table 1] are calculated as the average angular difference between the predicted pointing direction and the ground truth direction across all instances or frames in the dataset. We used a learning rate of 10^{-4} with the Adam optimizer and a batch size of 64.

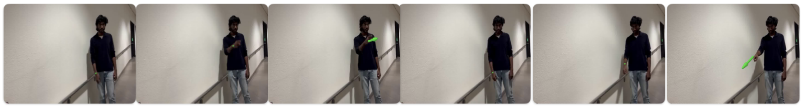


Fig 1.1 : Pointing gesture recognition with right hand and model fed as right in an indoor environment

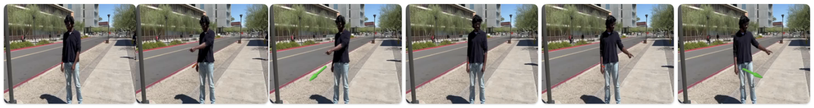


Fig 1.2 : Pointing gesture recognition with right hand but model fed as left in an outdoor environment

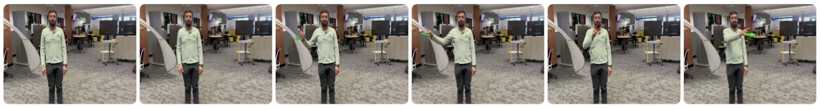


Fig 1.3 : Pointing gesture recognition with right hand but model fed as right in an indoor environment with noise

4 Our Contribution

Building upon our baseline "Deepoint" system, we have significantly augmented its capabilities by integrating it with a sophisticated object detection framework, creating a hybrid system that combines the strengths of both domains. This integration results in a more comprehensive and contextually aware object detection system.

Outputs from the Deepoint pointing gesture recognition network are directly fed into our object detection pipeline. This dual-network approach enables a contextual understanding of scenes. Rather than processing the entire image independently, our system utilizes the directional vectors from

Deepoint to guide the Faster R-CNN object detection network. This approach focuses the network's attention on areas most likely to contain relevant objects, drastically reducing the computational overhead of traditional exhaustive image searches and enhancing both processing speed and efficiency.

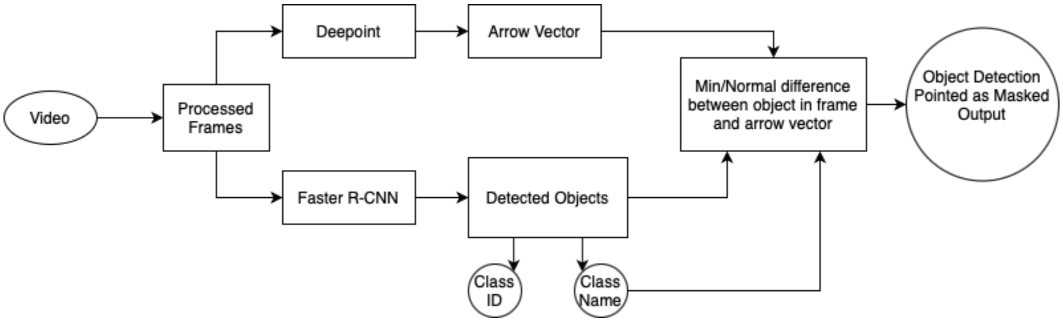

Video
Processed Frames
Deepoint
Arrow Vector
Faster R-CNN
Detected Objects
Class ID
Class Name
Min/Normal difference between object in frame and arrow vector
Object Detection Pointed as Masked Output

Figure 1: Overview of the Enhanced Object Detection Pipeline Integrating Human Pointing Gestures.

By customizing the Fast R-CNN model, which incorporates a robust ResNet50 backbone, our system dynamically prioritizes regions of interest based on real-time user input. This dynamic prioritization leads to more accurate detections as the system concentrates on areas identified by user pointing, improving the relevance and accuracy of the detection results. We have streamlined the integration between the object detection and visualization modules to ensure seamless and real-time processing. This optimization is essential for applications requiring immediate response, such as autonomous driving and surveillance. We've also enhanced error handling and data consistency between modules to improve the system's reliability and robustness.

Our enhanced model now delivers enriched visual feedback. It draws bounding boxes around detected objects and annotates these with the object's class name and confidence levels. The visualization includes additional elements like arrows or highlights that emphasize specific features or aspects of the detected objects, significantly enhancing the utility for end-users.

5 Results

In our comprehensive results analysis of the enhanced object detection system, we observed significant improvements in detection accuracy and efficiency, facilitated by the integration of the "Deepoint" pointing gestures with the Faster R-CNN model. The analysis of 2086 frames derived from 24 videos processed at 15 frames per second revealed insightful details about the system's performance in real-world scenarios.

1. **True Positive Rate:** The system achieved a high rate of true positives, with 1022 frames correctly identifying both the arrow direction and the detected object. This indicates a strong alignment between the pointing gesture input and the object detection output, reflecting high system reliability in scenarios where user interaction directly influences detection focus.
2. **False Positive and False Negative Rates:** There were 246 instances where the system erroneously detected objects that were not pointed at (false positives) and 212 cases where it failed to detect the correct object despite accurate pointing (false negatives). These metrics are crucial for understanding the limitations in the current model, particularly in terms of its sensitivity and specificity.

Overall Accuracy: The confusion matrix and subsequent calculations revealed that the system maintained a robust mean Average Precision (mAP) of 76%. This metric is particularly telling as mAP is a comprehensive measure that considers both precision and recall, providing a balanced view of the model's overall performance across various classes.

The standard deviation in the detection accuracies, particularly in the context of true positives and false negatives, was relatively low. This low variability in performance across multiple frames and scenarios suggests that the system behaves consistently under different conditions. The precise values of standard deviation were calculated based on the aggregation of results across all tested frames, providing a statistical basis for evaluating the model's reliability. The low standard deviations

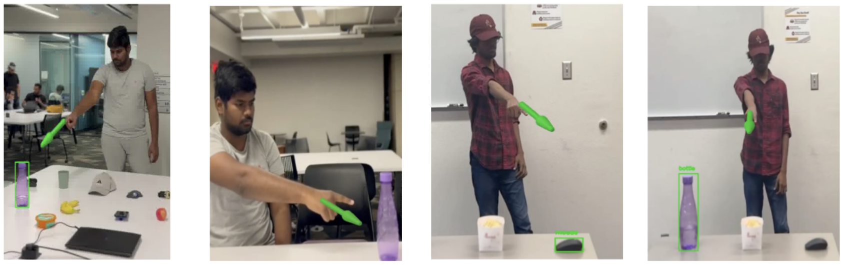
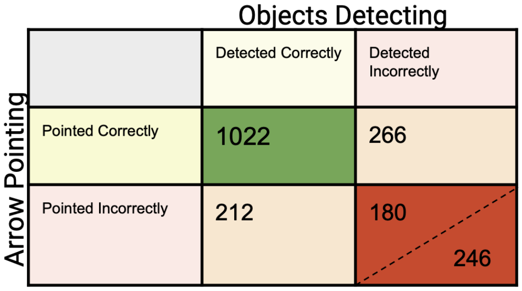


Figure 2: Model performance upon Pointing gesture arrow and object detection (a) True Positive (b) False Negative (c) False Positive (d) True Negative respectively



Arrow Pointing	Objects Detecting: Detected Correctly	Objects Detecting: Detected Incorrectly
Pointed Correctly	1022	266
Pointed Incorrectly	212	180 246

Figure 3: Confusion Matrix for the Enhanced Object Detection Model, illustrating the classification accuracy and misclassifications across different object categories.

combined with a high mean Average Precision allow us to claim victory in achieving a robust and reliable pointing gesture-enhanced object detection system. The consistent performance across a diverse set of video inputs and object scenarios underscores the system's effectiveness in real-world applications. The mean Average Precision comparison for observed classes from the COCO 2017 dataset, as outlined in Table 3, provided a benchmark against industry standards. This comparison not only highlighted the strengths of our system but also shed light on specific classes where the model could be further optimized.

Class Name	Faster R-CNN (Baseline mAP)	Our Model (mAP)
apple	70.1	72.6
banana	80.6	83.4
hat	78.2	80.4
laptop	69.9	66.4
bottle	49.9	52.8
computer mouse	76.3	75.9
chair	79.8	75.3
book	52.2	69.6

Table 3: mean Average Precision comparison for observed classes from COCO 2017 dataset

6 Conclusion

This project has effectively demonstrated how the integration of human pointing gestures with traditional object detection systems can significantly enhance the efficiency and accuracy of these systems. Our innovative approach not only optimizes computational resources by focusing on user-indicated areas but also enriches the system's contextual awareness, thereby facilitating more relevant and precise detections. This integration represents a significant advancement in the field of computer vision, suggesting substantial potential for further developments in interactive systems where human input can directly influence and improve machine perception and decision-making. The successful outcomes from this project lay a strong foundation for future research into creating more adaptive, intuitive, and user-focused detection systems.

References

[1] Cen, Kaiqi. “Study of Viola-Jones Real Time Face Detector.” (2016).

[2] Iandola, Forrest N., Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally and Kurt Keutzer. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size.” ArXiv abs/1602.07360 (2016): n. pag.

[3] Redmon, Joseph, Santosh Kumar Divvala, Ross B. Girshick and Ali Farhadi. “You Only Look Once: Unified, Real-Time Object Detection.” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015): 779-788.

[4] Law, Hei and Jia Deng. “CornerNet: Detecting Objects as Paired Keypoints.” International Journal of Computer Vision 128 (2018): 642 - 656.

[5] Masaaki Fukumoto, Kenji Mase, and Yasuhito Suenaga Dec. 1992 Real-time detection of pointing actions for a glove-free interface In ProcI APR Workshop on Machine Vision Applications, pages 473–476.

[6] Hiroki Watanabe, Hitoshi Hongo, Mamoru Yasumoto, and Kazuhiko Yamamoto. Detection and estimation of omni- directional pointing gestures using multiple cameras. In Proc. of IAPR Workshop on Machine Vision Applications, pages 345–348, Jan. 2000.

[7] Roberto Cipolla, Paul A. Hadfield, and Nicholas J. Hollinghurst. Uncalibrated stereo vision with pointing for a man–machine interface. In Proc. of IAPR Workshop on Machine Vision Applications, pages 163–166, 1994.

[8] Dai Fujita, Takashi Komuro, Michael M. Bronstein, and Carsten Rother. Three-dimensional hand pointing recognition using two cameras by interpolation and integration of classification scores. In Proc. of European Conference on Computer Vision Workshops, pages 713–726, Sept. 2015

[9] Roland Kehl and Luc Van Gool. Real-time pointing gesture recognition for an immersive environment. In IEEE International Conference on Automatic Face and Gesture Recognition, pages 577–582, Jan. 2004.

[10] Kai Nickel and Rainer Stiefelhagen. Pointing gesture recognition based on 3D-tracking of face, hands and head orientation. In Proc. of International Conference on Multimodal Interfaces, pages 140–146, Nov. 2003.

[11] Ren, Shaoqing, Kaiming He, Ross B. Girshick and Jian Sun. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.” IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (2015): 1137-1149.

[12] Shu Nakamura and Yasutomo Kawanishi and Shohei Nobuhara and Ko Nishino 2023 DeePoint: Visual Pointing Recognition and Direction Estimation.

[13] Dadhichi Shukla, O zgu r Erkent, and Justus Piater. Probabilistic detection of pointing directions for human-robot interaction. In Proc. of International Conference on Digital Image Computing: Techniques and Applications, pages 1–8, Nov. 2015.

[14] Ana Fernandez, Luca Bergesio, Ana M. Bernardos, Juan A. Besada, and Jose R. Casar. A kinect-based system to enable interaction by pointing in smart spaces. In Proc. of IEEE Sensors Applications Symposium, Sept. 2015.

[15] Bita Azari, Angelica Lim, and Richard Vaughan. Commodifying pointing in HRI: Simple and fast pointing gesture detection from RGB-D images. In Proc. of International Conference on Computer and Robot Vision, pages 174–180, May 2019.

[16] Naina Dhingra, Eugenio Valli, and Andreas Kunz. Recognition and localisation of pointing gestures using a RGB-D camera. In Constantine Stephanidis and Margherita Antona, editors, Proc. of HCI International 2020 - Posters, pages 205–212, July 2020.

[17] Shome S. Das. Precise pointing direction estimation using depth data. In Proc. of International Symposium on Robot and Human Interactive Communication, pages 202– 207, Aug. 2018.

[18] ShomeS.Das.A data set and a method for pointing direction estimation from depth images for human-robot interaction and VR applications. In Proc. of International Conference on Robotics and Automation, pages 11485–11491, May 2021.

[19] Stefan Mathe, Aleksis Pirinen, Cristian Sminchisescu; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2894-2902

[21] Shruti Jaiswal, Pratyush Mishra, and G.C. Nandi. Deep learning based command pointing direction estimation using a single RGB camera. In IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering, pages 1–6, Nov. 2018.