

Comparative Analysis of Heuristic Techniques for Vehicles movement detection

A BTP Report
By

**P.Rupesh Chowdary (S20210010173)
S.V.S.Apparao (S20210010208)
O . Khadar Basha (S20210010164)**

B24SMP03



INDIAN INSTITUTE OF INFORMATION TECHNOLOGY SRICITY

23-12-2024

Final Report



INDIAN INSTITUTE OF INFORMATION TECHNOLOGY SRICITY

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the BTP entitled **“Comparative Analysis of Heuristic Techniques for Vehicles movement detection”** in the partial fulfillment of the requirements for the award of the degree of B. Tech and submitted in the Indian Institute of Information Technology Sri City, is an authentic record of my own work carried out during the time period from January 2024 to December 2024 under the supervision of Dr. S Manipriya, Indian Institute of Information Technology Sri City, India.

The matter presented in this report has not been submitted by me for the award of any other degree of this or any other institute.

P. Rupesh Chowdary
25/12/24

(P. Rupesh Chowdary)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

S. Manipriya

(Dr. S Manipriya)



INDIAN INSTITUTE OF INFORMATION TECHNOLOGY SRICITY

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the BTP entitled **“Comparative Analysis of Heuristic Techniques for Vehicles movement detection”** in the partial fulfillment of the requirements for the award of the degree of B. Tech and submitted in the Indian Institute of Information Technology Sri City, is an authentic record of my own work carried out during the time period from January 2024 to December 2024 under the supervision of Dr. S Manipriya, Indian Institute of Information Technology Sri City, India.

The matter presented in this report has not been submitted by me for the award of any other degree of this or any other institute.

(S.V.S.Apparao)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

(Dr. S Manipriya)



INDIAN INSTITUTE OF INFORMATION TECHNOLOGY SRICITY

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the BTP entitled **“Comparative Analysis of Heuristic Techniques for Vehicles movement detection”** in the partial fulfillment of the requirements for the award of the degree of B. Tech and submitted in the Indian Institute of Information Technology Sri City, is an authentic record of my own work carried out during the time period from January 2024 to December 2024 under the supervision of Dr. S Manipriya, Indian Institute of Information Technology Sri City, India.

The matter presented in this report has not been submitted by me for the award of any other degree of this or any other institute.

O. Khadar Basha
25/12/2024

(O.Khadar Basha)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

S. Manipriya

(Dr. S Manipriya)

ABSTRACT

Accurate vehicle detection in traffic videos is crucial for effective traffic management, enabling real-time monitoring, congestion analysis, and traffic flow optimization. Our thesis project presents a novel framework, which uses heuristic techniques for vehicles movement detection to improve operational efficiency in urban mobility applications. To assess the effectiveness, a comparative analysis is conducted between three methods: Adaptive Optical flow, Adaptive Blob tracking, and Adaptive YOLO SORT. While most existing research focuses on deep learning approaches for motion boundary detection, which require extensive training on large datasets and result in high computational costs, this project introduces three alternative algorithms that avoid intensive training, offering robust performance for real-world traffic scenarios.

Based on the result analysis from the benchmark dataset, it is evident that adaptive Blob Tracking in MCVL exhibits superior computational efficiency of processing time of 0.005 seconds per frame and consistent memory usage of 6.3 kb irrespective of the complexity of the scene. It also shows high accuracy for detecting motion direction and boundaries for MCVL with MSE of 1221.734 and RMSE 34.953. However, adaptive Optical Flow and YOLO Sort for MCVL do not consistently demonstrate superiority over the alternatives.

Contents

0.1	Introduction.....
0.2	Data Set Description
0.3	Problem Statement
0.4	Contributions.....
0.5	Proposed Methodology.....
0.5.1	Road Surface Segmentation.....
0.5.2	Boundary detection Algorithms
	Adaptive Lucas Kanade Optical flow.....
	Adaptive Blob Tracking.....
	Adaptive YOLO and SORT.....
0.5.3	Dynamic Thresholding
0.6	Evaluation and Comparison of Results.....
0.6.1	Evaluation Metrics.....
0.6.2	Comparison of error rates.....
0.6.3	Time complexity Analysis
0.6.4	Memory usage Analysis.....
0.7	Graphical User Interface
0.8	Conclusion and Future works.....
0.8.1	Conclusion.....
0.8.2	Future Works.....
0.9	References.....

0.1 Introduction:

The efficiency of a traffic management application depends significantly on its ability to detect vehicles effectively from a traffic video. Hence vehicle detection techniques have to have an optimal balance on the computational time and space. This project aims to achieve this balance by integrating critical preprocessing steps, such as Road Surface Segmentation to determine vehicle direction and motion within the traffic scenes or videos. While existing research predominantly relies on deep learning for motion boundary detection, these approaches often entail extensive training on large datasets, leading to high computational costs and increased complexity. Therefore, this thesis gives a detailed analysis on the effectiveness of three distinct methods: optical flow, blob tracking, and YOLO sort, modified and made adaptive for detecting vehicle movements in traffic scenarios.

Based on the result analysis from the benchmark dataset, it is evident that adaptive Blob Tracking in MCVL exhibits superior computational efficiency of processing time of 0.005 seconds per frame and consistent memory usage of 6.3 kb irrespective of the complexity of the scene. It also shows high accuracy for detecting motion direction and boundaries for MCVL with MSE of 1221.734 and RMSE 34.953. However, adaptive Optical Flow and YOLO Sort for MCVL don't consistently demonstrate superiority over the alternatives.

0.2 Dataset Description:

- Vehicle tracking data is often gathered using technologies like loop detectors embedded in roadways, radar sensors, infrared sensors, or video cameras.
- UA-DETRAC is a challenging real-world benchmark for multi-object detection and tracking, widely used in Intelligent Transportation Systems.
- The dataset includes images from highway scenes and regular road scenarios, addressing diverse problems in traffic analysis and vehicle tracking.
- The DETRAC dataset has been chosen as the primary data source for this analysis due to its suitability for real-world multi-object tracking challenges.



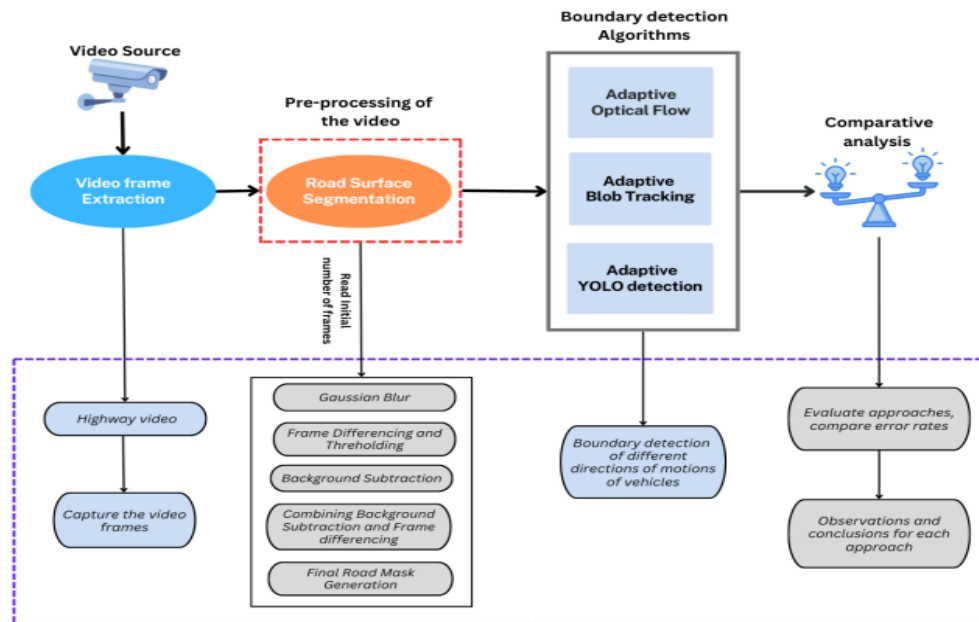
0.3 Problem Statement:

1. Using a novel preprocessing approach to accurately extract the road surface, effectively eliminating unnecessary disturbances and improving the clarity of traffic data.
2. Employ and adapt computer vision algorithms, including Optical Flow, Blob Tracking, YOLO, and SORT, to address the limitations of traditional deep learning methods, ensuring efficiency in real-time applications.
3. Analyze and compare the performance of these algorithms across diverse traffic scenarios, focusing on metrics such as computational time, memory usage, and error rates to derive actionable insights.
4. Automate the thresholding process during the preprocessing stage using dynamic methods like local thresholding and the Chow-Kaneko approach, enhancing adaptability to varying illumination conditions.

0.4 Contributions :

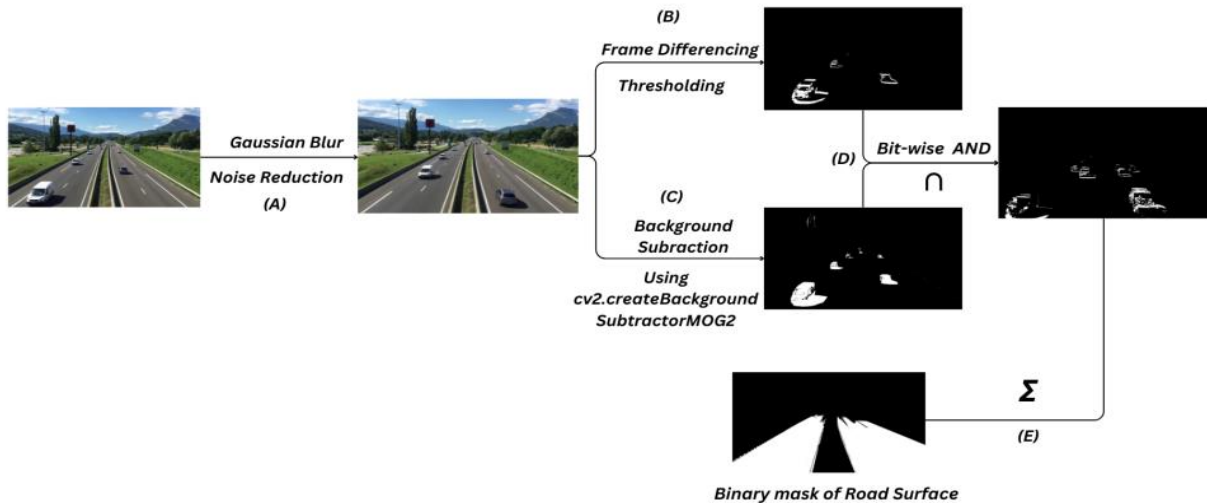
- **P. Rupesh Chowdary (S20210010173):** Implemented video pre-processing and adaptive Lucas-Kanade Optical flow algorithm, contributed to GUI development integrating all three algorithms, conducted comparative analysis, and implemented Chow-Kaneko dynamic thresholding across diverse video types.
- **S.V.S Apparao (S20210010208):** Developed adaptive Blob tracking approach, contributed to GUI integration, performed algorithmic comparison focusing on time complexities, and implemented Otsu's thresholding with accuracy analysis across various videos.
- **O. Khadar Basha (S20210010164):** Executed adaptive YOLO and SORT algorithm implementation, conducted memory usage analysis, and implemented Local Adaptive Thresholding with accuracy comparisons across different video categories.

0.5 Proposed Methodology:



0.5.1 Video Pre-processing: (Road Surface Segmentation):

Road surface segmentation focuses on extracting the road surface from few initial video frames by excluding background elements. Gaussian blur is first applied to reduce noise and enhance image clarity. Frame differencing and thresholding are then used to highlight changes between consecutive frames, identifying moving objects. The Background Subtractor MOG2 algorithm further refines this by separating moving objects from the static background. Combining these methods with bitwise-AND operations generates accurate binary masks, which are aggregated using a bitwise-OR operation to produce a binary road surface mask.



Workflow of Road Surface Segmentation Algorithm

0.5.2 Boundary Detection Algorithms:

Adaptive Optical Flow

- Optical flow estimates motion by analyzing the apparent movement of objects across consecutive frames.
- The project implements a modified Lucas-Kanade optical flow algorithm tailored for vehicle tracking.
- Video preprocessing is performed on the first 250 frames to generate a binary road surface mask.
- The binary mask is applied to all subsequent frames to focus on the road and eliminate irrelevant disturbances.
- Corner points in the initial frame are detected using `cv2.goodFeaturesToTrack()` as keypoints for tracking.

- Keypoint locations in subsequent frames are estimated using Lucas-Kanade optical flow (cv2.calcOpticalFlowPyrLK()).
- Consistency of keypoints is verified by selecting those with low error values and stable status.
- Trajectories are formed using points at consistent distances to ensure smooth tracking.
- The direction of vehicle movement is determined by analyzing the last two points of each trajectory.

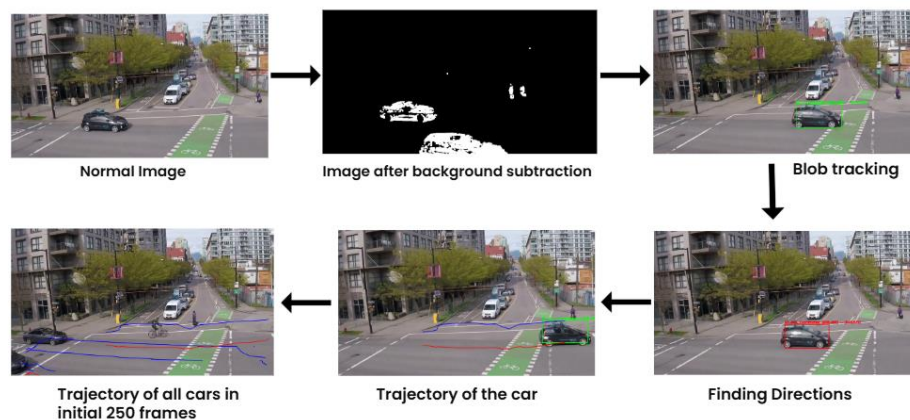
Work Flow of Algorithm :



Adaptive Blob Tracking

- Combines Blob Tracking and SORT (Simple Online and Realtime Tracking) to analyze vehicle movement and detect motion directions
- Begins with MOG2 (Mixture of Gaussians) for dynamic background subtraction to handle lighting changes and shadows effectively.
- Blob tracking detects connected pixel regions representing vehicles using OpenCV's SimpleBlobDetector.
- Detections are passed to the SORT tracker, which utilizes Kalman filtering and the Hungarian algorithm for motion prediction and data association.
- SORT assigns unique IDs to vehicles, ensuring consistent tracking across frames.

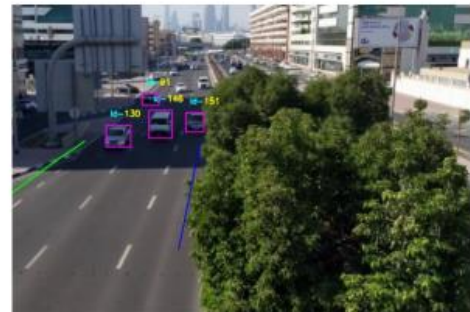
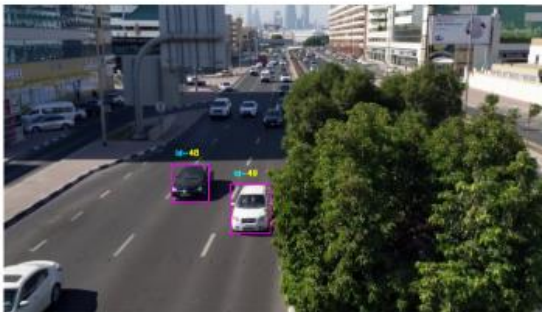
- Vehicle positions are updated for each frame, and their trajectories are analyzed.
- The direction of movement is determined by comparing current and previous positions of each vehicle.
- SORT maintains robustness, even during occlusions or temporary losses of vehicles.
- The integrated approach ensures efficient and accurate vehicle tracking.
- Provides a practical solution for intelligent transportation systems.



Adaptive YOLO and SORT:

- Combines YOLO (You Only Look Once) object detection with SORT (Simple Online and Real-Time Tracking) for vehicle detection and tracking.
- YOLO detects vehicles in each frame, providing bounding boxes and confidence scores.
- Detections are organized into an array containing bounding box coordinates and confidence values.
- SORT tracker is initialized to manage and update vehicle positions across frames.
- SORT integrates YOLO detections for real-time tracking of multiple vehicles.

- Movement direction is determined by comparing current and previous positions of tracked vehicles.
- A greater current y-coordinate indicates movement from top to bottom, and a lower value indicates the opposite.
- Changes in x-coordinates are analyzed for horizontal movement tracking.
- The algorithm ensures efficient and accurate vehicle tracking.
- Provides valuable insights for intelligent transportation systems.



0.5.3 Dynamic Thresholding:

- Manual thresholding uses a fixed threshold for all pixels in the image.
- Hence, it cannot deal with images containing, for example, a strong illumination gradient or weak illumination gradient
- Adaptive or Dynamic thresholding changes the threshold dynamically over the image
- This more sophisticated version of thresholding can accommodate changing lighting conditions in the image

Chow and Kaneko Dynamic Thresholding :

- Chow and Kaneko divide an image into an array of overlapping sub-images and then find the optimum threshold for each sub-image by investigating its histogram
- The image is divided into small overlapping windows of size $\text{window_size} \times \text{window_size}$. Each window represents a local region around a pixel.
- The threshold for each single pixel is found by interpolating the results of the sub-images.
- The threshold $T(i, j)$ for each pixel (i, j) is calculated using the Chow and Kaneko formula.
- If $I(i, j) > T(i, j)$ the pixel is set to 255 (white, foreground). Otherwise, it is set to 0 (black, background)

Local Adaptive Thresholding :

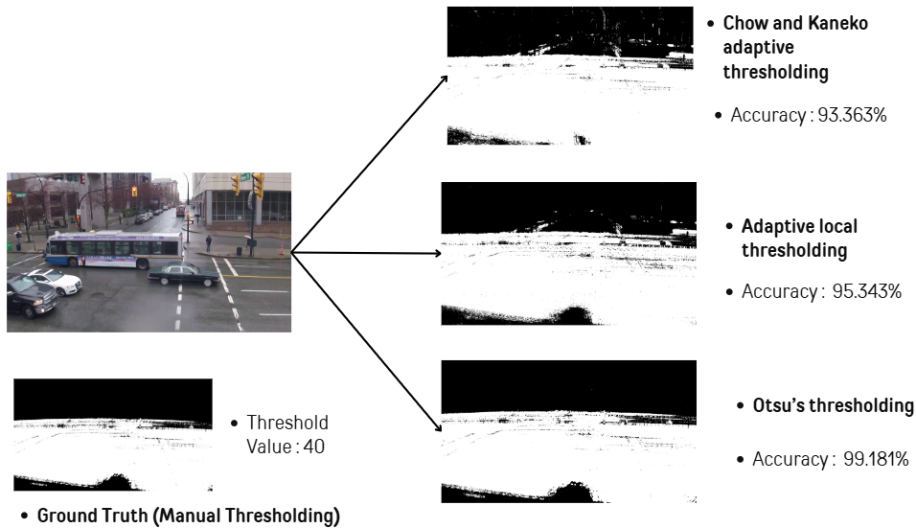
- This method statistically examines the intensity values of the local neighborhood of each pixel.
- .The size of the neighborhood has to be large enough to cover sufficient foreground and background pixels, otherwise a poor threshold is chosen. (Window size: $k \times k$)
- Within the local region, a Gaussian kernel is applied. A Gaussian kernel assigns more weight to pixels closer to the center of the region and less weight to pixels farther away.

- The weighted sum of pixel intensities within the local region is computed using the Gaussian weights. This gives the local threshold (T) for the pixel.
- If $I(i,j) > T$ the pixel is set to 255 (white, foreground). Otherwise, it is set to 0 (black, background)




Otsu's Thresholding :

- Compute the histogram of the grayscale image and the histogram represent the frequency of each intensity value (0–255 for 8-bit images).
- Normalize the histogram to calculate probabilities $P(i)$ of each intensity i .
- For each threshold t , Divide the histogram into two classes:
 1.) Class 1 (Background): Intensities $[0, t]$ 2.) Class 2 (Foreground): Intensities $[t+1, 255]$
- Compute the probabilities; mean intensity, between-class variance of each class by applying the respective formula.
- Identify the threshold that maximizes the between-class variance

Dynamic Thresholding Example



Accuracies of Road surface segmentation on different videos :

<u>Videos</u>	Chow and Kaneko adaptive thresholding	Adaptive local thresholding	Otsu's Thresholding
1.) 	74.164%	97.706 %	99.3588%
2.) 	96.029%	96.133%	98.523%
3.) 	90.751%	92.4%	98.949%

0.6 Evaluation and Comparison of Results:

0.6.1 Evaluation Metrics:

The project work primarily concentrates on prediction tasks, employing various approaches to forecast outcomes effectively. To evaluate the performance of these approaches in an objective manner, the study relies on standard evaluation metrics that provide clear benchmarks for comparison. The accuracy of the predictions is specifically quantified using Mean Square Error (MSE) and Root Mean Square Error (RMSE), which help measure the deviation between predicted and actual values in the analysis.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

MSE = mean squared error

n = number of data points




Y_i = observed values

\hat{Y}_i = predicted values

0.6.2 Comparison of error rates:

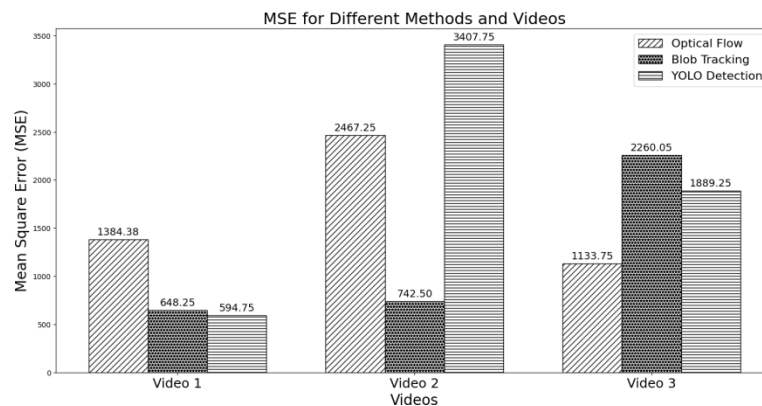
The performance of the proposed methods was evaluated using MSE, RMSE, and computational time. The DETRAC dataset was used for benchmarking.

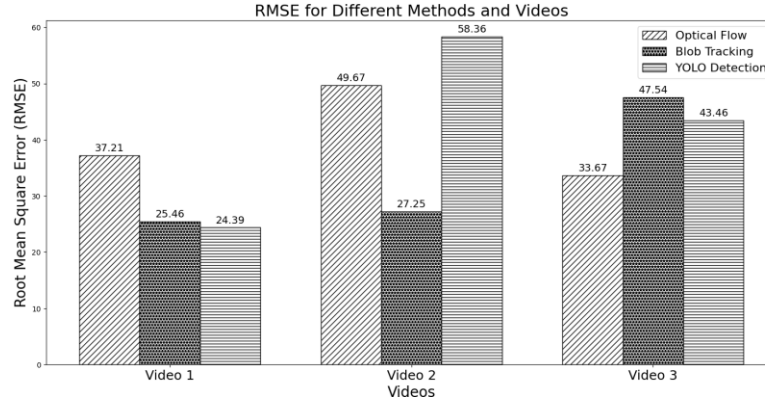
It is observed that the blob tracking approach performs well among the three methods evaluated. However, there are instances where the optical flow method outperforms the others. Each approach has its unique advantages and disadvantages. Additionally, variations in the error values were noted when different videos were analyzed.

Videos		Adaptive Optical Flow	Adaptive Blob Tracking	Adaptive YOLO and SORT
1.)		MSE : 1384.38 RMSE : 37.21	MSE : 648.25 RMSE : 25.46	MSE : 594.75 RMSE : 24.39
2.)		MSE : 2467.25 RMSE : 49.671	MSE : 742.5 RMSE : 27.25	MSE : 3407.75 RMSE : 58.36
3.)		MSE : 1133.75 RMSE : 33.67	MSE : 2260.05 RMSE : 47.54	MSE : 1889.25 RMSE : 43.46

*MSE: Mean Square Error *RMSE: Root Mean Square Error

One reason the blob tracking approach records minimal error is that the bounding boxes, or blobs, adjust as the vehicle moves farther from the camera. This adjustment allows the predicted boundary to closely match the actual value. In the optical flow approach, the boundary is detected using trajectory points derived from feature detection. As the car moves away from the camera, it becomes smaller, and fewer features are detected. In turn, the road boundary is not accurately detected, affecting the error.

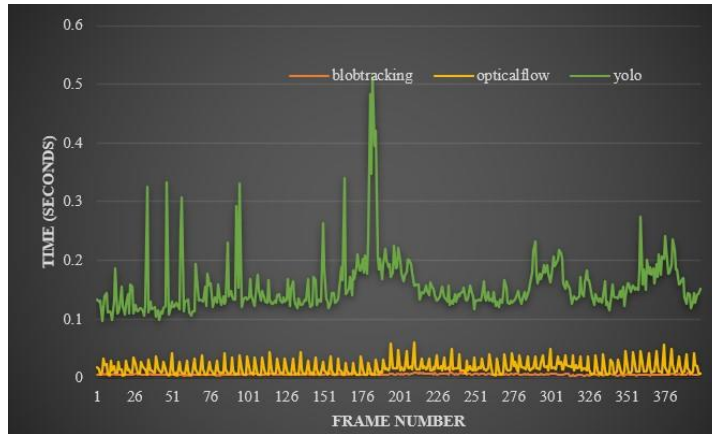




In the YOLO approach, the size of the bounding box remains almost fixed unless the vehicle size changes significantly. This rigidity causes the bounding box to sometimes exceed the vehicle's actual size, impacting the boundary detection and increasing the error. For better visualization, we created a graphical comparison of RMSE and MSE errors for the three approaches.

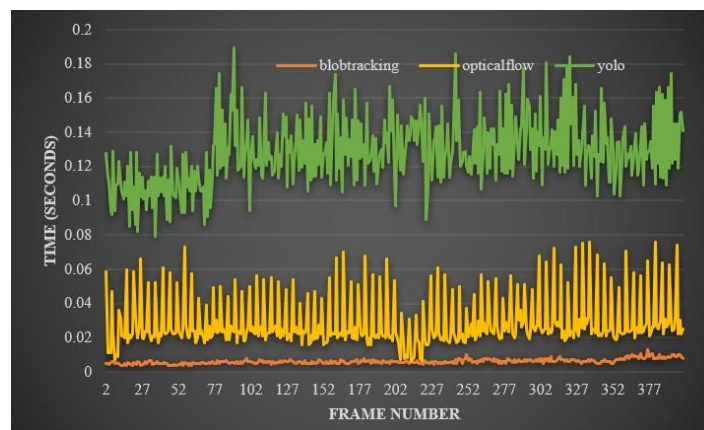
0.6.3 Time Complexity Analysis:

The computational time for the three adaptive algorithms is calculated by the processing time in seconds for each frame of the traffic video. While comparing the time complexities of the proposed method in well illuminated traffic video—Adaptive Blob Tracking, Adaptive Optical Flow, and Adaptive YOLO—measured in seconds per frame across a dataset, several insights emerge. Adaptive Blob Tracking consistently exhibits the lowest time complexity, with an average runtime of approximately 0.005 seconds per frame. This efficiency suggests its suitability for real-time applications where computational resources are constrained. Optical Flow, with an average runtime of 0.016 seconds per frame, demonstrates moderate performance, making it viable for scenarios where accuracy in motion detection is crucial but with a higher tolerance for processing time. Conversely, YOLO, despite its significantly higher average runtime of 0.145 seconds per frame, offers unparalleled accuracy and robustness in object detection tasks. This high computational demand aligns with its capability to handle complex scenes and a wide variety of objects, albeit at the cost of slower processing speeds compared to the other two algorithms.

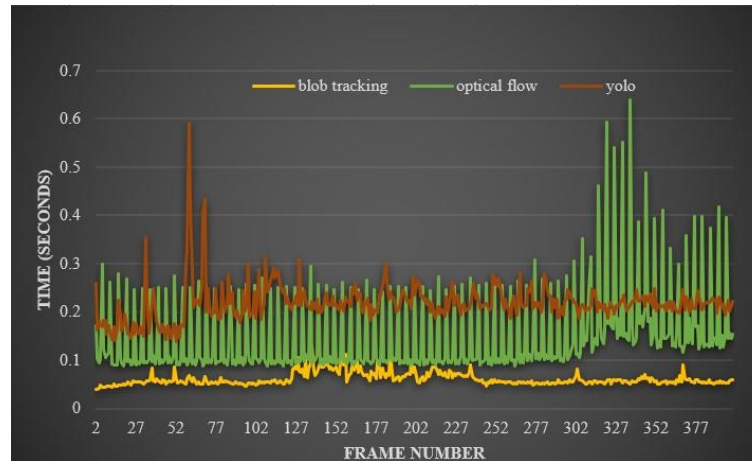


Comparative Analysis of Adaptive Algorithms by processing time for well illuminated Traffic Videos

For a moderate illuminated traffic video adaptive Blob tracking consistently shows the lowest execution times across the dataset, with an average of approximately 0.005 seconds per frame. This suggests it is the most computationally efficient among the three methods. Adaptive Optical flow, with an average of about 0.025 seconds per frame, demonstrates higher computational demands compared to blob tracking but remains relatively efficient. In contrast, Adaptive YOLO exhibits significantly higher execution times, averaging around 0.125 seconds per frame. This indicates YOLO is the most computationally intensive of the three, likely due to its comprehensive object detection capabilities that involve complex deep learning models.



Comparative Analysis of Adaptive Algorithms by processing time for moderately illuminated Traffic Videos

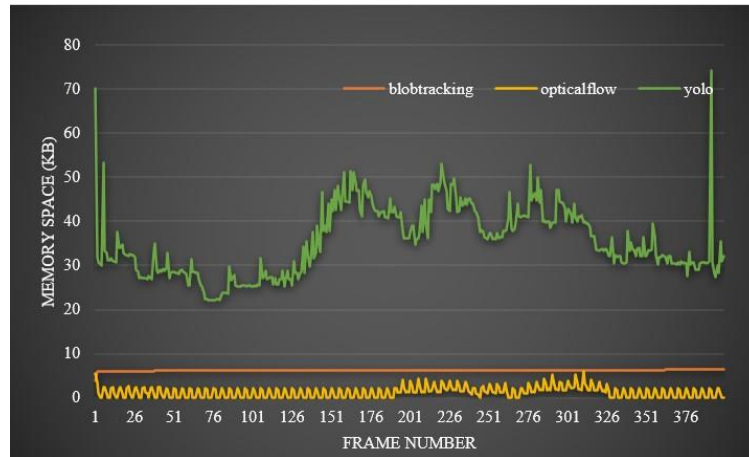


Comparative Analysis of Adaptive Algorithms by processing time for Night Traffic Videos

In night video, adaptive blob tracking shows the lowest average time complexity of approximately 0.055 seconds, optical flow estimation averages around 0.102 seconds, and YOLO object detection being the most computationally intensive with an average of 0.221 seconds. These measurements highlight YOLO's higher computational demands compared to blob tracking and optical flow, which are more lightweight tasks in terms of processing time. Understanding these time complexities is crucial for optimizing performance of MCVL.

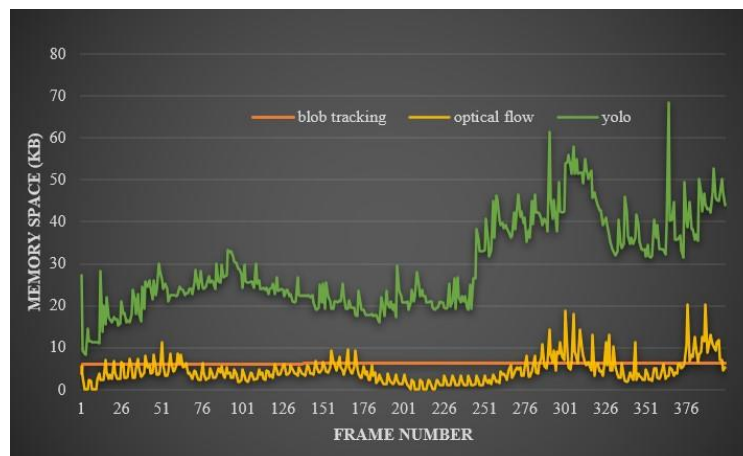
0.6.4 Memory usage Analysis:

In comparing the space complexities of adaptive blob tracking, optical flow, and YOLO algorithms in well-illuminated traffic videos, distinct patterns emerge in memory usage. Blob tracking demonstrates the most efficient memory usage among the three, peaking at 6.3 KB. This efficiency stems from its focus on individual object segmentation and frame-based tracking, requiring less memory compared to broader analysis methods. Optical flow follows closely with a maximum of 5.35 KB, emphasizing temporal coherence in pixel movements across frames without excessive memory overhead.



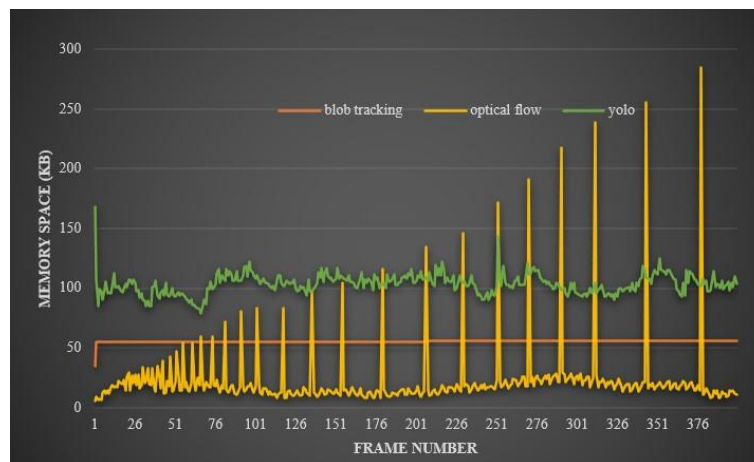
Comparative Analysis of Adaptive Algorithms with Memory Space allocation for well illuminated Traffic Videos

Conversely, YOLO, known for real-time object detection, exhibits significantly higher memory consumption, peaking at 52.96 KB. This increase is due to YOLO's deep neural network architecture, which processes entire images for accurate multi-object detection simultaneously. Therefore, while blob tracking and optical flow excel in memory efficiency suitable for well-illuminated traffic videos, YOLO's superior object detection performance comes with higher memory usage, illustrating trade-offs between functionality and resource consumption in computer vision applications.



Comparative Analysis of Adaptive Algorithms with Memory Space allocation for moderately illuminated Traffic Videos

The space complexity for the moderate traffic video shows that adaptive blob tracking shows the lowest space complexity, with measurements ranging mostly between 3.87 and 6.25 KB. Optical flow demonstrates moderate space usage, generally between 0 and 11.63 units, occasionally peaking up to 40.69 KB. In contrast, YOLO consistently exhibits the highest space complexity, ranging widely from 8.33 to 61.42 KB, indicating its significantly larger memory requirements compared to blob tracking and optical flow algorithms.



Comparative Analysis of Adaptive Algorithms with Memory Space allocation for Night Traffic Videos

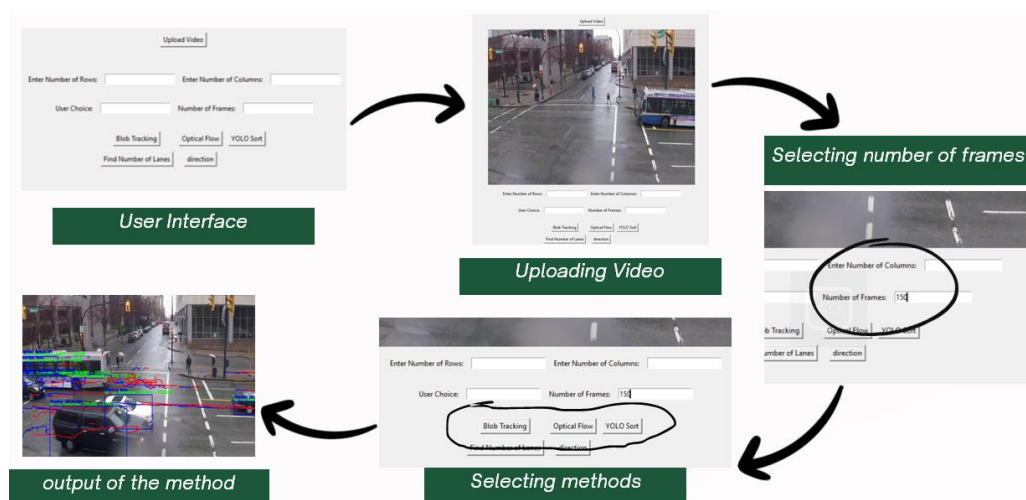
The night time video with poor illumination shows that blob tracking generally requires the least memory, with measurements ranging from approximately 34.84 KB to 55.61 KB. Optical flow's memory usage varies more widely, from around 5.55 KB to 256.04 KB. YOLO consistently demands the most memory, with values ranging from 78.80 KB to 145.84 KB. These variations highlight the differing memory demands of each algorithm, potentially influenced by factors such as image resolution, processing complexity, and specific implementation details during night time.

0.7 Graphical User Interface (GUI) :

- A GUI simplifies complex tasks, making it easy for users to interact with the system through buttons rather than typing commands.
- It offers different options, such as choosing algorithms and entering the number of frames.
- This makes it useful for both automatic and custom tasks
- Users can quickly choose and run algorithms without needing to change any code, which makes the process faster and easier



Workflow of GUI



0.8 Conclusion and Future works:

0.8.1 Conclusion:

- This research presents a comprehensive analysis of heuristic methods for vehicle detection.
- The results highlight the effectiveness of adaptive blob tracking in terms of accuracy and computational performance, while optical flow provides a balanced approach for low-resource environments.
- YOLO SORT, despite its higher computational demands, demonstrates strong potential for complex traffic scenarios requiring high precision.
- This makes an adaptable solution for diverse traffic management needs, enabling its application across various intelligent transportation systems.

0.8.2 Future Works:

1. **Expanding Detection Capabilities:** Incorporate additional motion directions and dynamic traffic scenarios to enhance the versatility of the adapted algorithms.
2. **Real-Time Implementation:** Explore the integration with edge-computing devices for real-time deployment in traffic monitoring systems.
3. **Integration with IoT:** Investigate the potential as part of a broader IoT-based intelligent transportation system, enabling seamless data exchange and decision-making.

4. **Longitudinal Studies:** Conduct longitudinal analyses on large-scale datasets to validate the framework's scalability and long-term reliability.

0.9 References :

1. Y. Ng et al., "Asynchronous Kalman Filter for Event-Based Star Tracking," ECCV Workshops, Springer, 2022. Hadi Ghahremannezhad, Hang Shi, Chengjun Liu, "Automatic Road Detection in Traffic Videos", *IEEE 4th International Conference on Image Processing, Applications and Systems (IPAS) pp., 2021.*
2. Huansheng Song, Haoxiang Liang* , Huaiyu Li, Zhe Dai and Xu Yun (2019) "Vision-based vehicle detection and counting system using deep learning in highway scenes". *European Transport Research Review*
3. Salman Qasim, Kaleem Nawaz Khan, Miao Yu; Muhammad Salman Khan "Performance Evaluation of Background Subtraction Techniques for Video Frames". *2021 International Conference on Artificial Intelligence (ICAI)*
4. Radhakrishnan, M (2013). Video object extraction by using background subtraction techniques for sports applications. *Digital Image Processing, 5(9), 91–97.*
5. "Moving Object Detection and Segmentation using Frame Differencing and Summing Technique" Gopal Thapa, Kalpana Sharma, M.K.Ghose, *September 2014 International Journal of Computer Applications 102(7):20-25*
6. Estevao Gedraite, M.Hadad January 2011 "Investigation on the effect of a Gaussian Blur in image filtering and segmentation". *ELMAR, 2011 Proceedings*
7. H.Yedjour, January 2021 Optical Flow Based on Lucas-Kanade Method for Motion Estimation. In : *Artificial Intelligence and Renewables Towards an Energy Transition (pp.937-945)*
8. Dhara Patel, Saurabh Upadhyay, Volume 61– No.10, January 2013 Optical Flow Measurement using Lucas kanade Method. *International Journal of Computer Applications (0975 – 8887)*
9. Real-Time Vehicle Detection Based on Improved YOLO v5 Yu Zhang, Zhongyin Guo, Jianqing Wu. *September 2022 Sustainability 14(19):12274*
10. "Vehicle Detection and Tracking using YOLO and Deep SORT" Muhammad Azhad Bin Zuraimi; Fadhlan Hafizhelmi Kamaru Zaman *2021 IEEE 11th IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE)*
11. Mahmoud Abouelyazid (2023) "Comparative Evaluation of SORT, DeepSORT, and ByteTrack for Multiple Object Tracking in Highway Videos". *Vol. 8 No. 11 (2023): IJSICS-November*

12. Karl Leyven Leonida, Karla Veronica Sevilla, Cyrel O. Manlises "A Motion-Based Tracking System Using the Lucas-Kanade Optical Flow Method". *2022 14th International Conference on Computer and Automation Engineering (ICCAE)*
13. Y. Ng, Y. Latif, T.-J. Chin, and R. Mahony, "Asynchronous kalman filter for event-based star tracking," in *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part I. Springer, 2023, pp. 66–79*
14. A. Chaturvedi and A. Shukla, "Automatic detection of satellite images using blob detection and boundary tracking techniques", *2020*.
15. S. Li, J. Chen, W. Peng, X. Shi and W. Bu, "A vehicle detection method based on disparity segmentation", *Multimedia Tools Appl.*, vol. 82, no. 13, pp. 19643-19655, May 2023.
16. A. A. Kumar et al., "A Comparative Study of Various Filtering Techniques", *ICOEI, 2021*.
17. N. M. Al-Shakarji, F. Bunyak, G. Seetharaman and K. Palaniappan, "Multi-object tracking cascade with multi-step data association and occlusion handling", *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1-6, 2018.
18. Enric Meinhardt-Llopis¹, Javier S´anchez², Daniel Kondermann³ "Horn–Schunck Optical Flow with a Multi-Scale Strategy", *Image Processing On Line on 2013–07–19*
19. K.C Hari; Sushil Shrestha; Manish Pokharel, "Video Object Motion Tracking using Dense Optical Flow Techniques", *2023 International Conference on Informatics and Computing (ICIC)*
20. Shuman Guo, Shichang Wang "A Review of Deep Learning-Based Visual Multi-Object Tracking Algorithms for Autonomous Driving", *Applied Sciences* 12(21):10741, October 2022
21. Chakradhara Panda, "Object Detection and Tracking using Faster R-CNN", September 2019, *International Journal of Recent Technology and Engineering (IJRTE)*
22. Sankaranarayanan, M., Mala, C., Mathew, S. "Efficient vehicle detection for traffic video-based intelligent transportation systems applications using recurrent architecture". *Multimed Tools Appl* 82, 39015–39033, 2023.