Real-Time Monitoring of Door Status in Public Transit Systems

R12942102, Hao Huang
M112022222, Kai-Chen Yang
R12942009, Po-Yi Liao

I. Introduction

In public transit systems, the status of vehicle doors is a critical aspect of passenger safety and operational efficiency. Timely and accurate detection of door states—whether they are opening, closing, or fully open/closed—is essential for ensuring that passengers are not caught in doors and that vehicles do not depart with doors improperly closed. Traditionally, this monitoring is conducted manually or with simple sensors, but such methods can be prone to errors and do not provide real-time analysis or visual confirmation.

This report introduces a method for monitoring of door status in public transit systems using computer vision techniques. Specifically, we leverage the capabilities of the YOLOv5 model for object detection and tracking to analyze video footage from onboard cameras. The process involves converting video into individual frames, manually annotating these frames to label door positions, and applying various data augmentation techniques to enhance model training.

II. Methodology

Our technique involves two distinct phases. First, we use an object detection model to identify the door's position and determine whether it is open or closed. In the second phase, we analyze the detected open/close status over time to ascertain whether the door is in the process of opening or closing.

A. Object Detection

In this part,we utilized the YOLOv5 model to initially determine on a broad scale whether the door in each frame is open or closed.

1) Data Preprocessing: First, we convert the video into frames and manually annotate the door's position as shown in Fig 1. Each frame will have a corresponding label file. Next, we perform data augmentation, including techniques such as mosaic augmentation, random affine transformations (scaling, translation, rotation, and shearing), color jitter, horizontal flipping, scale and aspect ratio adjustments, cutout, MixUp, HSV augmentation, random grayscale conversion, and adding slight noise to bounding box coordinates. Finally, we split the data into training, validation, and test sets based on an 0.8, 0.1, 0.1 ratio.

2) Training and Results: For the training phase, we use a pretrained YOLOv5x model, which is one of the larger and more powerful versions of the YOLOv5 family. YOLOv5x is known for its high accuracy in object detection due to its deeper architecture and larger number of parameters. We set the number of epochs to 50 for this training.

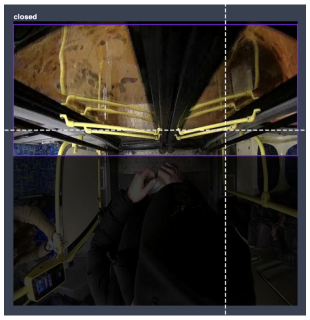


Fig. 1: Bounding Box Annotation.

Fig 2. shows the training process of the model, highlighting that losses (box, objectness, classification) decrease over time, indicating improved learning. Metrics such as precision, recall, and mean Average Precision (mAP) show high performance, suggesting the model effectively detects and classifies objects. Validation metrics reflect similar trends, with some variability, pointing to overall successful training.

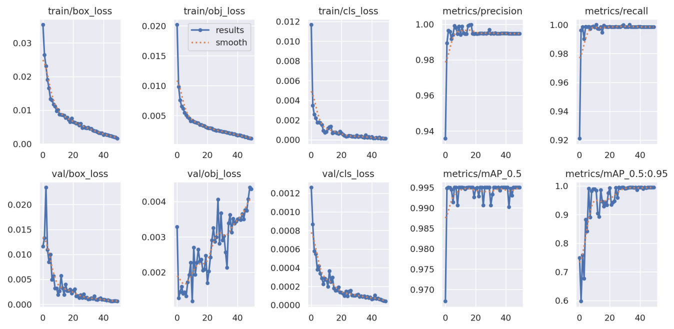


Fig. 2: Training Process.

When using this model to evaluate the door's open/close status, we found that the door's direction in the frame is crucial. Upon observation, in the sample videos, the doors are either at the top or bottom of the frame. However, in Test video 09.mp4, the door is on the left side of the frame, resulting in poor performance, with an average confidence level below 0.1 as shown in Fig 3(a).

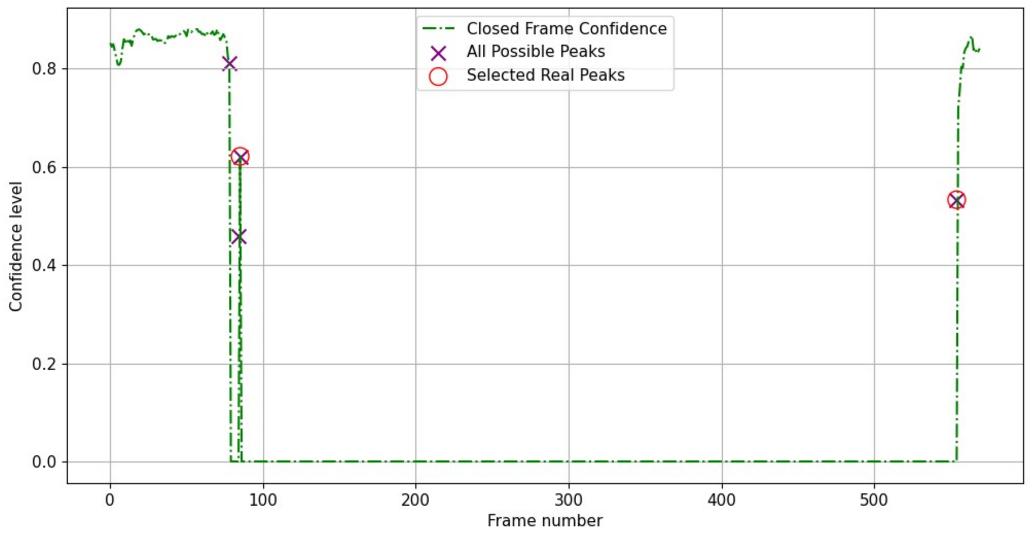
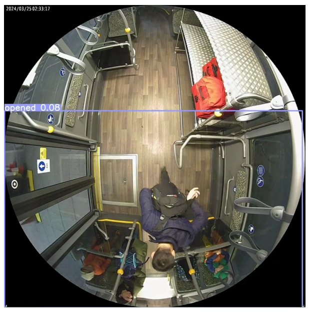
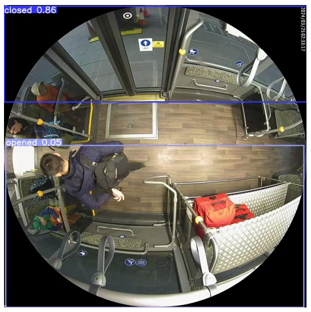


Fig. 4: Closed Door Confidence Level Over Frames with Peaks



(a)the door is on the left of the frame.



(b)the door is on the top of the frame (rotated).

Fig. 3: Comparison of door detection results based on door orientation in the frame.

To address this, we rotated 09.mp4 clockwise and observed a significant improvement, with the average confidence level rising above 0.8 as shown in Fig 3(b). We also attempted to rotate the training data clockwise but found that the generated model still did not perform well. Therefore, we decided to handle this issue by rotating the input video during processing.

B. Movement Analysis

In this part, using the trained model from the previous step, we can easily determine the open/close status of the door in each frame of the video. Our model is more effective at detecting when the door is closed; hence, we primarily rely on the confidence level associated with the door being closed. The closed frame confidence level is illustrated in Fig. 4. By setting a threshold, we classify the door as closed if the confidence level exceeds this threshold. Conversely, if the confidence level is below the threshold, we infer that the door may be open.Additionally, we have implemented other thresholds to filter out data with excessively low confidence levels, thus helping to smooth the curve and enhance the reliability of the analysis.

However, the previous steps only allow us to determine if the door is open or closed and do not differentiate between the transitional states of opening and closing. To address this issue, we use the closed door confidence level to identify frames where there is a change in the door's state. When there is a significant difference in confidence levels between consecutive frames, we treat these frames as all possible peaks, as shown by the purple crosses in Fig. 4. Among these possible peaks, some may result from model misjudgment. We use Algorithm 1 to remove anomalous peaks and identify the real peaks, as indicated by the red circles in Fig. 4. Below steps is the brief descriptions of Algorithm 1.

1) Boundary Check: We first check if any peaks are at the beginning or end of the video. If there are not enough frames around a peak to make an informed decision, then that peak is considered invalid.
2) Initial State Assumption: Since we can assume that the door is closed at the start of the video, the first peak we need to find should indicate an opening. By calculating the count of real peaks (initially assumed to be 0) and taking modulo 2, we can determine whether we are looking for an opening or closing peak.
3) State Consistency Around Peaks: For a valid opening peak, all frames to the right should indicate the door is open; conversely, for a closing peak, all frames to the left should indicate the door is open. Based on this, we decide whether to place the observation window on the left or right side of the peak.
4) Window Observation: By observing the states within the

window—where open is represented as 1 and closed as 0—we calculate the average to estimate the proportion of closed states within the window. If there are too many closed states (i.e., the average is too low), then this peak is also not considered genuine.

These steps ensure that only significant changes in door status are recorded as true peaks, enhancing the accuracy of our frame-by-frame analysis of door opening and closing actions. This method is critical in maintaining high precision in our model's performance and delivering reliable results in practical applications.

Algorithm 1 Generate Opening and Closing Frames

```
1:  Input: All possible peak frame indices
2:  Initialize: window_size ← 10, real_peak_i ← 0, real_peak
    ← [], opening_frames ← [], closing_frames ← []
3:  function IS_VALID_PEAK(peak_index,real_peak_i)
4:      if peak_index is too close to the start or end then
5:          return False
6:      end if
7:      if real_peak_i is even then
8:          Check if the right window indicates the door is
    open
9:          if too many closed states or nearby peaks are found
    then
10:             return False
11:         end if
12:     else
13:         Check if the left window indicates the door is open
14:         if too many closed states or nearby peaks are found
    then
15:             return False
16:         end if
17:     end if
18:     return True
19: end function
20: for each peak_index in all possible peak frame indices do
21:     if IS_VALID_PEAK(peak_index,real_peak_i) then
22:         Add peak to real_peak
23:         Increment real_peak_i
24:     end if
25: end for
26: for each real peak in real_peak do
27:     if peak count is even then
28:         Add to opening_frames
29:     else
30:         Add to closing_frames
31:     end if
32: end for
33: Output: opening_frames, closing_frames
```

III. Conclusion

In summary, our algorithm utilizes a trained model to initially predict whether a video frame indicates a door opening or closing. It then employs a series of conditions to sift through and identify the true peaks, ultimately determining the frames for opening and closing. Looking forward, we believe improvements can be made by applying more pre-processing to the videos. For example, we could filter out dark or black areas of the video (as most doors have black parts), then use edge detection to identify edges. Furthermore, after detecting edges, we could use morphological operations (such as Dilation and Erosion) to enhance the integrity of these edges. The goal is to minimize the presence of irrelevant objects in the frame as much as possible, which would simplify the visual data significantly.

References

[1] João Gaspar Ramôa, "Real-time 2D–3D door detection and state classification on a low-power device," https://link.springer.com/article/10.1007/s42452-021-04588-3#Sec12, 2021.

[2] "How to Train Custom Data in YoloV5," https://docs.ultralytics.com/yolov5/tutorials/train_custom_data/

[3] "Yolov5," https://github.com/ultralytics/yolov5?tab=readme-ov-file