https://doi.org/10.31449/inf.v49i23.9880

Enhanced Basketball Target Tracking in Panoramic Vision Systems via Advanced YOLO and Camshift Algorithm Integration

Jiaojiao Lu
Sport Institute, Yantai Institute of Science and Technology, Yantai, 265600, China
E-mail: jiaojiaoluj@outlook.com

Keywords: target, track, detection, MeanShift, YOLOv8, camshift

Received: June 16, 2026

The panoramic vision basketball robot can not only provide a 360° panoramic view, but also obtain real-time position and distance information of the basketball. To improve the target tracking performance of panoramic vision basketball robots, a target detection algorithm is first designed as the basis for subsequent robot target tracking. In object detection, the You Only Look Once version 8 is adopted. A bidirectional feature pyramid network and an improved cross stage partial 2-fusion are introduced for optimization. Based on the improved MeanShift algorithm, namely the continuous adaptive mean shift algorithm, target tracking is achieved. Subsequently, the research introduces Kalman filtering algorithm, target template update judgment, and fused features to optimize the target tracking algorithm. In terms of evaluation indicators, the study selects precision, recall, time consumption, central error, missed detection rate, and false detection rate, and conducted practical application tests. The precision, recall, and time consumption were 98.67%, 97.65%, and 54.21ms, respectively. Under obstructions, lighting changes, and background interference, the precision of the tracking algorithm was 95.67%, 96.47%, and 95.26%, respectively. The maximum central errors were 24 pixels, 21 pixels, and 22 pixels, respectively. The designed object detection and tracking algorithm has good performance and can provide solid technical support for the application of panoramic vision basketball robots in actual competitions.

Povzetek: Članek predstavi panoramskega košarkarskega robota združi izboljšan YOLOv8 za detekcijo z neprekinjenim adaptivnim MeanShiftom, Kalmanovim filtrom, posodobitvijo predloge in fuzijo značilk za robustno sledenje v težkih pogojih.

1 Introduction

With the development of artificial intelligence, computer vision, and robotics, intelligent robots are penetrating into various fields of society at an unprecedented speed [1-2]. In the field of sports competition, intelligent transformation has become the trend. From intelligent analysis in event broadcasting to athlete training assistance systems, intelligent technology is constantly improving the viewing and competitive level of sports events [3-4]. As a cutting-edge research direction in intelligent sports, panoramic vision basketball robots integrate panoramic vision technology and robot motion control technology. This robot can obtain real-time image information of the entire court through a panoramic camera, and provide data support for the robot to execute tactical decisions in basketball games with a wide field of view and fast perception ability [5]. In the panoramic vision basketball robot system, target tracking is a key technology for achieving autonomous decision-making and precise action execution of robots. Therefore, selecting the appropriate target tracking algorithm becomes particularly important. The commonly used target tracking methods currently include Mean Shift (MeanShift), Continuous Adaptive Mean Shift (Camshift), Channel and Spatial Features of Kernels (CSK), and Transformer target tracking algorithms [6-7]. In addition, many scholars have conducted research on target tracking algorithms. Wu et al. designed a composite cam driven micro hexapod crawling robot with visual perception and target tracking capabilities to expand large crawling robots to centimeter level. The robot took two micro motors as drivers and achieved trajectory tracking and motion target tracking through micro cameras and closed-loop control. The results showed that the robot could quickly crawl and turn in narrow spaces and outdoor environments [8]. Alymani et al. designed an intelligent mobile robot based on Red-Green-Blue (RGB) color object tracking for tracking colored objects. The results showed that the robot had a minimum average final error of 13.8cm and saved an average of 6.6s [9]. Kong et al. proposed a collision free tracking framework that combined improved guidance vector fields and disturbance rejection controllers for dynamic target tracking in obstacle environments. This framework transformed the elliptic integral curve into a straight line pointing towards the target point and constructed an adjustable parameter function. This method was effective in tracking virtual dynamic targets and dynamic quadruped robot targets in multi-obstacle environments [10]. Dai et al. proposed a visual servoing system based on adaptive images for tracking moving targets of incomplete wheeled mobile robots in obstacle environments. Meanwhile, a new constrained boundary function was designed based on pixel coordinates, which could deviate from zero. The method had good tracking performance and did not know feature height and target speed [11].

In summary, although existing methods have made some progress in the field of object detection and tracking, they still face many challenges in practical applications. In terms of object detection, although the existing You Only Look Once version 8 (YOLOv8) algorithm has high detection speed and accuracy, it has some shortcomings when dealing with complex scenes. For example, the YOLOv8 algorithm has limited feature extraction capabilities when dealing with multi-scale targets, especially when the target size changes significantly, resulting in a decrease in detection accuracy. In terms of target tracking, traditional MeanShift and Camshift algorithms have good real-time performance and robustness, but their tracking accuracy and stability are greatly affected when facing complex scenes such as occlusion, lighting changes, and background interference. For example, MeanShift algorithm is prone to losing targets when they are occluded, while Camshift algorithm is more sensitive to background interference and prone to misjudgment. Therefore, the research objective is to propose an improved object detection and tracking algorithm to enhance the performance of panoramic vision basketball robots in complex scenes. To achieve this objective and optimize the target tracking performance of panoramic vision basketball robots, a target detection model based on the improved YOLOv8 is designed, and a target tracking model based on the improved Camshift is designed. The research aims to improve the shortcomings of MeanShift and Camshift algorithms, enhance the target tracking accuracy and stability of robots in panoramic vision, and help sports intelligence move towards new heights. The innovation of the research is as follows. Compared with existing studies, the YOLOv8 algorithm introduces a bidirectional feature pyramid network and an improved cross stage partial 2-fusion module, which preserves most high-order information by expanding the receptive field to improve model accuracy. In target tracking, color features and texture features are integrated, and a dynamic template update mechanism is combined to solve the problem of traditional algorithms easily losing targets after occlusion.

2 Methods and materials

To improve the target tracking performance of panoramic vision basketball robots, the research starts with target detection, and designs a detection method based on the improved YOLOv8 algorithm. Then, it is taken as the basis for subsequent robot target tracking. In target tracking, the MeanShift algorithm is improved.

2.1 Design of robot object detection method based on improved YOLOv8

Robot target detection and tracking are important technologies in computer vision. Object detection provides initial target state information for target tracking. Once the target is detected, the robot's target tracking function begins to take effect [12-13]. Therefore, to optimize the target tracking performance of panoramic vision basketball robots, the research first designs a target detection method based on the improved YOLOv8. Then, a target tracking method based on the improved MeanShift is constructed. YOLOv8 is a real-time object detection algorithm that not only adopts an anchor free detection method, but also directly predicts the bounding box and category probability of objects in the image through a single forward propagation. It has fast speed, high accuracy, and high efficiency [14-15]. The YOLOv8 mainly has a backbone network, neck structure, and head structure. The backbone network extracts deep semantic information and feature maps from input images to provide basic feature support for subsequent object detection. The neck structure fuses feature of different scales, while the head structure is responsible for target classification and localization [16-17]. However, the YOLOv8 algorithm also has certain shortcomings, such as high computational and memory requirements, making its deployment in panoramic vision basketball robots difficult. Therefore, the YOLOv8 is optimized. Figure 1 displays the improved YOLOv8.

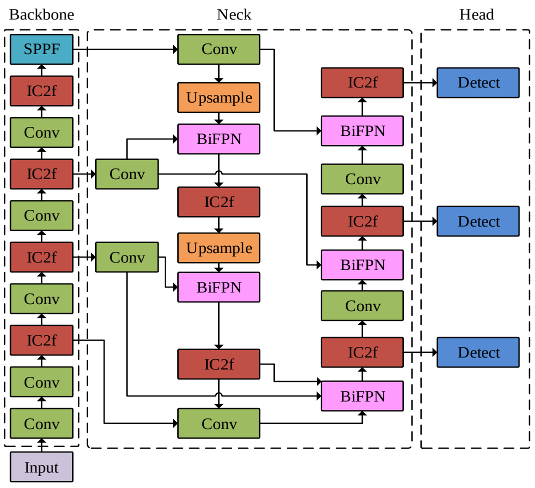


Figure 1: The structure of the improved YOLOv8

From Figure 1, the structure mainly includes Bidirectional Feature Pyramid Network (BiFPN), convolutional layer, up-sampling, Improved Cross stage partial 2-Fusion (IC2f), and Spatial Pyramid Pooling-Fast (SPPF). Among them, the BiFPN module captures target features of different scales through bidirectional information flow, and dynamically adjusts the contribution of different scale features through weighted feature fusion, thereby improving detection accuracy while reducing the number of parameters. The SPPF module can capture feature information at multiple scales through max pooling operations with different scales, and then concatenate these feature maps at different scales to form richer feature representations, thereby improving the model's detection ability for targets. Convolutional layers are mainly responsible for feature extraction, while up-sampling is mainly responsible for multi-scale feature fusion and detail enhancement. IC2f enhances the model's ability to capture complex features by expanding the receptive field of each feature while, retaining most of the high-order information and increasing the convolution frequency of high-order features. In addition, the IC2f module also avoids the negative impact of low-order features on detection accuracy by limiting the degree of intervention of low-order features.

IC2f is the first improvement of the YOLOv8 algorithm to enhance its feature extraction capability and avoid the increased computational and memory burden caused by the low order features. Figure 2 displays the IC2f module.

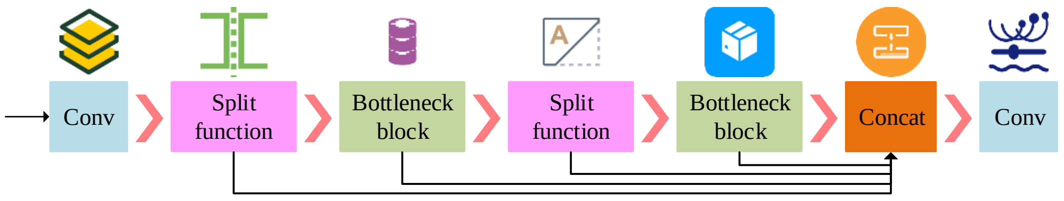


Figure 2: The structure of the IC2f

From Figure 2, the IC2f mainly consists of convolutional layers, split functions (Split), bottleneck blocks, and concatenation functions. The split function needs to divide the input tensor into multiple sub-tensors along a specified dimension, and these sub-tensors need to be processed in parallel to improve computational efficiency. In the IC2f module, the function of the bottleneck block is to efficiently process the segmented sub-tensors, reduce the computational burden by reducing the number of features, and enhance the feature representation ability through the feature extraction layer. The function of the connection function is to recombine the processed sub-tensors into a complete feature tensor. This merging process not only preserves the feature information of each sub-tensor, but also enhances the model's understanding of global information through feature fusion. The convolutional layer is the foundation of the IC2f module, responsible for extracting key information from input features. The improved IC2f module is mainly reflected in the convolution module, which expands the receptive field of each feature and preserves most high-order information. It also increases the convolution operation frequency of high-order features and constrains the intervention degree of low order features to improve model accuracy. In addition, when shortcut connections are set, the bottleneck block A is shown in equation (1) [18].

$$A(b) = b + Conv2(Conv1(b)) \tag{1}$$

In equation (1), b represents the input tensor. $Conv1$ represents the first convolutional layer. $Conv2$ is the second convolutional layer. When no shortcut connection is set, the bottleneck block is shown in equation (2) [19].

$$A(b) = Conv2(Conv1(b))$$

(2)

The introduced BiFPN is the second improvement of YOLOv8 algorithm. BiFPN is an efficient multi-scale feature fusion network that allows features to be fused in both top-down and bottom-up directions. It has strong feature fusion ability, high computational efficiency, robustness, and flexibility, and has extensive applications in object detection and image segmentation [20-21]. Therefore, the research introduces it into the head network of YOLOv8 algorithm to improve feature interaction while reducing parameter count. Figure 3 displays the BiFPN [22].

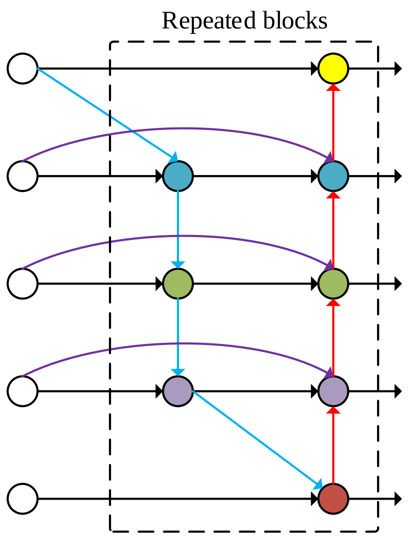


Figure 3: The structure of BiFPN

From Figure 3, BiFPN is mainly composed of nodes of different colors, top-down paths, bottom-up paths, and weighted feature fusion. Nodes of different colors have their own functions. For example, yellow nodes transmit semantic information to the next layer and receive fused features from the next layer in a bottom-up process to achieve bidirectional information exchange. Green nodes fuse features of different scales to enhance the detection ability of medium scale targets. The fused output feature C is shown in equation (3) [23].

$$C = \sum_D \frac{E_D \times F_D}{g + \sum_D E_D}$$

(3)

In equation (3), F_D signifies the D-th input feature. E_D represents the weight corresponding to the D-th input feature. g is a small positive number used to prevent the denominator from being zero.

$\sum_D E_D$ is the sum of all weights used to normalize the weighted sum.

Therefore, the improved YOLOv8 can achieve more efficient feature fusion and higher detection accuracy through BiFPN, while avoiding redundant feature extraction and computation. In the improved YOLOv8 algorithm, the pseudocode of SPPF is shown in Figure 4.

```
import torch
import torch.nn as nn

class SPPF(nn.Module):
    def __init__(self, in_channels, out_channels):
        super().__init__()
        mid_channels = out_channels // 4
        self.conv1 = nn.Conv2d(in_channels, mid_channels, 1, 1)
        self.pool = nn.MaxPool2d(3, 1, 1)
        self.conv2 = nn.Conv2d(mid_channels * 4, out_channels, 1,
1)

    def forward(self, x):
        x = self.conv1(x)
        x1 = x  # 1x1 pooling
        x2 = self.pool(x)  # 3x3 pooling
        x3 = self.pool(x2)  # 5x5 equivalent
        x4 = self.pool(x3)  # 7x7 equivalent
        return self.conv2(torch.cat([x1, x2, x3, x4], dim=1))
```

Figure 4: Pseudocode of SPPF in improved YOLOv8 algorithm

From Figure 4, the pseudocode employs the PyTorch framework and defines an SPPF class that inherits from nn. Module. In this pseudocode, SPPF first adjusts the channel through 1×1 convolution, and then simulates pooling effects of different sizes through multiple stacking of 3×3 pooling layers. Finally, the features of the four branches are connected and convolved to effectively capture multi-scale features.

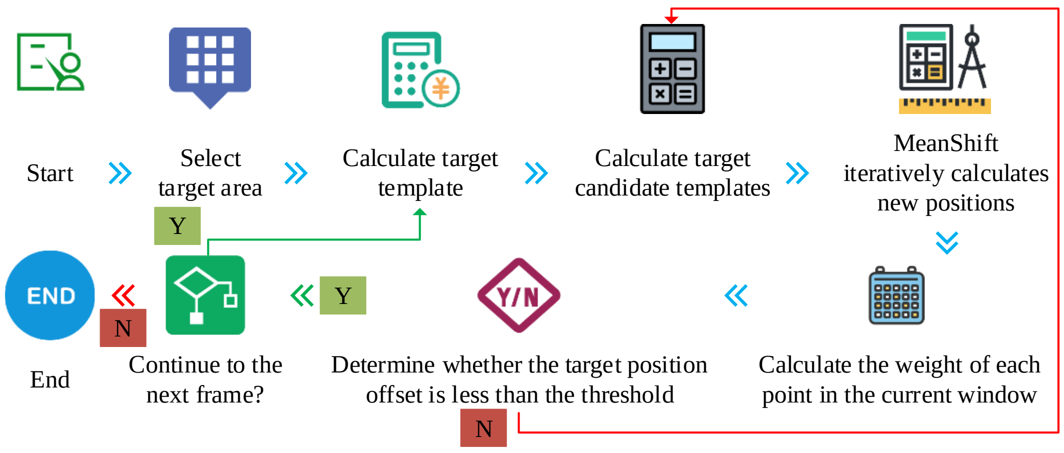

Start
Select target area
Calculate target template
Calculate target candidate templates
MeanShift iteratively calculates new positions
Calculate the weight of each point in the current window
Determine whether the target position offset is less than the threshold
Y/N
N
Y
Continue to the next frame?
Y
N
END
End

Figure 5: The main process of MeanShift algorithm

2.2 Design of robot target tracking method based on improved MeanShift

To improve the target tracking performance of panoramic vision basketball robots, a target detection method based on the improved YOLOv8 is designed to provide initial target state information for subsequent target tracking. To achieve target tracking, the MeanShift algorithm is improved. MeanShift algorithm is an unsupervised learning algorithm based on kernel density estimation. It can update the appearance model of the target in real-time in target tracking and predict the position of the target in the next frame based on its position and features in the current frame. It has good real-time performance, robustness to noise, no need to pre-set the number of clusters, and strong adaptability to non-spherical clustering [24-25]. Figure 5 displays the main process of MeanShift.

From Figure 5, the main process of MeanShift algorithm includes calculating the target template and target candidate template, calculating the new position of MeanShift iteration, and determining whether the target position offset is less than the threshold. In one-dimensional kernel density space, the expression of kernel density estimation $h(I)$ is shown in equation (4) [26].

$$h(I)=\frac{1}{J}\sum_{k=1}^{K}L_M\left(I-I_k\right) \tag{4}$$

In equation (4), I is the data point. K signifies the total I. k signifies the sequence number of I. $L(I)$ signifies the kernel function. M represents the bandwidth of $L(I)$. In the multidimensional kernel density space, a radial kernel function $L(I)'$ is used, which is expressed as equation (5).

$$L(I)'=N_{l,p}l\left(\|I\|^2\right) \tag{5}$$

In equation (5), $l(\cdot)$ signifies the contour coefficient of $L(I)'$. $N_{l,p}$ is the normalization constant. p is the data dimension. At this time, the kernel density estimation $h_{M,L}(I)$ is shown in equation (6) [27].

$$h_{M,L}(I)=\frac{N_{l,p}}{JM^p}\sum_{k=1}^{K}l\left(\left\|\frac{I-I_k}{M}\right\|^2\right) \tag{6}$$

In equation (6), $\|I-I_k\|$ represents the Euclidean distance between point I and data point I_k. However, the MeanShift algorithm also has certain problems, such as fixed window size limitations, difficulty in adapting to changes in target scale, high computational resource consumption, and difficulty in meeting real-time tracking requirements. Therefore, an improved algorithm for MeanShift is proposed, namely Camshift algorithm [28]. Camshift is a target tracking algorithm based on MeanShift algorithm, which mainly characterizes the target through color histogram and continuously updates the position, size, and other information of the target to continuously track the target [29]. The Camshift algorithm can not only adaptively adjust the window, but also resist the situation where the target is partially occluded, while having high computational efficiency. Figure 6 displays the main process of Camshift.

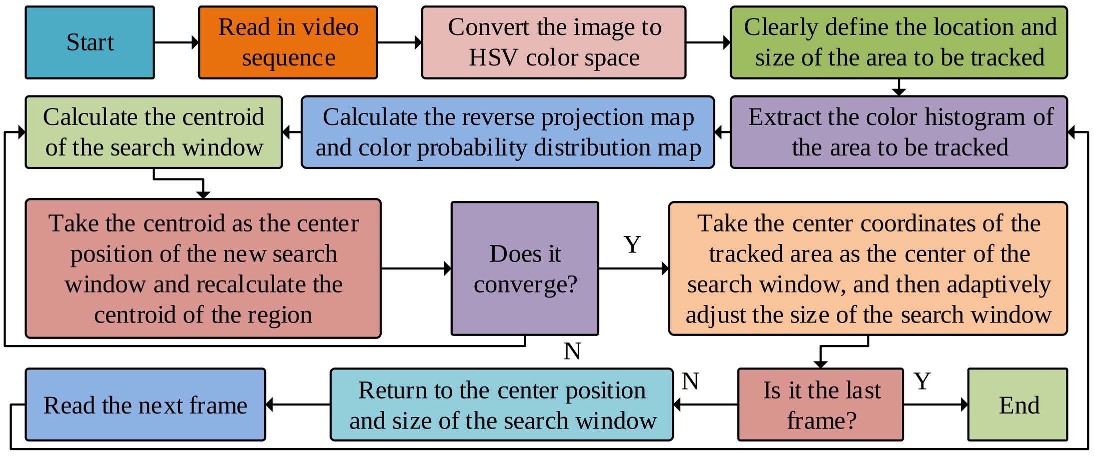

Start
Read in video sequence
Convert the image to HSV color space
Clearly define the location and size of the area to be tracked
Calculate the centroid of the search window
Calculate the reverse projection map and color probability distribution map
Extract the color histogram of the area to be tracked
Take the centroid as the center position of the new search window and recalculate the centroid of the region
Does it converge?
Y
Take the center coordinates of the tracked area as the center of the search window, and then adaptively adjust the size of the search window
N
Read the next frame
Return to the center position and size of the search window
N
Is it the last frame?
Y
End

Figure 6: The main process of Camshift algorithm

From Figure 6, the key steps of Camshift algorithm are color space conversion and histogram back-projection. Color space conversion is the process of converting an image from the RGB color space to the Hue-Saturation-Value (HSV) color space to reduce the impact of lighting changes on the detection target. The HSV color space can help distinguish targets of different colors. Even with other interference, it can achieve good object detection results based on color features. Figure 7 displays the HSV color space model.

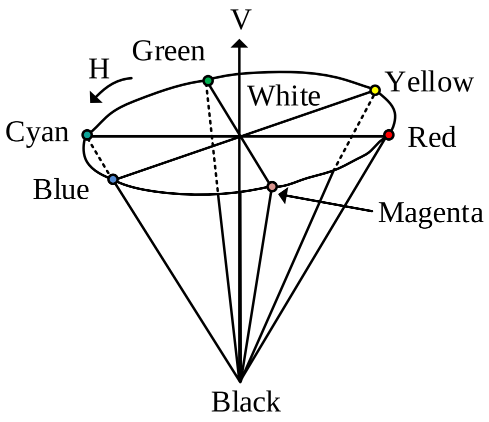


Figure 7: The model of HSV color space

From Figure 7, in the HSV color space model, hue signifies the type of color, and the range is usually 0-360°. For example, green is around 60-180°. Saturation represents the purity of a color, that is, the proportion of gray components in the color, and the value range is generally from 0 to 100%. Brightness represents the brightness of a color, with a range of 0-100%. When the brightness is 0, the color is black. The color space conversion is shown in equation (7).

$$H = \begin{cases} 0^{\circ} & (if \;\; V = 0, S = 0\,) \\ 60^{\circ} \times \left(\dfrac{G - B}{S} \right) & (if \;\; V = R\,) \\ 60^{\circ} \times \left(\dfrac{B - R}{S} + 2 \right) & (if \;\; V = G\,) \\ 60^{\circ} \times \left(\dfrac{R - G}{S} + 4 \right) & (if \;\; V = B\,) \end{cases} \tag{7}$$

After color space conversion, a distributed histogram can be obtained, and histogram back-projection creates a color probability distribution map to help quickly locate the target area. The distribution probability $\hat{Q}$ is shown in equation (8).

$$\begin{cases} \hat{Q} = \left\{ \hat{Q}_r \right\}, r = 1, 2, 3, \ldots, T \\ \sum \hat{Q}_r = 1 \end{cases} \tag{8}$$

In equation (8), T represents the total number of eigenvalues r. $\hat{Q}_r$ represents the distribution probability of r. The expression of $\hat{Q}_r$ is shown in equation (9).

$$\hat{Q}_r = N_M \sum_{k=1}^{K} l \left(\left\| \frac{(X_k, Y_k) - (X_0, Y_0)}{M} \right\|^2 \right) W \left[Z\left((X_k, Y_k) \right) - r \right] \tag{9}$$

In equation (9), (X_k, Y_k) represents the coordinates of the pixel point. N_M represents the normalization coefficient. $W(\cdot)$ is a one-dimensional delta function. Z signifies the chromaticity value of the coordinate point. In addition, the histogram back-projection is presented in equation (10).

$$a\left(X_k,Y_k\right)=\sum_{k=1}^{K}\hat{Q}_r W\left[Z\left(\left(X_k,Y_k\right)\right)-r\right]$$

(10)

However, although Camshift algorithm solves the shortcomings of MeanShift algorithm, it still has certain problems, such as sensitivity to background interference, poor robustness under lighting changes, and the need to strengthen the ability to handle obstruction problems. An improved Camshift is designed. Figure 8 displays the main process of the improved Camshift.

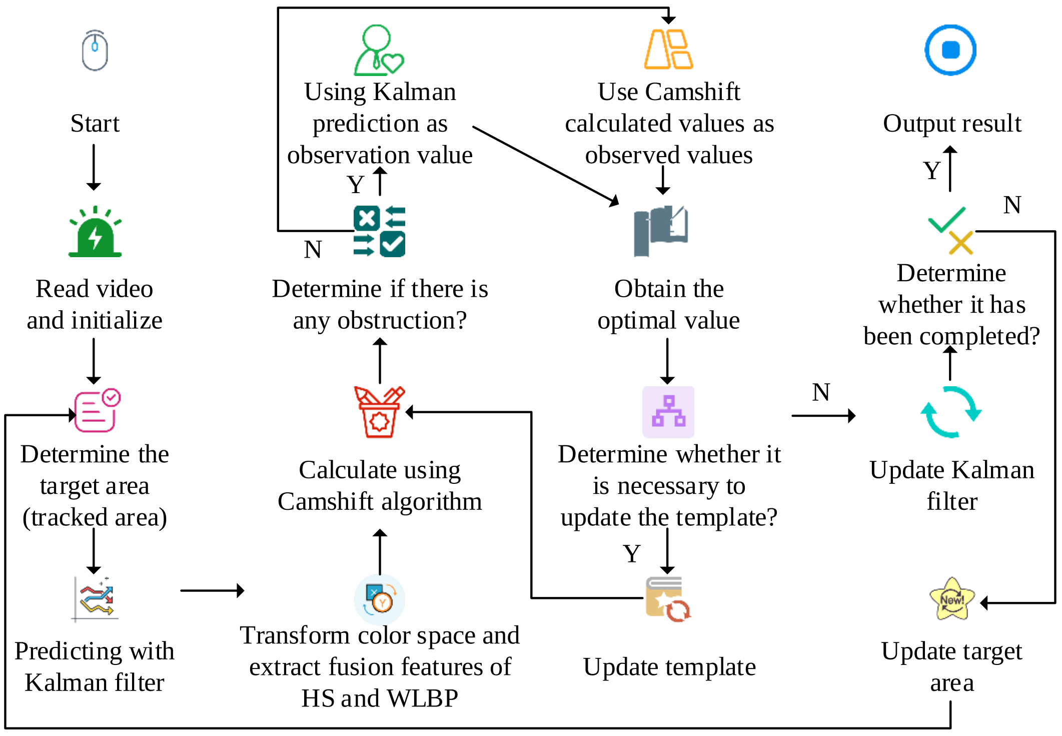

Start
Read video and initialize
Determine the target area (tracked area)
Predicting with Kalman filter
Using Kalman prediction as observation value
Y
N
Determine if there is any obstruction?
Calculate using Camshift algorithm
Transform color space and extract fusion features of HS and WLBP
Use Camshift calculated values as observed values
Obtain the optimal value
Determine whether it is necessary to update the template?
Y
N
Update template
Output result
Y
N
Determine whether it has been completed?
Update Kalman filter
Update target area

Figure 8: The main process of the improved Camshift

From Figure 8, the core steps of the improved Camshift include using Kalman filters for obstruction prediction, extracting the fused feature, updating the target template, and updating the Kalman filter. Extracting the fused feature is the first improvement of Camshift algorithm to obtain more complete color features. The specific improvement details involve combining chromaticity and saturation components to generate a template HS to obtain richer color information, and extracting texture features. Meanwhile, the research adaptively fuses HS and Weighted Local Binary Pattern (WLBP) features. LBP features are mainly used to describe the local texture information of an image, while WLBP is an improved method that introduces the concept of weights based on LBP features. Its advantage is that it can highlight important features, adapt to diverse textures, and reduce the influence of noise [30-31]. Therefore, to obtain WLBP features, wavelet decomposition method is adopted in the study. The reverse projection after feature fusion is shown in equation (11).

$$f=\frac{t_{HS}}{t_{WLBP}+t_{HS}}f_{HS}+\frac{t_{WLBP}}{t_{WLBP}+t_{HS}}f_{WLBP}$$

(11)

In equation (11), t_{HS} and t_{WLBP} are histograms of HS and WLBP features, respectively. f_{HS} represents the similarity coefficient of HS color features. f_{WLBP} represents the similarity coefficient of texture features. f represents the similarity coefficient of features, which is the contribution of each feature to the final fusion result. The reverse projection pseudocode after feature fusion is shown in Figure 9.

```
def back_projection_fused(HS_hist, WLBP_hist, f_HS, f_WLBP):
    # Calculate fused histogram
    fused_hist = (f_HS * HS_hist + f_WLBP * WLBP_hist) / (f_HS
+ f_WLBP)
    # Back projection: map each pixel to fused probability
    prob_map = histogram_backprojection(fused_hist)
    return prob_map
```

Figure 9: Reverse projection pseudocode after feature fusion

From Figure 9, the reverse projection pseudocode after feature fusion first takes the similarity coefficient of the features to weight the histograms of different features to obtain the fused histogram. Afterwards, based on the fused histogram, the reverse projection operation is performed and probability mapping results are generated, thereby realizing the reverse projection process after feature fusion. The expression of f_{HS} is shown in equation (12).

$$f_{HS} = \sum_{r_1=1}^{T} \sqrt{\beta_{HS,r_1} \delta_{HS,r_1}} \qquad (12)$$

In equation (12), δ_{HS} represents the target template for HS color features. β_{HS} is a candidate template for HS color features. The expression of f_{WLBP} is shown in equation (13).

$$f_{WLBP} = \sum_{r_2=1}^{T} \sqrt{\beta_{WLBP,r_2} \delta_{WLBP,r_2}} \qquad (13)$$

In equation (13), δ_{WLBP} and β_{WLBP} represent the target template and candidate template for texture features, respectively. The update judgment of the target template is the second improvement of the Camshift algorithm to avoid the continuous accumulation of bias under obstruction. This process requires comparing the similarity j with a pre-set threshold i. The update method is shown in equation (14).

$$\eta^{(\theta+1)} = \begin{cases} \eta_{\lambda}^{(\theta)}, \ if \ i \geq j \\ \eta^{(\theta)}, \ otherwise \end{cases} \qquad (14)$$

In equation (14), $\eta^{(\theta+1)}$ represents the updated target template, and i belongs to $[0,1]$. $\eta^{(\theta)}$ and $\eta_{\lambda}^{(\theta)}$ respectively represent the current target template and the range of better candidate targets.

The introduced Kalman filtering algorithm is the third improvement of Camshift algorithm to enhance the tracking performance. The Kalman filtering algorithm has good real-time performance, low computational complexity, high accuracy, and strong robustness, and has extensive applications in the field of target tracking [32-34]. Therefore, the study takes Kalman filtering algorithm to predict whether there is obstruction. When implemented, the Kalman filter is mainly used to predict the position and velocity of the target, providing reliable estimates when the target is occluded or lost, to compensate for the shortcomings of the Camshift algorithm in occlusion situations. The Camshift algorithm provides real-time observation data for the Kalman filter, ensuring real-time and accurate tracking. Specifically, in each frame, a Kalman filter is first used for prediction to obtain the predicted position and velocity of the target. Then, the improved Camshift algorithm is taken to calculate the actual position of the target. If the target is not occluded, the result of Camshift algorithm will be used as the observation value for updating the Kalman filter. If the target is obstructed, the predicted value of the Kalman filter will be used as the observation value. In this way, the Kalman filter and the improved Camshift algorithm complement each other. The pseudocode for occlusion prediction using Kalman filter is shown in Figure 10.

```
def kalman_occlusion_predict(obs, is_occluded):
    pred_state = trans_mat @ state
    pred_cov = trans_mat @ cov @ trans_mat.T + noise

    if is_occluded:
        upd_state = pred_state
    else:
        innov = obs - obs_mat @ pred_state
        gain = pred_cov @ obs_mat.T @ inv(obs_mat @ pred_cov @ 
obs_mat.T + obs_noise)
        upd_state = pred_state + gain @ innov

    state[:] = upd_state
    return upd_state
```

Figure 10: Pseudocode for occlusion prediction using Kalman filter

From Figure 10, the pseudocode implements Kalman filter occlusion prediction. It first completes state and covariance prediction based on transition matrices and other factors. If the target is occluded, the state is updated directly with the predicted value. If the target is not occluded, the observation is used to correct the predicted value, reflecting the tracking logic in conjunction with the Camshift algorithm.

3 Results

To verify the designed object detection algorithm and object tracking algorithms, experimental equipment, experimental datasets, and comparison methods are set up. In addition, in terms of evaluation indicators, the study selects precision, recall, Intersection over Union (IoU) ratio, and memory usage. Meanwhile, the study also evaluates the performance in specific practical applications.

3.1 Performance validation of object detection algorithms

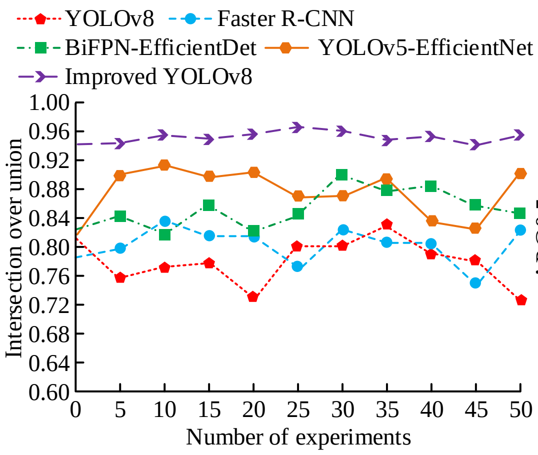
To validate the designed object detection algorithm, the well-known Common Objects in Context (COCO2017) dataset is taken. The COCO2017 dataset collects over 200,000 images and is separated into training, testing, validation sets, and unlabeled images [35]. The study takes the Windows 10 operating system. The Central Processing Unit (CPU) is an Intel Core i5-12600KF, with a dynamic acceleration frequency of 4.9GHz and maximum memory support of 128GB. In terms of comparison methods, YOLOv8, Fast Region-based Convolutional Neural

Network (Faster R-CNN), YOLOv5-EfficientNet combined YOLOv5 with Efficient Neural Network (EfficientNet), and BiFPN-EfficientDet combined BiFPN with Efficient Object Detection (EfficientDet) are selected. In addition, the model has 200 iterations and uses a stochastic gradient descent optimizer. In terms of evaluation indicators, the study adopts precision, recall, IoU ratio, and mean Average Precision@0.5 (mAP@0.5), CPU utilization, and memory usage, etc. Table 1 displays the ablation experiment results.

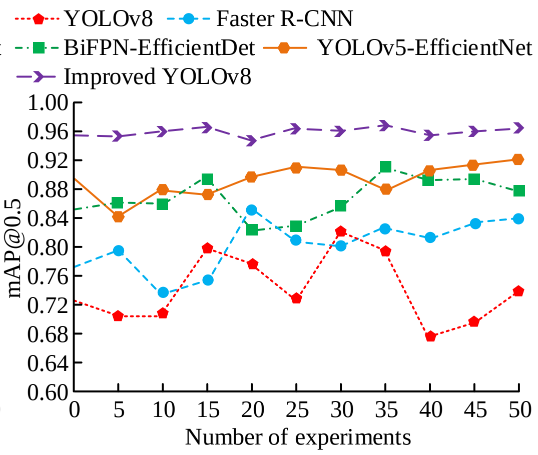
Table 1: Experimental results of ablation based on improved YOLOv8 detection model

YOLOv8	IC2f	BiFPN	Index		
			Precision	Recall	Time consumption
√	√	√	98.67%	97.65%	54.21ms
√	√	×	96.43%	95.73%	58.98ms
√	×	√	95.02%	94.38%	61.76ms
√	×	×	88.64%	87.05%	87.84ms

From Table 1, the precision, recall, and time consumption values of the detection model combined with YOLOv8, IC2f, and BiFPN were 98.67%, 97.65%, and 54.21ms, respectively, indicating significantly better performance than other combinations. For example, in terms of precision, the values of the YOLOv8, the model combining YOLOv8 and IC2f, as well as the model combining YOLOv8 and BiFPN were 88.64%, 96.43%, and 95.02%, respectively, which were 10.03%, 2.24%, and 3.65% lower than 98.67%. This may be because the designed the IC2f module, which improves the feature extraction ability. Therefore, it can more accurately identify target objects in basketball games, improving precision and recall. Meanwhile, the BiFPN structure also optimizes the multi-scale feature fusion, further improving the precision and recall rates. The IoU ratio and mAP@0.5 of different models are shown in Figure 11.



(a) Comparison of intersection over union



(b) Comparison of mAP@0.5

Figure 11: Comparison of IoU and mAP@0.5 among different models

From Figure 11 (a), the maximum IoU of the improved YOLOv8 was0.967, and the minimum value was 0.949. The overall IoU ratio of the model was above 0.94, indicating a high overlap between the predicted bounding boxes and the true bounding boxes, and the detection results were more accurate. In addition, the maximum IoU of YOLOv8, Faster R-CNN, YOLOv5-EfficientNet, and BiFPN-EfficientDet was 0.834, 0.846, 0.903, and 0.916, respectively, which were 0.133, 0.121, 0.064, and 0.051 lower than 0.967, respectively. As shown in Figure 11 (b), in terms of the mAP@0.5, the improved YOLOv8 performed better, with a maximum value of 0.971 and a minimum value of 0.952. In addition, the maximum mAP@0.5 values of these four comparison models were 0.823, 0.854, 0.912, and 0.923, which were 0.148, 0.117, 0.059, and 0.048 lower than 0.971, respectively. This indicates that the designed object detection model has higher accuracy, stronger generalization ability, and better model performance, which can effectively detect targets in basketball games. The CPU utilization and memory usage among different models are shown in Figure 12.

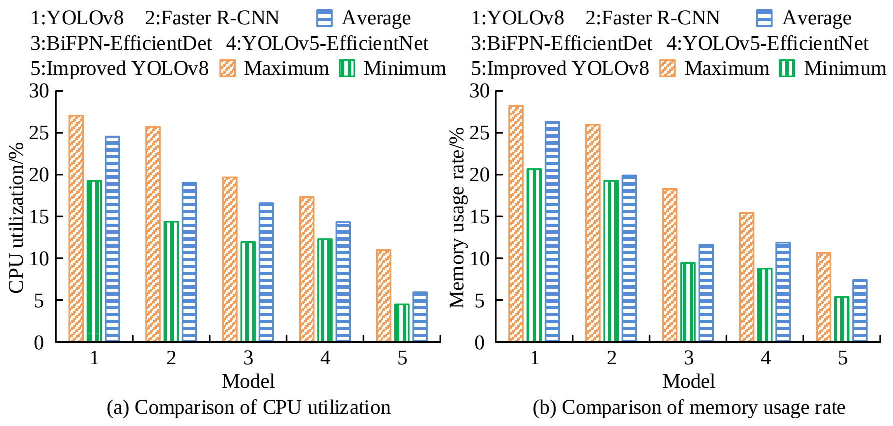


Figure 12: Comparison of CPU utilization and memory usage among different models

According to Figure 12 (a), the maximum values of the improved YOLOv8 and the four comparison models were 11.05%, 27.03%, 25.65%, 19.76%, and 17.43%, respectively. Compared with the maximum CPU utilization of the improved YOLOv8, the maximum CPU utilization of the four comparison models were 15.98%, 14.60%, 8.71%, and 6.38% higher, respectively. This indicates that the CPU utilization of the improved YOLOv8 is lower, which may be due to the IC2f module, which constrains the intervention of low order features and reduces the computational and memory burden. From Figure 12 (b), the memory usage of the improved YOLOv8 was still quite impressive. The maximum memory usage was 10.98%, and the minimum value was 5.76%. Meanwhile, the maximum memory usage rates of the four comparison models were 28.67%, 26.04%, 18.13%, and 15.78%, respectively, which were 17.69%, 15.06%, 7.15%, and 4.80% higher than 10.98%. This indicates that the designed object detection model has lower memory usage and better performance, making it more suitable for panoramic vision basketball robots. To further demonstrate the performance, it is applied to specific basketball games. The selected basketball game is a campus basketball game held by a certain university, with a total of 18 games and the detection target set as basketball. In basketball games, the average time consumption, missed detection rate, and false detection rate of different models are compared, as presented in Figure 13.

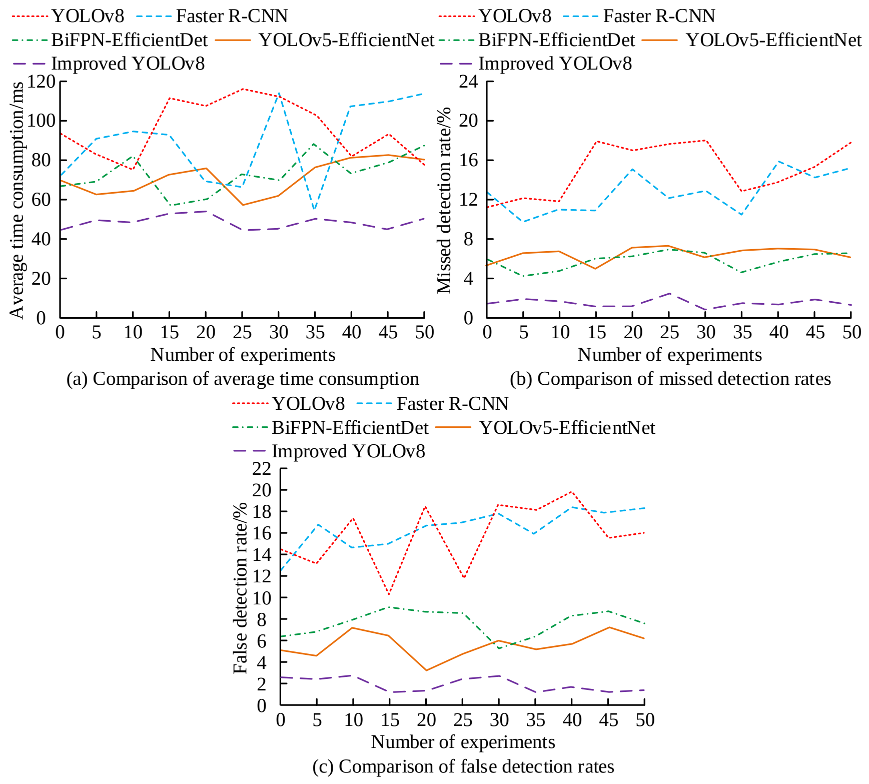


Figure 13: Comparison of average time consumption, missed detection rate, and false detection rate among different models

From Figure 13 (a), in the comparison of the average time consumption, the maximum average time consumption of the improved YOLOv8 was 55.17ms, and the minimum was 42.09ms. The maximum values of the four comparison models were 117.94ms, 115.32ms, 87.94ms, and 81.03ms, respectively, which were 62.77ms, 60.15ms, 32.77ms, and 25.86ms higher than 55.17ms. This indicates that the improved YOLOv8 has lower time consumption in practical applications and can detect basketball targets. In Figure 13 (b), the proposed object detection model performed better in the missed detection rate, with a maximum value of 2.21% and a minimum value of 1.03%. The performance of BiFPN-EfficientDet and YOLOv5-EfficientNet models was second, with maximum missed detection rates of 6.98% and 7.45%, respectively. The Faster R-CNN and YOLOv8 models performed at the bottom, with maximum missed detection rates of 15.98% and 18.04%, respectively. The designed object detection model can detect basketball more completely. From Figure 13 (c), the maximum false detection rates of the improved YOLOv8 and the four comparison models were 2.89%, 20.04%, 18.55%, 9.04%, and 7.33%, respectively. Compared with the maximum false detection rate of the improved YOLOv8, the maximum false detection rates of the four comparison models were 17.15%, 15.66%, 6.15%, and 4.44% higher, respectively. This indicates that the designed object detection model has better performance and higher detection accuracy in basketball detection.

3.2 Performance verification of target tracking algorithm

To verify the performance of the target tracking algorithm, the research also takes the Windows 10 operating system and Intel Core i5-12600KF processor. The Online object Tracking Benchmark (OTB) dataset is adopted. The OTB dataset contains multiple video sequences for target tracking, covering different scenarios and target characteristics. Therefore, the research selects data with obstruction, lighting changes, and background interference in this dataset. MeanShift algorithm, Camshift algorithm, CSK algorithm, M-KCF combining MeanShift algorithm and Kernelized Correlation Filter (KCF), and S-HOG combining Staple algorithm and Histogram of Oriented Gradient (HOG) features are taken for comparison. In addition, the threshold of Kalman filtering is 10, and the iteration is 300. Evaluation indicators include precision, central error, and average frame rate. Table 2 displays the ablation experimental results of target tracking algorithm.

Table 2: The ablation experimental results based on improved Camshift

Camshift	Extract fusion features	Template update judgment	Kalman	Having obstruction		Light changes		Background Interference	
				Precision/%	Recall/%	Precision/%	Recall/%	Precision/%	Recall/%
√	√	√	√	95.67	96.01	96.47	96.09	95.26	95.89
√	√	√	×	93.89	93.86	94.00	93.22	92.15	93.89
√	√	×	×	91.65	91.98	91.86	91.96	91.32	91.65
√	×	×	×	86.88	86.26	85.65	87.22	87.07	86.88

From Table 2, the target tracking algorithm combining Camshift, fusion feature extraction, template update judgment, and Kalman filtering algorithm had better performance. Its precision under obstruction, lighting changes, and background interference was 95.67%, 96.47%, and 95.26%, respectively, and the recall was 96.01%, 96.09%, and 95.89%, significantly better than those of other combinations. For example, when there was obstruction, the precision of Camshift, the model combined Camshift and extracted fusion features, as well as the model combined Camshift, extracted fusion features, and template update judgment was 86.88%, 91.65%, and 93.89%, respectively, which were 8.79%, 4.02%, and 1.78% lower than 95.67%, respectively. The target tracking algorithm has better performance, showing good performance on data with obstruction, lighting changes, and background interference. The central error of different models under different conditions is shown in Figure 14.

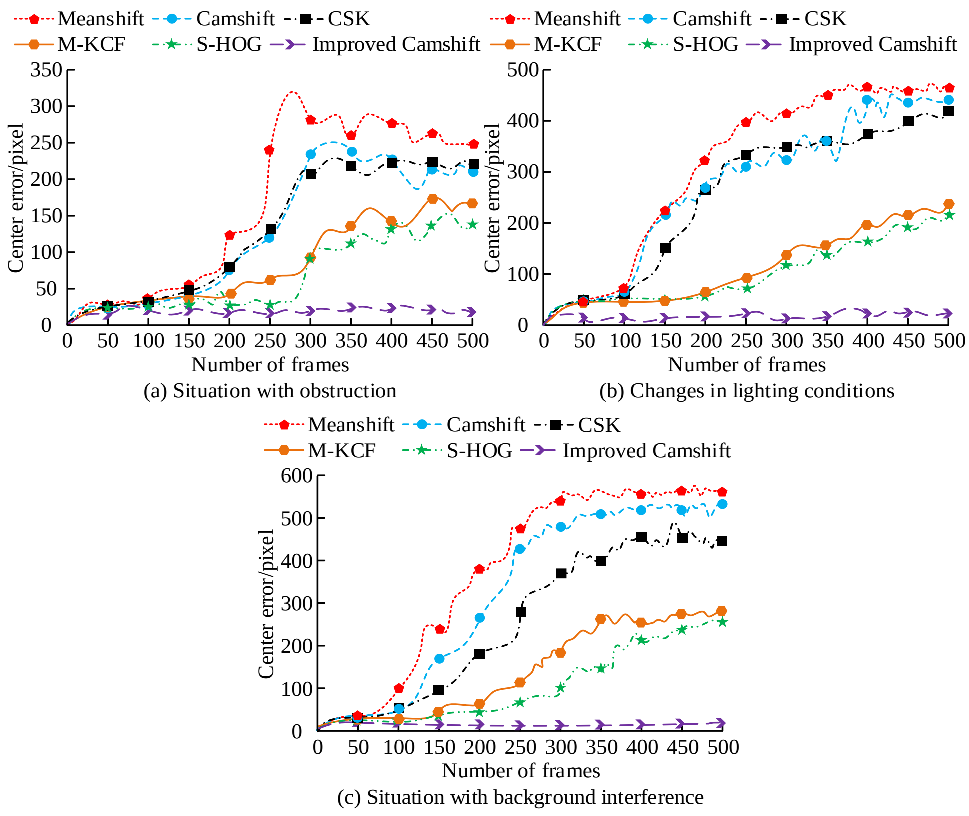


Figure 14: Comparison of central errors of different models under different conditions

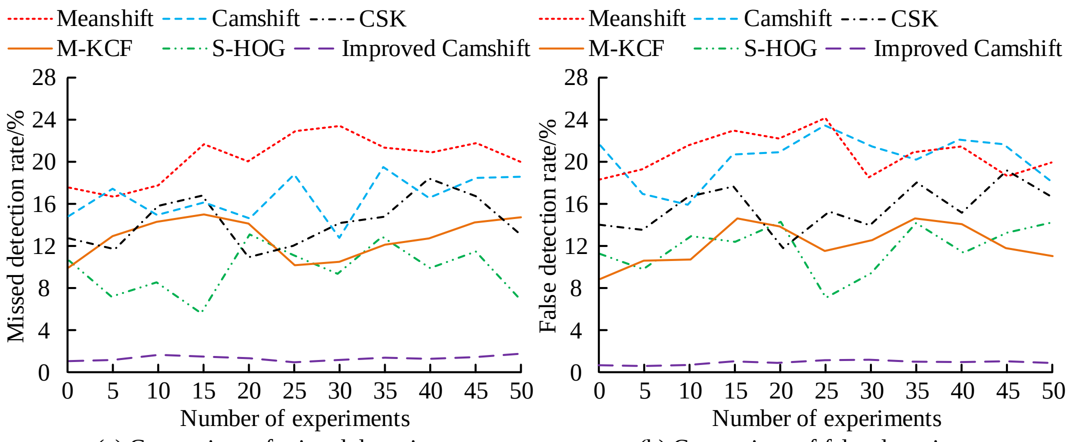
From Figure 14 (a), when there was obstruction, the maximum error value of the target tracking algorithm was 24 pixels. In addition, the maximum errors of MeanShift algorithm, Camshift algorithm, CSK algorithm, M-KCF algorithm, and S-HOG algorithm were 320 pixels, 250 pixels, 237 pixels, 175 pixels, and 153 pixels, respectively, which were 296 pixels, 226 pixels, 213 pixels, 151 pixels, and 129 pixels higher than 24 pixels. As the frame rate increased, the error variation of the improved Camshift was relatively stable and basically maintained within 25 pixels. The error trend of the five comparison methods increased with the increase of frame rates. From Figure 14 (b), when there was lighting changes, the target tracking algorithm designed in the research still had smaller errors, followed by the S-HOG algorithm and M-KCF algorithm, and finally the CSK algorithm, Camshift algorithm, and MeanShift algorithm. The maximum errors of the six algorithms were 21 pixels, 219 pixels, 234 pixels, 427 pixels, 453 pixels, and 479 pixels, respectively. From Figure 14 (c), the target tracking algorithm designed in this study still had lower errors in the background interference. The maximum error values of this algorithm and five comparison algorithms were 22 pixels, 267 pixels, 289 pixels, 497 pixels, 533 pixels, and 587 pixels, respectively. Compared with the maximum error value of the improved Camshift, the maximum error values of the five comparison algorithms were 245 pixels, 267 pixels, 475 pixels, 511 pixels, and 565 pixels higher, respectively. In summary, the improved Camshift has better performance and smaller central errors, and can more accurately track targets on the basketball court. The average frame rate comparison of different algorithms under different conditions is presented in Table 3.

Table 3: Comparison of average frame rates of different algorithms in different situations/Frame Per Second (FPS)

Algorithm	Situation with obstruction Number of experiments 1	2	3	4	5	6	7	8
MeanShift	28.21	25.15	25.40	29.77	26.34	29.33	29.38	28.50
Camshift	37.48	32.12	30.44	39.84	37.04	31.90	38.19	30.11
CSK	50.34	41.62	53.73	44.03	51.34	45.00	42.63	51.14
M-KCF	51.95	50.27	54.12	52.94	50.18	53.79	54.65	52.35
S-HOG	64.88	65.18	67.50	66.89	60.98	68.66	60.76	62.17
Improved Camshift	84.66	89.89	85.04	81.76	80.69	81.05	85.58	87.70

Algorithm	Changes in lighting conditions							
	Number of experiments							
	1	2	3	4	5	6	7	8
MeanShift	26.47	24.18	24.62	28.38	26.43	26.03	27.32	28.91
Camshift	36.23	37.14	34.65	42.71	37.37	38.07	41.80	42.01
CSK	52.48	55.83	56.66	53.68	55.05	56.77	56.56	56.41
M-KCF	50.54	44.50	44.69	50.68	45.73	46.85	52.77	49.29
S-HOG	70.69	70.44	65.07	71.94	64.28	68.68	64.68	70.87
Improved Camshift	85.87	84.59	89.78	80.61	80.10	82.15	80.30	80.63
Algorithm	Situation with background interference							
	Number of experiments							
	1	2	3	4	5	6	7	8
MeanShift	27.46	23.26	25.54	27.70	27.18	27.11	25.57	27.71
Camshift	56.91	57.08	51.80	56.53	54.17	54.30	54.74	52.63
CSK	56.91	57.08	51.80	56.53	54.17	54.30	54.74	52.63
M-KCF	43.08	44.48	43.51	44.12	52.79	52.07	48.29	50.03
S-HOG	71.87	70.27	69.93	73.22	67.30	69.28	68.29	71.51
Improved Camshift	87.76	93.02	88.95	80.11	89.01	88.93	94.41	85.16

From Table 3, in the case of obstruction, the maximum average frame rate of the improved Camshift was 89.89FPS, the minimum was 80.69FPS, and the average was 84.55FPS, which was significantly higher than those of the comparison algorithms. At this time, the average frame rates of MeanShift algorithm, Camshift algorithm, CSK algorithm, M-KCF algorithm, and S-HOG algorithm were 27.76FPS, 34.64FPS, 47.48FPS, 52.53FPS, and 64.63FPS, respectively, which were 56.79FPS, 49.91FPS, 37.07FPS, 32.02FPS, and 19.92FPS lower than 84.55FPS. In addition, the target tracking algorithm still performs better on average frame rate under varying lighting conditions and background interference. The average frame rate of the improved Camshift is higher, which can process more image frames per unit time and is more suitable for basketball sports scenarios that require fast response. To further verify the target tracking algorithm, it is also applied to a basketball campus game held in a certain university, and the detection target is also set as a basketball. In basketball games, the missed and false detection rates of different target tracking models are shown in Figure 15.



(a) Comparison of missed detection rates (b) Comparison of false detection rates

Figure 15: Comparison of missed detection rate and false detection rate of different target tracking models

From Figure 15 (a), in terms of the missed detection rates, the maximum value of the improved Camshift was 1.73%, and the minimum value was 1.09%. Meanwhile, the maximum missed detection rates of the five comparison algorithms were 23.74%, 19.56%, 18.54%, 14.98%, and 13.65%, respectively, which were 22.01%, 17.83%, 16.81%, 13.25%, and 11.92% higher than 1.73%. As shown in Figure 15 (b), the target tracking algorithm designed in the study still performed better in terms of false detection rate, with a maximum value of 1.56%. The maximum false detection rates of the five comparison algorithms were 24.09%, 23.41%, 19.32%, 15.09%, and 14.77%, respectively, all of which were higher than that of the improved Camshift. Overall, the designed target tracking algorithm performs better in actual basketball target tracking, with lower missed detection rate and false detection rate.

4 Discussion

To improve the target tracking performance of panoramic vision basketball robots, corresponding target detection algorithms and target tracking algorithms are designed. The detection model combined with YOLOv8, IC2f, and BiFPN had better performance. The time consumption of this model was only 54.21ms. The precision was 98.67%, which was 10.03%, 2.24%, and 3.65% higher than that of YOLOv8, the model combining YOLOv8 and IC2f, and the model combining YOLOv8 and BiFPN, respectively. This is because the IC2f module in the improved YOLOv8 can retain most high-order information and constrain the intervention of low order features, and BiFPN has strong feature fusion ability. This result also indicates that the algorithm has achieved a good balance between accuracy and real-time performance, providing reliable technical support for real-time object detection of basketball robots. When there was obstruction, the maximum average frame rate of the improved Camshift was 89.89FPS, while the average frame rates of the five comparison algorithms were 27.76FPS, 34.64FPS, 47.48FPS, 52.53FPS, and 64.63FPS, respectively. The designed tracking algorithm still has higher average frame rate when there are changes in lighting and background interference. This may be because the algorithm obtains more complete color features through fusion feature extraction, and avoids the continuous accumulation of bias in obstruction situations by updating the judgment based on the target template. This also exhibits the stability and robustness in dynamic environments.

Compared with existing research, the improved YOLOv8 object detection and Camshift object tracking algorithms designed in this study have good performance. Li Z et al. designed an improved YOLOv8 algorithm based on BiFPN and MobileNetv3 to detect coal gangue, achieving a precision rate of 99.9% [36]. This is similar to the introduced BiFPN in YOLOv8. Fiyad H M N et al. designed an improved method based on CamShift and Kalman filtering algorithm to achieve real-time single target tracking, achieving high tracking accuracy on both static and moving objects [37]. This is similar to the Kalman filtering algorithm used in CamShift.

Although the study has achieved good experimental results, there are still some limitations. Firstly, although the improved YOLOv8 performs well in feature extraction and multi-scale feature fusion, the detection accuracy may decrease when dealing with extreme lighting changes or highly dynamic scenarios. Future research can add data samples with extreme lighting changes and highly dynamic environments to the training data to learn more diverse features, or explore new feature extraction methods and network structures. Secondly, the improved Camshift algorithm still needs further improvement in tracking stability when facing targets that move rapidly or have significant shape changes. Future research could consider integrating more feature information and further optimizing the update mechanism of target templates, as well as combining deep learning methods to improve existing algorithms. In addition, the experiments are mainly based on specific basketball game scenarios, and the generalization ability of the algorithm has not been fully validated in more sports scenarios. Future research can conduct experimental verification in other sports scenarios and apply the algorithm to object detection and tracking tasks in other non-sports fields to further validate the generalization ability and generality.

5 Conclusion

To enhance the target tracking performance of panoramic vision basketball robots, a target detection algorithm based on improved YOLOv8 and a target tracking algorithm based on improved Camshift were designed, and good performance was achieved. In the improved YOLOv8 object detection algorithm, IC2f module and BiFPN structure were introduced in the research, which increased its detection accuracy to 98.67%. Compared with the YOLOv8 model that did not introduce these modules, its accuracy improved by 10.03%. This indicates that the IC2f module and BiFPN structure can improve detection accuracy. In the improved Camshift target tracking algorithm, due to the introduced fusion feature extraction, template update judgment, and Kalman filter, the algorithm achieved the highest accuracy of 95.67%, 96.47%, and 95.26% in occlusion, lighting changes, and background interference, which was significantly higher than other comparison algorithms and Camshift algorithms without these modules. The designed object detection algorithm and tracking algorithm can provide good technical support for the practical application of panoramic vision basketball robots.

References

[1] M. G. Majeed, W. Hameed, N. H. Haroon, S. R. Abdul Kadeem, H. M. Salman, and S. Kadry. “Optimizing Resource Management in Physical Education through Intelligent 5G-Enabled Robotic Systems,” Fusion: Pract. Appl., vol. 13, no. 1, pp. 162-174, May, 2023, DOI: 10.54216/FPA.130113.

[2] L. Xu. “Application analysis of sports robots based on pose recognition and action feature analysis,” Int. J. Syst. Assurance Eng. Manag., vol. 14, no. 2, pp. 519-528, April, 2023, DOI: 10.1007/s13198-021-01245-1.

[3] M. M. Sadr, T. Saheb, and A. Farahani. “A Mapping and Visualization of the Role of Artificial Intelligence in Sport Industry,” Res. Sport Manag. Mark., vol. 5, no. 1, pp. 44-56, January, 2023, DOI: 10.22098/rsmm.2023.13064.1241.

[4] W. Chen and X. Huang. “RETRACTED ARTICLE: Research on reliability of sports intelligent training system based on hybrid wolf pack algorithm and IoT,”

Soft Comput., vol. 27, no. 14, pp. 10189-10197, April, 2023, DOI: 10.1007/s00500-023-08247-0.

[5] M. A. Arif, A. Zhu, H. Mao, and Y. Tu. “Panoramic visual system for spherical mobile robots,” Robotica, vol. 42, no. 7, pp. 2089-2107, February, 2024, DOI: 10.1017/S0263574724000043.

[6] C. Zha, S. Luo, and X. Xu. “Infrared multi-target detection and tracking in dense urban traffic scenes,” IET Image Process., vol. 18, no. 6, pp. 1613-1628, February, 2024, DOI: 10.1049/ipr2.13053.

[7] M. Zhai, H. Zhang, L. Wang, D. Xiao, Z. Gu, and Z. Li. “Special Vehicle Target Detection and Tracking Based on Virtual Simulation Environment and YOLOv5-Block+ DeepSort Algorithm,” Comput. Mater. Continua, vol. 81, no. 2, pp. 3241-3260, April, 2024, DOI: 10.32604/cmc.2024.056241.

[8] Y. Wu, H. Deng, H. Liu, X. Zhao, and Y. Fang. “NKhex: A New Miniature Hexapod Crawling Robot with Visual Perception and Target Tracking,” IEEE Trans. Ind. Electron., vol. 72, no. 1, pp. 693-702, January, 2025, DOI: 10.1109/TIE.2024.3409901.

[9] M. Alymani, M. E. Karar, and H. I. Shehata. “A Practical Study of Intelligent Image-Based Mobile Robot for Tracking Colored Objects,” Comput. Mater. Continua, vol. 80, no. 8, pp. 2181-2197, August, 2024, DOI: 10.32604/cmc.2024.052406.

[10] S. Kong, J. Sun, A. Luo, W. Chi, C. Zhang, S. Zhang, and J.Yu. “A Collision-Free Target Tracking Controller with Uncertain Disturbance Rejection for Quadruped Robots,” IEEE Trans. Intell. Vehicles, vol. 9, no. 1, pp. 670-680, January, 2024, DOI: 10.1109/TIV.2023.3296669.

[11] S. L. Dai, J. Liang, K. Lu, and X. Jin. “Adaptive image-based moving-target tracking control of wheeled mobile robots with visibility maintenance and obstacle avoidance,” IEEE Trans. Control Syst. Technol., vol. 32, no. 2, pp. 488-501, March, 2024, DOI: 10.1109/TCST.2023.3331553.

[12] J. Yan, J. Lin, X. Yang, C. Chen, and X. Guan. “Cooperation detection and tracking of underwater target via Aerial-Surface-Underwater vehicles,” IEEE Trans. Autom. Control, vol. 70, no. 2, pp. 1068-1083, February, 2025, DOI: 10.1109/TAC.2024.3447976.

[13] Y. Liu, B. An, S. Chen, and D. Zhao. “Multi-target detection and tracking of shallow marine organisms based on improved YOLO v5 and DeepSORT,” IET Image Process., vol. 18, no. 9, pp. 2273-2290, April, 2024, DOI: 10.1049/ipr2.13090.

[14] S. Pal, A. Roy, P. Shivakumara, and U. Pal. “Adapting a Swin Transformer for License Plate Number and Text Detection in Drone Images,” Artif. Intell. Appl., vol. 1, no. 3, pp. 145-154, April, 2023, DOI: 10.47852/bonviewAIA3202549.

[15] S. Li, M. Wang, Y. Zhou, Q. Su, L. Liu, and T. Wu. “IG-YOLOv8: Insulator Guardian Based on YOLO for Insulator Fault Detection,” IEEJ Trans. Electr. Electron. Eng., vol. 20, no. 4, pp. 537-547, November, 2025, DOI: 10.1002/tee.24221.

[16] Y. Chen, Q. Liu, X. Jiang, Y. Wei, X. Zhou, J. Zhou, and H. Xing. "FEW-YOLO: a lightweight ripe fruit detection algorithm in wolfberry based on improved YOLOv8: Chen et al," J. Food Meas. Charact., vol. 19, no. 7, pp. 4783-4795, May, 2025, DOI: 10.1007/s11694-025-03290-x.

[17] H. Liu, W. Gu, W. Wang, Y. Zou, H. Yang, and T. Li. "Persimmon fruit detection in complex scenes based on PerD-YOLOv8: Liu et al," J. Food Meas. Charact., vol. 19, no. 7, pp. 4543-4560, May, 2025, DOI: 10.1007/s11694-025-03268-9.

[18] H. Zhang, G. Li, D. Wan, Z. Wang, J. Dong, S. Lin, and H. Liu. “DS-YOLO: A dense small object detection algorithm based on inverted bottleneck and multi-scale fusion network,” Biomimetic Intelligence and Robotics, vol. 4, no. 4, pp. 64-75, December, 2024, DOI: 10.1016/j.birob.2024.100190.

[19] Q. Zhang, C. Wang, H. Li, S. Shen, W. Cao, X. Li, and D. Wang. “Improved YOLOv8-CR Network for Detecting Defects of the Automotive MEMS Pressure Sensors,” IEEE Sens. J., vol. 24, no. 16, pp. 26935-26945, August, 2024, DOI: 10.1109/JSEN.2024.3419806.

[20] P. I. Jun, N. Hou, and G. Zhi. “Lightweight human pose estimation algorithm by integrating CA and BiFPN,” J. Graphics, vol. 44, no. 5, pp. 868-878, December, 2023, DOI: 10.11996/JG.j.2095-302X.2023050868.

[21] S. Liu, Y. Yang, T. Cao, and Y. Zhu. “A BiFPN-SECA detection network for foreign objects on top of railway freight vehicles,” Signal, Image and Video Processing, vol. 18, no. 12, pp. 9027-9035, September, 2024, DOI: 10.1007/s11760-024-03527-0.

[22] Y. Peng, K. Xu, Z. Heng, C. Feng, W. Wen, and W. Ruo. "Recognition and location of coal gangue based on BiFPN and ECA attention mechanism," Int. J. Coal Prep. Util., vol. 44, no. 8, pp. 1173-1185, October, 2024, DOI: 10.1080/19392699.2023.2270920.

[23] P. Yu, Y. Lin, Y. Lai, S. Cheng, and P. Lin. “Dense log end face detection method using the hybrid of BiFPN and YOLOv5s,” J. Forestry Eng., vol. 8, no. 1, pp. 126-134, November, 2023, DOI: 10.13360/j.issn.2096-1359.202204006.

[24] D. Y. Bernanda, D. N. A. Jawawi, S. Abd Halim, and F. Adikara. “Natural Language Processing for Requirement Elicitation in University Using Kmeans and Meanshift Algorithm,” Baghdad Science Journal, vol. 21, no. 2, pp. 561-567, May, 2024, DOI: 10.21123/bsj.2024.9675.

[25] Y. Zhang and Y. C. Chen. “Linear convergence of the subspace constrained mean shift algorithm: from Euclidean to directional data,” Inf. Inference: A Journal of the IMA, vol. 12, no. 1, pp. 210-311, June, 2023, DOI: 10.1093/imaiai/iaac005.

[26] S. Devaraj and B. Sridharan. “Deep Perona-Malik Diffusive Mean Shift Image Classification for Early Glaucoma and Stargardt Disease Detection,” Malaysian J. Comput. Sci., vol. 36, no. 1, pp. 14-39,

January, 2023, DOI: 10.22452/mjcs.vol36no1.2.

[27] D. Xiong, and F. Xu. "A robust diagnostic approach in mean shifts for multivariate statistical process control," J. Stat. Comput. Simulation, vol. 95, no. 3, pp. 507-524, November, 2025, DOI: 10.1080/00949655.2024.2431856.

[28] J. Wang, Z. Jia, H. Lai, and F. Shi. "A real time target face tracking algorithm based on saliency detection and Camshift," MULTIMED TOOLS APPL., vol. 82, no. 28, pp. 43599-43624, April, 2023, DOI: 10.1007/s11042-023-14889-x.

[29] L. S. K. Vatsavai, and K. S. V. Mantena. "Camshift Algorithm with GOA-Neural Network for Drone Object Tracking," ISI., vol. 28, no. 2, pp. 491-498, April, 2023, DOI: 10.18280/isi.280226.

[30] P. D. Barua, T. Keles, M. Kuluozturk, M. A. Kobat, S. Dogan, M. Baygin, and U. R. Acharya. "Automated asthma detection in a 1326-subject cohort using a one-dimensional attractive-and-repulsive center-symmetric local binary pattern technique with cough sounds," Neural Comput. Appl., vol. 36, no. 27, pp. 16857-16871, June, 2024, DOI: 10.1007/s00521-024-09895-5.

[31] J. C. Sekhar, P. J. Josephson, A. Chinnasamy, M. Maheswari, S. Sankar, and R. R. Kalangi. "Automated face recognition using deep learning technique and center symmetric multivariant local binary pattern," Neural Comput. Appl., vol. 37, no. 1, pp. 263-281, November, 2025, DOI: 10.1007/s00521-024-10447-0.

[32] C. Zhiyuan, Z. Zhichao, and S. U. I. Qingyang. "Online identification of key parameters of secondary edges in underwater WPT system based on unscented Kalman filtering algorithm," Proc. CSEE, vol. 44, no. 11, pp. 4470-4479, May, 2024, DOI: 10.13334/j.0258-8013.pcsee.230708.

[33] C. Zhu, S. Wang, C. Yu, H. Zhou, and C. Fernandez. "An improved proportional control forgetting factor recursive least square-Monte Carlo adaptive extended Kalman filtering algorithm for high-precision state-of-charge estimation of lithium-ion batteries," J. Solid State Electrochem., vol. 27, no. 9, pp. 2277-2287, May, 2023, DOI: 10.1007/s10008-023-05514-w.

[34] B. Han. "Distributionally robust Kalman filtering with volatility uncertainty," IEEE Trans. Autom. Control, vol. 70, no. 6, pp. 4000-4007, June, 2025, DOI: 10.1109/TAC.2024.3522192.

[35] Y. Fu and S. Gao. "Research on Lightweight Model of Multi-person Pose Estimation Based on Improved YOLOv8s-Pose," J. Front. Comput. Sci. Technol., vol. 19, no. 3, pp. 682-692, July, 2025, DOI: 10.3778/j.issn.1673-9418.2403059.

[36] Z. Li, S. Fu, L. Xu, M. Bo, Z. Xin, and J. Qing. "Research on gangue target detection algorithm based on MBI-YOLOv8," J. Graphics., vol. 45, no. 6, pp. 1301-1312, May, 2024, DOI: 10.11996/JG.j.2095-302X.2024061301.

[37] H. M. N. Fiyad, H. M. B. Metwally, M. A. El-Hameed, and M. A. H. Abozied. "Improved Real Time Target Tracking System Based on Cam-Shift and Kalman Filtering Techniques," J. Appl. Res. Technol., vol. 21, no. 2, pp. 297-308, June, 2023, DOI: 10.22201/icat.24486736E.2023.21.2.1565.