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

[Author name redacted]
[School information redacted]

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: Electrical Engineering; signal and image processing; computer vision; machine learning; AI model; fast object detection and tracking, table tennis, trajectory tracking, sport 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 Unit for golf self-coaching (Renò et al., 2017; MTA, 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 the above improvements.

Hardware and software improvements over the past decade have made it possible for student researchers to implement such systems. Convolutional Neural Networks (CNN), a type of deep learning neural network architecture, are commonly used for object classification and detection. CNNs evolved from Artificial Neural Networks (ANN) 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 leap 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 amount 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 (Carey et al. ??).

• *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; JRTIP 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's (2024) study of table tennis ball tracking, he 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 pixelwise 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 models and the open-source image processing library OpenCV for ball detection in images from multiple cameras (IEEE CP 2025). The detected balls, in each time frame from multicamera 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, which provides an estimation of a system’s state and uses past data to update this state, is applied on 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).

• 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 as long as they didn't step out of the cameras' ranges. A 6x8 chessboard was also placed on the table for each camera to capture it 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.

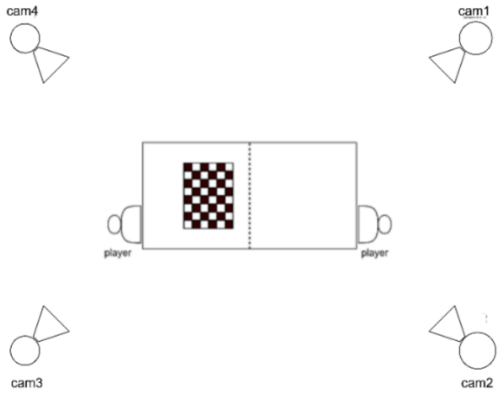

Figure 1. A diagram showing the capture setup with camera, chessboard and player locations.

• 2.2. *Data Acquisition & Labeling*

Recorded two sets of 5-minute videos from each camera. 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. 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, interference 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.

Camera 1 Camera 2 Camera 3 Camera 4

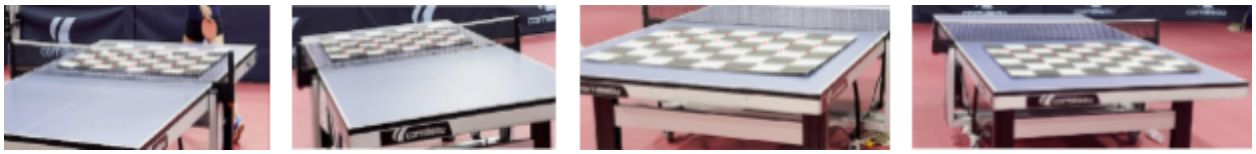

Figure 2. Hand-labeled inner-corner of the chessboard from different cameras.

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 validating 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. Prior to training, the images needed to be resized to the model's input dimensions.

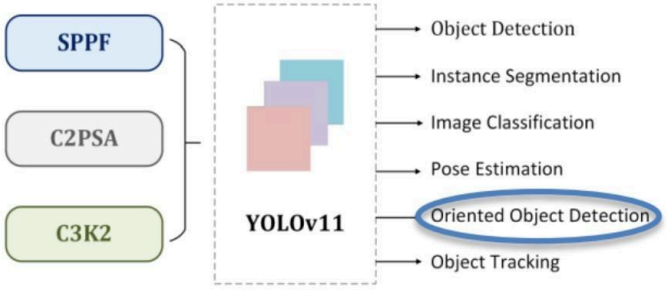

Figure 3. YOLO11 architecture adapted (Khanam and Hussain 2024).

• 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.

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

Equation 2. Precision reflects model's ability to identify correct objects, is the proportion of correct positive detection 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 model's ability to find all relevant objects, and it's the fraction of true positives to all objects.

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

Equation 4: F1-score evacuates 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.

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

Equation 5: Mean Average Precision is derived from precision and recall with a set counfidence 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 @ 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 was defined assuming a constant velocity model since, most of the time, the ball was traveling at a constant velocity. 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.
With the five-fold-cross-validation method, the model was trained for 50 epochs in each fold, and the performances across epochs are displayed in Figure 4; all metrics reached asymptote with 50 epochs. The performance metrics are displayed in Figure 5. The same metrics for one-fold trained models are illustrated in Figure 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 negative and 3 false positive detections. In comparison, the one-fold method resulted in varied models' performance.

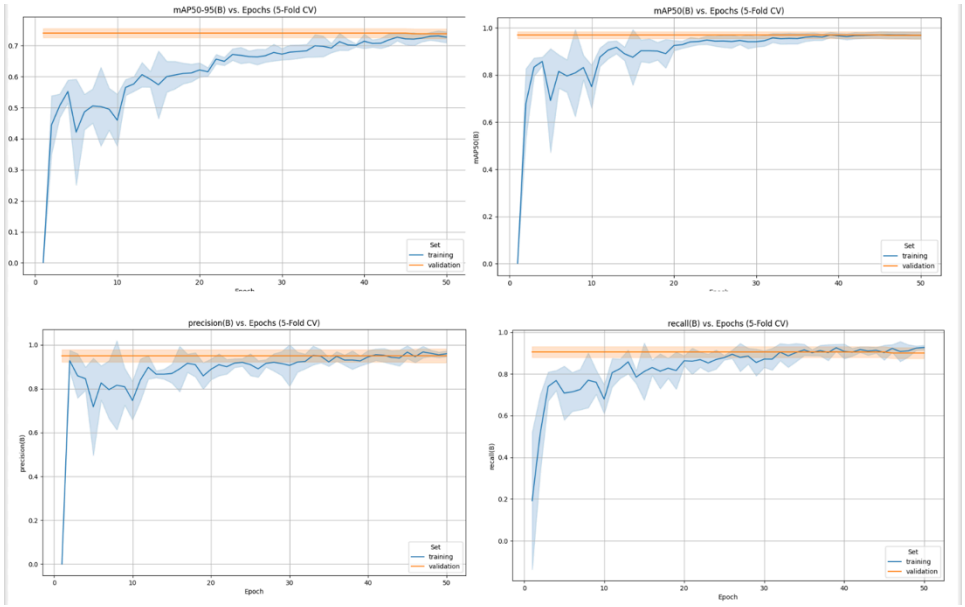

Figure 4. Training and validation performance curves across epochs for model trained with five-fold-cross-validation: mean Average Precision averaged over IoU thresholds from 0.5

to 0.95(top-left), mean Average Precision averaged at IoU thresholds of 0.5(top-right), precision(bottom-left), and recall(bottom-right).

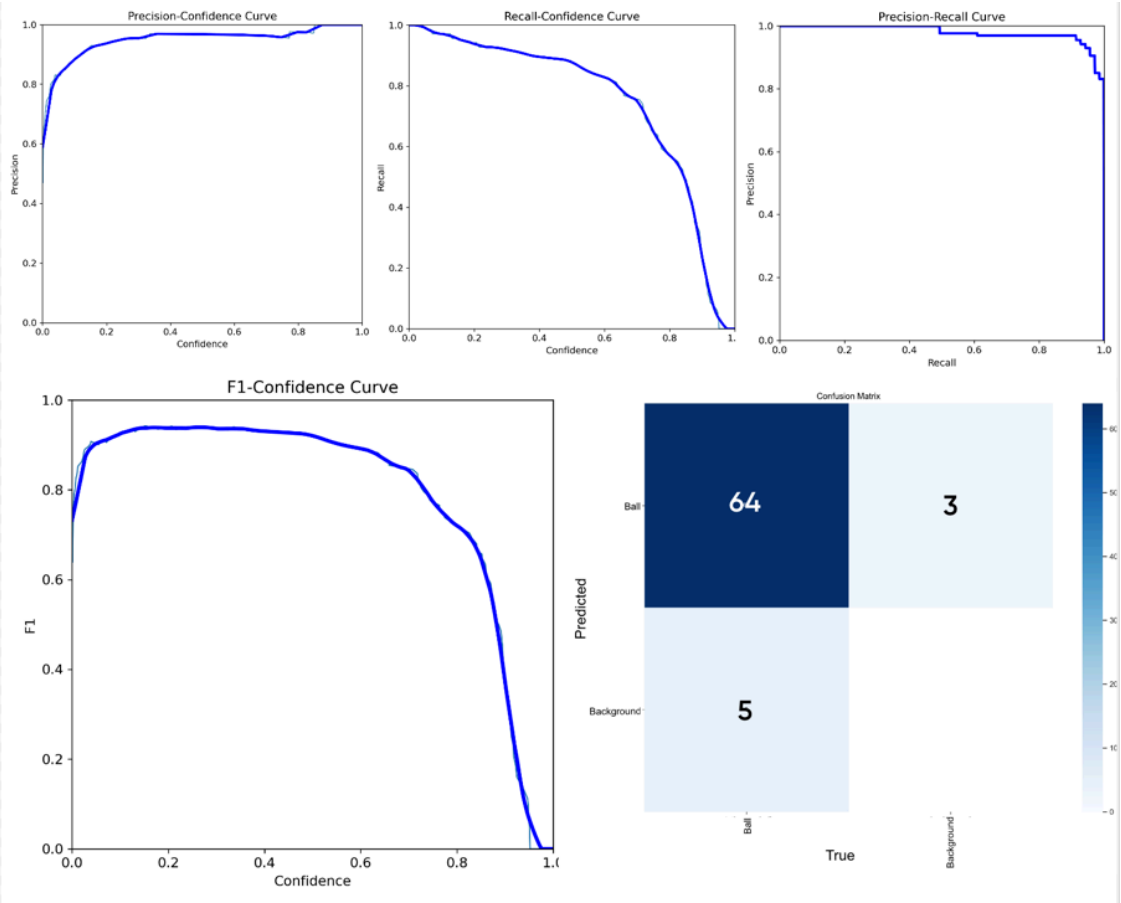

Figure 5. Precision-Recall Curve(top-left), Recall-Confidence(top-middle), Precision-Recall Curve(top-right), F1-Confidence Curve(bottom-left), and Confusion Matrix(bottom-right) for model trained with five-fold-cross-validation method.

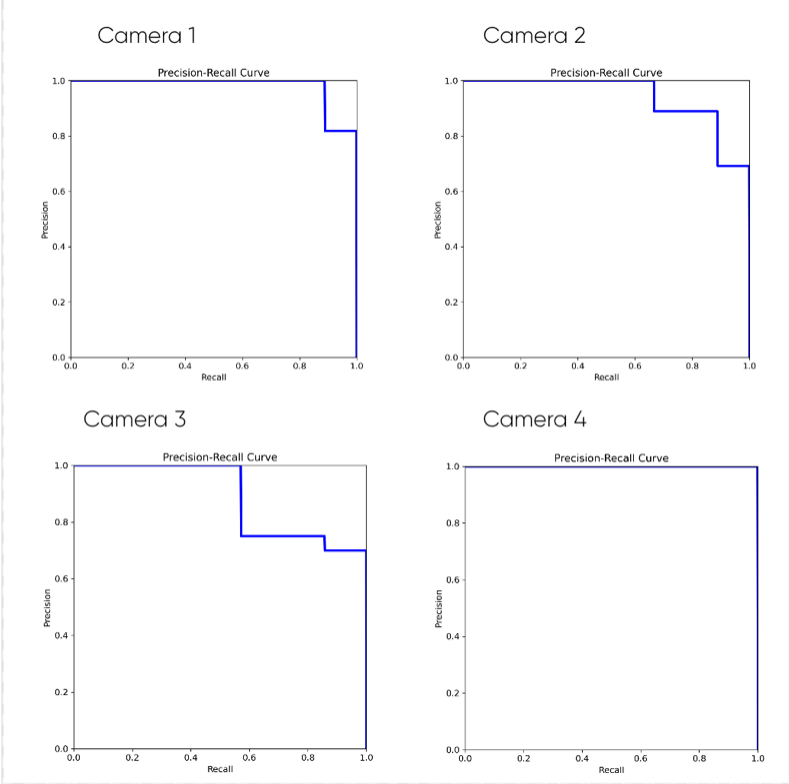

Figure 6. The Precision-Recall Curve for each camera model trained with one-fold method

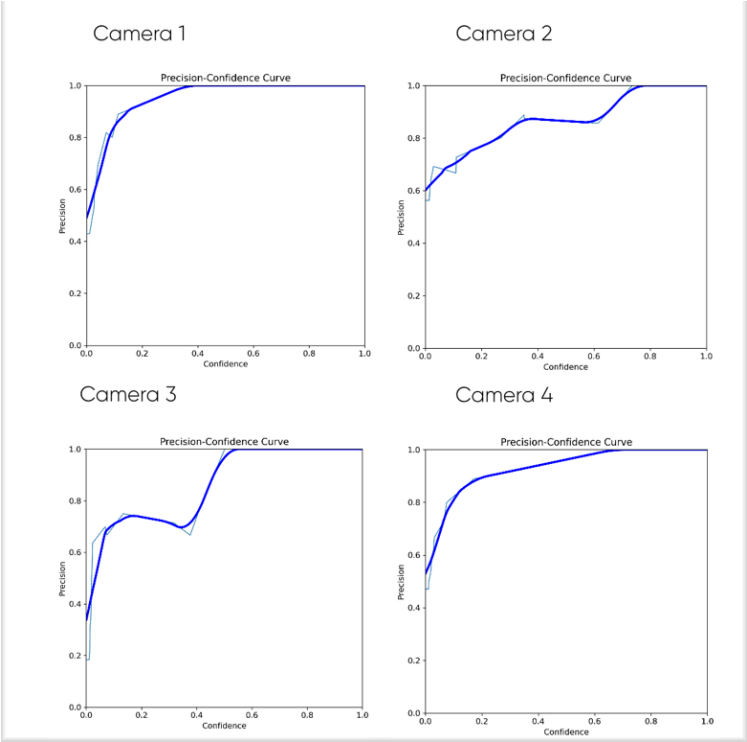

Figure 7. The Precision-Confidence Curve for each camera model trained with one-fold method

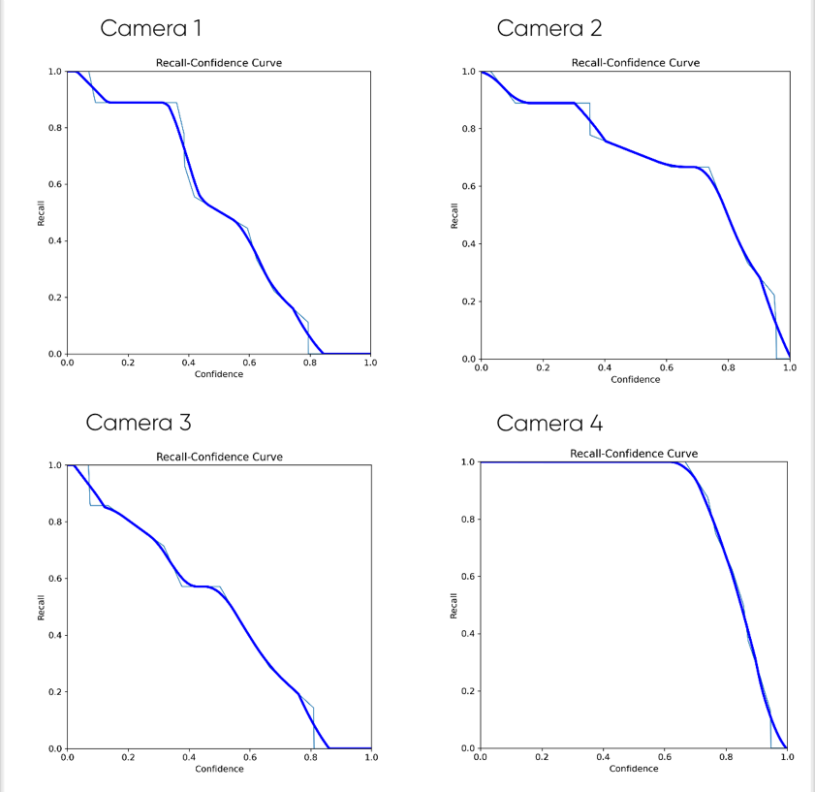

Figure 8. The Recall-Confidence Curve for each camera model trained with one-fold method

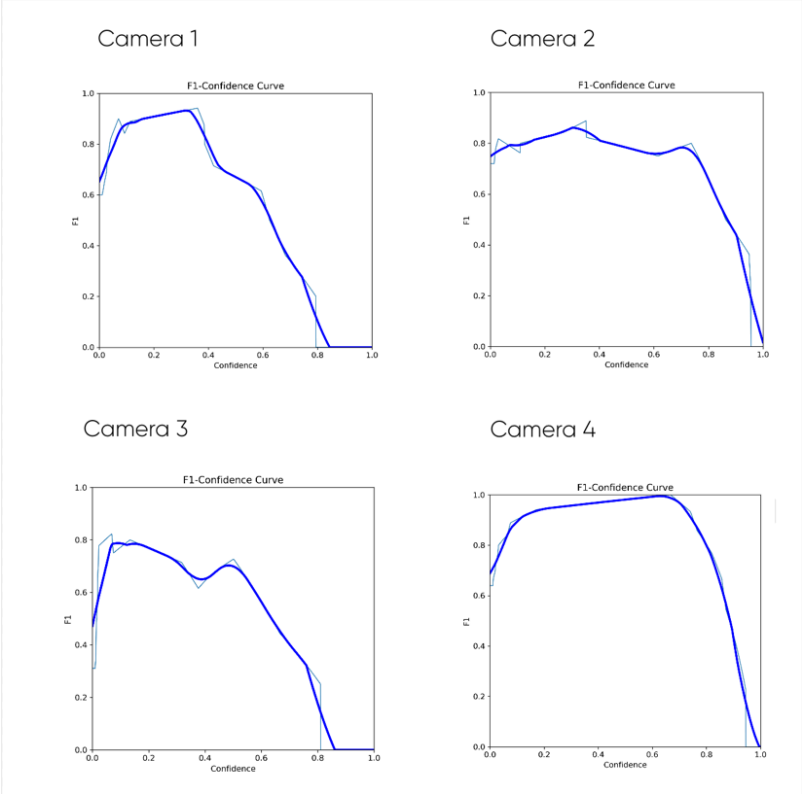

Figure 9. The F1-Confidence Curve for each camera model trained with one-fold method

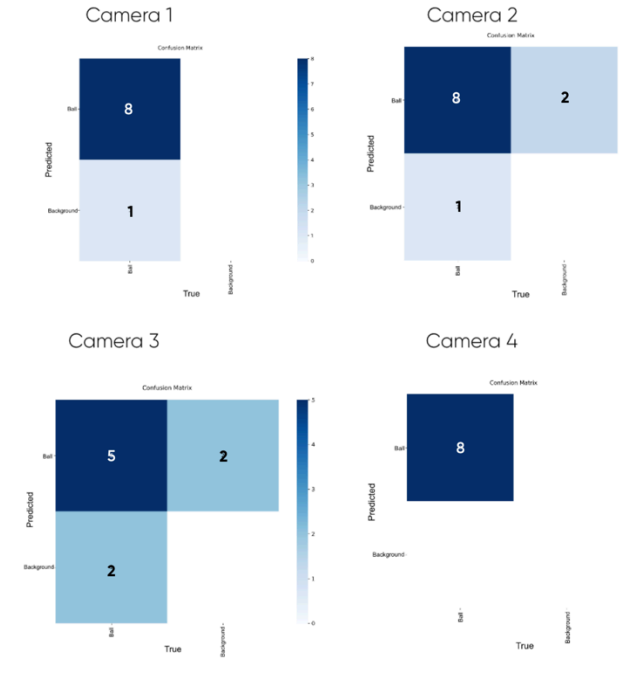

Figure 10. The Confusion Matrix for each camera model trained with one-fold method

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 success detections from images of clear view(left most), motion blur(middle), and partial occlusion(rightmost).

4.2 Model Inference Result

To evaluate the model's performance, inference was ran on Inference_set to test the model trained with five-fold-cross-validation method; accuracy score and dice score were listed in Table 2.

Table 2. Detection accuracy and dice score for five-fold cross-validation for inference.

	Accuracy Score	**Dice Score**
Five-Fold-Cross-Validation	0.7200	0.7651

• *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 12 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 12. Reconstructed chessboard inner-corner through camera calibration and triangulation algorithms on camera 4 image.

• 4.3 3D *Tracking Performance*

Examples of detected ball locations and Kalman filter predicted ball locations are shown in Figure 13.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 label, 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 3. The maximum, minimum, mean, and RMS of the detection and prediction errors are calculated and listed in Table 4. The histograms of detection error and prediction error distribution are shown in Figure 14.

Figure 13. Tracked trajectory with ten balls, Kalman Filter predicted balls are in white circles. the detected balls are in green circles.

Table 3. 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 4. 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 five-fold-cross-validation method trained model.

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

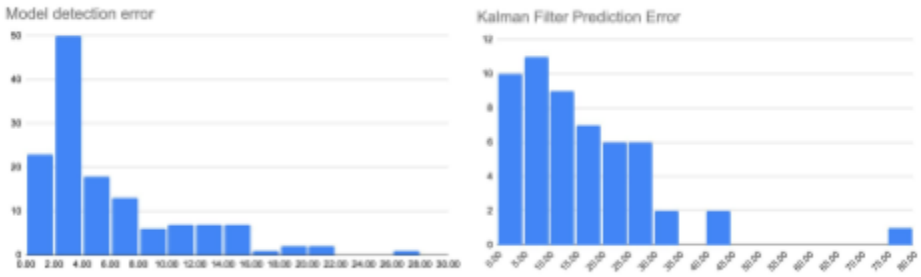

Figure 14. Histogram displaying the error distribution of the detection error for the model detected and Kalman filter predicted coordinates in 200 consecutive 1280x720 frames in Tracking_set, using five-fold-cross-validation trained model.

5. *Discussion*

5.1 Interpolation 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.

Using the one-fold method to train the model from four cameras resulted in different model performance: models for cameras 1 and 4 have a better F1 score, precision, recall, mAP, and detection result, as shown in Figures 6-10. This could come from the different angles of each camera. Cameras 1 and 4 both have the 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 YOLO11-obb model was chosen for its superior performance on the testing.
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 and filled in the gaps with the Kalman filter predictions.

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 a 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 YOLO11-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 could be optimized in position and size.
- AI modeling training on a stream of images: investigate the possibilities to incorporate temporal information in the AI model training for a 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.

Acknowledgements

The support and guidance of this research from my mentor, [name redacted], 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 on the logistics of my research.

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. TensorFlow: A System for Large-Scale Machine Learning. arXiv May 31, 2016. https://doi.org/10.48550/arXiv.1605.08695

Bankhead, P. et al. QuPath: Open source software for digital pathology image analysis. *Scientific Reports* (2017). https://doi.org/10.1038/s41598-017-17204-5

Carey, M. R.; Johnscm, D. S.; Tarjan, R. E. TRIANGULATING A SIMPLE POLYGON

Gomez-Gonzalez, S.; Neumann, G.; Schölkopf, B.; Peters, J. Using Probabilistic Movement Primitives for Striking Movements. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*; 2016; pp 502–508. https://doi.org/10.1109/HUMANOIDS.2016.7803322

Gomez-Gonzalez, S.; Nemmour, Y.; Schölkopf, B.; Peters, J. Reliable Real-Time Ball Tracking for Robot Table Tennis. *Robotics* **2019**, 8 (4), 90. https://doi.org/10.3390/robotics8040090

Hu, M.-C.; Chang, M.-H.; Wu, J.-L.; Chi, L. Robust Camera Calibration and Player Tracking in Broadcast Basketball Video. *IEEE Transactions on Multimedia* **2011**, 13 (2), 266–279. https://doi.org/10.1109/TMM.2010.2100373

Huang, Y.; Scholkopf, B.; Peters, J. Learning Optimal Striking Points for a Ping-Pong Playing Robot. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*; IEEE: Hamburg, Germany, 2015; pp 4587–4592. https://doi.org/10.1109/IROS.2015.7354030

IEEE Conference Publication | IEEE Xplore. *A brief introduction to OpenCV* https://ieeexplore.ieee.org/document/6240859 (accessed 2025-11-01).

Journal of Real-Time Image Processing (2025). *Real-time GPU color-based segmentation of football players.* https://link.springer.com/article/10.1007/s11554-011-0194-9 (accessed 2025-10-31)

Khanam, R.; Hussain, M. YOLOv11: An Overview of the Key Architectural Enhancements. arXiv October 23, 2024. https://doi.org/10.48550/arXiv.2410.17725

Krizhevsky, A.; Sutskever, I.; Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc., 2012; Vol. 25

Le Cun, Y.; Matan, O.; Boser, B.; Denker, J. S.; Henderson, D.; Howard, R. E.; Hubbard, W.; Jacket, L. D.; Baird, H. S. Handwritten Zip Code Recognition with Multilayer Networks. In *10th International Conference on Pattern Recognition [1990] Proceedings*; 1990; Vol. ii, pp 35–40 vol.2. https://doi.org/10.1109/ICPR.1990.119325

Multimedia Tools and Applications (2025). *A sensor-aided self coaching model for uncocking improvement in golf swing* https://link.springer.com/article/10.1007/s11042-013-1359-2 (accessed 2025-07-10).

Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*; IEEE: Las Vegas, NV, USA, 2016; pp 779–788. https://doi.org/10.1109/cvpr.2016.91

Renò, V.; Mosca, N.; Nitti, M.; D'Orazio, T.; Guaragnella, C.; Campagnoli, D.; Prati, A.; Stella, E. A Technology Platform for Automatic High-Level Tennis Game Analysis. *Computer Vision and Image Understanding* **2017**, *159*, 164–175. https://doi.org/10.1016/j.cviu.2017.01.002

Welch, G. An Introduction to the Kalman Filter. **1997**

Wong, K. C. P.; Dooley, L. S. High-Motion Table Tennis Ball Tracking for Umpiring Applications. In *IEEE 10th INTERNATIONAL CONFERENCE ON SIGNAL PROCESSING PROCEEDINGS*; 2010; pp 2460–2463. https://doi.org/10.1109/ICOSP.2010.5657001

Zhang, Z. A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2000**, *22* (11), 1330–1334. https://doi.org/10.1109/34.888718

Zhang, Y.; Chen, Z.; Wei, B. 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)*; 2020; pp 1312–1316. https://doi.org/10.1109/ICCC51575.2020.9345010

Zou, T.; Wei, J.; Yu, B.; Qiu, X.; Zhang, H.; Du, X.; Liu, J. Fast Moving Table Tennis Ball Tracking Algorithm Based on Graph Neural Network. *Sci Rep* **2024**, *14* (1), 29320. https://doi.org/10.1038/s41598-024-80056-3

Baider, L., Uziely, B., & Kaplan De-Nour, A. (1994). Progressive muscle relaxation and guided imagery in cancer patients. *General Hospital Psychiatry*, *16*(5), 340–347. https://doi.org/10.1016/0163-8343(94)90021-3

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

Final Recommendation: Accept with Minor Revisions

This paper presents a well-structured and methodologically sound study on 3D trajectory reconstruction of table tennis balls using a multi-camera setup, YOLO11-OBB for detection, and Kalman filtering for temporal smoothing. The work is highly relevant to the fields of computer vision and sports analytics, demonstrating both technical rigor and practical applicability. The author has clearly articulated the problem, contextualized it within existing literature, and provided a comprehensive description of the methodology, experiments, and results. The use of oriented bounding boxes (OBB) for detecting elliptical ball images under motion blur is particularly innovative and well justified. The integration of Kalman filtering effectively addresses issues of noise and occlusion, resulting in a robust tracking pipeline. The paper is well-organized, and the writing is generally clear and professional and making a valuable contribution to the field.

Review Feedback and Recommendations:

The paper is logically structured in general, and most sections are easy to follow. The use of figures and tables to illustrate results, setup, and performance metrics is very helpful. However, to achieve the standard as an accepted Convergence Journal paper, some revisions are required:

1. Clarify the Contribution in the Introduction (Section 1.4):

While the contribution is stated, it could be more sharply differentiated from prior work. The author can briefly but explicitly summarize what specific gap in the related work (Section 1.3), such as adding some context for the combination of OBB for shape accuracy on a fast-spinning object with a Kalman filter in 3D space for occlusion handling.

2. Expand on the Kalman Filter Implementation (Section 3.4):

The description of the Kalman filter is currently quite high-level. To enhance reproducibility and depth, the author can add a brief mention of the state vector dimensions (e.g., [x, y, z, vx, vy, vz]) and the values chosen for the process and measurement noise covariance matrices, even if they were empirically tuned. This adds clarity to the "constant velocity model" assumption.

3. Refine the Results and Discussion Sections:

In Section 4.3, the detection accuracy is 67.5%. This is a good result given the challenges, but the discussion should briefly address the 32.5% of frames where detection failed. Were these primarily due to occlusion, motion blur, or other factors? A sentence or two analyzing the main causes of missed detections would strengthen the critical analysis.

4. Revision for the Reference section:

There are several areas in the reference part where accuracy, consistency, and relevance need to be improved to meet academic publication standards. The author should diligently search for the complete details of the incomplete or incorrectly formatted references (Carey et al., the 2025 papers). A standard citation style (e.g., APA, IEEE) is required to follow (Check the Convergence Journal submission guidelines) and apply it consistently to every reference. This includes the order of elements (Author, Year, Title, Journal/Conference, Volume, Issue, Pages, DOI) and the use of italics.

5. Formatting:

Some minor problems need to be improved, such as figure titles need be remain on the same pages, the figures need to be checked with higher resolution quality, and remain the same size.

Keep the title on the same page as the figure:

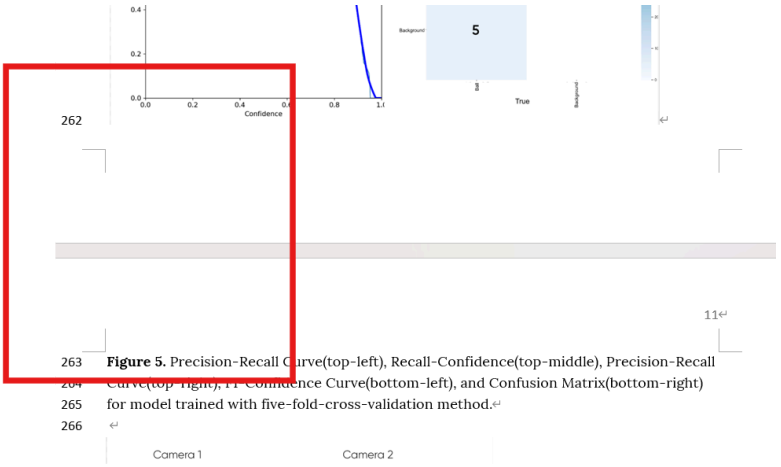

Figure 5. Precision-Recall Curve(top-left), Recall-Confidence(top-middle), Precision-Recall Curve(top-right), F1-Confidence Curve(bottom-left), and Confusion Matrix(bottom-right) for model trained with five-fold-cross-validation method.

Remain the same size/higher resolution quality:

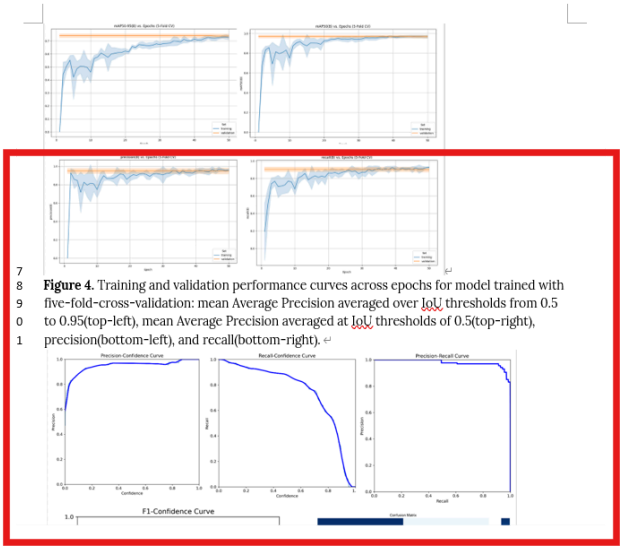

Figure 4. Training and validation performance curves across epochs for model trained with five-fold-cross-validation: mean Average Precision averaged over IoU thresholds from 0.5 to 0.95(top-left), mean Average Precision averaged at IoU thresholds of 0.5(top-right), precision(bottom-left), and recall(bottom-right).

Decision: Accept with major revisions (if lack of generalizability cannot be explained) or accept with minor revisions (if lack of generalizability can be explained well in the Discussion section)

Overall, the paper is technically quite strong with a robust literature review, but I have some concerns about the generalizability of the model to other datasets (as seen by performance drops in Tables 2 and 3). I would recommend an "Accept with revisions" decision, leaning towards major revisions since the lack of generalizability is a big concern, but also if the authors add a satisfactory explanation to the Discussion I could be convinced it should be accepted without any new analyses.

This paper presents an original computer vision approach to modeling the movement of physical objects in 3D space, and benefits from a very robust discussion of previous literature as well as discussion of tradeoffs between training/measurement strategies.Their tracking pipeline is well thought out and the use of a custom-trained YOLO11-OBB detector, and Kalman filtering is novel.

My primary concern is that the authors highlight an impressive 97.6% mAP on the test split, but performance drops significantly on the on the inference set (72% detection accuracy, Table 2) and during 3D tracking (58–67% frame-level detection accuracy, Table 3).

This suggests the model may be overfitted, the training set doesn't generalize to other datasets, or perhaps the real-world video tracking may have blurring or resolution issues. The authors should explicitly address this performance drop in order to better show whether "this combined approach provides a robust and scalable solution for advanced ping pong analytics" (as they argue in their abstract).

Furthermore, the discussion section is rather short and it would be helpful to expand on some of the initial discussion of the contributions of this paper that they set up in the Intro, as well as

adding a section to discuss the performance drop and generalizability of the model.

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

[redacted]

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: Electrical Engineering; signal and image processing; computer vision; machine learning; AI model; fast object detection and tracking, table tennis, trajectory tracking, sport 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 Unit for golf self-coaching (Renò et al., 2017; ~~MTA~~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 the above improvements.

Hardware and software improvements over the past decade have made it possible for student researchers to implement such systems. Convolutional Neural Networks (CNN), a type of deep learning neural network architecture, are commonly used for object classification and detection. CNNs evolved from Artificial Neural Networks (ANN) 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 leap 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 amount 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 (Carey et al., 1978~~22~~).

- ***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; ~~JRTIP~~ 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, he 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 pixelwise 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 (~~IEEE CP~~Culjak et al., 2025). The detected balls, in each time frame from multicamera 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, which provides an estimation of a system's state and uses past data to update this state, is applied on 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~~s~~, 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 ~~a better~~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~~the application of Kalman filtering effectively tracks the trajectory at the presence of occlusion without losing generalization~~, 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 as long as they didn't step out of the cameras' ranges. A 6x8 chessboard was also placed on the table for each camera to capture it 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.

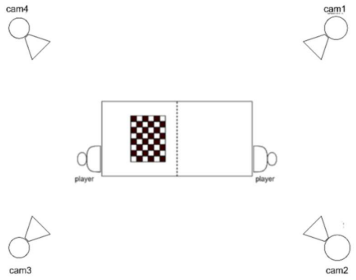

Figure 1. A diagram showing the capture setup with camera, chessboard and player locations.

- ***2.2. Data Acquisition & Labeling***

Recorded two sets of 5-minute videos from each camera. 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. 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, interference 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.

Camera 1 Camera 2 Camera 3 Camera 4

Figure 2. Hand-labeled inner-corner of the chessboard from different cameras.

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 validating 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. Prior to training, the images needed to be resized to the model's input dimensions.

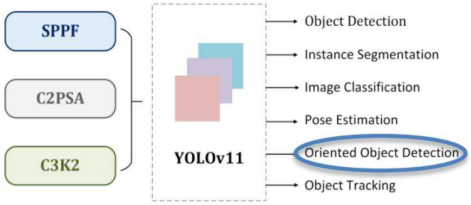

Figure 3. YOLO11 architecture adapted (Khanam and Hussain, 2024).

• *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, is the proportion of correct positive detection 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's the fraction of true positives to all objects.

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

Equation 4: F1-score evacuates 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.

$$mAP = \frac{1}{Q} \cdot \sum_{q=1}^{Q} AP(q),$$

$$[\![where\ Q\ is\ the\ number\ of\ classes, and\ AP(q)\ is\ the\ average\ precision\ for\ a\ specific\ class.]\!]$$

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 @ 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.

With the five-fold-cross-validation method, the model was trained for 50 epochs in each fold, and the performances across epochs are displayed in Figure 4; all metrics reached asymptote with 50 epochs. The performance metrics are displayed in Figure 5. The same metrics for one-fold trained models are illustrated in Figure 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 negative and 3 false positive detections. In comparison, the one-fold method resulted in varied models' performance.

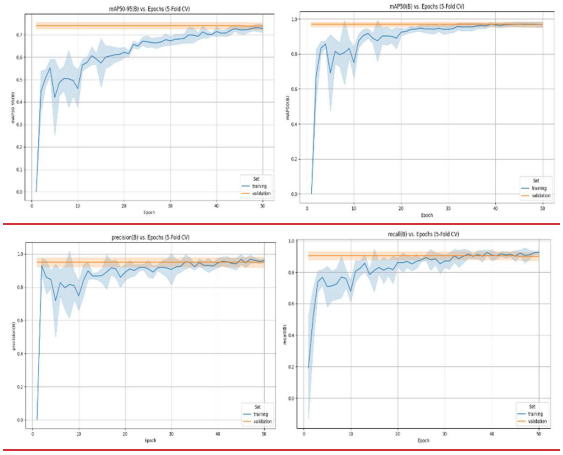

Figure 4. Training and validation performance curves across epochs for model trained with five-fold-cross-validation: mean Average Precision averaged over IoU thresholds from 0.5 to 0.95(top-left), mean Average Precision averaged at IoU thresholds of 0.5(top-right), precision(bottom-left), and recall(bottom-right).

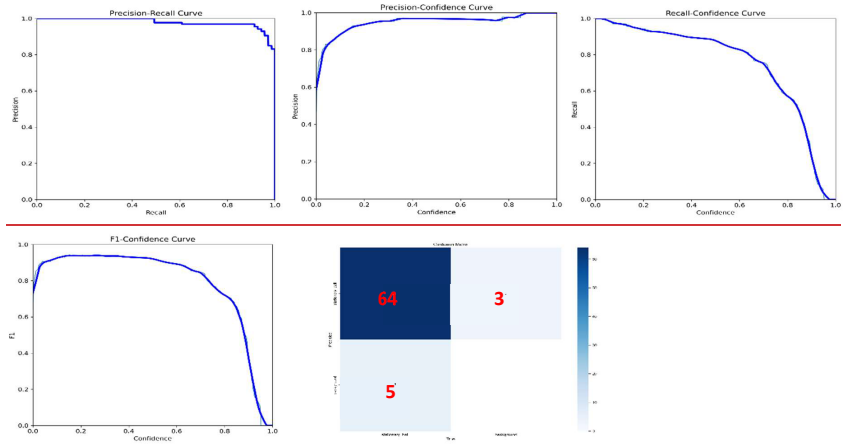

Figure 5. Precision-Recall Curve(top-left), Recall-Confidence(top-middle), Precision-Recall Curve(top-right), F1-Confidence Curve(bottom-left), and Confusion Matrix(bottom-right) for model trained with five-fold-cross-validation method.

Formatted: Indent: Left: 0", Hanging: 0.01", Right: 0"

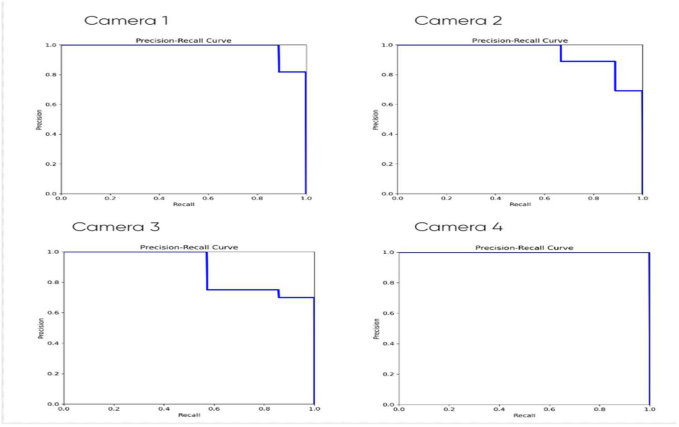

Figure 6. The Precision-Recall Curve for each camera model trained with one-fold method

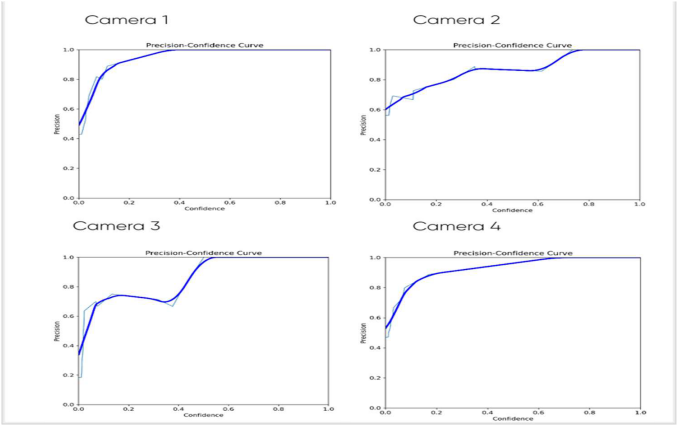

Figure 7. The Precision-Confidence Curve for each camera model trained with one-fold method

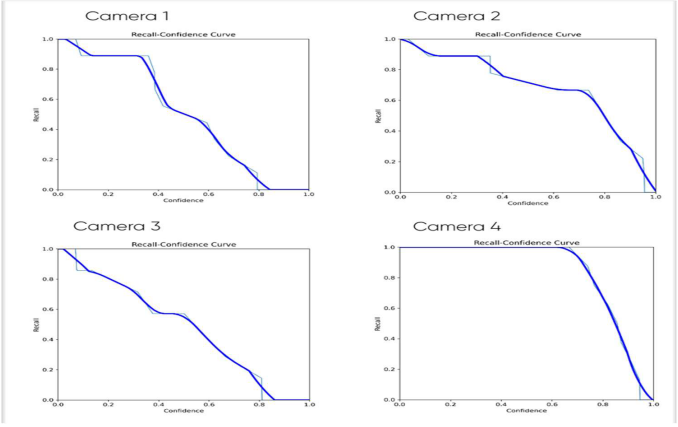

Figure 8. The Recall-Confidence Curve for each camera model trained with one-fold method

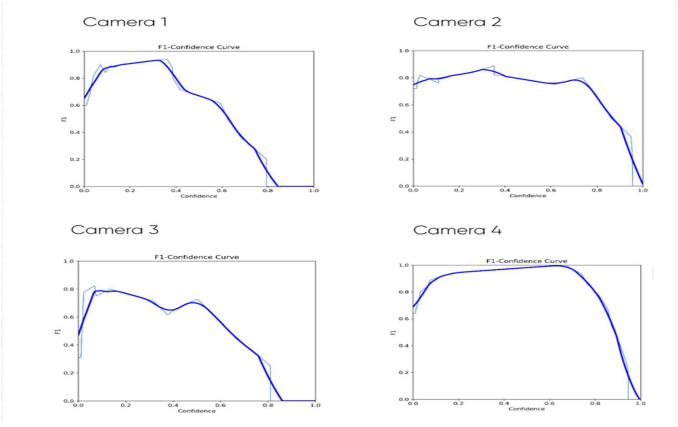

Figure 9. The F1-Confidence Curve for each camera model trained with one-fold method

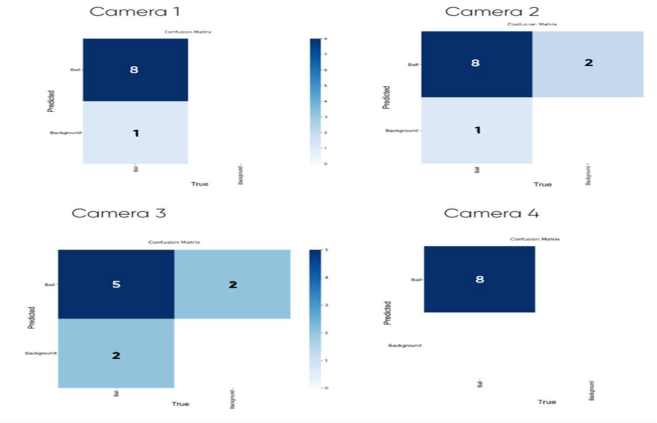

Figure 10. The Confusion Matrix for each camera model trained with one-fold method

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

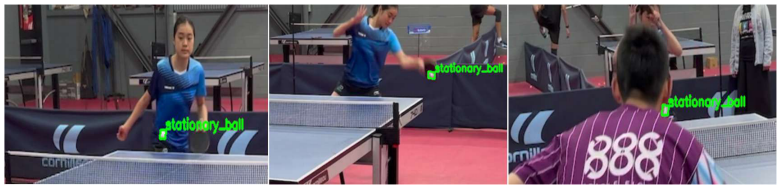

Figure 11: The images demonstrate success detections from images of clear view(left most), motion blur(middle), and partial occlusion(rightmost).

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

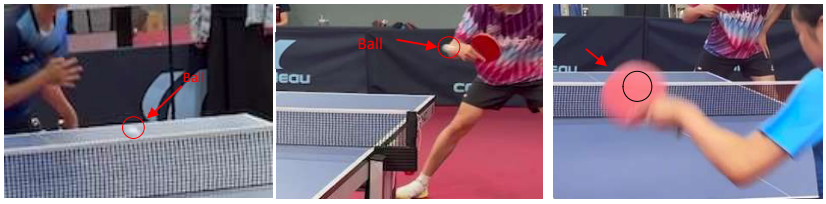

Figure 12: The images demonstrate failed detection from background interference(left 2~~3~~) and ball occlusion(right most). The red and black marks indicate where the ball is in each frame.

For Tracking_set, the confidence_threshold was set to 0.25, 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 that in both camera 4 and camera 1 are also listed. With 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, 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.

	Cam4	**Cam1**	**Cam2**	**Cam3**	**Cam4 & cam1**	**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 ~~ran~~run on Inference_set to test the model trained with five-fold-cross-validation method. For 100 images in Inference set, there were 74 true positive detections, 3 false positive detections and 7 false negative detections.~~;~~ ~~accuracy~~F1 score and dice score were listed in Table 2.

Table 3~~2~~. ~~Detection accuracy~~F1 and dice score for five-fold cross-validation for inference.

	F1~~Accuracy~~ Score	Dice Score
Five-Fold-Cross-Validation	0.93677~~200~~	0.7651

• *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~~2~~ 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~~2~~. Reconstructed chessboard inner-corner through camera calibration and triangulation algorithms on camera 4 image.

• *4.3~~3~~ 3D Tracking Performance*

Examples of detected ball locations and Kalman filter predicted ball locations are shown in Figure 14~~3~~. 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 label, 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 3. The maximum, minimum, mean, and RMS of the detection and prediction errors are calculated and listed in Table 4. The histograms of detection error and prediction error distribution are shown in Figure 15~~4~~.

Figure 1~~43~~. Tracked trajectory with ten balls, Kalman Filter predicted balls are in white circles~~,~~ the detected balls are in green circles.

Table ~~43~~. 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 ~~54~~. 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 five-fold-cross-validation method trained model.

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

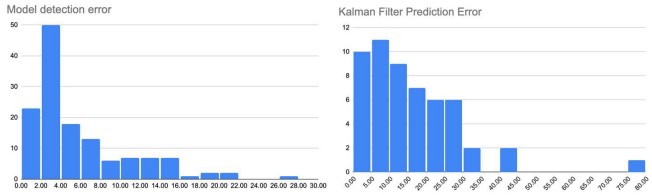

Figure 154. Histogram displaying the error distribution of the detection error for the model detected and Kalman filter predicted coordinates in 200 consecutive 1280x720 frames in Tracking_set, using five-fold-cross-validation trained model.

5. Discussion

5.1 Interpolation 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 detected ball mostly overlapped with the ball's location.

Using the one-fold method to train the model from four cameras resulted in different model performance: models for cameras 1 and 4 have a better F1 score, precision, recall, mAP, and detection result, as shown in Figures 6-10. This could come from the different angles of each camera. Cameras 1 and 4 both have the 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~~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 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, each camera achieved approximately 80% detection rate, demonstrating the effectiveness of YOLO11m-OBB model considering only a small set with 400 images was used for training. By increasing the ~~size of 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 5 with mean at 8.8 pixel and RMS at 9.5 pixel 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 a 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 could be optimized in position and size.
- AI modeling training on a stream of images: investigate the possibilities to incorporate temporal information in the AI model training for a 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.

Acknowledgements

The support and guidance of this research from [redacted], 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 ~~on~~in the logistics of my research.

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., … Zheng, X. (2016). *TensorFlow: A system for large-scale machine learning* (No. 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, 16878. https://doi.org/10.1038/s41598-017-17204-5

Carey, M. R., Johnscm, D. S., & Tarjan, R. E. (n.d.). *TRIANGULATING A SIMPLE POLYGON*.

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. (n.d.). *A brief introduction to OpenCV*.

Gomez-Gonzalez, S., Nemmour, Y., Schölkopf, B., & Peters, J. (2019). Reliable Real-Time Ball Tracking for Robot Table Tennis. *Robotics*, 8(4), Article 4. https://doi.org/10.3390/robotics8040090

Gomez-Gonzalez, S., Neumann, G., Schölkopf, B., & Peters, J. (2016). Using probabilistic movement primitives for striking movements. *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, 502–508. 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., Scholkopf, B., & Peters, J. (2015). Learning optimal striking points for a ping-pong playing robot. *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4587–4592. https://doi.org/10.1109/IROS.2015.7354030

Khanam, R., & Hussain, M. (2024). YOLOv11: *An Overview of the Key Architectural Enhancements*. (No. 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. *Advances in Neural Information Processing Systems, 25*. https://proceedings.neurips.cc/paper_files/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html

Formatted: Heading 2, Indent: Hanging: 0.01", Space After: 8 pt
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Font: Not Bold
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic

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. *10th International Conference on Pattern Recognition [1990] Proceedings, ii*, 35–40 vol.2. 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* (No. 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. (2001). *An introduction to the Kalman filter* (Technical Report TR 95-041). University of North Carolina at Chapel Hill.

Wong, K. C. P., & Dooley, L. S. (2010). High-motion table tennis ball tracking for umpiring applications. *IEEE 10th INTERNATIONAL CONFERENCE ON SIGNAL PROCESSING PROCEEDINGS*, 2460–2463. 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. *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, 1312–1316. 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), 29320. https://doi.org/10.1038/s41598-024-80056-3

~~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. TensorFlow: A System for Large-Scale Machine Learning. arXiv May 31, 2016. https://doi.org/10.48550/arXiv.1605.08695~~

~~Bankhead, P. et al. QuPath: Open source software for digital pathology image analysis. *Scientific Reports* (2017). https://doi.org/10.1038/s41598-017-17204-5~~

~~Carey, M. R.; Johnsem, D. S.; Tarjan, R. E. TRIANGULATING A SIMPLE POLYGON~~

Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Font: Italic
Formatted: Line spacing: Multiple 1.15 li
Formatted: Font: Italic
Formatted: Normal, Indent: First line: 0", Space After: 0 pt

Gomez-Gonzalez, S.; Neumann, G.; Schölkopf, B.; Peters, J. Using Probabilistic Movement Primitives for Striking Movements. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*; 2016; pp 502–508. https://doi.org/10.1109/HUMANOIDS.2016.7803322

Gomez-Gonzalez, S.; Nemmour, Y.; Schölkopf, B.; Peters, J. Reliable Real-Time Ball Tracking for Robot Table Tennis. *Robotics* **2019**, *8* (4), 90. https://doi.org/10.3390/robotics8040090

Hu, M.-C.; Chang, M.-H.; Wu, J.-L.; Chi, L. Robust Camera Calibration and Player Tracking in Broadcast Basketball Video. *IEEE Transactions on Multimedia* **2011**, *13* (2), 266–279. https://doi.org/10.1109/TMM.2010.2100373

Huang, Y.; Scholkopf, B.; Peters, J. Learning Optimal Striking Points for a Ping-Pong Playing Robot. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*; IEEE: Hamburg, Germany, 2015; pp 4587–4592. https://doi.org/10.1109/IROS.2015.7354030

IEEE Conference Publication | IEEE Xplore. *A brief introduction to OpenCV* https://ieeexplore.ieee.org/document/6240859 (accessed 2025-11-01).

Journal of Real-Time Image Processing (2025). *Real-time GPU-color-based segmentation of football players*. https://link.springer.com/article/10.1007/s11554-011-0194-9 (accessed 2025-10-31)

Khanam, R.; Hussain, M. YOLOv11: An Overview of the Key Architectural Enhancements. arXiv October 23, 2024. https://doi.org/10.48550/arXiv.2410.17725

Krizhevsky, A.; Sutskever, I.; Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc., 2012; Vol. 25

Le Cun, Y.; Matan, O.; Boser, B.; Denker, J. S.; Henderson, D.; Howard, R. E.; Hubbard, W.; Jacket, L. D.; Baird, H. S. Handwritten Zip Code Recognition with Multilayer Networks. In *10th International Conference on Pattern Recognition [1990] Proceedings*; 1990; Vol. ii, pp 35–40 vol.2. https://doi.org/10.1109/ICPR.1990.119325

Multimedia Tools and Applications (2025). *A sensor-aided self coaching model for uncocking improvement in golf swing* https://link.springer.com/article/10.1007/s11042-013-1359-2 (accessed 2025-07-10).

Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*; IEEE: Las Vegas, NV, USA, 2016; pp 779–788. https://doi.org/10.1109/cvpr.2016.91

Renò, V.; Mosca, N.; Nitti, M.; D'Orazio, T.; Guaragnella, C.; Campagnoli, D.; Prati, A.; Stella, E. A Technology Platform for Automatic High-Level Tennis Game Analysis. *Computer Vision and Image Understanding* **2017**, *159*, 164–175. https://doi.org/10.1016/j.cviu.2017.01.002

Welch, G. An Introduction to the Kalman Filter. **1997**

Wong, K.-C. P.; Dooley, L. S. High-Motion Table Tennis Ball Tracking for Umpiring Applications. In *IEEE 10th INTERNATIONAL CONFERENCE ON SIGNAL PROCESSING PROCEEDINGS*; 2010; pp 2460–2463. https://doi.org/10.1109/ICOSP.2010.5657001

Zhang, Z. A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2000**, *22* (11), 1330–1334. https://doi.org/10.1109/34.888718

Zhang, Y.; Chen, Z.; Wei, B. 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), 2020; pp 1312–1316. https://doi.org/10.1109/ICCC51575.2020.9345010

Zou, T.; Wei, J.; Yu, B.; Qiu, X.; Zhang, H.; Du, X.; Liu, J. Fast Moving Table Tennis Ball Tracking Algorithm Based on Graph Neural Network. *Sci Rep* **2024**, *14* (1), 29320. https://doi.org/10.1038/s41598-024-80056-3

Baider, L., Uziely, B., & Kaplan De-Nour, A. (1994). Progressive muscle relaxation and guided imagery in cancer patients. *General Hospital Psychiatry*, *16*(5), 340–347. https://doi.org/10.1016/0163-8343(94)90021-3

Response to review feedbacks and Recommendations:

Thank you so much for your encouraging review on my paper and helpful recommendations for improvement, here are the actions I have taken to address each recommendation:

1. Clarify the Contribution in the Introduction (Section 1.4):

Added explicit summarization on the contribution of current work in Section 1.4

2. Expand on the Kalman Filter Implementation (Section 3.4):

Added state vector dimension in the description of Kalman filter, and the values of process and measurement noise covariance with a brief explanation of why the process noise covariance is higher.

3. Refine the Results and Discussion Sections:

I went over my result and revised the result for inference, as in my original paper, the pictures with no ball were not taken out from the accuracy score calculation, so Table 3 was updated with F1-score. For the detection loss cases, I added Table 2 with number of frames with ball detection for each camera, as well as explanations on the causes for detection failure. I added several images to illustrate the cases of ball occlusion and background interference. I have also calculated the number of frames with more than two camera detection, and stated that with ideal camera calibration, the detection accuracy will be 20% better than 67.5% which only used camera 1 and camera 4 for 3D reconstruction.

4. Revision for the Reference section:

Corrected all my citations in the paper as well as the reference section.

5. Formatting:

Figure titles are made sure to be on the same page as the Figure, and figures are kept as the same size.

Response to review feedbacks and Recommendations:

Thank you so much for your encouraging review on my paper and insightful feedback, here are the actions I have taken to address your primary concern:

I went over my result and revised the result for inference, as in my original paper, the pictures with no ball were not taken out from the accuracy score calculation, so Table 3(Table 2 in the original paper) was updated with F1-score of 94.6%. For the detection loss cases, I added Table 2 with number of frames with ball detection for each camera, as well as explanations on the causes for detection failure. I added several images to illustrate the cases of ball occlusion and background interference. I have also calculated the number of frames with more than two camera detection, and stated in the discussion section that with ideal camera calibration, the detection accuracy will be 20% better than 67.5% which only used camera 1 and camera 4 for 3D reconstruction. The main contributing factor for lower detection rate was ball occlusion, which should be mitigated by images from 4 cameras by design. However, due to the non-ideal condition for camera calibration, which is one of the main things for future improvement, I can only use data from camera 1 and 4 for reliable 3 reconstruction.

As before, this paper takes a novel and original approach to tracking motion of an object in a real-world scenario, with wide ranging potential applications to both table tennis itself and other types of real-world motion tracking objectives. The primary contribution is the use of YOLO11-OBB detector and Kalman filtering to predict object location. Additionally, the authors have done a good job addressing some of the concerns from prior reviews, including detailing how their unique contributions improve on previous work, updating references and tables, and addressing some weaknesses in their analyses (e.g., results excluding occlusion).

While the approach is novel and the conclusions seem appropriate, there are a few issues that the authors should address prior to accepting this paper for publication:

1. The authors provide a solid explanation for why their approach should be better than previous approaches, but they don't actually directly compare their approach versus previous methods. Ideally, they would apply previous methods OR have a baseline performance from a simplistic model applied to their dataset to directly show that their method is better. In the absence of that, though, a more rigorous discussion of how the results compare to results in past papers (Is F1 score on the inference set better? Are there other ways this method could be quantified as "better"?) could address this issue. Without a baseline it's hard to evaluate how "good" these results are.
2. The authors note that a major limitation is the poor calibration of the cameras, but don't actually go into much detail about why their calibration was insufficient or how they would improve on this in the future. What could be done differently to address this limitation? For instance, they mention that "the chessboard for camera calibration could be optimized in position and size". Optimized in what way?
3. The data acquisition summary is incomplete and confusing. Table 1 shows that there were 325 images used, but two 5 minute videos with 60 frames per second should yield tens of thousands of images. The image characteristics column indicates that some of the images were selected at "random" and some were "consecutive". Could the authors provide more details on how these images were selected, why they didn't use the full dataset (split into train_set and inference_set), and whether there was any overlap in which images were randomly selected to be part of the train_set and inference_set (e.g., random selection without replacement).