Convergence
A Journal for Young Researchers

ORIGINAL RESEARCH

Received: 11-9-25; Accepted: 12-30-25;
Managing Editor: Jonas Katona;
Action Editor: Jonas Katona

Real-Time 3D Trajectory Reconstruction of Table Tennis Balls Using a Multi-Camera YOLO11-OBB Pipeline with Kalman Filtering

Yishiuan Lin

Lexington High School, Lexington, Massachusetts, United States of America

Abstract

In table tennis, the spin of the ball is crucial to the game and yet hard to observe either in court or on TV. A vision system that can track the table tennis ball position is the first step toward a system that can track the ball with spin direction and strength information, which can be used to enhance the viewing experience. Accurate, real-time 3D tracking of high-speed ping pong balls is difficult due to their velocity, spin, and frequent occlusions. In this paper, we present a complete pipeline utilizing a four-camera setup. We employ chessboard-based calibration for precise 3D space reconstruction. A custom-trained YOLO11-Oriented Bounding Boxes (OBB) model detects the ball in each 2D camera feed. These 2D detections are then triangulated to reconstruct the ball's 3D position. A Kalman filter is applied to the resulting 3D trajectory data to smooth noise, predict state, and robustly handle tracking through occlusions. Our system demonstrates high-fidelity 3D tracking, showing significant improvement in trajectory smoothness and consistency due to the Kalman filter. The YOLO11-OBB model achieves 97.6% mAP on our test dataset. This combined approach provides a robust and scalable solution for advanced ping pong analytics.

Keywords: computer vision, You Only Look Once (YOLO), oriented bounding boxes (OBB), object detection, 3D trajectory reconstruction, multi-camera calibration, triangular calibration, Kalman filtering, sports analytics

1. Introduction

1.1. Motivation & Background

In recent years, advancements in computer vision and motion sensing have been applied in various sports. For example, video analysis for tennis performance and sensor-based intelligent Inertial Measurement Units for golf self-coaching (Renò et al., 2017; Chun, S. et al., 2025). There is growing interest in applying these technologies to table tennis, as it is well anticipated that motion data on the ball and/or player can improve the viewing experience, provide data-backed tactical analysis, improve training efficiency, and ultimately benefit the athlete's competition performance. Ball tracking is an

essential enabling technology for all these improvements.

Hardware and software improvements over the past decade have made it possible for student researchers to implement such systems. Convolutional Neural Networks (CNNs), a type of deep learning neural network architecture, are commonly used for object classification and detection. CNNs evolved from Artificial Neural Networks (ANNs) and received the name "Convolutional Neural Network" when Yann LeCun proposed a convolutional structure for handwritten zip code recognition (Le, 1990). However, the traditional activation function with the back-propagation training algorithm faced convergence and overfitting issues with deep CNN models. The breakthrough came when Krizhevsky et al. (2012) proposed an architecture, AlexNet, with ReLU activation functions, dropout techniques, and data augmentation, which enabled the existing back-propagation algorithm to successfully train an unprecedentedly deep and large CNN. The training of such a CNN model required a large training dataset, considering it contained 60 million parameters. Consequently, the training would have been computationally slow and long on CPUs. The breakthrough came when NVIDIA's GPU, which consists of a large number of computing units that can operate in parallel to efficiently compute complex matrices and other calculations essential for AI model training, was repurposed, accelerating the training speed by many orders of magnitude. TensorFlow's release by Google in 2015 enabled CUDA-accelerated tensor multiplication (Abadi et al., 2016). In addition to the success of hardware advancements, increasingly capable CNNs were developed. The You Only Look Once (YOLO) model was proposed by Redmon et al. (2016) for object detection. Unlike traditional detection methods of region proposal followed by classification, YOLO reframed object detection as a single regression problem, reaching real-time performance with high accuracy. Starting from these models, transfer learning is often applied so that models with pretrained weights from large-scale datasets can be trained with a much smaller dataset.

Furthermore, the availability of open-source vision libraries, such as OpenCV, allows researchers to quickly implement well-known camera calibration algorithms where multiple cameras can be used jointly for 3D imaging. Zhang's Camera Calibration Method acquires each camera's intrinsic parameters and the relationships between the cameras (Zhang, 2000). With these parameters, the same points from multiple camera views are merged into a single point in 3D space by triangulation; a 3D point can be mapped to any camera's image by using these parameters (Garey et al., 1978).

1.2. The Problem: Challenges in Ping Pong Tracking

The above technologies have been applied successfully for object tracking in other papers (Zhang et al., 2020; Montañés Laborda et al., 2025; Hu et al., 2010). However, tracking table tennis balls presents the following challenges: 1) High-Speed: the ball travels at extreme velocities; 2) Small size: it is a small object in a large vision field; 3) Spin: the ball's spin (topspin, backspin) affects its trajectory, and its high rotation can cause motion blur; 4) Occlusion: frequent and abrupt occlusion by the players' bodies, paddles, and the net.

1.3. Related Work

There is some research on table tennis ball tracking, yet robust tracking in complex scenarios remains limited. The study by K. C. P. Wong and Laurence S. Dooley (2010) used color-segmentation thresholding techniques along with spatial and temporal evaluations to detect and track a table tennis ball for umpiring in table tennis. While some heuristics-based ball detection methods, which typically look for features such as ball shape, color, and rapid movement characteristics, perform well in controlled imaging conditions, they lack the generalizability and robustness of CNN models like YOLO, which use

many layers of learned convolutional filters to detect objects with varied background conditions (Huang et al., 2015; Gomez-Gonzalez et al., 2016). In Zou Tianjian et al.'s (2024) study of table tennis ball tracking, they used the unique movement pattern of the ball and combined it with detection and discrimination methods to perform the tracking. This approach reportedly achieved certain success in terms of accuracy and robustness for different scenarios; however, there were many manually tuned empirical parameters, which made it difficult to translate to different use case scenarios. Furthermore, the system was computationally expensive, which resulted in slow processing speeds. Another study by Sebastian Gomez-Gonzalez et al. (2019) utilized multiple cameras to detect and track the trajectory of a table tennis ball in real time. It was intended to support a robot table tennis system. The proposed vision system uses semantic segmentation to return a pixel-wise classification of object versus background by running a small convolutional unit. The authors use a simple algorithm to post-process the classification image: the point with the highest probability of being the target is found first, and its neighbors with probability scores above a certain threshold are also identified as part of the target. The object detection is fast and computationally efficient. However, the tradeoff for the high speed is that with training done on small segments of the images, it is more prone to false detections and will need information from more cameras to eliminate the impact of false positive detections.

1.4. Our Contribution

To address these problems, this study proposes a table tennis ball tracking vision system, which leverages YOLO11-oriented bounding box (OBB) models and the open-source image processing library OpenCV for ball detection in images from multiple cameras (Culjak et al., 2025). The detected balls in each time frame from multi-camera views are fused to form a 3D point. To incorporate temporal information, Kalman filtering, a linear prediction algorithm first introduced by Rudolf E. Kálmán in 1960 that provides an estimation of a system's state and uses past data to update this state, is applied to the series of detected 3D points such that, in the absence of YOLO detections, the predicted value can be used for trajectory tracking (Welch, 1997). Compared to the models used in previous work, our choice of YOLO11-OBB model better suits the detection of table tennis balls in motion. Considering the shape of the captured ball is often blurred into an ellipse with varying orientations due to high speed and spin, OBB fits the contour more precisely, leading to better detection. Additionally, unlike the heuristic methods of temporal smoothing, the application of Kalman filtering effectively approximates the trajectory even when the ball is occluded, suitable for a wide range of scenarios and backgrounds.

1.5. Paper Structure

Section 2 details our data acquisition and camera calibration method. Section 3 describes the detection and tracking models. Section 4 presents our experimental results, followed by a discussion in Section 5 and our conclusion in Section 6.

2. Methodology: System and Data

This project was developed using Python 3.10.16 with the following key packages: ultralytics 8.3.185, opencv-python 4.11.0.86, and shapely 2.1.1.

2.1. Hardware and System Setup

The data collection for this research was conducted at a local table tennis club, with court use consent from the club

manager. Four cameras with a resolution of 1280x720 pixels, running at 60 frames per second (FPS), were set up at each corner of the table. Two participants were given table tennis paddles and balls and stood on opposite sides of the table. Participants were allowed to move freely if they didn't step out of the camera's range. A 6x8 chessboard was also placed on the table for each camera to capture before playing. Due to the table setup, the net on the table could not be removed; therefore, the chessboard view from cam1 and cam2 is partially occluded by the net. The setup is illustrated in Figure 1.

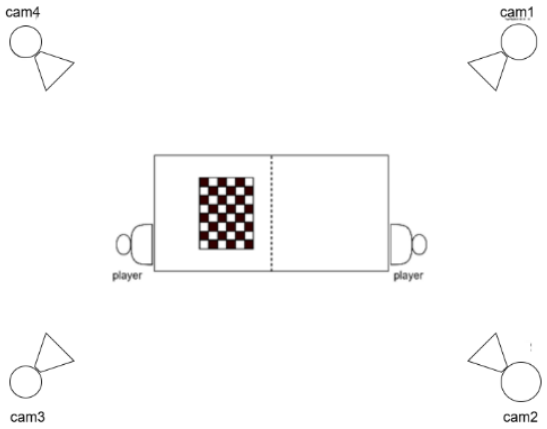


Figure 1: A diagram showing the capture setup with camera, chessboard, and player locations.

2.2. Data Acquisition & Labeling

Two sets of 5-minute videos from each camera were recorded. In total, 8 sets of videos were collected. The recorded videos were split into individual image files through frame extraction. Table 1 shows the details of the data sets generated. The Training and Inference sets were randomly selected across four cameras without overlapping, while the Tracking set was taken from a short segment with continuous play. Due to the amount of hand labeling involved for the Tracking set, only 200 frames per camera were annotated, as it was sufficient to demonstrate the effectiveness of the tracking pipeline. QuPath, an open-source software for image analysis, was used for generating the labels using oriented bounding boxes (Bankhead et al., 2017).

Table 1: Table with details about the data sets used for training, inference, and tracking.

Data Set Name	Number of images (from each camera)	Image characteristics	Usage
Train_set	100	random	Hand-labeled with oriented bounding box, used for YOLO model training. Randomly split into train-validate-test with ratio 7:2:1

Inference_set	25	random	Hand-labeled with oriented bounding box, used for YOLO model detection validation
Tracking_set	200	consecutive	Hand-labeled the center of the ball, used for tracking

2.3. Multi-Camera Calibration

A precise spatial relationship between all cameras is essential for 3D reconstruction. Functions in OpenCV: findChessboardCorners and calibrateCamera were used to find intrinsic (focal length, principal point) and extrinsic (rotation, translation) parameters for each camera relative to a world coordinate system (e.g., a corner of the table). Standard planar patterns, such as chessboard patterns with known dimensions, are often used for the calculation of these parameters.

However, findChessboardCorners is very sensitive to the quality and size of the chessboard. Since the chessboard used was drawn on cardboard, the function failed to detect the inner corners in the image. To resolve this, the inner corner coordinates of the chessboard image were hand-labeled using QuPath. Only three rows of inner corners were labeled to avoid using data occluded by the net. The coordinates were further adjusted to align more precisely using linear fitting. The adjusted coordinates were then used in the camera calibration algorithm to generate intrinsic, rotation, and translation matrices for each camera with respect to one camera as the reference camera. The images from each camera with hand-labeled inner corners are shown in Figure 2.

Figure 2: Hand-labeled inner corner of the chessboard from different cameras: camera 1 (leftmost), camera 2 (second-left), camera 3 (second-right), camera 4 (rightmost).

3. Methodology: 3D Tracking Pipeline

3.1. 2D Ball Detection: YOLO11-OBB

The Ultralytics YOLO11m-OBB model with 20.9 million pretrained weights was used to perform transfer learning on the Train_set data. The Ultralytics YOLO models by default apply image augmentation techniques (e.g., mosaic, HSV adjustment, etc.) during training. A high-level graphical representation of the architecture of YOLO11 is shown in Figure 3. We chose the oriented bounding box model over a standard, axis-aligned one because it provided a tighter fit to the ball, which was often blurred into an ellipse due to high spin and velocity. The images and their labels in Train_set for each camera were randomly separated into three sets—training, validation, and testing—for training and validation of the YOLO11m-OBB model to detect table tennis balls. As a result, four models were obtained at the end. This training method is

referred to as the one-fold method in the rest of the paper. Additionally, a five-fold-cross-validation method was used to obtain a single model by training with images from all four cameras. All images in Train_set were randomly split into five equal folds. Four of the folds were used to train the model, while the remaining fold was used to validate the training. This process iterates five times until each fold has been used to validate. Before training, the images needed to be resized to the model's input dimensions.

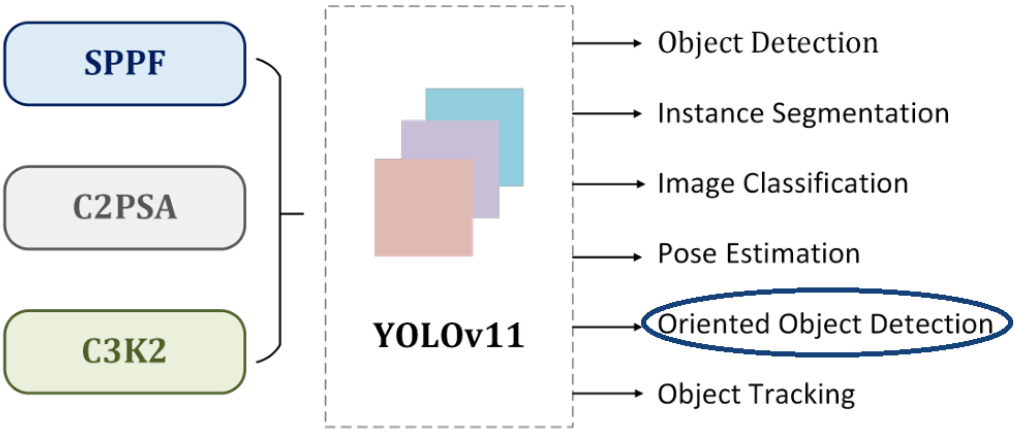


Figure 3: YOLO11 architecture adapted (Khanam and Hussain, 2024).

3.2. Model Validation

After training, images from Inference_set, extracted from the same videos, were used to run inference with the trained model. The detected bounding box was compared with the human-labeled bounding box to determine the model's performance. The models were evaluated using standard metrics such as Dice Score, Precision, Recall, F1-score, and Mean Average Precision (mAP), as shown in equations 1, 2, 3, 4, and 5.

$$Dice\ Score = \frac{2\times|A\cap B|}{(|A|+|B|)},\ [\![where\ |A| is\ prediction\ area,\ |B| is\ actual\ area,\ and\ |A\cap B| is\ the\ intersection\ area.]\!]$$

Equation 1: Dice score quantifies the overlap between the predicted bounding box and the actual bounding box.

$$Precision = \frac{TP}{TP+FP},\ [\![where\ TP\ is\ true\ positive\ detections,\ FP\ is\ false\ positive\ detections.]\!]$$

Equation 2: Precision reflects the model's ability to identify correct objects and is the proportion of correct positive detections out of all detections by the model.

$$Recall = \frac{TP}{TP+FN},[\![where\ TP\ is\ true\ positive\ detections,\ FP\ is\ false\ negative\ detections.]\!]$$

Equation 3: Recall reflects the model's ability to find all relevant objects, and it is the fraction of true positives to all objects.

$$F_1\ score = 2\cdot\frac{precision\cdot recall}{precision+recall}$$

Equation 4: F1-score evaluates the accuracy of the model's detection by combining precision and recall into a single value. It's only high when both precision and recall are high.

$$mAP = \frac{1}{Q} \bullet \sum_{q=1}^{Q} AP(q), \quad [\![where\ Q\ is\ the\ number\ of\ classes,\ and\ AP(q)\ is\ the\ average\ precision\ for\ a\ specific\ class.]\!]$$

Equation 5: Mean Average Precision is derived from precision and recall with a set confidence threshold for IoU, and it's calculated as the area under the precision-recall curve.

3.3. 3D Position Triangulation

The models trained from both one-fold and five-fold-cross-validation methods were used to detect the ball at a 0.25 confidence threshold in images originating from the four different cameras in Tracking_set. For each frame, if the ball was detected, the location of the ball (x,y) was calculated as the average of the bounding box coordinates along the x and y axes. Among the four images from different cameras captured at the same time, if more than two images contained detections, triangulation was run on each pair of images with detected balls to merge their detected 2D coordinates into a 3D position. The triangulation algorithm used the intrinsic, rotation, and translation matrices for each camera from the camera calibration algorithm. The results of these 3D positions were averaged as the final 3D position of the detected ball. A 3D trajectory of the ball was obtained by concatenating the 3D positions over the consecutive frames, overlaying them on camera four.

3.4. Temporal Smoothing: The Kalman Filter

The raw triangulated 3D points contained noise from detection inaccuracies and calibration errors. Furthermore, occlusions created gaps in the data. To address these issues, a Kalman filter was adopted to filter out the detection noise and predict the ball location for missing detections. The state vector [x,y,z,vx,vy,vz] was defined assuming a constant velocity model, since most of the time, the ball was traveling at a constant velocity. We used processNoiseCov(Q) = 0.03 and measurementNoiseCov(R) = 0.01, making Q larger so that the Kalman filter responds faster to ball movement changes. The state model first predicted the ball's next position, then a new 3D point from triangulation was used to update the internal state. When no 3D point was measured (i.e., the ball was occluded), the filter's state was propagated using only the prediction step, allowing the system to maintain a valid track and reacquire the ball post-occlusion.

4. Experiments and Results

4.1. 2D Detection Performance

We experimented with labeling the table tennis ball with a standard bounding box, but found the detection result was poor, and using the OBB model with oriented bounding boxes yielded greater than 90% mAP. We believe that when the ball is stretched and diagonal (with respect to the image axes), an axis-aligned box contains much more of the background than the oriented bounding box. Therefore, in such conditions, the axis-aligned bounding box model is trained with information mixed with ball and background and will require more training samples to reach the same F1 score as the OBB, which, on the other hand, is trained with just the ball information.

Using the five-fold cross-validation method, the model was trained for 50 epochs per fold, and performance across epochs is shown in Figure 4; all metrics reached an asymptote at 50 epochs. The performance metrics are displayed in Figure 5. The same metrics for one-fold trained models are illustrated in Figures 6-10.

The five-fold cross-validation training method demonstrated strong performance. It achieved an F1 score of 0.94 at a confidence threshold of 0.266 and a mean Average Precision (mAP) of 0.976 at a 0.5 Intersection over Union (IoU) threshold. The model correctly identified and classified 64 balls with 5 false negatives and 3 false positives. In comparison, the one-fold method resulted in varied model performance.

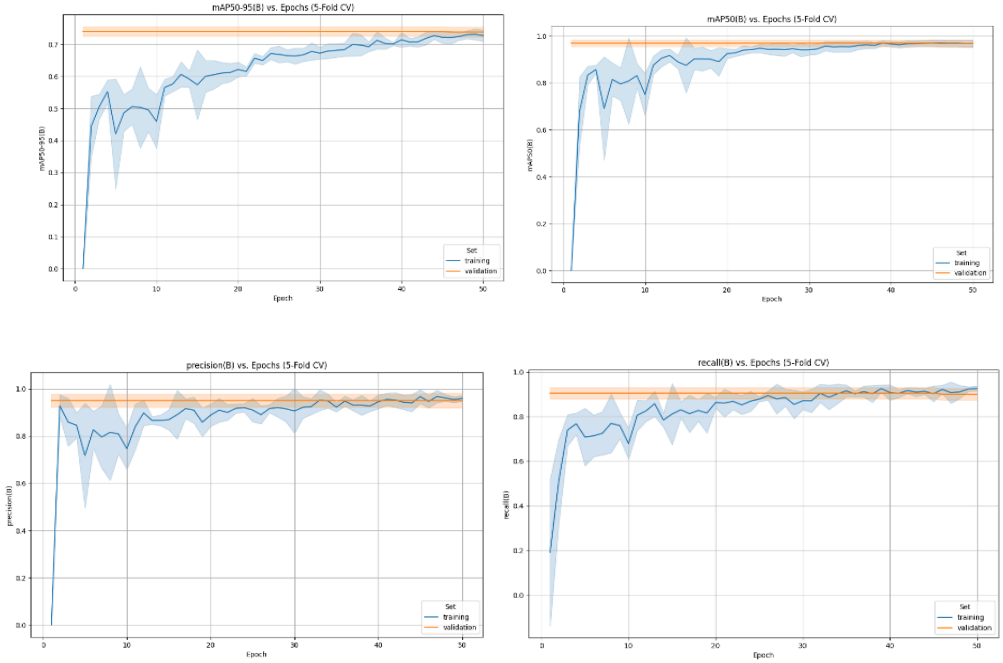


Figure 4: Training and validation performance curves across epochs for the model trained with five-fold-cross-validation: mean Average Precision averaged over IoU thresholds from 0.5 to 0.95 (top-left), mean Average Precision averaged at IoU thresholds of 0.5 (top-right), precision (bottom-left), and recall (bottom-right).

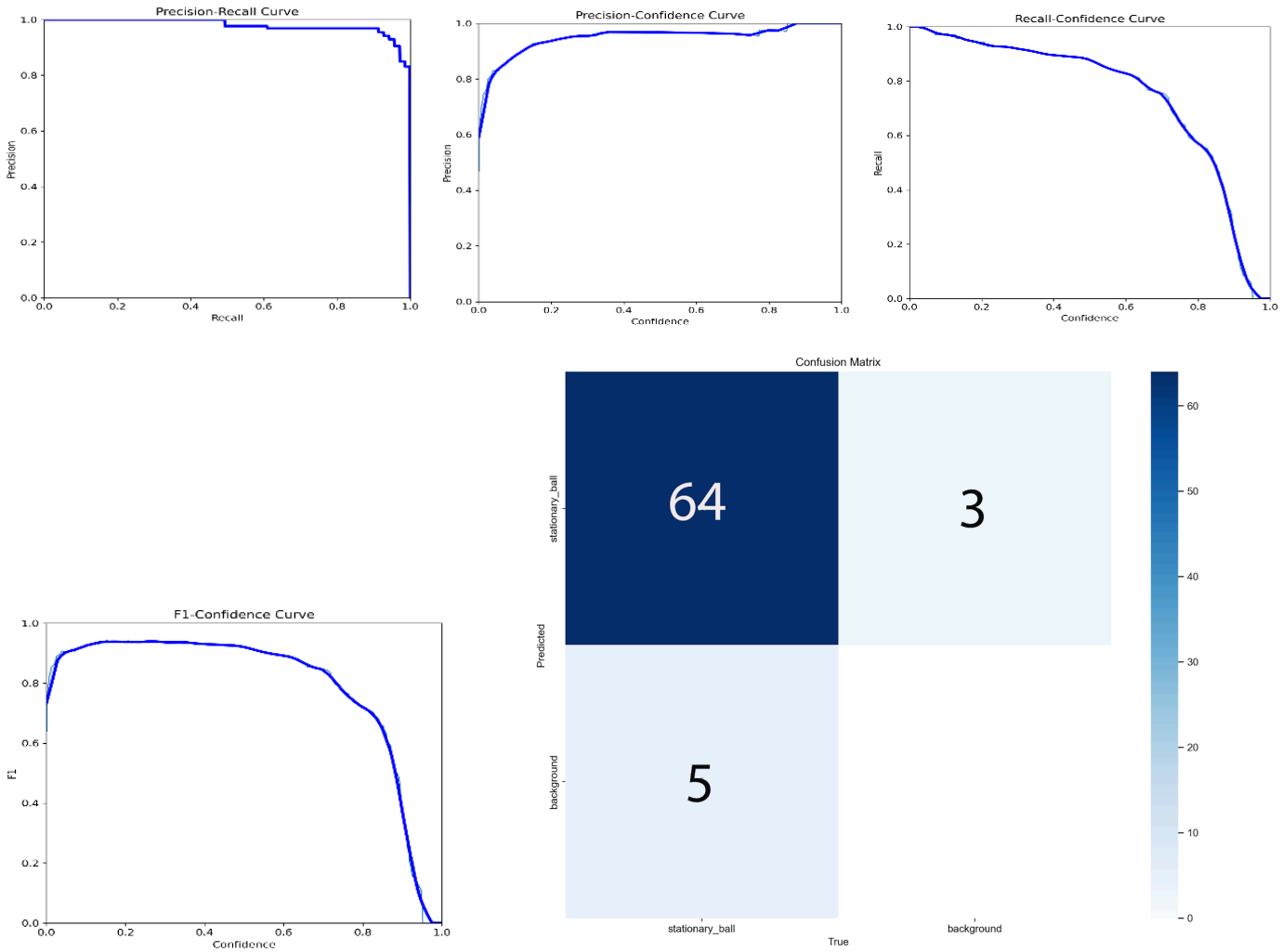


Figure 5: Precision-Recall Curve (top-left), Recall-Confidence (top-middle), Precision-Recall Curve (top-right), F1-Confidence Curve (bottom-left), and Confusion Matrix (bottom-right) for a model trained with five-fold cross-validation.

Camera 1 Camera 2

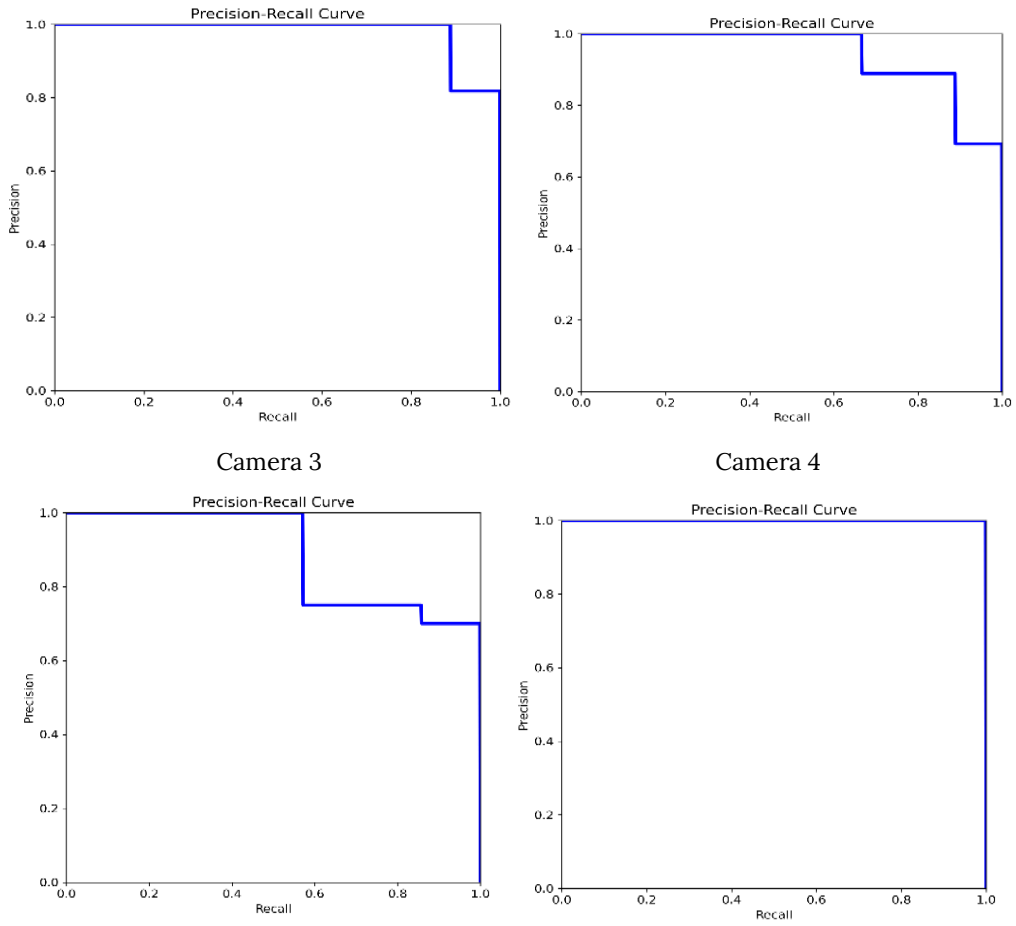


Figure 6: The Precision-Recall Curve for each camera model trained with one fold.

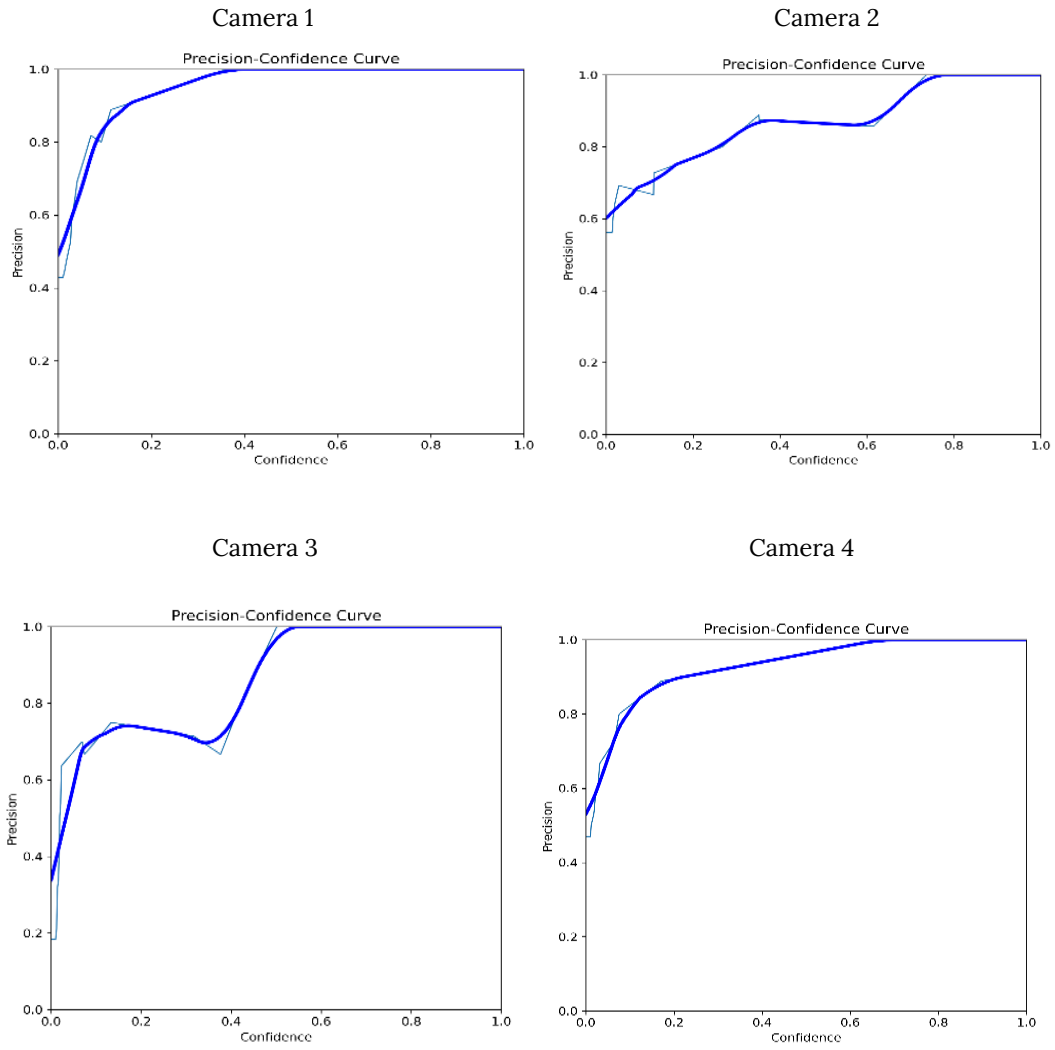


Figure 7: The Precision-Confidence Curve for each camera model trained with one fold.

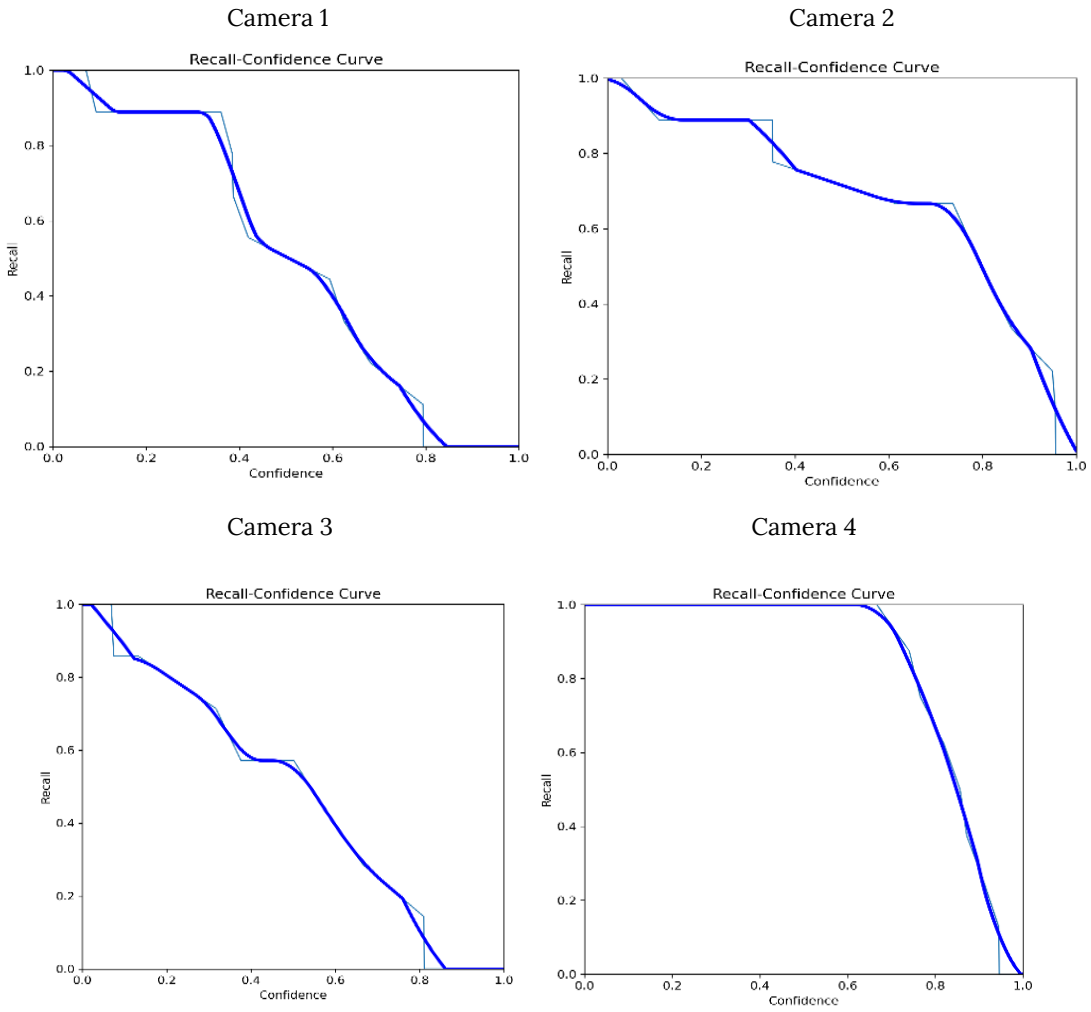


Figure 8: The Recall-Confidence Curve for each camera model trained with one fold.

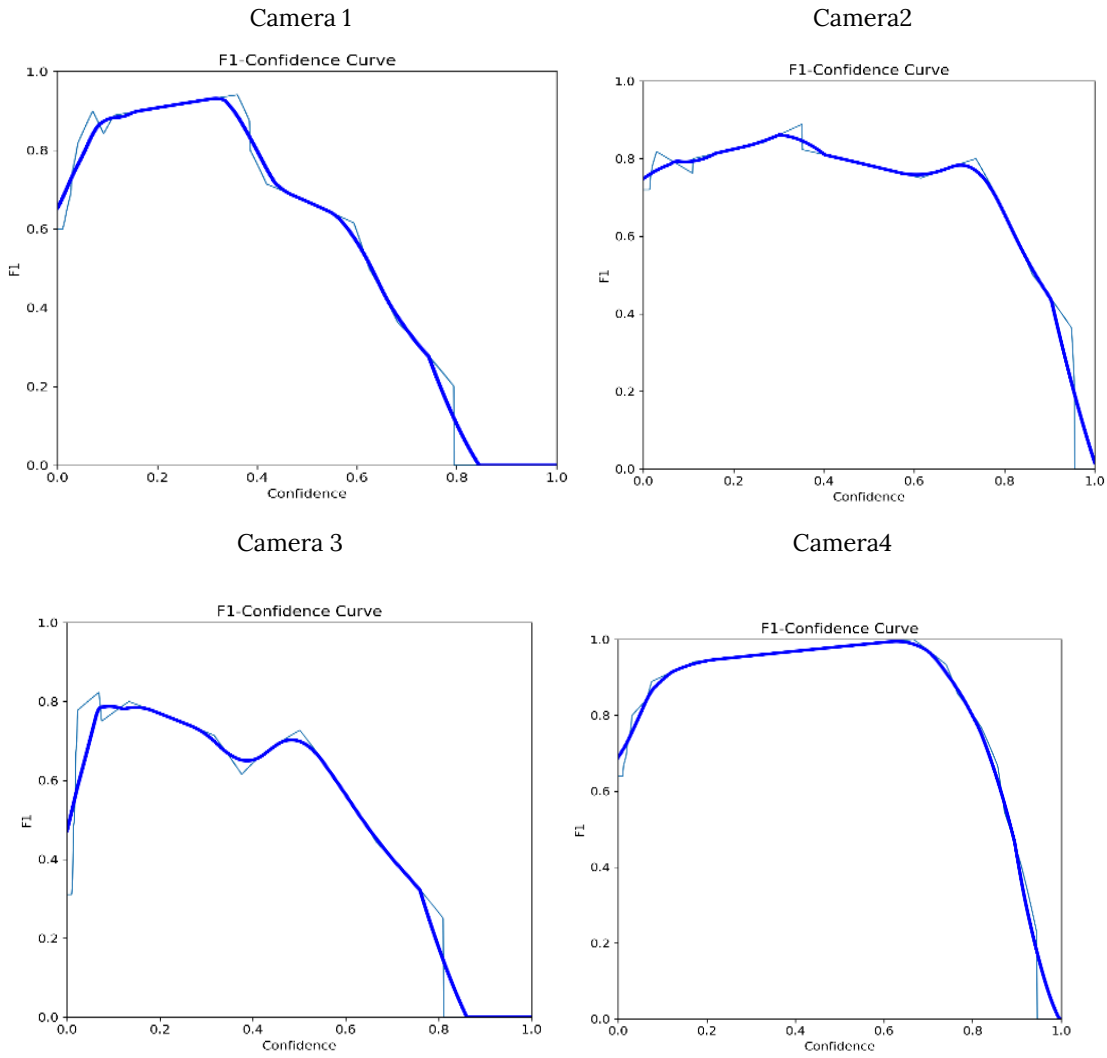
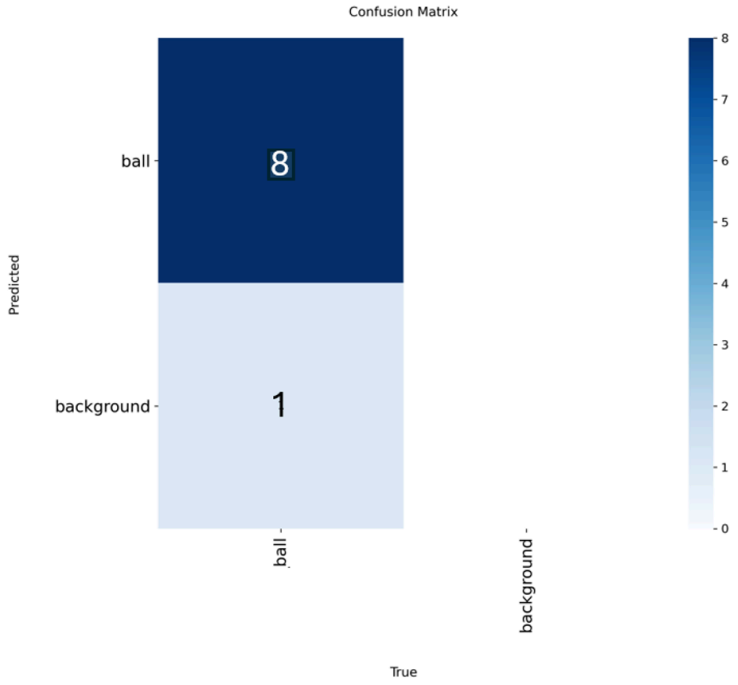
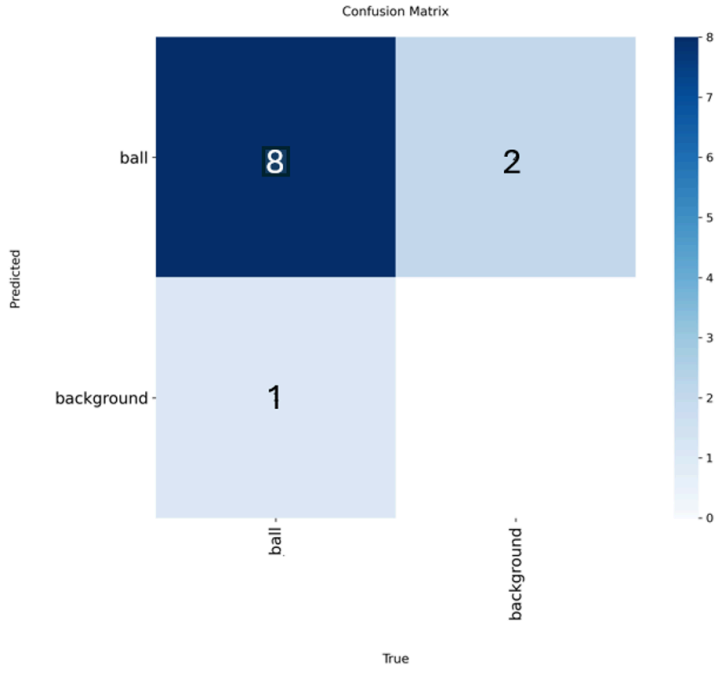


Figure 9: The F1-Confidence Curve for each camera model trained with one fold.

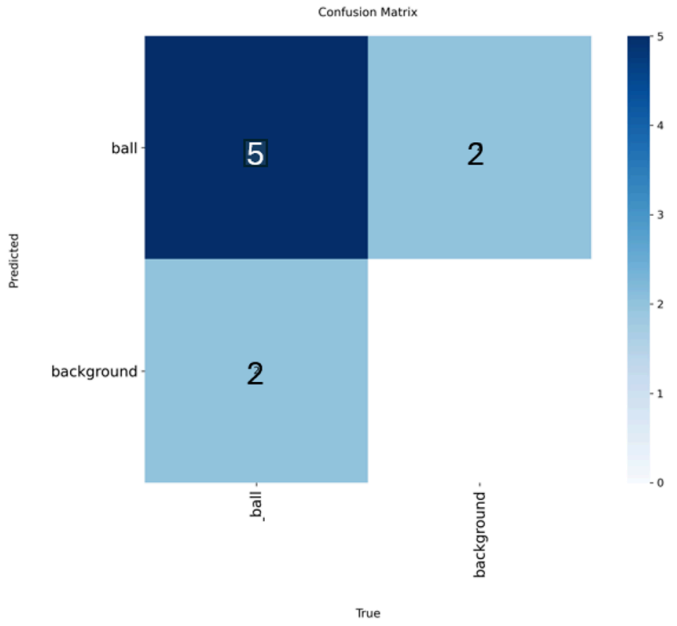
Camera 1



Camera 2



Camera 3



Camera 4

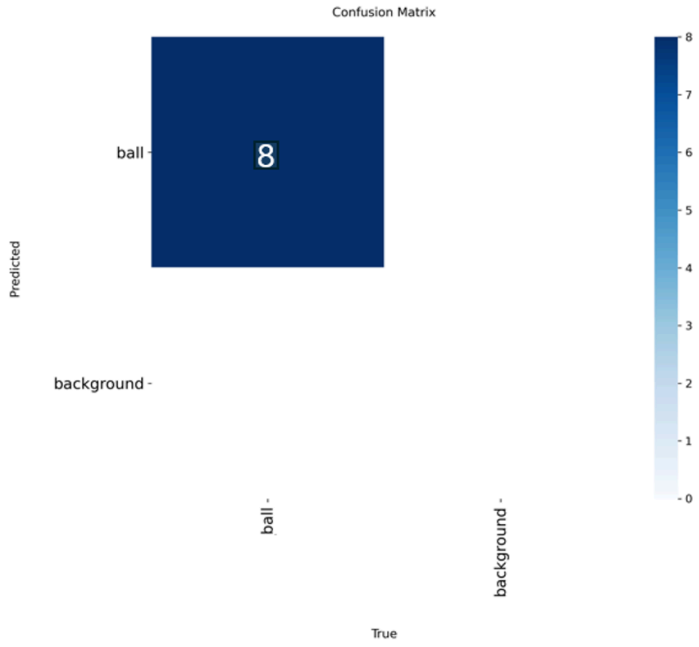


Figure 10: The Confusion Matrix for each camera model trained with one fold.

Example images of successful detection of clear shot, ball with motion blur, and partial occlusion are shown in Figure 11.

Figure 11: The images demonstrate successful detections from images of clear view (leftmost), motion blur (middle), and partial occlusion (rightmost).

Example images of failed detection due to occlusion and background interference are shown in Figure 12.

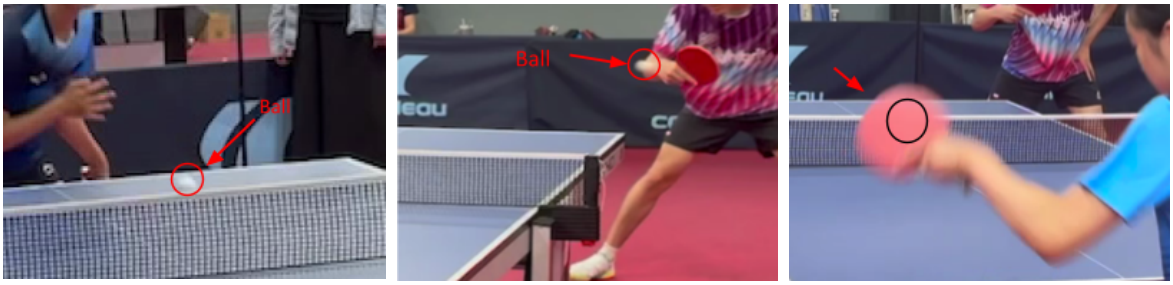


Figure 12: The images demonstrate failed detection from background interference (left 2) and ball occlusion (rightmost). The red and black marks indicate where the ball is in each frame.

For Tracking_set, the confidence_threshold was set to 0.25, and the number of frames with ball detection for each camera is listed in Table 2. Since the 3-D reconstruction needs ball detection from two cameras, the number of frames with ball detection in more than two cameras, and those in both camera 4 and camera 1 are also listed. With a 0.25 confidence threshold, there was no false detection across all cameras. Lowering the confidence threshold to 0.1 will result in 5% more frames with ball detection; however, there will be a 0.5% chance of false detection. With the Kalman filter in the pipeline to effectively fill in the missed points in the trajectory, a 0.25 confidence threshold was used.

Table 2: Number of frames with Ball detection: each camera, both camera 1 and camera 4, and more than two cameras.

	Cam 4	Cam 1	Cam 2	Cam 3	Cam 4 & cam 1	More than two cameras
Number of frames with ball detection	168	158	133	138	135	175

Total number of frames with balls not occluded	190	200	161	164	N/A	N/A
Detection Rate	88%	79%	83%	84%		

4.2. Model Inference Result

To evaluate the model's performance, inference was run on Inference_set to test the model trained with the five-fold-cross-validation method. For 100 images in Inference_set with a confidence level of 0.25, there were 74 true positive detections, 3 false positive detections, and 7 false negative detections. F1 score and dice score were listed in Table 3. The performance was compared to the YOLOv8 performance as well as the GNN model performance reported by Zou et al. (2024) in Table 4, where the side-facing view result was chosen because this angle was most comparable to the data collected in this research.

Table 3: F1 and dice score for five-fold cross-validation for inference.

	F1 Score	**Dice Score**
Five-Fold-Cross-Validation	0.9367	0.7651

Table 4: Performance comparison with YOLOv8 and GNN model

	Precision	**Recall**	**F1 Score**	**Training data size**
GNN(side facing)	0.862	0.954	0.906	237150
YOLOv8 (confidence @0.5)	0.924	0.647	0.761	237150
YOLO11m-OBB(confidence @0.25)	0.961	0.914	0.9367	400

4.3. Camera Calibration Result

The reconstructed 3D coordinates of the chessboard inner corners using the camera calibration result were converted to 2D coordinates and overlaid on camera four in Figure 13 to validate the intrinsic, rotational, and translation matrix generated by the camera calibration algorithm. The reconstructed coordinates align very well with the chessboard inner corners, showing the accuracy of the calibration results.

Figure 13: Reconstructed chessboard inner corner through camera calibration and triangulation algorithms on camera 4 image.

4.4. 3D Tracking Performance

Examples of detected ball locations and Kalman filter predicted ball locations are shown in Figure 14. The white circles are the trajectory points from the Kalman Filter prediction. Without applying the Kalman Filter, the trajectory would be jittery with gaps.

The tracking performance is defined by the detection accuracy and tracking error. For each frame, the tracking error is calculated by finding the distance between the actual ball location, obtained from hand labeling, and the detected or predicted ball location. The detection accuracy comparison between the models trained by one-fold and five-fold-cross-validation methods is shown in Table 5. The maximum, minimum, mean, and RMS of the detection and prediction errors are calculated and listed in Table 6. The histograms of detection error and prediction error distribution are shown in Figure 15.

Table 5: Ball tracking performance comparison between models trained by one-fold and five-fold cross-validation methods out of 200 consecutive frames in Tracking_set.

	Detected Frames	Total Frames	Detection Accuracy
One-fold	116	200	58%
Five-Fold-Cross-Validation	135	200	67.5%

Table 6: Statistics of detection error for the model-detected and Kalman filter-predicted coordinates against actual ball coordinates in 200 consecutive 1280x720 frames in Tracking_set, using a five-fold cross-validation method trained model.

	Maximum error (pixel)	Minimum error (pixel)	Mean error (pixel)	RMS error (pixel)
Detected (135 frames)	27.973	0.707	5.872	5.163
Kalman filter Predicted (65 frames)	78.118	2.550	16.439	13.444
Combined	78.118	0.707	8.759	9.529

Figure 14: Tracked trajectory with ten balls, Kalman Filter-predicted balls are in white circles, the detected balls are in green circles.

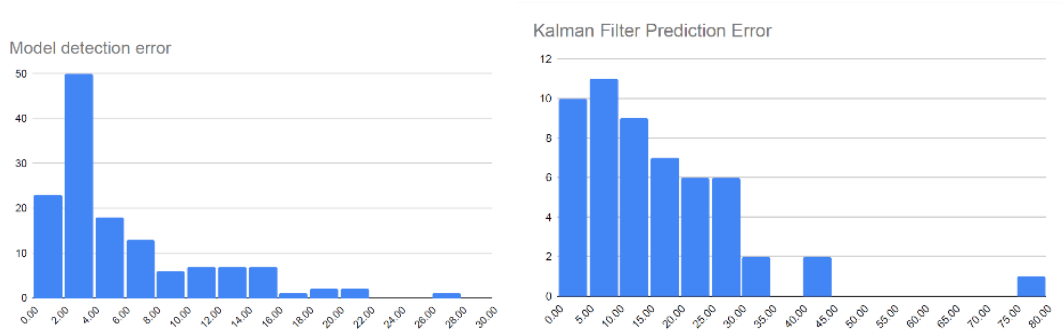


Figure 15: Histogram displaying the error distribution of the detection error for the model-detected and Kalman filter-predicted coordinates in 200 consecutive 1280x720 frames in Tracking_set, using a five-fold cross-validation trained model.

5. Discussion

5.1. Interpretation of Results

The combination of a high-precision OBB detector (to get accurate 2D centers), a well-calibrated camera, and a robust temporal filter (to handle 3D noise) was key to tracking the fast-moving table tennis ball trajectory.

When we ran the inference with Inference_set, the dice score/IoU was 76.51%, which indicated that the area of the detected ball mostly overlapped with the ball's location. The inference performance was overall better compared with both YOLOv8 and the GNN model reported by Zou et al. (2024) with a much smaller training set, as shown in Table 4, which demonstrated the superior effectiveness of the OBB detector.

Using the one-fold method to train the model from four cameras resulted in different model performance: models for cameras 1 and 4 have better F1 score, precision, recall, mAP, and detection results, as shown in Figures 6-10. This could come from the different angles of each camera. Cameras 1 and 4 both have a better view with minimum occlusion, while cameras 2 and 3 both have more than half of the table blocked. It is also possible that due to the small training set, the training data for cameras 2 and 3 are more difficult compared to that of cameras 1 and 4.

The five-fold-cross-validation method trained all four cameras' data into a single model, and all images were used in four iterations of the training, resulting in a better-performing model compared to the one-fold training method with a single iteration on fewer training images.

For camera calibration, we found that the reconstruction result was affected by the choice of reference camera. Since the images of the chessboard in cameras 1 and 2 are further away and behind the net, using cameras 1 and 2 as the reference led to poor results. So instead, the calibration used camera 4 as the reference frame. For the final tracking implementation, the five-fold-cross-validation-trained YOLO11m-OBB model was chosen for its superior performance on the testing. We found that the loss of ball detection mainly comes from the occlusion, which was listed in Table 2 for each camera. The

other reason for the failed detection was background interference displayed in Figure 12. When the ball overlapped with similar colored patterns, arm, net, or table edges, the model often failed to detect the ball. We calculated the detection rate of each camera without occlusion as shown in Table 2, and each camera achieved approximately 80% detection rate, demonstrating the effectiveness of the YOLO11m-OBB model, considering only a small set with 400 images was used for training. By increasing the training size, the model will be more robust. Due to the non-ideality of the camera calibration, the only reliable 3D position detection came from the triangulation result of cameras 1 and 4. Including all camera views led to degraded detection performance. Thus, for the final ball tracking, we only used the 3D position triangulated from cameras 1 and 4 with 67.5% detection accuracy and filled in the gaps with the Kalman filter predictions. However, assuming perfect camera calibration, the number of frames with 3D position detected will increase to 175, as shown in Table 2. Since, ideally, the 3D position can be triangulated from any two cameras with ball detection, the detection accuracy will increase to 87.5%. The final trajectory error in Table 6, with a mean of 8.8 pixels and RMS at 9.5 pixels in a 1280x720 image, shows the effectiveness of Kalman filtering.

5.2. Limitations

The system's accuracy is highly dependent on the quality of the initial calibration. Tracking can still be lost during prolonged occlusions. The camera's frame rate (FPS) is another limitation; performance could be improved with higher FPS, though with the tradeoff of increased computing power. In addition, the size of the training set is relatively small due to the time constraint. The training data size can be increased to achieve better model performance.

6. Conclusion and Future Work

6.1. Conclusion

We have demonstrated a highly effective pipeline for 3D tracking of ping pong balls, combining the YOLO11m-OBB detector with a Kalman filter for temporal consistency. Our results show a reliable tracking system that consistently tracks the movement of the table tennis ball in the field of view.

6.2. Future Work

Camera Calibration: The chessboard for camera calibration was too small and only covered a small portion of the camera's view field, with the board being flat on the table. It could be optimized by using a larger chessboard pattern drawn on both sides of the board and placed vertically above the table.

AI modeling training on a stream of images: Investigate the possibilities to incorporate temporal information in the AI model training for more robust ball detection.

Spin Analysis: Investigate a direct correlation between the OBB's orientation angle (θ) and the ball's physical spin, potentially enabling real-time spin-rate detection.

Player Tracking: Integrate this ball-tracking pipeline with a 2D/3D pose estimation model to track player movements for full-game analysis.

Speed optimization: Run on NVIDIA GPU to fully leverage the speed advantage of YOLO11 detector, further optimization for deployment on edge devices for on-site, low-latency analytics.

7. References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., & Zheng, X. (2016). *TensorFlow: A system for large-scale machine learning* (arXiv:1605.08695). arXiv. https://doi.org/10.48550/arXiv.1605.08695

Bankhead, P., Loughrey, M. B., Fernández, J. A., Dombrowski, Y., McArt, D. G., Dunne, P. D., McQuaid, S., Gray, R. T., Murray, L. J., Coleman, H. G., James, J. A., Salto-Tellez, M., & Hamilton, P. W. (2017). QuPath: Open source software for digital pathology image analysis. *Scientific Reports*, 7, Article 16878. https://doi.org/10.1038/s41598-017-17204-5

Chun, S., Kang, D., Choi, H. R., & Lee, S. (2014). A sensor-aided self-coaching model for uncocking improvement in golf swing. *Multimedia Tools and Applications*, 72, 253–279. https://doi.org/10.1007/s11042-013-1359-2

Culjak, I., Abram, D., Pribanić, T., Dzapo, H., & Cifrek, M. (2012). A brief introduction to OpenCV. In *35th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* (pp. 1725–1730). IEEE. https://mipro-proceedings.com/sites/mipro-proceedings.com/files/upload/sp/sp_008.pdf

Garey, M. R., Johnson, D. S., Preparata, F. P., & Tarjan, R. E. (1978). Triangulating a simple polygon. *Information Processing Letters*, 7(4), 175–179. https://doi.org/10.1016/0020-0190(78)90062-5

Gomez-Gonzalez, S., Nemmour, Y., Schölkopf, B., & Peters, J. (2019). Reliable real-time ball tracking for robot table tennis. *Robotics*, 8(4), Article 90. https://doi.org/10.3390/robotics8040090

Gomez-Gonzalez, S., Neumann, G., Schölkopf, B., & Peters, J. (2016). Using probabilistic movement primitives for striking movements. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)* (pp. 502–508). IEEE. https://doi.org/10.1109/HUMANOIDS.2016.7803322

Hu, M.-C., Chang, M.-H., Wu, J.-L., & Chi, L. (2011). Robust camera calibration and player tracking in broadcast basketball video. *IEEE Transactions on Multimedia*, 13(2), 266–279. https://doi.org/10.1109/TMM.2010.2100373

Huang, Y., Schölkopf, B., & Peters, J. (2015). Learning optimal striking points for a ping-pong playing robot. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 4587–4592). IEEE. https://doi.org/10.1109/IROS.2015.7354030

Khanam, R., & Hussain, M. (2024). *YOLOv11: An overview of the key architectural enhancements* (arXiv:2410.17725). arXiv. https://doi.org/10.48550/arXiv.2410.17725

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25* (NIPS 2012). https://proceedings.neurips.cc/paper_files/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html

Le Cun, Y., Matan, O., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., Jacket, L. D., & Baird, H. S. (1990). Handwritten zip code recognition with multilayer networks. In 10th International Conference on Pattern Recognition Proceedings (Vol. 2, pp. 35–40). IEEE. https://doi.org/10.1109/ICPR.1990.119325

Montañés Laborda, M. A., Torres Moreno, E. F., Martínez del Rincón, J., & Suescun García, R. (2012). Real-time GPU color-based segmentation of football players. *Journal of Real-Time Image Processing*, 7, 267–279. https://doi.org/10.1007/s11554-011-0194-9

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You only look once: Unified, real-time object detection* (arXiv:1506.02640). arXiv. https://doi.org/10.48550/arXiv.1506.02640

Renò, V., Mosca, N., Nitti, M., D'Orazio, T., Guaragnella, C., Campagnoli, D., Prati, A., & Stella, E. (2017). A technology platform for automatic high-level tennis game analysis. *Computer Vision and Image Understanding*, 159, 164–175. https://doi.org/10.1016/j.cviu.2017.01.002

Welch, G., & Bishop, G. (1995). *An introduction to the Kalman filter* (Technical Report TR 95-041). University of North Carolina at Chapel Hill. https://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf

Wong, K. C. P., & Dooley, L. S. (2010). High-motion table tennis ball tracking for umpiring applications. In *IEEE 10th International Conference on Signal Processing Proceedings* (pp. 2460–2463). IEEE. https://doi.org/10.1109/ICOSP.2010.5657001

Zhang, Y., Chen, Z., & Wei, B. (2020). A sport athlete object tracking based on deep sort and Yolo V4 in case of camera movement. In 2020 *IEEE 6th International Conference on Computer and Communications* (ICCC) (pp. 1312–1316). IEEE. https://doi.org/10.1109/ICCC51575.2020.9345010

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11), 1330–1334. https://doi.org/10.1109/34.888718

Zou, T., Wei, J., Yu, B., Qiu, X., Zhang, H., Du, X., & Liu, J. (2024). Fast moving table tennis ball tracking algorithm based on graph neural network. *Scientific Reports*, 14(1), Article 29320. https://doi.org/10.1038/s41598-024-80056-3

Code and Data Availability

All datasets, models, scripts, and videos used or referred to in this study are hosted in the following public repository:

http://drive.google.com/drive/folders/1iYzuSSWz2hfVgEZShBB31eBI0nlLX8eZ?usp=sharing.

Acknowledgements

The support and guidance of this research from my mentor, Marcin Kedziera, is greatly appreciated. His directions on how to form the research framework and how to use online tools to make the research process more efficient are of great help to me. I am also deeply grateful for my parents' support in the logistics of my research.

Author Biography

Yi-shiuan Lin is an 18-year-old senior at Lexington High School. Being a passionate table tennis player since age 7, she has been on the USA Junior and Adult National Table Tennis Team since 2021. When she is not on the court, she spends her time invested in STEM, specifically in the field of electrical engineering. She is interested in exploring how technology can positively impact the community.

Mentor Contribution Statement

Marcin Kedziera, a PhD candidate in artificial intelligence at the University of Edinburgh, was Yishiuan Lin's sole mentor throughout the project. He worked closely with Yishiuan throughout the duration of the project and met with her on a weekly basis. Initially, he taught several classical computer vision concepts and deep learning techniques to Yishiuan, which Yishiuan then adapted to her dataset. Marcin also supervised the writing of the final paper via three separate rounds of written feedback. All ideas, data acquisition, analyses, and writing were performed by Yishiuan.