Speed Estimation using Lucas Kanade Optical Flow

Project Report

by

Rohith Puvvala Subramanyam

Paper link:

https://ieeexplore-ieee-org.libaccess.sjlibrary.org/stamp/stamp.jsp?tp=&arnumber=9486316

QR code to paper:

Code: https://github.com/Rohithps98/object-detection-speed-estimation

Problem definition:

The project aims to develop an advanced system for real-time object detection and accurate speed estimation in video streams. Object detection is crucial for applications like surveillance, autonomous navigation, and traffic management, while speed estimation provides essential information for tracking and predicting object motion. However, seamlessly integrating these capabilities poses significant challenges.

Robust and accurate object detection is a primary challenge. The system needs to detect multiple object classes in real time while maintaining high precision and recall rates, even in challenging environments with occlusions and varying lighting conditions.

Accurate speed estimation is also essential. The system must track objects across frames and estimate their velocities reliably, accounting for factors like occlusions, varying appearances, and complex trajectories.

Seamlessly integrating object detection and speed estimation is another challenge. The system needs efficient algorithms that combine the outputs of both modules, ensuring consistency and reliability in the results. Handling issues like object ID switches and track fragmentation is crucial for maintaining accurate representations of object velocities over time.

The project aims to overcome these challenges and develop an advanced system that excels in real-time object detection and accurate speed estimation. Successful completion will enable applications such as robust surveillance systems, intelligent transportation systems, and autonomous vehicles, contributing to advancements in various domains.

Project objectives:

1. **Develop an object detection system using the YOLOv5 model:** The primary objective is to build a robust and efficient object detection system that can accurately detect and classify objects within a video stream. This involves training the YOLOv5 model on a suitable dataset, fine-tuning the model parameters, and optimizing its performance to achieve high accuracy and real-time processing speed.

2. **Implement Lucas-Kanade optical flow for speed estimation:** The second objective is to integrate the Lucas-Kanade optical flow algorithm into the project pipeline. By utilizing the optical flow technique, the project aims to estimate the speed and motion vectors of objects detected in the video frames. This information will enable the calculation of object velocities, which can be useful for tracking objects and analyzing their movements.

3. **Integrate the object detection and speed estimation modules:** The third objective is to seamlessly combine the object detection and speed estimation components into a single

coherent system. This integration will involve designing an efficient pipeline that takes video input, performs object detection using YOLOv5, applies Lucas-Kanade optical flow to estimate object speeds, and produces the final output with annotated objects and their corresponding velocities.

4. **Evaluate the performance of the system:** The final objective is to thoroughly evaluate the developed system's performance in terms of object detection accuracy, speed estimation accuracy, and real-time processing capabilities. This evaluation will involve testing the system on various video datasets, measuring the detection and speed estimation accuracies against ground truth, and analyzing the system's overall efficiency and robustness. By achieving these objectives, the project aims to provide a comprehensive solution for real-time object detection and speed estimation, with potential applications in surveillance, traffic management, and autonomous systems.

Architecture

Yolov5

Yolov5 is the fifth version of the YoLo object detection system, based on a regression network. In contrast to the usual “regional candidate network”, YOLO unifies two steps of candidate region generation and recognition and considers the recognition task as a direct regression problem. As a result, speed and accuracy in the majority of object recognition tasks have been increased. YOLOv5 is composed primarily of “CSPDarknet53, spatial pyramid pooling (SPP), and a path aggregation network (PANet)”.

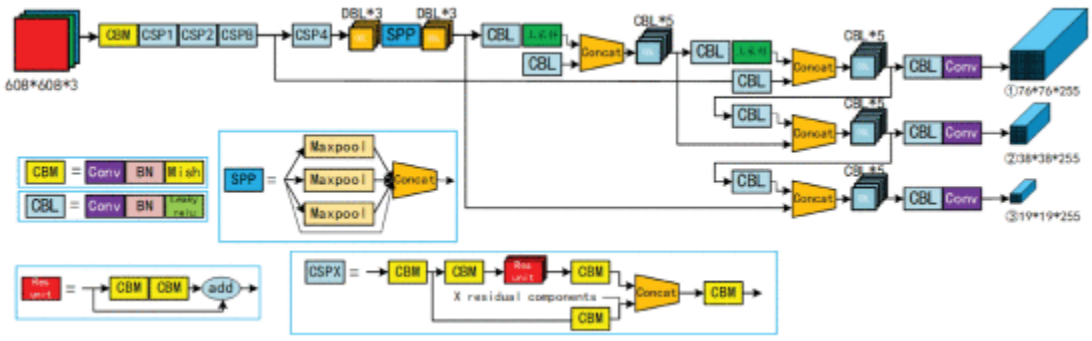


Fig1. Architecture of Yolov5

The most important feature extraction network of this object detection model is “CSPDarknet53”, a feature pyramid is formed with the help of features extracted. The feature pyramid structure makes use of SPP and PANet. “SPP” is the maximum pooling of the result of convolution in the “P5” layer. The process of pooling has four pooling layers which are scaled, and whose kernel sizes at the maximum pooling are 11,155,989,1313. The receptive area can be efficiently enlarged after the SPP, and significant context components can be separated. “PANet” is a convolution, upsampling, feature layer fusion, and downsampling-based circular pyramid structure. Following the “PANet”, various feature layers are fully merged, which can boost defect feature extraction capacities significantly. Finally, the YOLOv5 head predicts crack identification outcomes using the three feature layers processed by “PANet”, which is helpful to determine whether objects exist in the three preceding frames of each feature layer or not, as well as the type of objects. This model performs poorly while implemented in a situation where multiple objects were present, in such situations this model requires higher computation power, and as a result, the speed of the object detection is reduced.

Google proposed the initial version of the “lightweight network MobileNet, MobileNetv1, in 2017”. The most notable feature is that it suggests “depth-separated convolution”. Its convolution module joins 33 Depthwise convolutions with 11 Pointwise convolutions before adding BN and ReLU activation functions after each convolution. Yolov5 mainly improves on earlier versions in two ways:

1. Extension to work “MobileNetV2” eliminates Linear Bottlenecks structure's second point-to-point convolution and replaces it with a point-to-point convolution.
2. “Mobilenetv2” inverted residuals are designed using the ResNet structure. It uses 1x1 convolution to increase the dimension to a 3x3 network structure, and then uses 1x1 convolution to decrease the dimension after the 3x3 network structure.

Lucas Kanade optical flow

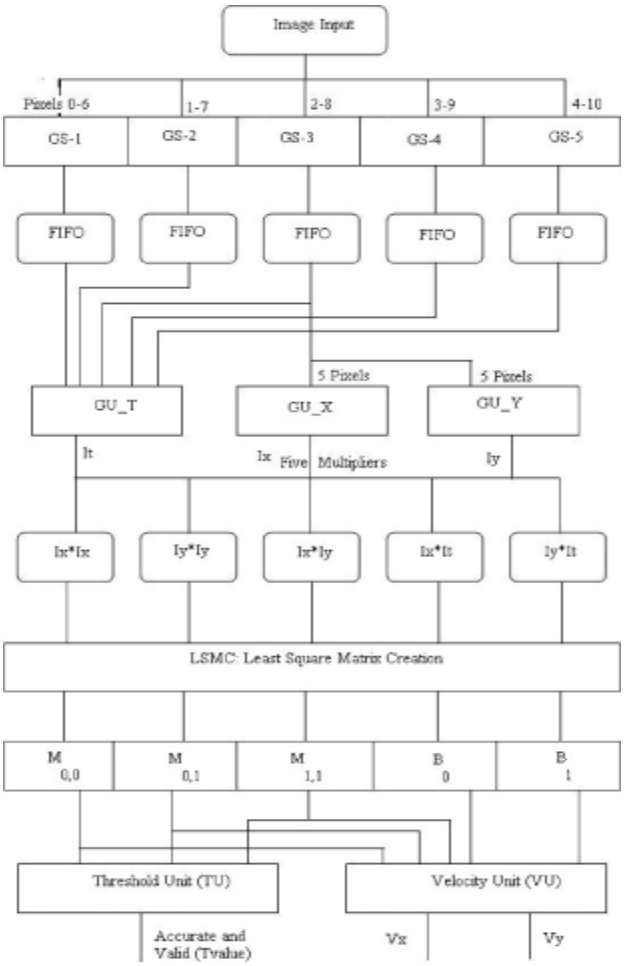


Fig2. The architecture of Lucas Kanade Optical Flow

Inside the smoothing blocks is the pipelined parallel dataflow. The pixels are subsequently passed from the “FIFO blocks” to the gradient computation “GU block”. “GU block” is a basic weighted convolution of pixels and kernel elements. The outputs of the gradient are later binary multiplied and transmitted to the matrix construction step (LSMC block). The optical velocity flow is then estimated using the least square matrix elements and transmitted to the “VU block”.

The suggested architecture processes a clustered collection of pixels in one flow, with a block of 25 25 pixels. To begin execution, the pipelined design requires 11 picture input frames. “Pixels from frames 0-6, 1-7, 2-8, 3-9, and 4-10 are routed to GS1, GS2, GS3, GS4, and GS5 blocks”, as shown in Fig. 2. The GS block's duty is to transport image pixels for computations in three

dimensions (x, y, and t), and it also includes a multiplexer, a de-multiplexer, and FIFO registers to enable simultaneous computation by the GS blocks. In this stage, the convolution operation is the sum of the picture pixels and coefficient products.

The FIFO 3 register block is specially designed with the ability to send data in the x and y directions to the GU blocks GUx and GUy. The slope values are calculated from the GUs-X, Y, and T, respectively. The values are sent to the multiplier block which calculates the products "(Ix2, Iy2, Ix Iy, Iy It and Ix It)" pixel by pixel. After that, the articles are moved to the LSMC stage. The array elements are fed into the threshold or velocity unit, which calculates the final optical flow values as well as a valid accurate signal. The verified TU signal can tell the difference between acceptable speed values and several erroneous starts.

Methodology

The methodology for the project involves a systematic approach to implement and integrate the object detection using YOLOv5 and speed estimation using Lucas-Kanade optical flow. The following steps outline the methodology for the project:

Data Collection and Preparation: Gather a suitable dataset consisting of annotated images or video frames for object detection training. Ensure the dataset represents diverse object classes and scenarios relevant to the application. Preprocess the dataset by resizing, normalizing, and augmenting the images to enhance training performance and generalization.

Object Detection using YOLOv5: Implement the YOLOv5 architecture using a deep learning framework such as PyTorch. Train the YOLOv5 model on the prepared dataset using appropriate loss functions and optimization techniques. Fine-tune the model by adjusting hyperparameters, conducting cross-validation, and optimizing the model's performance. Evaluate the trained model using evaluation metrics such as precision, recall, and mAP to assess its object detection accuracy.

Lucas-Kanade Optical Flow Implementation: Implement the Lucas-Kanade optical flow algorithm using libraries such as OpenCV or custom implementations. Apply the optical flow algorithm to track the objects detected in consecutive video frames. Estimate object velocities based on the computed optical flow vectors, accounting for potential challenges like occlusions and camera motion. In order to provide approximate, but fast results, we made a few assumptions

such as assuming that the width of a car is 3 meters. Using this assumption, we calculate the real-world velocity of an object as d = x * wide/w where x = displacement of an object given by optical flow; wide = width of the car as we made an assumption of 3 meters; w = width of the bounding box. We then calculated the velocity of an object using the formula v = d * fps, where fps is the number of frames per second.

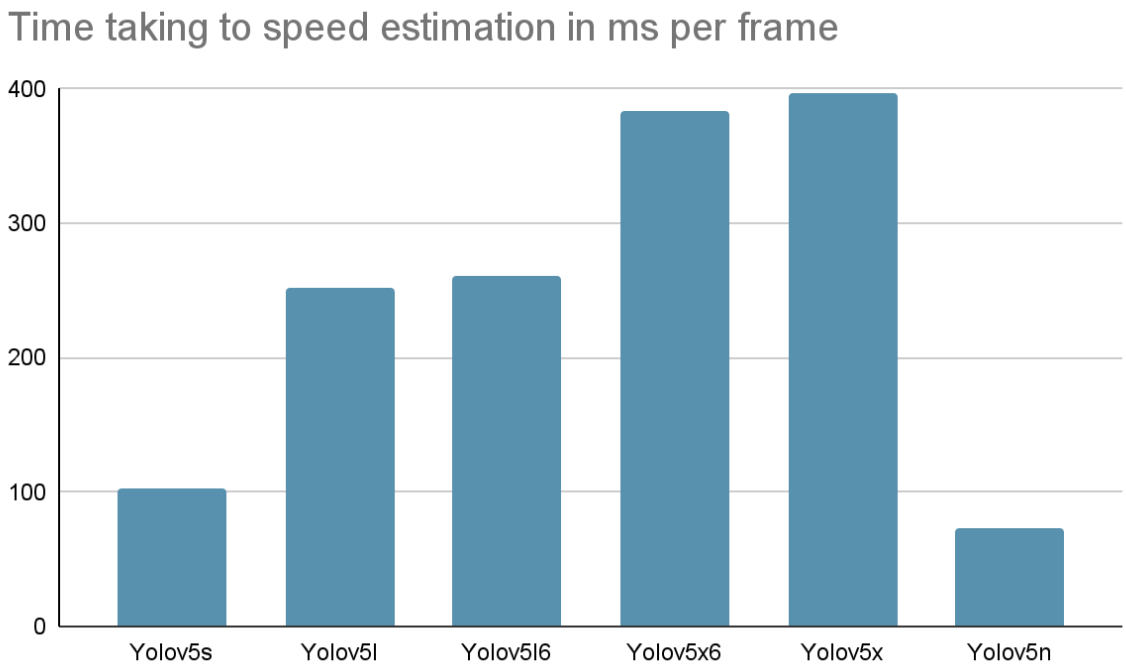
System Integration: Design and implement a pipeline that seamlessly integrates object detection using YOLOv5 and speed estimation using Lucas-Kanade optical flow. Develop algorithms to associate detected objects across frames, ensuring consistency in object tracking and speed estimation. Handle challenges such as object ID switches, track fragmentation, and occlusion handling to maintain accurate representations of object velocities.

Results

Evaluation

For the evaluation purpose, multiple Yolov5 models like Yolov5s, Yolov5x, Yolov5x6, Yolov5l, Yolov5n, Yolov5l6 models were used. The average time taken to speed estimation is calculated and plotted on a bar graph. From the below graph, it can be seen that the time taken to speed

estimation is less for the Yolov5n model which has a lesser number of layers compared to bigger models Yolov5x6 and Yolov5x. So for the real-time experience, Yolov5n and Yolov5s models are preferred.



Discussion

- Depth estimation, and speed estimation, are all approximate. It is enough for a human rider to interpret the results.
- Currently, we can only use this system on cars/trucks as obstacles because we make the assumption that obstacles are approx. 3m wide. We will need to update the system to hold a data structure that has information about the true width of objects based on their classification by object detection.
- We need to make this system work in real time on a resource-constrained device. Performance optimizations on object detection and breakdown of the cause for high latency experiments need to be done as future work

References

[1] L. Shao, Z. Fan, J. Li, and H. Liu, "Research on Road Vehicle Detection based on improved Yolov5s," 2022 *IEEE International Conference on Mechatronics and Automation (ICMA)*, 2022, pp. 428-433, DOI: 10.1109/ICMA54519.2022.9856288.

[2] S Woo, J Park, J. Y Lee and I. S Kweon, "Cbam: Convolutional block attention module", *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[3] Jin, X., Li, Z., & Yang, H. (2021, October). "Pedestrian Detection with YOLOv5 in Autonomous Driving Scenario". In *2021 5th CAA International Conference on Vehicular Control and Intelligence (CVCI)* (pp. 1-5). IEEE.

[4] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). "You only look once: Unified, real-time object detection". In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).

[5] Han, X., Chang, J., & Wang, K. (2021). "You only look once: Unified, real-time object detection". *Procedia Computer Science*, *183*, 61-72.

[6] H. V. and K. R., "Real Time Pedestrian Detection Using Modified YOLO V2," *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, 2020, pp. 855-859, doi: 10.1109/ICCES48766.2020.9138103.

[7] Z. -J. Liu, Y. -M. Li, M. A. Berwo, Y. -M. Wang, Y. -H. Li and N. Yang, "Vehicle Detection Based on Improved Yolov5s Algorithm," *2022 3rd International Conference on Information Science, Parallel and Distributed Systems (ISPDS),* 2022, pp. 295-300, doi: 10.1109/ISPDS56360.2022.9874217.

[8] L. Y. Siong, S. S. Mokri, A. Hussain, N. Ibrahim and M. M. Mustafa, "Motion detection using Lucas Kanade algorithm and application enhancement," *2009 International Conference on Electrical Engineering and Informatics*, 2009, pp. 537-542, doi: 10.1109/ICEEI.2009.5254757.

[9] Yanfeng Chen and Q. Wu, "Moving vehicle detection based on optical flow estimation of edge," *2015 11th International Conference on Natural Computation (ICNC)*, 2015, pp. 754-758, doi: 10.1109/ICNC.2015.7378085.

[10] X. Xing, Y. Yongjie and X. Huang, "Real-Time Object Tracking Based on Optical Flow," *2021 International Conference on Computer, Control and Robotics (ICCCR),* 2021, pp. 315-318, doi: 10.1109/ICCCR49711.2021.9349376.

[11] S. Srinivas and M. A. Suresh, "Faster Depth Estimation for Situational Awareness on Urban Streets," *2021 IEEE International Conference on Big Data (Big Data),* 2021, pp. 917-926, doi: 10.1109/BigData52589.2021.9671783.

[12] T. R. S. Kalyan and M. Malathi, "Architectural implementation of high speed optical flow computation based on Lucas-Kanade algorithm," 2011 3rd International Conference on Electronics Computer Technology, 2011, pp. 192-195, doi: 10.1109/ICECTECH.2011.5941885.