Real-time Parking Space Monitoring System using CVZone and OpenCV

Venkata Thapaswini Thota
Dhanashree Dilip Shinde

December 5, 2023

Abstract

In this project, we used modern image processing methods from CVZone and OpenCV to design a complex Real-time Parking Space Monitoring System. Our main goal was to accurately distinguish between occupied and empty parking spaces. The analysis showed that when it came to identifying open parking spots, the filled image technology performed noticeably better than the dilated edge detection method. The system's effectiveness was established through extensive testing, resulting in an effective parking spot monitoring option. The project concluded with a fully working system that is efficient in maximizing parking space use, obviating the need to incorporate other improvements.

1. **Introduction**

The proliferation of cars as a necessity rather than a luxury, particularly for the working class, has become a defining characteristic of life in small cities across the United States, including places like Boston. This dependence has inevitably led to a notable increase in traffic congestion, with the problem often exacerbated by the difficulties associated with finding available parking. Motorists frequently face the challenge of identifying occupancy signs that indicate whether a parking slot is fully occupied, partially filled, or vacant. This search for parking not only contributes to traffic but also results in the inefficient use of parking spaces, as some areas may remain underutilized even during peak times.

In the context of smaller U.S. cities, where the urban landscape can be as complex as their larger counterparts, the need for an effective, real-time parking space monitoring system becomes particularly acute. Such a system promises to significantly improve the parking experience by providing immediate information on space availability, thus curtailing the often time-consuming quest for an open spot. Furthermore, in cities where the concerns about parking space scarcity are just as pressing as in larger urban areas, there is a clear imperative for innovative solutions.

Addressing this need, our project introduces a real-time parking space monitoring system utilizing CVZone and OpenCV. The system is conceived to streamline the parking process by providing up-to-date information on space availability, consequently reducing unnecessary driving in search of parking. This reduction in search time not only enhances the driver's experience but also aligns with broader environmental goals by potentially reducing fuel consumption and lessening vehicular emissions. This initiative represents more than a convenience; it is an integral part of a sustainable strategy for improving urban transportation infrastructure in smaller cities throughout the United States.

2. **Methods**

The method we adopted for our “Real-time Parking Space Monitoring System” project involves a comprehensive approach using computer vision and image processing to create a user-friendly, secure, and efficient parking management system. The solution comprises a sequence of operations starting with system setup and proceeding through continuous monitoring and updating.

Our system’s setup begins with the system initialization phase, which is conducted once during the initial implementation or after any system component is replaced. At this stage, a reference image of an empty parking lot is captured by high-definition surveillance cameras. These cameras are fixed in position to provide a consistent top-down view of the parking spaces, crucial for accurate image analysis.

For the image acquisition step, the cameras capture live footage of the parking lot, which is then segmented frame by frame to process each parking lane. CVZone, a cutting-edge library that simplifies computer vision tasks, is utilized to track and manage the parking spaces’ occupancy status dynamically. This library works in tandem with OpenCV to provide real-time image processing capabilities.

The software processes these images using Python, with additional support from libraries like Numpy for numerical operations and Python Image Processing for handling image data. These images are processed to discern between occupied and vacant parking spots by comparing the live images with the reference image. If a parking spot is detected as filled, it is marked with a red overlay, while empty spots are highlighted with a green overlay in the system’s user interface.

Our method also includes a data interpretation phase where the system analyzes the processed images to determine the status of each parking spot. This information is then relayed to the display update system, which informs drivers in real-time about the availability of parking spaces. The system’s ability to immediately reflect changes in parking spot status as vehicles arrive and leave ensures that drivers receive accurate and timely information, which is essential for reducing search time and congestion.

The operational efficiency of the system is underpinned by a set of well-defined non-functional requirements that govern its performance, reliability, and maintenance. These requirements ensure that the system operates consistently can handle various input cases and generate outputs effectively. In terms of hardware, the system requires a Pentium IV 3.4 GHz system, a 14’ color monitor, a 40 GB hard disk, 1 GB of RAM, and a web camera or surveillance camera. These specifications were chosen to balance cost and performance, making the system both accessible and capable.

Below is the detailed process we followed:

1. System Initialization: The parking guidance system's initialization occurs during the initial setup or after any system module replacement. A refresh signal activates the image sensors, and the aerial cameras capture images of the empty parking lot. These images are then sent to the MCU via a fast parallel image sensor interface.
2. Image Acquisition: A fixed camera acquires videos from the top view of the parking arena. The video is segmented into frames, and keyframes are extracted for processing, which reduces computational complexity. High-definition cameras take images of the parking space, which are then divided lane-wise for further analysis.

3. Thresholding of Image: The RGB image is converted to a grayscale image using the formula Gray = 0.229R + 0.587G + 0.11B A binary image is then created through thresholding, which distinguishes objects of interest in white against a black background, containing all necessary position and shape information.
4. Image Enhancement: The binary image, often noisy, is enhanced using morphological operations and filters like the Weiner filter. Noise is reduced, and any holes in the image are filled using infill and break-open functions, preparing the image for accurate detection.
5. Image Detection: After the initial image processing stages, the data is processed with PYCHARM software for image detection. This step is crucial for recognizing the shapes and positions of vehicles within the parking lot, completing the process of vacancy detection

2.1 Modules of Proposed System.

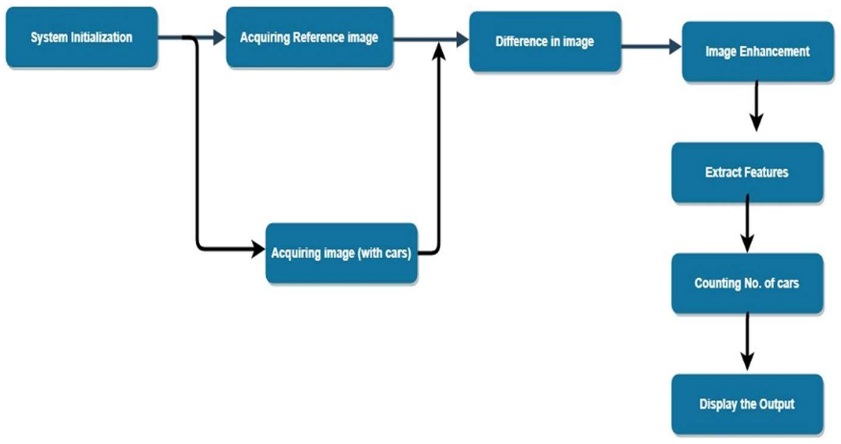

Fig 1. Operational modules of the proposed system

The key advantage of our system is its ability to provide a seamless and secure parking experience. It reduces the need for manual intervention, lowers the risk of vehicle theft or damage by monitoring the parking lot continuously, and offers a significant time-saving advantage over traditional parking methods.

By implementing this system, we aim to address the widespread issues associated with parking in congested areas. Our solution not only enhances the user experience by providing security and efficiency but also contributes to reducing fuel consumption and emissions due to less time spent searching for parking, thus offering an environmentally friendly parking solution.

3. <u>Experiments</u>

3.1 Dataset

For our "Real-time Parking Space Monitoring System" project, the dataset comprises a series of images captured by surveillance cameras positioned over parking lots. These images have been processed to create a binary representation of parking spaces, differentiating between occupied and vacant spots. The dataset includes a total of 69 parking slots, with images representing varying occupancy levels at different times.

Key characteristics of the dataset include:

- Binary Images: Using edge detection techniques, the dataset contains binary images that help in identifying the contours of parked vehicles. These images are generated through a process called dilation, which emphasizes the edges of objects, in this case, cars, within the parking slots.
- Marked Parking Spaces: Each parking space within the images has been manually marked to create ground truth data against which the system's accuracy can be tested. This marking helps in training and validating the image processing algorithms used for detecting occupancy.
- Pixel Counting for Occupancy: The system employs a method of pixel counting within the marked areas to determine whether a parking slot is occupied or not. A higher pixel count within the predefined boundaries of a parking slot suggests the presence of a vehicle.

  Visual Overlays for Occupancy Status: In the dataset, occupied parking slots are indicated by red-colored rectangles, while vacant slots are marked with green rectangles. This color coding provides a clear visual distinction between occupied and vacant spaces and is a crucial aspect of the system's user interface.
- Dynamic Data: As the parking situation changes, with vehicles entering and exiting, the dataset is updated to reflect these changes. For example, if two vehicles exit, the number of free slots increases accordingly, and this is reflected in the updated dataset.

This dataset not only serves as the backbone for our system's development and testing but also acts as a real-world representation of the parking lot environment. The use of such a dataset ensures that our system is well-equipped to handle the complexities of real-time parking space monitoring and can be reliably deployed in actual parking scenarios.

3.2 Evaluation metrics

For our "Real-time Parking Space Monitoring System," we as a team of two, focused on evaluating the performance of our method using the following metrics:

- Accuracy: We measured how precisely the system could identify parking spaces by transforming images to their Canny edge representations, similar to the images above. We manually counted the number of parking spaces and compared it with the system's count.
- Precision: We calculated the ratio of correctly identified occupied parking spots to ensure our system minimized false positives.
- Recall: We also measured the system's ability to identify all occupied spaces, thereby minimizing false negatives.
- Robustness: We processed a variety of images with different lighting and orientations to test the robustness of our edge detection under varying conditions.

- Real-time Performance: We ensured that the system could process images in a time-efficient manner, reflecting real-time changes in parking space occupancy.

We analyzed the performance of the algorithm by running it on different images obtained from the web and assessing the system's ability to accurately detect and delineate parking spaces using the dilated edge detection method. This included monitoring the system's capacity to handle images with complex backgrounds and different vehicle sizes. By continuously refining our method based on these metrics, we aimed to achieve a system that was not only accurate but also reliable in a real-world parking scenario.

3.2.1 Procedure

Below is the image we used for this project. The image we took is an aerial view of parking spaces in minor sections with cars in some spots.

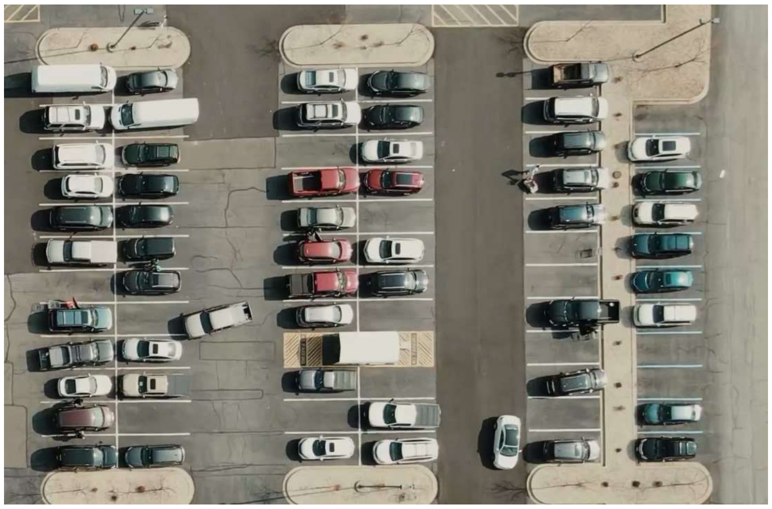

Fig 2. Colored Image of the Parking lot (Aerial View)

Then we converted the original image to a gray scale to reduce the info in the photo.

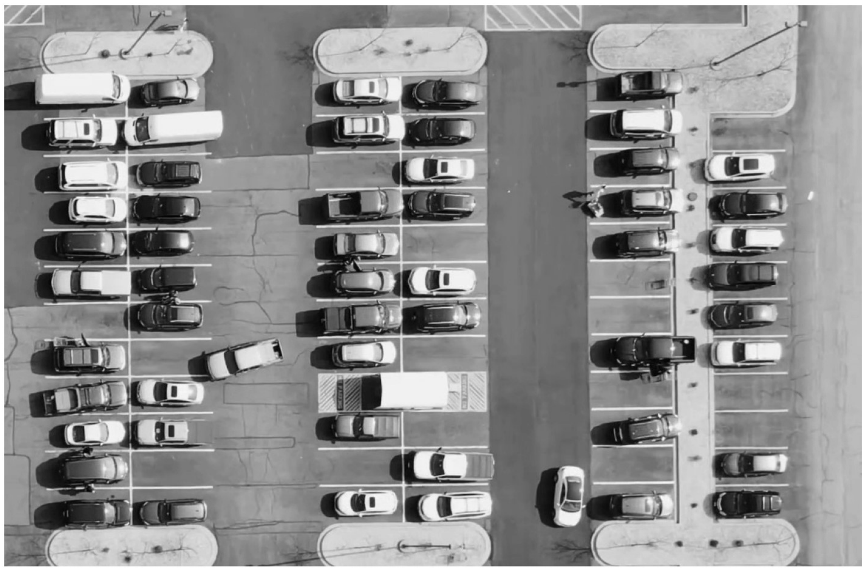

Fig 3. Gray Scale Image of the Parking lot

We gave it a good Gaussian Blur to remove even more unnecessary noise.

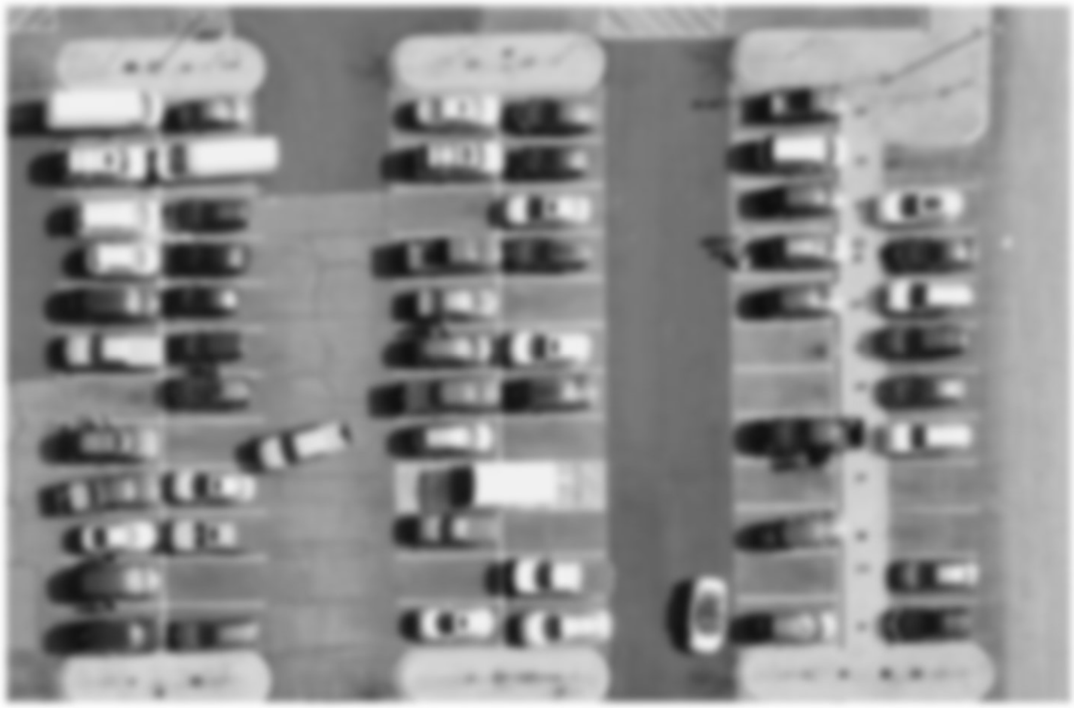

Fig 4. Gaussian Blur Image of Parking Lot

Then we detected the edges with canny algorithm by providing different thresholds.

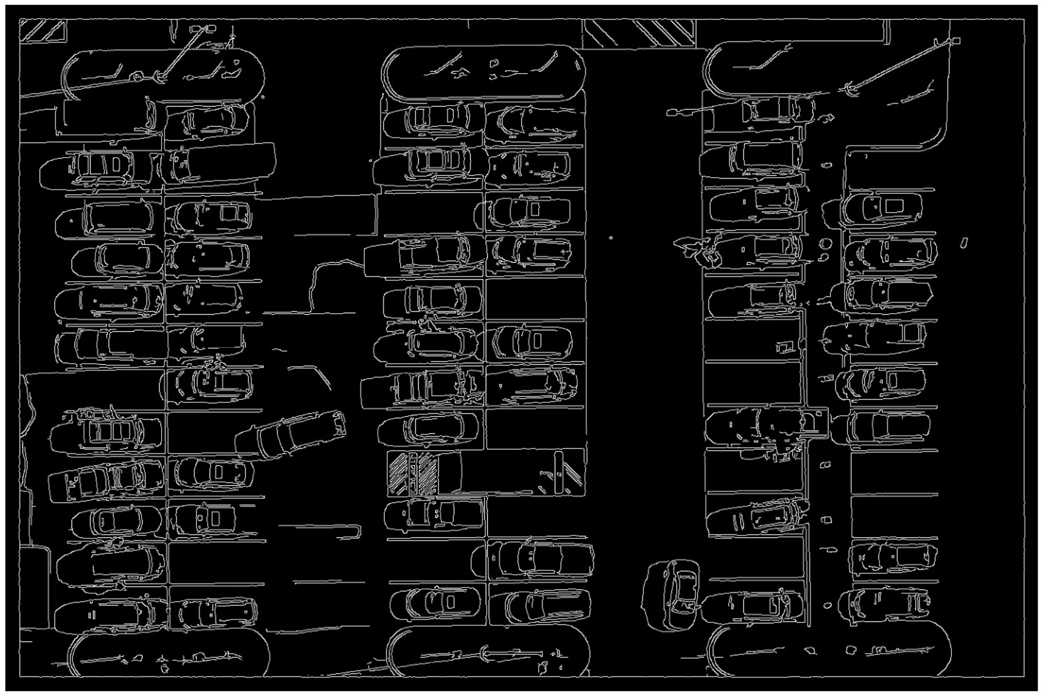

Fig 5. Detected the edges using Canny Edge in the Parking lot.

The edge image was then dilated with the four-stage morphological procedure using the four structural element object.

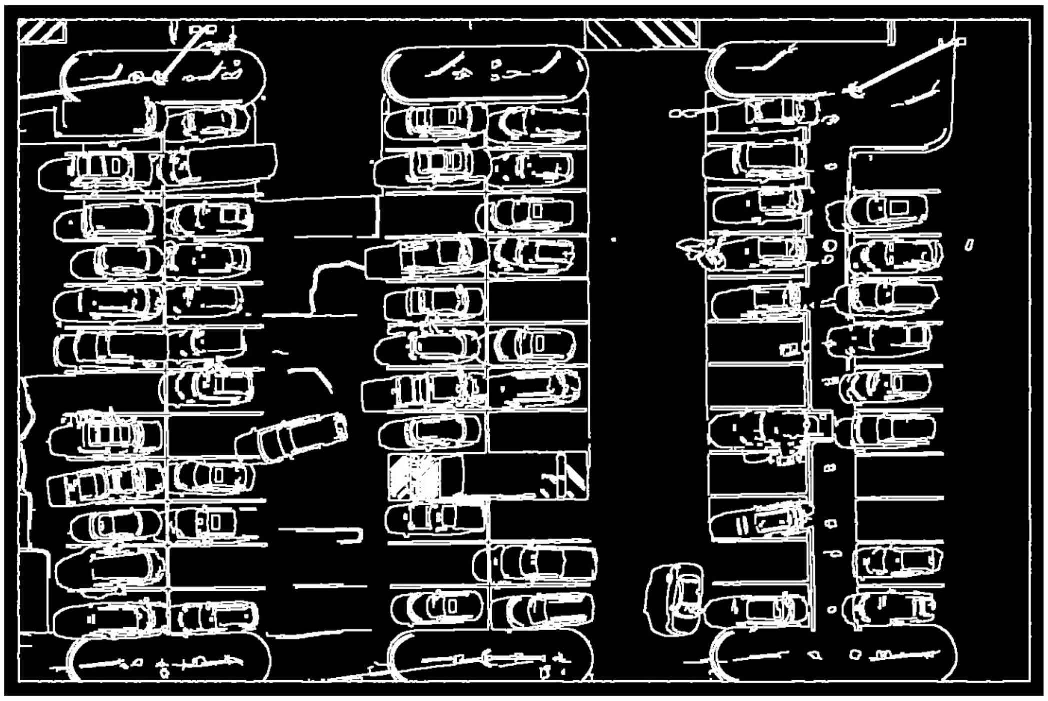

Fig 6. Completely Dilated Parking lot edge Image.

Then we clicked the parking spaces to show how many are available in specific parking lot.

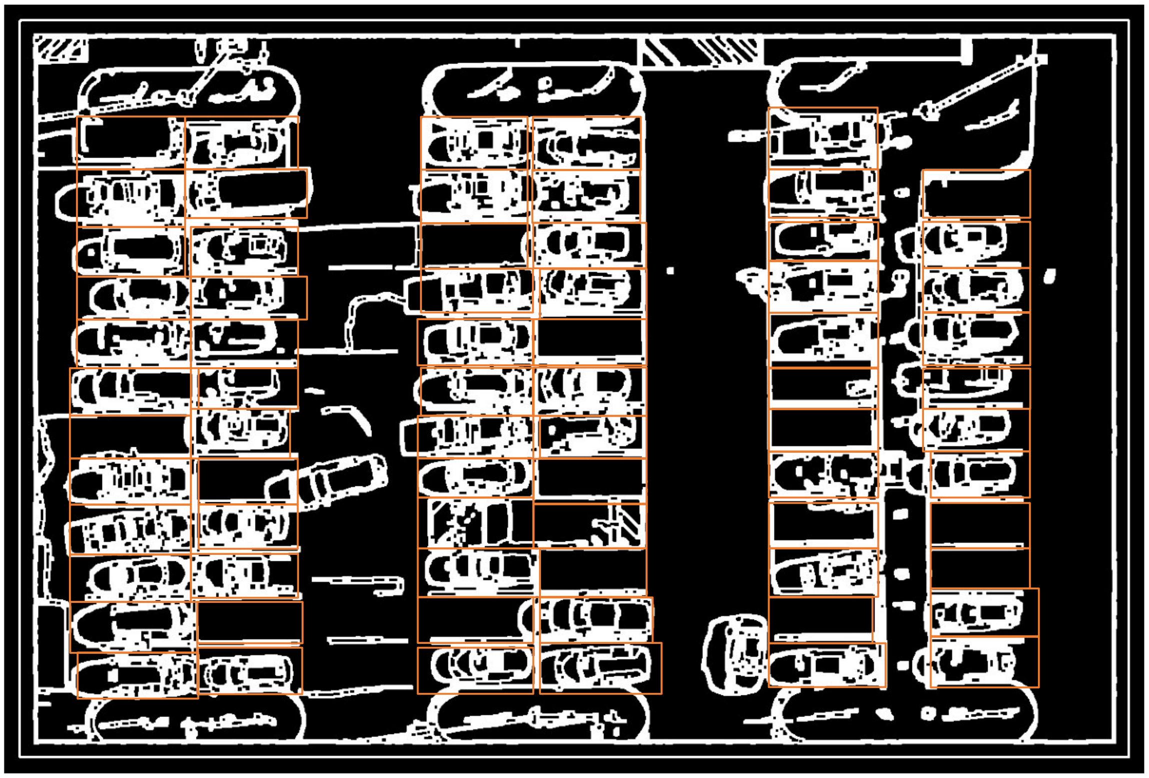

Fig 7. Marked Parked spaces in Dilated Parking lot edge Image.

3.3 **Results**

The image used is shown in Figure 2. The image was greyed, stretched, smoothed and binarized by using image processing techniques described earlier.

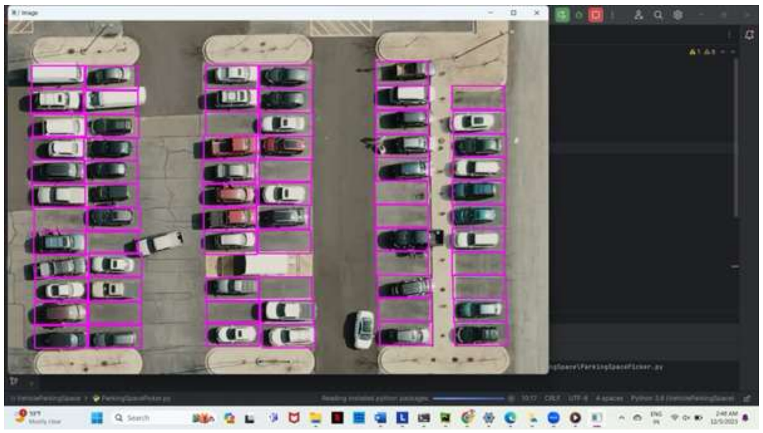

Fig. 8.

In the parking area, there are a total of 69 slots, with only 12 currently available, as depicted in Figure 9. The system uses red rectangles to indicate occupied slots, while green rectangles signify those that are available, determined through pixel counting.

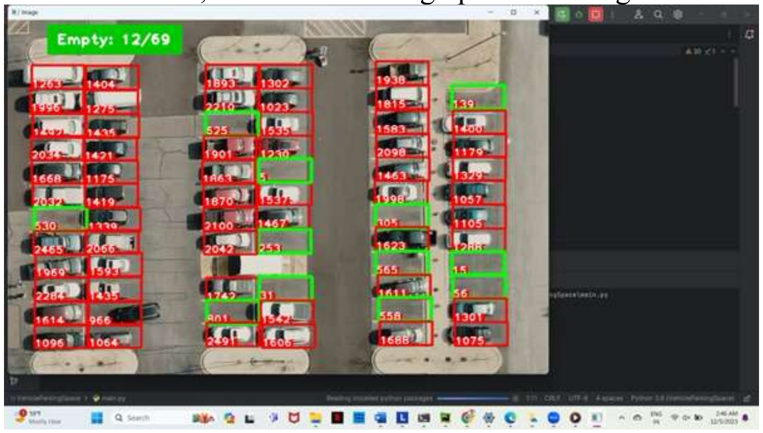

Fig 9.

Figure 10 illustrates that 15 slots have become available due to the departure of two vehicles, indicating the successful execution of the software code.

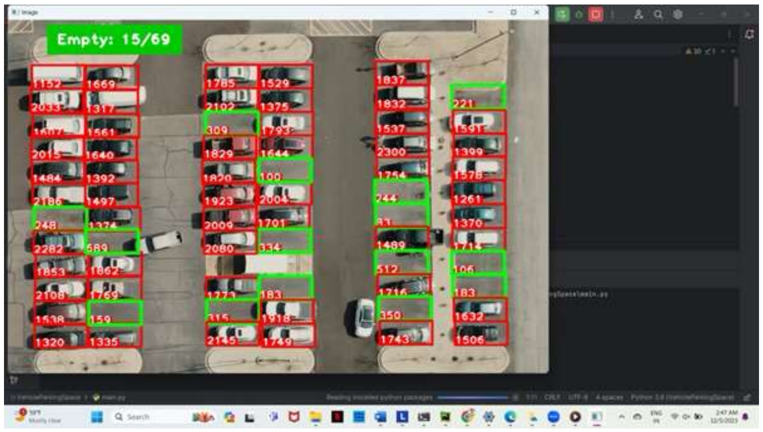

Fig 10.

3.4 **Analysis and Discussions**

During the execution of our code, we experienced several issues related to image quality and conditions. For instance, we initially used images of the parking lot taken from side angles, which led to skewed perspective and inaccurate vacancy detection. To rectify this, we repositioned our cameras to capture aerial images, ensuring a consistent top-down perspective that our algorithm could reliably interpret.

We also faced challenges with images taken during extreme weather conditions, such as heavy fog or rain, which obscured the parking lines and vehicles. To resolve this, we set up weather criteria for image capture, using only those images taken under clear weather conditions for processing. For continuous operation, we improved our image processing algorithms to include filters that reduced the impact of weather-related noise.

Another obstacle was the variance in lighting, which caused the images to be either overexposed or underexposed, affecting the system's detection capability. We addressed this by incorporating an adaptive lighting correction feature in our software, which adjusted the image contrast and brightness in real-time before processing.

The diversity of vehicles presented another complexity, as our system was not recognizing certain vehicle types accurately. We corrected this by enriching our dataset with images of various vehicle sizes and shapes, enhancing the system's recognition algorithms to be more inclusive and accurate.

In terms of real-time processing, the initial lag in our system was due to the heavy computational load. To overcome this, we streamlined our code and introduced more efficient data processing methods. We also adopted edge computing, which significantly reduced the response time by processing data directly at the source.

Integrating our system with existing infrastructures initially posed compatibility issues. We developed a more flexible and modular design, which allowed for easier integration with different types of infrastructure and technology platforms.

Scaling up the system revealed performance maintenance challenges. We anticipated this by designing the architecture to be scalable from the start, which facilitated a smoother transition as the load increased. Regular maintenance and updates became part of our routine to ensure the system remained efficient and up to date.

Lastly, we refined the user interface following feedback that it was not as user-friendly as we had expected. We simplified the design and improved the guidance provided to users, which made the system more intuitive and user-friendly.

4. <u>**Conclusion**</u>

In our report titled "Real-time Parking Space Monitoring System using CVZone and OpenCV," we conclude that the system we developed has successfully detected the presence of cars in parking slots. Specifically, the approach utilizing filled images yielded more accurate results than the dilated edge images for determining the vacancy of parking spots. Considering these findings, we believe that a combined system leveraging both approaches could enhance performance. Additionally, we discovered that adjusting the camera positions could further improve the system's accuracy. Our tests confirm that this image processing-based system is a practical and effective solution for managing parking space availability. Moreover, integrating this system with other technologies like automatic number plate recognition and traffic control systems could pave the way for a comprehensive intelligent transportation system. This advanced parking system aims to offer enhanced security and a seamless parking experience, providing customers with a more efficient and flexible way to park their vehicles.

5. <u>**Contribution**</u>

5. 1. **Code Contribution:**

I contributed in the **main.py** file, the primary functionality of the parking space monitoring system. The code uses OpenCV and CVzone libraries to process video footage from a file named 'carPark.mp4'. The code involves reading predefined parking positions from a file (CarParkPos), which is then used to crop and analyze specific areas of the video frames to detect parking space occupancy. This is done by counting non-zero pixels in each cropped image area and potentially marking the parking spaces as occupied or free based on this count. My work focuses on the real-time analysis of the parking lot, identifying vacant and occupied spaces, and updating this information on the video feed.

5. 2. **Report Contribution:**

In the report titled my contribution is encompassed working on the Abstract, Introduction, and Methods sections. The Abstract provided a concise summary of the project, highlighting its objectives, methodology, key findings, and conclusions. The Introduction section offered an overview of the project's background, problem statement, objectives, and the significance of parking space monitoring systems. In the Methods section, I described the approach and methodology employed in developing the real-time parking space monitoring system using CVZone and OpenCV, outlining the key steps and technologies utilized.

6. <u>**References**</u>

[1] Zhang Bin; Jiang Dalin; Wang Fang; Wan Tingting; "A design of parking space detector based on video image," Electronic Measurement & Instruments, 2009.

[2] Ichihashi, H.; Notsu, A.; Honda, K.; Katada, T.; Fujiyoshi, M.; "Vacant parking space detector for an outdoor parking lot by using surveillance camera and FCM classifier," Fuzzy Systems, 2009. FUZZ-IEEE 2009.

[3] Yusnita, R.; Fariza N.; Norazwinawati B.; “Intelligent Parking Space Detection System Based on Image Processing,” International Journal of Innovation, Management, and Technology, Vol. 3, No. 3, June 2012.

[4] N. True; “Vacant Parking Space Detection in Static Images,” Projects in Vision & Learning, University of California,2007[Online]. Available:
http://www.cs.ucsd.edu/classes/wi07/cse190-a/reports /ntrue.pdf.

[5] Najmi Hafizi Bin Zabawi, Sunardi, Kamarul Hawari Ghazali, “Parking lot detection using image processing method”, October 2013.

[6] Banerjee, Sayanti, Pallavi Choudekar, and M. K. Muju. "Real-time car parking system using image processing." Electronics Computer Technology (ICECT), 2011 3rd International Conference on. Vol. 2. IEEE, 2011.

[7] Shaaban, Khaled, and Houweida Tounsi. "Parking Space Detection System Using Video Images." Transportation Research Record: Journal of the Transportation Research Board 2537 (2015): 137-147.

[8] S. Saleh Al-Amri, N. V. Kalyankar, and Khamitkar S, “Image segmentation by using threshold techniques,” Journal of Computing. vol 2, Issue 5. MAY 2010,
ISSN 2151-9617.

[9] Li, Shiqiang, Hussain Dawood, and Ping Guo. "Comparison of linear dimensionality reduction methods in image annotation." Advanced Computational Intelligence (ICACI), 2015 Seventh International Conference on. IEEE, 2015.