CTU Journal of Innovation and Sustainable Development
ISSN 2588-1418 | e-ISSN 2815-6412

DOI:10.22144/ctujoisd.2026.012

An attempt at fall detection on an embedded device based on YOLOv8n-pose

Dinh-Tu Nguyen[1], Loc-Dinh Tran[1], Van-Minh Huynh[1], Hoai-Tan Nguyen[2], and Chi-Ngon Nguyen[3*]

[1]*Faculty of Mechanical Engineering, Can Tho University of Technology, Viet Nam*

[2]*Faculty of Mechanical Engineering, College of Engineering Technology, Can Tho University, Viet Nam*

[3]*Faculty of Automation Engineering, College of Engineering Technology, Can Tho University, Viet Nam*

**Corresponding author (ncngon@ctu.edu.vn)*

Article info.

Received 25 Mar 2025
Revised 28 Apr 2025
Accepted 07 Nov 2025

Keywords

Fall detection, real-time, surveillance camera, YOLOv8n-pose

ABSTRACT

Human-action recognition aims to identify the actions performed by individuals. Due to the broad spectrum of human activities, action recognition covers a wide range. Among all, fall detection is a critical aspect of surveillance, particularly in environments where individuals are at risk. Throughout the years, several sensors, data types, and classification techniques have been investigated to address this issue. This paper proposes a lightweight fall-detection system that processes image sequences in real time. This system is deployed on an embedded device, specifically the Jetson Nano. Our goal is to construct a comprehensive dataset that accurately detects falls in various lighting conditions. The proposed system is constructed using YOLOv8n-pose, which has been trained to identify people using widely recognized dataset. Our methodology includes the design and implementation of YOLOv8n-pose, data collection, and rigorous testing to ensure the accuracy of real-time fall detection using a surveillance camera. The experimental results show that high detection over 90% accuracy and acceptable timing capabilities are achieved.

1. INTRODUCTION

Falls are a significant health concern, particularly for the elderly and individuals with neurodegenerative disorders such as Parkinson's disease. According to recent statistics, between 45% and 68% of people with Parkinson's disease will fall each year (Lima et al., 2022). Falls are one of the major causes of emergency room visits and hospitalizations for people with Parkinson's disease (Paul et al., 2017; Dorsey & Bloem, 2024; Herbers et al., 2024). Furthermore, nearly one million people in the U.S. are living with Parkinson's disease, a number expected to rise to 1.2 million by 2030 (Marras et al., 2018).

Recently, there are a lot of commercial cameras, which applied smart detection (relying on cloud services for motion detection, person detection, zone selection). However, it has some drawbacks when compared to programmable detection systems, such as limited customization (usually offers only basic setting), lack of smart detection, cloud dependency, and limited scalability/integration.

Given these drawbacks of commercial cameras, the need for effective fall detection in programmable systems is more critical than ever. This paper presents an approach to real-time fall detection using embedded devices and surveillance cameras. Our method leverages the power of YOLO model, a

state-of-the-art machine learning model for human pose estimation, to accurately detect falls in real-time (Yadav et al., 2023; Dong & Du, 2024).

By implementing this system, we aim to provide a cost-effective, efficient, and reliable solution to fall detection, ultimately improving the safety and quality of life for those at risk. The following sections will detail our methodology, experimental setup, and results.

2. RELATED WORK

Fitriawan and colleagues proposed a low-cost fall detection system using Raspberry Pi, a 3-axis accelerometer, GSM, and a GPS module. The system compares acceleration with determined upper fall threshold values to detect early fall events. When a fall is detected, the system sends an alert message along with location information to the responsible person (Fitriawan et al., 2024), (Arulalan et al., 2024).

However, Raspberry Pi while cost-effective, may not provide the computational power necessary for more complex fall detection algorithms (De Miguel et al., 2017; Arulalan et al. 2024). In comparison, the Jetson Nano was faster and more powerful. Therefore, the Jetson Nano is a more suitable choice for real-time fall detection.

In terms of object detection models, YOLOv5 has been widely used due to its speed, simplicity, and accuracy (Wang et al., 2023). The results achieved real-time accurate fall detection using the improved YOLOv5s model, but the speed and volume of the algorithm were still deficient.

On the other hand, YOLOv8 is one of the latest model in the YOLO family, which is faster and more accurate than YOLOv5 (Selcuk & Serif, 2023; Sohan et al., 2024). It provides a unified framework for training models for performing object detection, instance segmentation, and image classification. A project on GitHub has focused on training YOLOv8 on a Falling Dataset with the goal of enabling real-time fall detection. This project aimed to harness the capabilities of YOLOv8, a cutting-edge object detection model, to enhance the efficiency of real-time fall detection.

In conclusion, based on the advantages of Jetson Nano over Raspberry Pi and the superior performance of YOLOv8 over YOLOv5, we choose to use Jetson Nano and YOLOv8n-pose for our project. We believe this combination will provide a more efficient and accurate real-time fall-detection system.

3. METHODOLOGY

3.1. System overview

The YOLOv8n-pose model was utilised to detect falls in real-time from a surveillance camera. The model, renowned for its speed and accuracy, has been optimised in its YOLOv8n-pose version to accurately detect fall events (Chang et al., 2024).

Roboflow was employed for data preprocessing and augmentation. It provides robust tools for efficient data preparation, including data splitting into training, validation, and testing sets, and applying data augmentation techniques like image flipping and brightness enhancement (Alexandrova et al., 2015).

Google Colab, a cloud-based Jupyter notebook environment by Google was used to train the model. Google Colab allows us to leverage Google's powerful GPU for quick and efficient model training (Bisong, 2019).

Proposed system was designed to operate in real-time. Upon detecting a fall, the proposed system automatically sends an alert, ensuring the safety of users in the field of view (FOV) of the surveillance camera.

Figure 1. Schematic of the proposed system

The process of the project is as follows:

• Step 1: Images are captured by the surveillance camera.

• Step 2: Images are uploaded to the cloud for storage.

• Step 3: Python and the YOLOv8n-pose model analyze the images to detect falls.

– Training the pose model with YOLOv8n-pose using the Ultralytics framework was a straightforward task.

– In this research, Ultralytics framework was used to provide training option through command-line interface (CLI).

– Python was used to perform the model in Pycharm (Python Integrated development enviroment - IDE) in order to inference on the test images.

• Step 4: Results are shown on a display connected to a powerful Jetson Nano for fast and clear visualization.

3.2. YOLOv8n-pose

YOLOv8n-pose has several unique features that make it better suited for fall detection than YOLOv8. Firstly, YOLOv8n-pose is designed for pose estimation, which is crucial because it involves determining a person's position and orientation in an image. Secondly, YOLOv8n-pose uses an anchor-free detection system, which can improve accuracy by directly predicting a person's boundaries in an image rather than relying on predefined bounding boxes. Lastly, YOLOv8n-pose has modified convolutional blocks, which can enhance the model's ability to learn from image data and improve its performance in detecting falls.

3.3. Roboflow

To optimize the system, we leveraged a Roboflow dataset for training YOLOv8, an image recognition model. However, we added this dataset with nighttime images to ensure effective fall detection in low-light condition environments. Additionally, we meticulously hand-labelled all images to guarantee accuracy for our specific project requirements. This meticulous data preparation approach yielded a dataset perfectly suited for real-time fall detection, even under challenging nighttime conditions.

Figure 2. Representative images from the dataset

We applied data augmentation to strengthen our dataset, improving fall detection robustness and generalizability.

Figure 3. Dataset tuning options

As illustrated in Figure 3, we adopted methods such as increasing the image brightness by 25%, elevating the bounding box brightness by 25%, and boosting the bounding box exposure increase between -10% and 10%. Following the data augmentation, the final dataset comprised 4226 images. These were derived from 2975 images in the training set, 845 in the validation set, and 406 in the test set. The images in the dataset were categorized into two labels: "normal" and "fall".

Figure 4. Representative images from the dataset using augmentations

3.4. Embedded Devices

For our real-time fall detection system (Figure 5), we chose the cost-effective Jetson Nano 2GB (as listed in Table 1), known for its AI prowess and powerful processing, paired with the Imou Ranger SE A23 camera (as listed in Table 2). The Jetson Nano seamlessly analyzes live video, empowering the system with the speed and performance needed for accurate fall detection.

Figure 5. The Jetson Nano and the Imou Ranger SE A23 camera

Table 1. NVIDIA Jetson Nano 2GB specification

Specification	Details
CPU	Quad-core ARM, 1.43 GHz
GPU	128-core NVIDIA Maxwell
RAM	2GB
Storage	microSD
Connectivity	802.11ac wireless (Gigabit Ethernet)
Operating system	Ubuntu 18.04
I/O	40 pin Header (GPIO, I2C, I^2S, SPI, UART) 12 pin Header (Power and related signals, UART) 4 pin Fan Header
Dimensions	100 mm × 80 mm × 29 mm

The Imou Ranger SE A23 camera was selected for its high resolution, effective night vision, and 360-degree pan-tilt capabilities. Images captured by this camera are then fed into the YOLOv8n-pose model. This model excels at identifying keypoints on objects within an image, particularly human body parts. This allows us to determine the position and pose of individuals in the video, enabling real-time fall detection.

Table 2. Imou Ranger SE A23 specification

Specification	Details
Aspect ratio	16:9
IR distance (m)	10 m
Zoom	16X Digital Zoom
Storage capacity	microSD (up to 256 GB)
Connectivity	Wifi 2.4 GHz and USB 3.0
Maximum resolution	1920 × 1080
View	360° coverage
Frame rate	Up to 25/30 fps frame rate

4. RESULTS AND DISCUSSION

In this section, we compare the results of two models: YOLOv8n-pose with augmentation and YOLOv8n without augmentation. Both models are trained with the same dataset, consisting of 4226 images, with the following parameters:

• Epochs = 100: The number of iterations over the entire dataset during training. Each epoch is a complete pass through the dataset.

• ImgSz = 640: The size of the input image for the model. All images are resized to 640 x 640 pixels before being fed into the model.

• Lr = 0.0001: The initial learning rate. This crucial parameter determines the speed at which the model's weights are updated during training.

The purpose of this training is to observe differences in posture or action-label identification between the two models. This helps us better understand each model's performance and capabilities, enabling us to make the best choice for specific needs. Through comparison, we can analyse and evaluate factors such as accuracy, processing speed, and detection capabilities of each model.

4.1. Chart after training

The YOLOv8-Pose model, once trained, will produce two main types of graphs: "Box" and "Pose".

"Box" chart in YOLOv8 is commonly used for object detection. During training, the model learns to predict bounding boxes around objects within the image. Each bounding box includes the x and y coordinates of its center, its width and height, and the type of object inside it. For instance, in an image of a person walking, the "Box" chart would create a box around the person, including information about the center position, the size of the box, and potentially the label "Person walking."

"Pose" chart in YOLOv8 refers to pose estimation. Pose estimation involves determining the location of specific points on an object, often called keypoints. These keypoints can represent different parts of the object, such as human body joints, road surface markers, or other prominent features. Example: In the same image of a person walking, the "Pose" chart wouldn't just create a bounding box; it would also determine the location of keypoints on the human body like knees, elbows, ankles, etc.

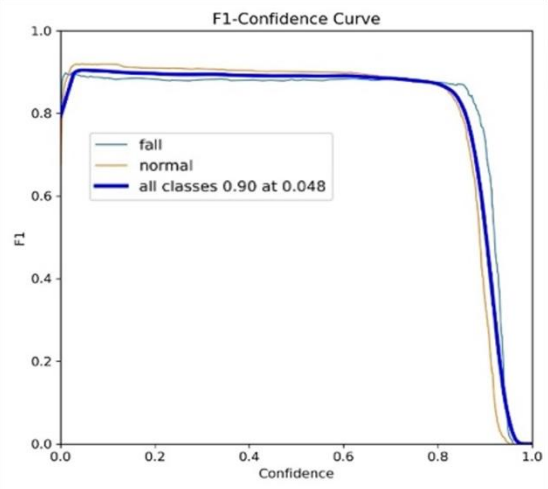

Figure 6. Relationship between F1 score and Confidence BoxF1_curve chart

The BoxF1_curve graph reveals the relationship between the F1 score and Confidence for the detection of "Fall" and "Normal" classes. Both curves show a significant rise and a plateau at high F1-score levels, indicating good model performance at certain Confidence thresholds. Optimal performance is noted when all classes achieve an F1 score of 0.90 at a Confidence level of 0.048.

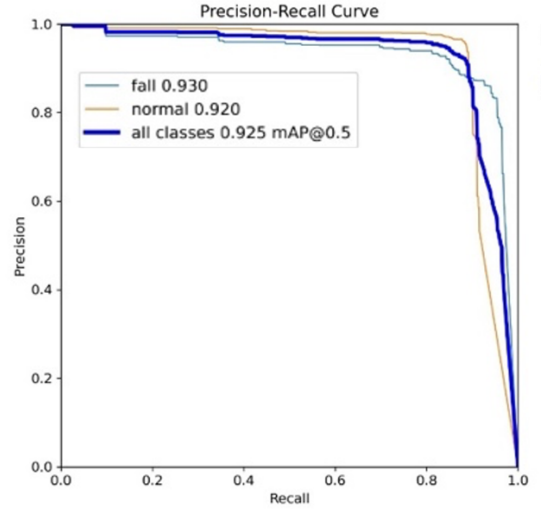

Figure 7. Relationship between Precision and Recall in the BoxPR curve chart

The BoxPR_curve graph shows the relationship between Precision and Recall for two classes. Specifically:

• The "Fall" curve (light blue): Initially, with low Recall, Precision is high. However, as Recall increases, Precision decreases. This is known as the trade-off between Precision and Recall. Often, increasing Precision reduces Recall and vice versa. Because higher Precision usually means fewer False Positives (predicted as Positive but actually Negative), which may lead to missing some true Positive cases, reducing Recall. To comprehensively evaluate the classification model's performance, we use the F1 score, the harmonic mean of Precision and Recall (Buckland et al. & Gey, 1994). The higher the F1 score, the better the model.

• The "Normal" curve (orange): Similar to the "fall" curve, the Precision for the "Normal" class also decreases as Recall increases. However, the Precision for the "Normal" class remains higher than that for the "fall" class as Recall increases, indicating the model's better ability to classify "Normal" cases accurately.

• Optimal Point (dark blue line): The optimal performance across all classes shows a mean Average Precision (mAP) of 0.925 at an IoU threshold of 0.5, indicating strong model accuracy.

4.2. Experimental results

Beyond basic ("normal" or "fall") object detection, our goal is to identify falls disguised as normal activity. We compare YOLOv8n (object type and position) with YOLOv8n-pose, pose analysis. This will expose the limitations of YOLOv8n and demonstrate the power of YOLOv8n-pose in detecting falls.

As shown in Figure 8, YOLOv8n's weakness is its inability to capture pose information. This comes from collapsing the file into an object of specified type and location within the frame. This mode of return should be especially evident in matters requiring deeper analysis, limited consideration such as a person sitting. In low light conditions, the disadvantage of YOLOv8n is even greater when it cannot decode posture, thereby lacking the ability to analyze posture.

On the contrary, YOLOv8n-pose gave better results. YOLOv8n-pose accurately identifies even sitting postures, even in low-light environments. This clear contrast highlights the important role that posture analysis plays in achieving robust fall detection systems.

(a) (b)

Figure 8. Reprensentative results of (a) YOLOv8n; (b) YOLOv8n-pose model for label "Normal''

(a) (b)

Figure 9. Representative results of (a) YOLOv8n; (b) YOLOv8n-pose model for label ''Fall''

Our in-depth analysis revealed striking discrepancies in fall detection performance between YOLOv8n and YOLOv8n-pose across both daytime and nighttime conditions.

Daytime Performance: In daylight conditions, YOLOv8n demonstrated limitations in accurately identifying falls. It primarily detected basic falling postures, essentially capturing a limited range of fall directions. In contrast, YOLOv8n-pose excelled at detecting falls with similar angles that YOLOv8n missed. This superior performance highlights YOLOv8n-pose's enhanced ability to capture posture nuances, enabling a more comprehensive understanding of an object's state. Moreover, as previously discussed, YOLOv8n-pose's strong fall detection performance during the day underscores its effective training in recognizing falls with precision.

Nighttime Performance: Similar shortcomings emerged with YOLOv8n in low-light conditions, as it failed to identify falls compared to YOLOv8n-pose, as exemplified in Figure 9(a). These results demonstrate YOLOv8n-pose's remarkable ability to not only accurately identify postures but also discern complex actions, a crucial capability for real-world fall detection and prevention systems.

4.3. Evaluation of experimental results

We evaluate the fall detection system's experimental results under two lighting conditions: full light (daytime) and low light (nighttime). Results were collected from five trials for each condition and analyzed for two labels: Normal and Fall.

The results are listed in the Table 3. These results indicate that the fall detection system performs well under both lighting conditions, although accuracy is higher during the day than at night. The slight decrease in accuracy at night suggests that lighting conditions affect the system's performance, but it remains reasonably effective across different scenarios. Note that, testing result in Table 3 was done by a laptop.

Table 3. Summary of average system accuracy under different lighting conditions

Test	Day Normal	Day Fall	Night Normal	Night Fall
Test 1	96%	95%	94%	92%
Test 2	94%	93%	97%	95%
Test 3	97%	96%	95%	93%
Test 4	95%	92%	93%	96%
Test 5	98%	94%	96%	91%
Test 6	93%	95%	92%	97%
Test 7	96%	94%	98%	94%
Test 8	95%	93%	94%	95%
Test 9	97%	96%	95%	92%
Test 10	94%	92%	97%	96%
Average of 10 tests	95.6%	94.1%	95.1%	94.2%

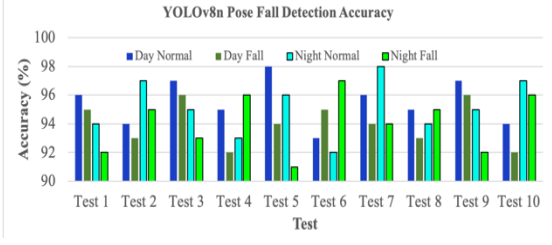

Figure 10. Performance evaluation of YOLOv8-pose in day and night condition

These results are also visualized in Figure 10, which facilitates easy comparison of accuracy across trials and lighting conditions. The figure clearly shows that despite variations in accuracy across trials, the system maintained a stable performance in both environments.

The analysis and comparison of these results help us better understand the system's performance in real-world conditions and guide future improvements.

4.4. Embedded devices deployment experiment

The experiments in this section are designed to evaluate the detection speed of the YOLOv8n-pose on an embedded device in practice. In our project, we used Jetson Nano as the main processing center for monitoring. However, the results were not as good as we expected. Specifically, we used a Jetson Nano 2GB as the main processor, but with 2GB of RAM, it could not meet our monitoring needs.

As shown in Figure 11, the test results showed that the average frames per second (fps) was only about 1.37, the preprocessing time was 9.2 ms, the inference time was 608 ms, and the post-processing time was 3.5 ms per image at a specific shape, and the frame was 2.0.

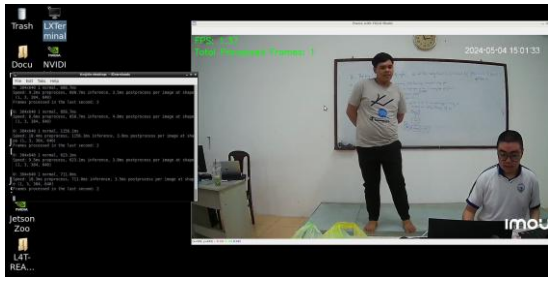

Figure 11. Representative result under YOLOv8n-pose using Jetson Nano as an embedded device

From these results, we see that improving the embedded device in the future is necessary. One way could be to switch to using at least Jetson Nano 4GB or stronger versions of Jetson, like Jetson Xavier NX. This not only enhances processing capabilities and reduces inference time, but also improves real-time fall detection, better meeting our monitoring needs.

5. CONCLUSION

In the process of implementing the study "Real-time fall detection for embedded devices based on YOLOv8n-pose", we conducted numerous tests with a real-time surveillance camera and the YOLOv8n-pose model on the Jetson Nano hardware platform. The results exceeded our expectations. Specifically, the "normal" and "fall" cases with complex forms were classified with good accuracy.

Previous studies have not fully explored the potential of combining YOLOv8n-pose's detection accuracy with the power of Jetson. The solution we propose aims to optimise fall detection and can be applied to larger applications such as alerting Parkinson's patients when they faint or in more severe fall cases.

Finally, we believe that with advancements in technology and the continuous development of AI models, the ability to detect falls will continue to improve, bringing practical benefits to the community and society.

CONFLICT OF INTEREST

The authors declare that they have no known conflict of interest.

REFERENCES

Arulalan, V., Muralidharan, C., Sahayaraj, K. K. A., & Garg, P. (2024, May). Vision based fall detection model using Raspberry Pi, In *International Conference on Innovations and Advances in Cognitive Systems* (pp. 369-382). Cham:Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-69201-7_28

Buckland, M., & Gey, F. (1994). The relationship between recall and precision. *Journal of the American Society for Information Science*, *45*(1), 12-19. https://doi.org/10.1002/(SICI)1097-4571(199401)45:1<12::AID-ASI2>3.0.CO;2-L

Dorsey, E. R., & Bloem, B. R (2024). Parkinson's disease is predominantly and enviromental disease. *Journal of Parkinson's Disease*, *14*(3), 451-465. https://doi.org/10.3233/JPD-230357.

Dong, C., & Du, G. (2024). An enhanced real-time human pose estimation method based on modified YOLOv8 framework. *Scientific Reports*, *14*(1), 8012. https://doi.org/10.1038/s41598-024-58146-z.

Herbers, C., Zhang, R., Erdman, A., & Johnson, M. D (2024). Distinguishing features of Parkinson's disease fallers based on wireless insole plantar pressure monitoring, *npj Parkinson's Disease*, *10*(1), 67. https://doi.org/10.1038/s41531-024-00678-2.

Marras, C., Beck, J. C., Bower, J. H., Roberts, E., Ritz, B., Ross, G. W., et al. (2018). Prevalence of Parkinson's disease across North America. *npj Parkinson's Disease*, *4*(1), 21. https://doi.org/10.1038/s41531-018-0058-0.

Paul, S. S., Harvey, L., Canning, C. G., Boufous, S., Lord, S. R., Close, J. C. T., & Sherrington, C. (2017). Fall-related hospitalization in people with Parkinson's disease, *European journal of neurology*, *24*(3), 523-529. https://doi.org/10.1111/ene.13238.

Sohan, M., Sai Ram, T., Reddy, R., & Venkata, C. (2024, January). A review on YOLOv8 and its advancements. In *International Conference on Data Intelligence and Cognitive Informatics* (pp. 529-545). Springer Nature Singapore. https://doi.org/10.1007/978-981-99-7962-2_39.

Selcuk, B., & Serif, T. (2023, August). A comparison of YOLOv5 and YOLOv8 in the context of mobile UI detection, In *International Conference on Mobile Web and Intelligent Information Systems* (pp. 161-174). Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-39764-6_11.

Wang, Y., Chi, Z., Liu, M., Li, G., & Ding, S. (2023). High performance lightweight fall detection with an improved YOLOv5s algorithm. *Machines*, *11*(8), 818. https://doi.org/10.3390/machines11080818.

De Miguel, K., Brunete, A., Hernando, M., & Gambao, E. (2017). Home camera-based fall detection system for the elderly. *Sensors*, *17*(12), 2864. https://doi.org/10.3390/s17122864.

Yadav, A., Chaturvedi, P. K., Rani, S., Tripathi, A. K., & Shrivastava, V. (2023). Object detection and tracking using YOLOv8 and DeepSORT. *Advancements in Communication and Systems*, 81-90. https://doi.org/10.56155/978-81-955020-7-3-7.

Lima, D. P., de-Almeida, S. B., Bonfadini, J. D. C., Carneiro, A. H. S., Luna, J. R. G. D., Alencar, M. S. D., & Braga-Neto, P. (2022). Falls in Parkinson's disease: the impact of disease progression, treatment, and motor complications. *Dementia & Neuropsychologia*, *16*(2), 153-161. https://doi.org/10.1590/1980-5764-DN-2021-0019.

Fitriawan, H., Purwiyanti, S., Faturrohman, E. A., Santoso, M. R. F., Darajat, A. U., & Gunawan, T. S. (2024, July). Development of a Low-Cost Fall Detection System for the Elderly with Accurate Detection and Real-Time Alerts. In *2024 IEEE 10th International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA)* (pp. 309-314). IEEE. https://doi.org/10.1109/ICSIMA62563.2024.10675589.

Alexandrova, S., Tatlock, Z., & Cakmak, M. (2015, May). RoboFlow: A flow-based visual programming language for mobile manipulation tasks. In *2015 IEEE international conference on robotics and automation (ICRA)* (pp. 5537-5544) IEEE. https://doi.org/10.1109/ICRA.2015.7139973.

Bisong, E. (2019). *Google colaboratory. In Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners* (pp. 59-64). Berkeley, CA: Apress.

Chang, L., & Zhang, Z. (2024, December). Drowning Detection Based on YOLOv8-Pose and Human Pose Estimation. In *2024 10th International Conference on Systems and Informatics (ICSAI)* (pp. 1-6). IEEE. https://doi.org/10.1109/ICSAI65059.2024.10893825.