

ORIGINAL RESEARCH

Open access

An Efficient Multi-Object Tracking Framework for Calves Based on YOLOv8s and Adaptive Rematching

Mikko Lahtinen^{1*}, Elina Salo¹, Juhani Virtanen²

Abstract

Health surveillance depends fundamentally on the reliable tracking of individual calves. Yet, prevailing multi-object tracking frameworks are prone to recurrent identification anomalies, particularly ID switches arising under occluded conditions. To overcome this limitation, we introduce YSD-BPTrack in this work, a novel multi-object tracking framework tailored for calves within occlusion-prone farm environments. The architecture is divided into two principal stages: detection and tracking. For the detection stage, DCNv4 is embedded into the YOLOv8s model to learn spatial deformations caused by occlusion, thereby strengthening detection robustness under such conditions. Meanwhile, the Star operation borrowed from StarNet is adopted to deliver high detection capability while keeping computational demands low. In the tracking stage, we first devise an original rematching mechanism (Rematching module) alongside a redesigned trajectory removal policy (Trajectory removal module). The Rematching module exploits expanded trajectory prediction boxes to re-establish correspondences with detection boxes during occlusion events, thereby cutting down on ID switch errors. The Trajectory Removal module, in turn, flexibly governs the timeout period for lost trajectories, minimizing the risk of premature trajectory deletion. Concretely, the framework we propose sets a new HOTA (Higher Order Tracking Accuracy) score of 91.6%, leading competing approaches in both tracking quality and runtime performance. Experimental evidence further demonstrates the advantage of YSD-BPTrack, with HOTA climbing by 17.6%, MOTA (Multiple Object Tracking Accuracy) by 13.9%, MOTP (Multiple Object Tracking Precision) by 1.8%, IDF1 (Identification F1 Score) by 15.4%, accompanied by reductions of 49.1% in parameters, 88.9% in IDSw (Identification Switches), and 39.2% in computational overhead relative to the other frameworks. Collectively, the presented multi-object tracking framework carries strong potential to redefine calf tracking practice.

Keywords Calves, Multi-object tracking, Occluded environments, Rematching algorithm

*Correspondence:

Mikko Lahtinen

mikko.lahtinen@helsinki.fi

¹ Department of Precision Livestock Tracking, Faculty of Agriculture, University of Helsinki, Helsinki, Finland

² Department of Multi-Object Tracking for Animal Monitoring, Faculty of Agriculture, University of Eastern Finland, Kuopio, Finland

Introduction

Cattle monitoring and tracking represent core activities within modern precision livestock management, underpinning operational efficiency, health oversight, and the collective welfare of the herd [1]. Calves, serving as a critical cohort within the herd's demographic makeup, exert direct influence on the overall yield and financial returns realized by the farmer [2]. Consequently, meticulous surveillance and tracing of their health and development are indispensable for sound farm administration. Tracking and observing calves permits the examination of their behavior, enabling timely identification of nascent health disorders and developmental anomalies in specific individuals. This paves the way for early remedial action, curbing both the propagation of illness and associated treatment expenditures. In addition, the precise documentation of behavioral rhythms offers livestock managers a key barometer of animal health status.

Nevertheless, conventional manual surveillance practices for gathering cattle information prove to be both laborious and slow, while also suffering from poor efficiency [3]. Alternatively, reliance on contact-type wearable sensors to log cattle activity can deliver measurements of moderate precision [4, 5]. Such instruments, affixed to the animal's body in some permanent fashion, inevitably impose physical irritation and generate long-term animal welfare reservations. Furthermore, under extensive farming operations, the steep investment in hardware plus vulnerabilities connected to equipment failure must be weighed [6].

As computing technology advances steadily, non-invasive cattle monitoring strategies grounded in deep learning and computer vision have surfaced as viable alternatives. Andrew *et al.* [7] and Xu *et al.* [8] harnessed Faster-RCNN [9] and Mask-RCNN [10] architectures to label and train overhead imagery datasets of Holstein cows, reporting detection accuracies peaking at 99.6% and 96.0%, respectively. That said, when compared against two-stage detectors like Faster-RCNN and Mask-RCNN, single-stage detectors not only streamline end-to-end implementation but also furnish higher inference speeds. The YOLO lineage stands as the exemplar of this single-stage advantage.

Jiang *et al.* [11] leveraged a refined YOLOv3 [12] model to deliver 93.7% detection accuracy on anatomically key cow regions captured in convoluted settings. Qiao *et al.* [13] deployed an improved YOLOv5 variant, recording 96.2% individual detection accuracy when evaluated on a bovine dataset. Shen *et al.* applied an enhanced YOLOv7 [14] model, attaining 93.0% accuracy in classifying cattle behavior inside densely populated datasets.

Despite the high detection and behavioral analysis accuracies reported in the aforementioned research, these efforts remain oriented toward single-frame detection and spatial localization, rather than toward maintaining persistent individual identity through time [15]. Continuous identity tracking is fundamental for analyzing and monitoring individual behavior.

Mar *et al.* [16] later engineered a multi-object tracking (MOT) solution that draws upon positional and visual signatures (e.g., color and CNN-derived features) to enable cow tracking. Huang *et al.* [17] outlined an optimized framework for cow tail detection and tracking built upon a refined SSD [18] architecture fused with a Kalman filter, reporting 96.97% tracking accuracy through improved Kalman filtering and Hungarian algorithm matching. Aye *et al.* [19] contributed a livestock tracking pipeline that combines YOLOv5 for black cattle detection with DeepSort [20] for real-time tracking. Su *et al.* [21] formulated a multi-object tracking method for black cattle centered on YOLOv7 and StrongSort [22]. Although these systems realize multi-object tracking of cattle, both DeepSort and StrongSORT embed Re-ID (Re-Identification) components that impose additional runtime and resource loads, tending to reduce processing speed and constrain real-time viability.

Zheng and Qin [23] put forward ByteTrack [24], likewise a multi-object cattle tracking algorithm paired with YOLOv7 but deliberately omitting the Re-ID module. Their experimental configuration achieved a throughput of 47 frames per second (FPS) and a HOTA (Higher Order Tracking Accuracy) metric of 67.6% [25]. While the removal of the Re-ID module in ByteTrack lowers hardware deployment costs and accommodates real-time detection requirements in livestock operations, difficulties such as ID swaps triggered by occlusions and missed detections within complex environments continue to pose problems.

YOLOv8 [26] delivers notable improvements over earlier versions, encompassing a more elaborate network design, refined training recipes, and advanced feature extraction backbones, thereby considerably lifting model capacity and ease of use, and securing gains in both detection accuracy and speed.

StarNet [27] makes use of the Star operation, also termed element-wise multiplication, where matching elements from two arrays (or vectors, matrices, etc.) are multiplied together. This operation implicitly maps the initial input representation from a lower-dimensional feature space into a higher-dimensional, nonlinear feature space while preserving computational parsimony and avoiding increased complexity. As a result, the network upholds computational efficiency yet acquires richer and more expressive feature representations.

DCNv4 (Deformable Convolution version 4) [28] belongs to a family of convolutional operations in deep learning that refines spatial sampling positions by learning supplementary offsets. This design bolsters a convolutional neural network's ability to capture geometric transformations, thereby sharpening feature extraction for objects subject to deformation or scale variation introduced by occlusion. Beyond this, DCNv4 removes softmax normalization to make spatial aggregation more dynamic and expressive. The resulting flexibility in fusing spatial information allows the network to harvest richer feature representations and to reliably pull out salient object cues across complex scenes, crowded surroundings, and occluded settings.

Drawing upon the merits of the methods described above, we put forward in this paper a multi-object tracking framework named YSD-BPTrack, purpose-built for calves. The core contributions of this work are summarized as follows.

- A novel multi-object tracking framework for calves, termed YSD-BPTrack, is introduced to counteract occlusion-driven abnormal ID switches and raise tracking fidelity. The refined design spanning both detection and tracking stages further empowers the framework to achieve high-precision tracking under limited resource budgets.
- To establish a dependable detection base for downstream tracking, we first embed the DCNv4 Head dynamic detection head into YOLOv8 during the detection stage. This substantially improves detection performance on calves undergoing occlusion, deformation, or scale shifts. Moreover, the adoption of StarNet secures standout detection capability while incurring lower computational costs. Quantitatively, the enhanced model retains 99.2% mAPval50 compared to YOLOv8, while parameter count drops by 49.1% and computational overhead shrinks by 39.2%.
- To sharpen tracking accuracy and cut down on occlusion-related abnormal ID switches, we first refine the KF trajectory prediction box for better trajectory prediction quality. In parallel, we optimize the trajectory-to-detection matching scheme and tailor the timeout interval for lost matching trajectories to suit cattle yard conditions, thereby suppressing trajectory matching errors. Notably, relative to the other frameworks, YSD-BPTrack raises HOTA by 16.5%, MOTA by 13.9%, MOTP by 1.8%, and IDF1 by 14.0%, while decreasing IDSw by 77.8%.

Materials and Methods

Dataset

Video material was gathered between July and October 2023 at the National Jinnan Cattle Genetic Resources and Gene Protection Center in Yuncheng City, China. Recordings were made with Hikvision cameras installed in the corners of the Jinan cattle activity zone. The cameras were attached to support pillars roughly 2.5 meters above ground inside the cattle shed, faced diagonally downward, and captured the daily activity of the animals under observation at the Jinan cattle farm without interruption. Collection ran from 0:00 to 24:00 and included sunny, overcast, and rainy weather. The ratio of sunny, cloudy, and rainy conditions across the datasets stood at 6:3:1. The footage covered cattle of multiple ages, from neonatal calves up to six months of age through to adult animals. All sample sequences were filmed in natural settings, as shown in **Figure 1**.



Figure 1. Example of data collection in the cattle farms.

Our dataset is partitioned into tracking and detection portions. For the tracking portion, 9 video clips, each lasting 60 seconds, were selected. These clips span assorted calf behaviors and diverse occlusion intensities. **Table 1** provides detailed information regarding weather, level of obstruction, and behavior. Additionally, examples of these calf behaviors appear in **Figure 2**. The DarkLabel tool was employed to annotate individual-specific data per calf, including ID, bounding box center coordinates, width, height, and related attributes. The annotation followed the MOT17 format and was ultimately stored in a txt file. The annotated tracking portion is depicted in **Figure 3b**.

Table 1. Composition of the calf tracking dataset.

No	Weather	Occlusion	Behavior
01	Rainy	High	Walking, Running, Standing
02	Sunny	Normal	Walking, Running
03	Sunny	Low	Walking
04	Cloudy	Normal	Walking, Running
05	Sunny	Normal	Running, Standing
06	Cloudy	High	Nursing, Walking
07	Sunny	High	Running, Lying
08	Cloudy	Normal	Diarrhea
09	Cloudy	Normal	Diarrhea

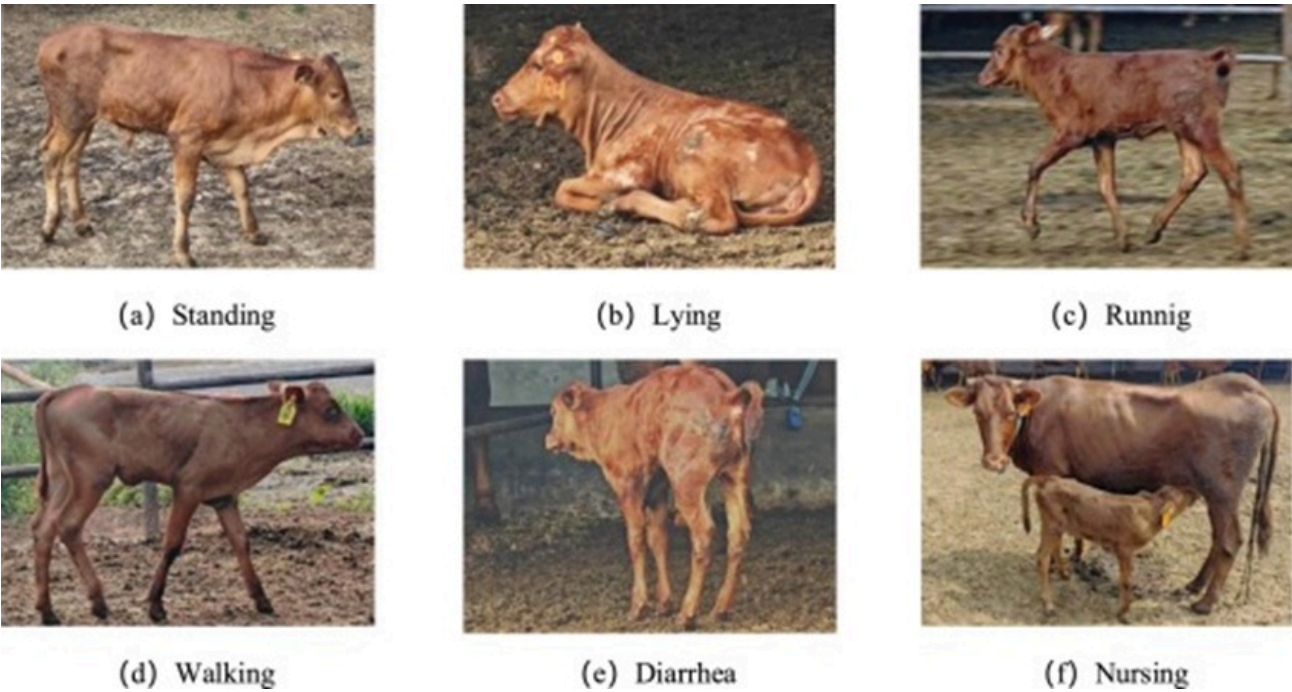


Figure 2. Some examples of calf behaviors.

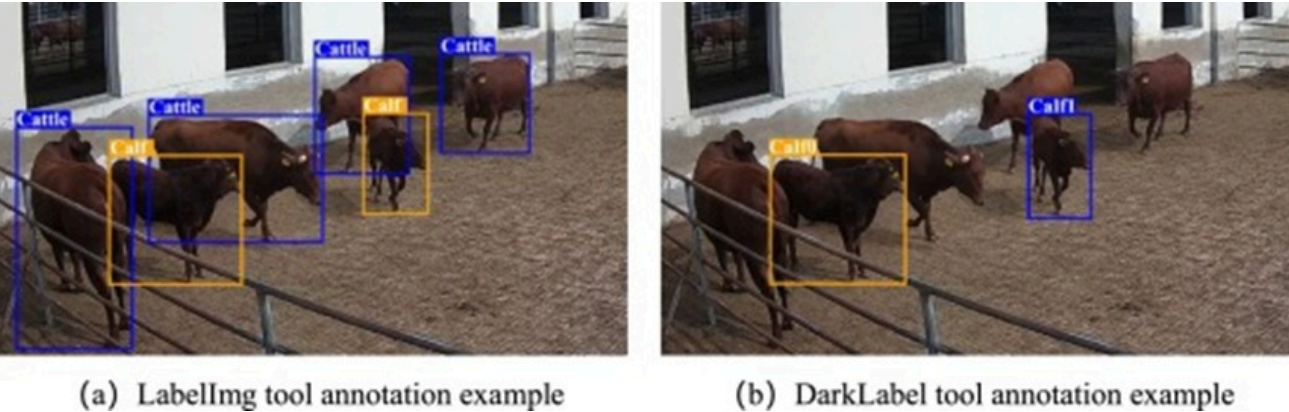


Figure 3. Some examples of annotated images.

To assemble an effective detection dataset, highly repetitive and blurry frames were first manually stripped from the sequentially captured videos, leaving a total of 4,809 images. Both calves and adult cattle were then annotated using the Labelling tool, following the annotation convention shown in Figure 3a. The detection dataset was subsequently divided into training, validation, and test portions in a 7:2:1 proportion. Of these images, 3,393 served for training, 944 for validation, and the remainder were set aside for testing.

Flowchart of the proposed approach

The pipeline of the proposed multi-object tracking framework for calves is laid out in Figure 4. Following dataset compilation, the continuously acquired videos are preprocessed, and segments where calves encounter varying levels of occlusion are selected. These segments are then annotated to record the ID and bounding box coordinates of each calf throughout the video sequence. The annotated data are next fed into the trained YOLOv8s-SD detection model to produce detection bounding boxes. Those detection outputs are subsequently handed over to our proposed BytePro for further steps, including object prediction, association, and tracking, ultimately yielding the final tracking outputs. The result is an efficient multi-object tracking framework suited for fast and accurate calf tracking.

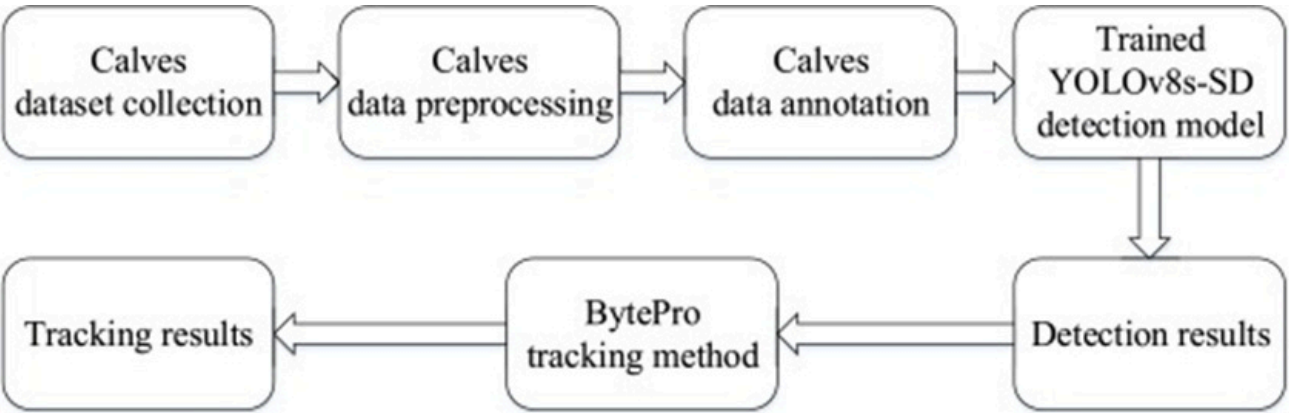
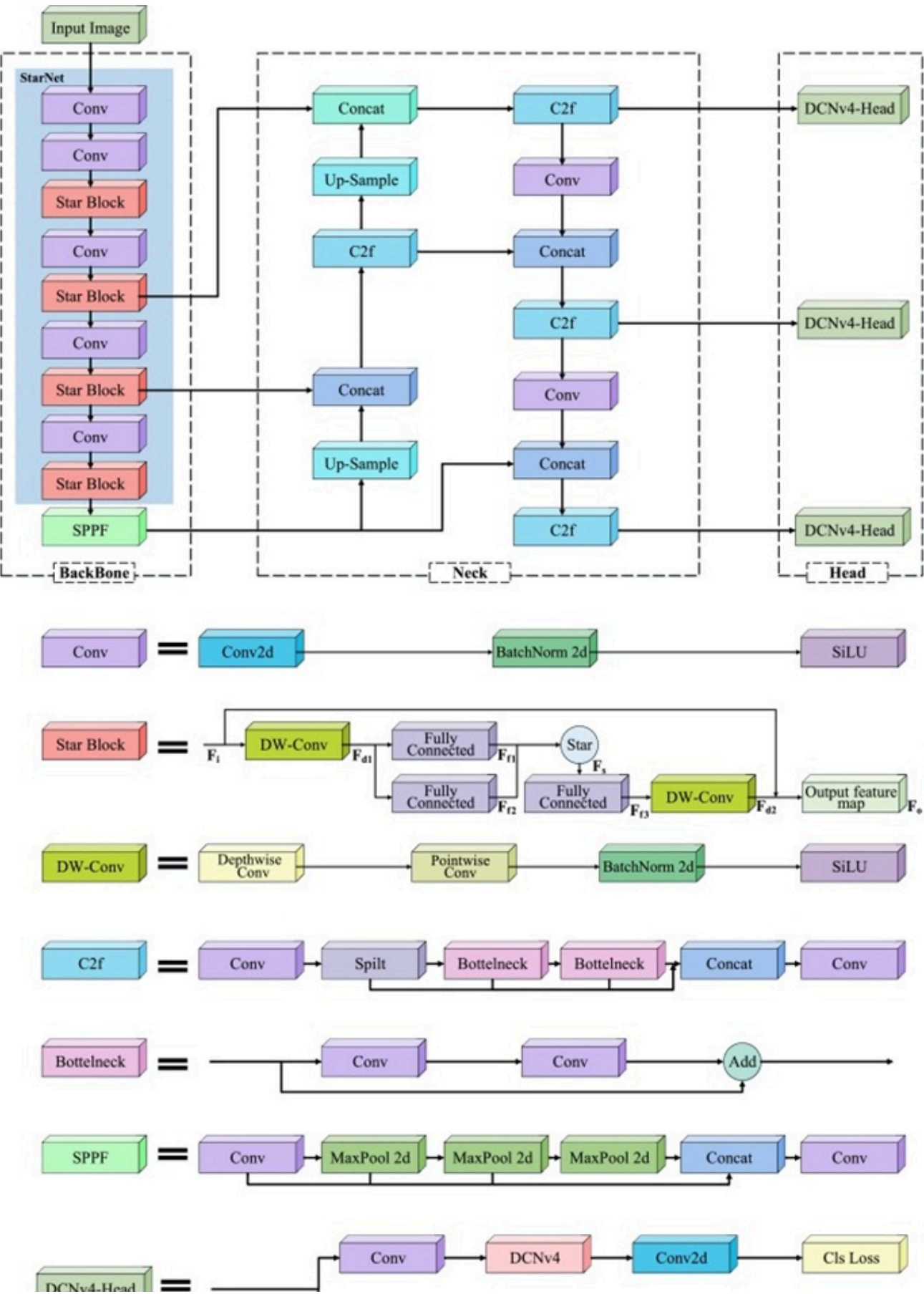
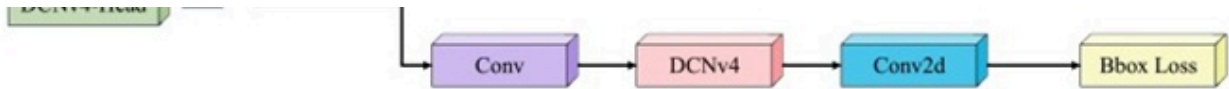


Figure 4. The flowchart of the proposed YSD-BPTrack framework.

YOLOv8s-SD: the object detection for calves

The detection module of YSD-BPTrack rests upon YOLOv8s and incorporates StarNet along with DCNv4 units into the original design. This updated detection module is designated YOLOv8s-SD for clarity. The detailed architecture of YOLOv8s-SD is shown in Figure 5, comprising three core sections: backbone, neck, and head.





BACKBONE

The backbone of YOLOv8s-SD relies on StarNet [27], which follows a 4-stage hierarchical structure. Every stage houses a Conv unit together with a Star Block.

In detail, the Conv unit brings together a Conv2d layer, a BatchNorm2d layer, and a Sigmoid Linear Unit (SiLU) activation function.

As for the processing sequence inside the Star Block of StarNet, the incoming feature map F_i first travels through a DW-Conv unit to produce an intermediate feature map F_{d1} . F_{d1} is then routed along two independent Fully Connected layers to yield feature maps F_{f1} and F_{f2} . These two maps, F_{f1} and F_{f2} , undergo the Star operation to form feature map F_s . Afterwards, F_s advances through yet another Fully Connected layer to produce feature map F_{f3} . F_{f3} is then processed by a DW-Conv unit to obtain feature map F_{d2} . Finally, a residual link between F_{d2} and the original feature map F_i generates the output feature map, F_o .

Turning to the Star operation within the Star Block, it blends features originating from different subspaces through element-wise multiplication. In essence, the Star operation is capable of projecting inputs into an exceptionally high-dimensional, nonlinear feature space. A closer examination of this operation is presented below.

The Star operation is commonly expressed as $(W_1^T X + B1) * (W_2^T X + B2)$, with $W1$ and $W2$ as weight matrices, X the input feature vector, and $B1$, $B2$ as bias vectors. To lighten the computation, the weight matrix and bias can be folded into a single composite, giving $W' = \begin{bmatrix} W \\ B \end{bmatrix}$. In a similar vein, the linear transformation can be absorbed into matrix operations by expanding the input feature vector X to $X' = \begin{bmatrix} X \\ 1 \end{bmatrix}$. Accordingly, the compact Star operation becomes $(W_1'^T X') * (W_2'^T X')$.

Eq. 1 lays out the simplified expanded computation.

$$\begin{aligned} (W_1'^T X') * (W_2'^T X') &= \left(\sum_{i=1}^{d+1} w_1^i x^i \right) * \left(\sum_{j=1}^{d+1} w_2^j x^j \right) \\ &= \sum_{i=1}^{d+1} \sum_{j=1}^{d+1} w_1^i w_2^j x^i x^j \\ &= a_{(1,1)} x^1 x^1 + \dots + a_{(4,5)} x^4 x^5 + \dots + a_{(d+1,d+1)} x^{d+1} x^{d+1} \end{aligned} \quad (1)$$

Where $x \in (d + 1) \times 1$ dimensions, d represents the number of input channels, and i, j index the channels, and $a \in \frac{(d+2)(d+1)}{2}$ dimensions indicates the coefficient for each corresponding item. Eq. 2 shows the specific calculation process of a .

$$a_{(i,j)} = \begin{cases} w_1^i w_2^j & \text{if } i = j, \\ w_1^i w_2^j + w_1^j w_2^i & \text{if } i \neq j. \end{cases} \quad (2)$$

From Eq. 1 above, the Star operation can lift an input spanning $d + 1$ dimensions into an output of $\frac{(d+2)(d+1)}{2} \approx \left(\frac{d}{2}\right)^2$ dimensions. Hence, the Star operation can dramatically amplify the dimensionality of the input features.

The preceding analysis centers on the Star operation's computation within a single layer. Across multiple layers, the output dimensionality grows exponentially with network depth while incurring no additional computational cost. For the l -th Star operation specifically, the per-layer output dimensions follow [Eq. 3](#).

$$\begin{aligned} O_1 &= \sum_{i=1}^{d+1} \sum_{j=1}^{d+1} w_{(1,1)}^i w_{(1,2)}^j x^i x^j && \in \mathbb{R} \left(\frac{d}{2}\right)^{2^1} \\ O_2 &= W_{2,1}^T O_1 * W_{2,2}^T O_1 && \in \mathbb{R} \left(\frac{d}{2}\right)^{2^2} \\ O_3 &= W_{3,1}^T O_2 * W_{3,2}^T O_2 && \in \mathbb{R} \left(\frac{d}{2}\right)^{2^3} \\ &\dots && \\ O_l &= W_{l,1}^T O_{l-1} * W_{l,2}^T O_{l-1} && \in \mathbb{R} \left(\frac{d}{2}\right)^{2^l} \end{aligned} \quad (3)$$

According to [Eq. 3](#), starting from a d -dimensional input, after the l -th Star operation, the output feature space expands to $\left(\frac{d}{2}\right)^{2^l}$ dimensions.

Unlike conventional convolutional neural networks, which attain high-dimensional feature mappings by inserting extra parameters at each layer, the Star operation accomplishes implicit dimensional expansion without introducing additional parameters. As the backbone of YOLOv8s-SD, StarNet thus raises detection efficiency while preserving detection performance.

Neck

The neck architecture within YOLOv8s-SD takes inspiration from YOLOv7 while swapping out C3 units for C2f counterparts. In essence, the C2f unit strings together successive feature fusion stages, making it possible to seamlessly consolidate feature representations originating from disparate scales. For a deeper discussion of the neck design applied in this study, readers may consult the YOLOv7 documentation [\[14\]](#).

Head

To strengthen detection performance on obstructed targets, our first proposal is to embed DCNv4 [\[29\]](#) into the decoupled detection head of YOLOv8s-SD. More precisely, DCNv4 is appended right after the opening convolutional layer within every branch of the head. The mechanism by which DCNv4 raises detection quality for occluded objects lies in its injection of supplementary offsets during convolution. These offsets effectively displace spatial sampling locations, permitting more accurate representations of deformed object characteristics.

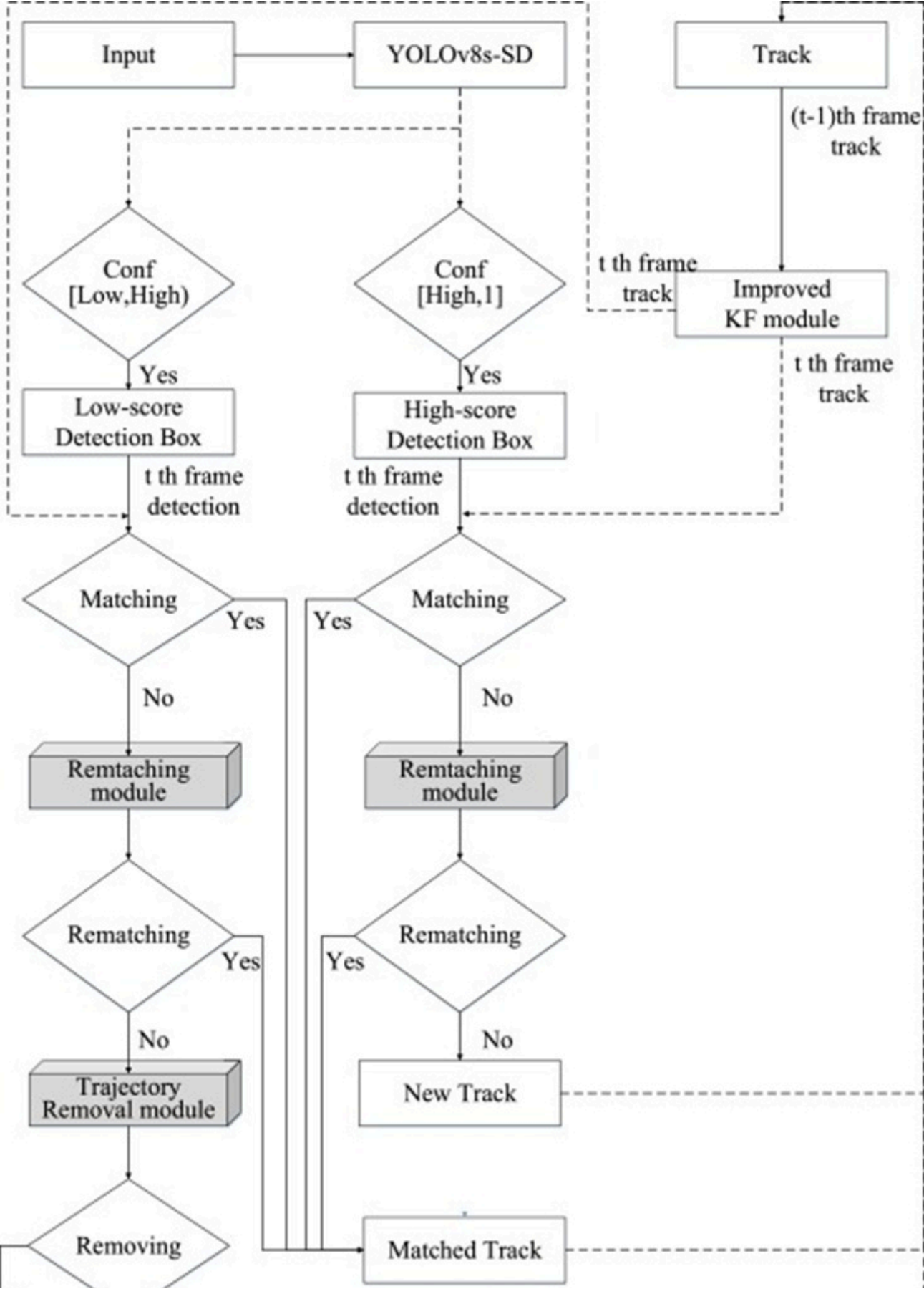
The definition of DCNv4 runs as follows:

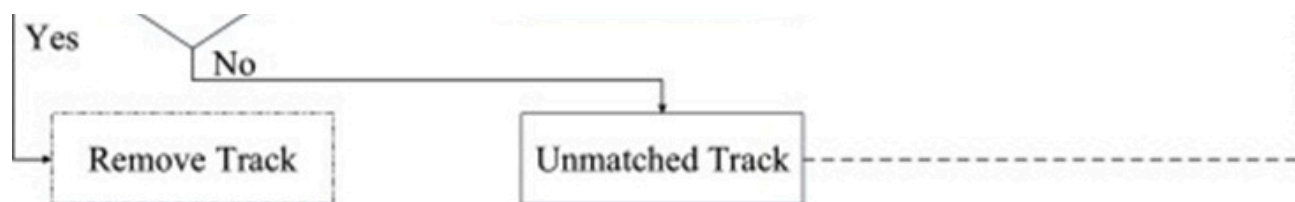
$$y(p_0) = \sum_{g=1}^G \sum_{k=1}^K w_g m_{gk} x_g(p_0 + p_k + \Delta p_{gk}) \quad (4)$$

Where G gives the total count of aggregation groups, K specifies how many sampling points belong to each aggregation group, w_g corresponds to the learnable weight assigned to the g -th group, m_{gk} denotes the modulation scalar tied to the k -th sampling point inside the g -th group, p_k marks the fixed offset associated with the k -th sampling point, and Δp_{gk} captures the learned offset term for the k -th sampling point belonging to the g -th group.

BytePro: the multi-object tracking for calves

The tracking segment of our YSD-BPTrack architecture is anchored on ByteTrack [24]. Here, we augment ByteTrack with an Improved KF module, an original Rematching module, and a newly designed Trajectory Removal module, collectively giving rise to a fresh calf tracking methodology christened BytePro. The overall schematic of BytePro is laid out in **Figure 6**. The explanation that follows traces the tracking of one calf as a running example.





(Conf) together with bounding boxes for the detected instances. Objects whose detection confidence satisfies $\text{Conf} \in [\text{Low}, \text{High})$ get categorized as low-score detection boxes, whereas those meeting $\text{Conf} \in [\text{High}, 1]$ are tagged as high-score detection boxes, under the constraint $0 \leq \text{Low} < \text{High} \leq 1$. In our setup, Low equals 0.3 and High equals 0.6.

Moving forward, the Track originating from frame $(t-1)$ issues a predicted Track box for frame t by way of the Improved KF module. This predicted Track box at frame t is subsequently aligned against the detection box (from either the high-score or low-score pool) produced for frame t by the YOLOv8-SD detection stream. A successful alignment yields a Matched Track as output. Should alignment fail, the Rematching module is triggered to attempt another round of matching. A successful rematch likewise produces a Matched Track. Otherwise, the protocol depends on whether the detection box belongs to the low-score or high-score category, as expanded below.

Inside the high-score pathway, a rematching failure prompts the high-score detection box to be spawned as a fresh Track. In the end, conditional on the matching or rematching results, either the Matched Track or the New Track gets adopted as the track output for the current frame. This Matched Track or New Track is also leveraged for forecasting the track position in frame $(t+1)$.

Symmetrically, within the low-score pathway, if rematching remains unsuccessful, the low-score detection box is discarded, and the linked Track at this instant is forwarded into the Trajectory Removal module for further assessment. Should the prescribed removal conditions be triggered, this Track corresponding to frame t gets eliminated. If not, it is reclassified as an Unmatched Track. Ultimately, based on the matching or rematching outcomes, only the Matched Track is permitted to serve as the track output at the present frame. Both the Matched Track and Unmatched Track are subsequently drawn upon for track prediction at frame $(t+1)$.

What follows unpacks the detailed workings of the Improved KF module, the Rematching module, and the Trajectory Removal module.

The improved KF (Kalman filter) module

Because ByteTrack was originally popularized for pedestrian tracking, some of its design choices may prove unsuitable for calf tracking scenarios. Adjustments are therefore warranted. A point worth highlighting is that one of the parameters populating the Kalman filter state vector is the aspect ratio. For the task of calf tracking addressed here, the aspect ratio exhibits shortcomings. This stems from the fact that calves adopt movement patterns markedly different from humans, featuring distinctive gait signatures. The practical consequence is an imprecise estimation of the predicted box width.

To remedy this, the Improved KF module [30] swaps out the aspect ratio and inserts the width directly into the Kalman Filter state vector. This structural choice further allows the prediction box to gauge the calf's positional coordinates with greater fidelity, thereby translating into superior tracking outcomes.

The track prediction routine under the improved KF can be spelled out as follows.

For each entity being tracked, a state vector x is formulated to capture both the location and the motion of the tracked object. It is parameterized by the center coordinates (l, r) , the width w , the height h , and their corresponding rates of change $\dot{l}$, $\dot{r}$, $\dot{w}$, and $\dot{h}$. Formally, this reads $x = [l, r, w, h, \dot{l}, \dot{r}, \dot{w}, \dot{h}]^T$, and such an x is commonly referred to as a track.

Within the Kalman filtering framework, x_{t-1} is first derived from x_t .

$$\bar{x}_t = Fx_{t-1} \quad (5)$$

Here, Δt is treated as the unit time step, and in our implementation, Δt is fixed to 1.

Thereafter, x_t is produced by refining $\bar{x}_t$ against z_t taken from the matching frame.

Above, x_{t-1} stands for the state vector belonging to the tracked object at frame $t-1$, $\bar{x}_t$ signals the predicted state vector at frame t , and F designates the state transition matrix.

Here, Δt is treated as the unit time step, and in our implementation, Δt is fixed to 1.

Thereafter, x_t is produced by refining $\bar{x}_t$ against z_t taken from the matching frame.

$$x_t = \bar{x}_t + K(z_t - H\bar{x}_t) \quad (6)$$

In these expressions, H is the measurement matrix, z_t symbolizes the observed state vector at frame t written as $[l, r, w, h]^T$, $H\bar{x}_t$ refers to the observation vector computed from $\bar{x}_t$, $(z_t - H\bar{x}_t)$ encodes the measurement residual separating the actual observation from the predicted observation, x_t corresponds to the polished state vector at frame t , and K is the Kalman gain. Further particulars regarding K are supplied below.

Effectively, K regulates the balance between trusting the prediction and trusting the observation, as formulated in [Eq. 7](#). To compute K , an 8×8 covariance matrix P is introduced; this matrix models the degree of uncertainty surrounding the tracked object's position and velocity components.

$$K = \frac{\bar{P}_t H^T}{H \bar{P}_t H^T + R} \quad (7)$$

Here, $\bar{P}_t$ stands for the predicted covariance matrix at frame t , while R corresponds to the measurement noise covariance matrix.

In [Eq. 7](#), $\bar{P}_t$ is obtained starting from P_{t-1} .

$$\bar{P}_t = F P_{t-1} F^T + Q \quad (8)$$

Here, Q designates the system noise matrix, and P_{t-1} refers to the covariance matrix at frame $t-1$.

The updated covariance matrix P_t for frame t is then derived from $\bar{P}_t$.

$$P_t = (I - KH) \bar{P}_t \quad (9)$$

Here, I is the identity matrix, and $(I - KH)$ acts as the correction multiplier.

To close the loop, x_t is handed over to the next frame so the predicted state vector for that upcoming frame can be computed. Readers seeking additional insight into the KF variant employed here are encouraged to look up the ByteTrack reference [24].

The rematching module

In the course of calf tracking, occlusions frequently break the correspondence between tracks and detections. Worse still, once a calf becomes entirely hidden from view, its motion pattern tends to be erratic and hard to anticipate. For a calf suffering a broken match, ByteTrack's recourse is linear extrapolation grounded on the immediately preceding frame. This simplistic fallback can prevent the calf from being correctly re-associated with its original track identity when it re-emerges from occlusion, thereby precipitating a track ID swap.

The Rematching module turns to enlarged track prediction boxes to restore correspondence with target detection boxes. It adaptively broadens the track prediction box in proportion to the number of frames for which the track has remained unmatched, thereby strengthening its capacity to reconnect with detected targets. Through such a mechanism, ID transitions are substantially reduced, most notably for calves that completely vanish behind obstacles and later reappear.

The rematching logic can be structured as detailed below.

As visually conveyed in **Figure 7**, with the center position (c_x, c_y) , the width w , and the height h of a track prediction box P_i , this box can be compactly written as $P_i = [c_x, c_y, w, h]$. In cases where tracks fail to be matched to any detection box D_j , P_i gets widened by Δw pixels on both its left and right sides and extended by Δh pixels on its top and bottom sides, thus producing an enlarged prediction box $\overline{P}^i$, in line with **Eq. 10**.

$$\overline{P}^i = [c_x, c_y, w + 2\Delta w, h + 2\Delta h] \quad (10)$$

Here, Δw captures the expansion applied horizontally, and Δh captures the expansion applied vertically, with their definitions supplied in **Eqs. 11 and 12**.

$$\Delta w = w \times \left(1 + \frac{1}{1+e^{-a}}\right) \quad (11)$$

$$\Delta h = h \times \left(1 + \frac{1}{1+e^{-a}}\right) \quad (12)$$

Here, a counts how many consecutive frames the track has gone unmatched; as soon as the track secures a match, a is returned to 0.

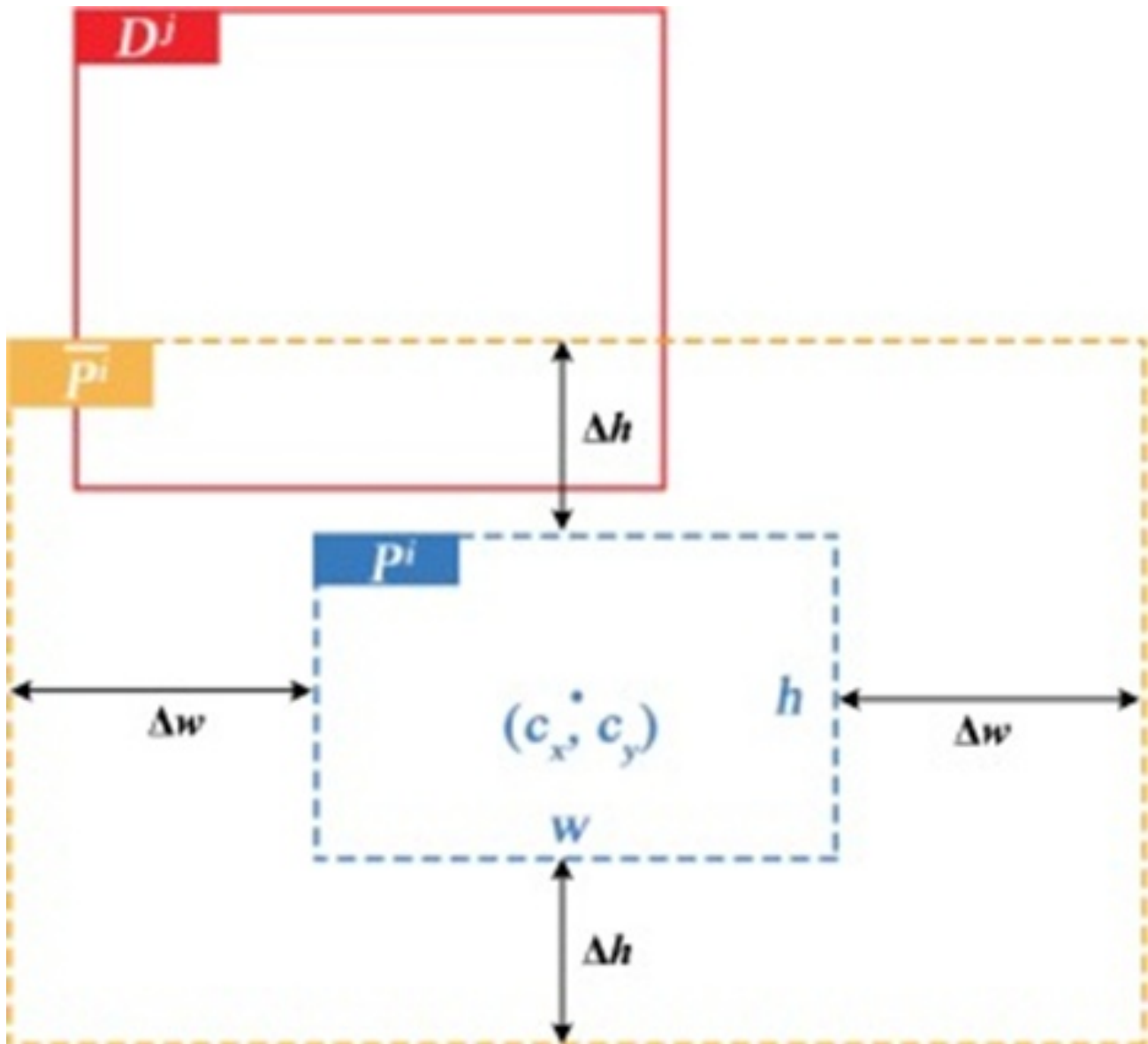


Figure 7. Extended track prediction box. The red box represents the detection box, the blue box represents the prediction box, and the yellow box represents the expanded prediction box.

Following the expansion, the matching rate M_r between $\bar{P}^i$ and D^j is quantified.

$$M_r = \frac{\text{Intersection}(\bar{P}^i, D^j)}{\text{Area}(D^j)} \in [0, 1] \quad (13)$$

In this notation, i indexes the track, j indexes the detection box, $\text{Intersection}(\bar{P}^i, D^j)$ computes the overlapping region between $\bar{P}^i$ and D^j , and $\text{Area}(D^j)$ yields the area encompassed by D^j .

The value of M_r then steers the rematching decision between $\bar{P}^i$ and D^j into one of two distinct cases, enumerated in [Eq. 14](#).

$$i^*, j^* = \begin{cases} (\arg \max_{i,j} M_r) & \text{if } 0 < M_r < 1 \\ (\arg \min_{i,j} (\text{center}(\overline{P^i}) - \text{center}(D^j))) & \text{if } M_r = 1 \end{cases} \quad (14)$$

Here, $(\arg \max_{i,j} M_r)$ picks out the track index i and detection box index j that jointly achieve the highest M_r , while $(\arg \min_{i,j} (\text{center}(\overline{P^i}) - \text{center}(D^j)))$ identifies the indices i and j that give the smallest Euclidean distance between the center of P_i and the center of D_j .

In practical terms, when handling rematching between tracks and detection boxes, if a track encounters several detection boxes whose matching rates satisfy $0 < M_r < 1$, the D_j that maximizes M_r is linked for rematching with the track represented by P_i^- .

Furthermore, when a track encounters multiple detection boxes all yielding $M_r=1$, the D_j positioned at the shortest Euclidean distance from the track represented by P_i^- is chosen for rematching.

The trajectory removal module

Under realistic cattle yard conditions, calves are prone to suffer brief occlusions caused by herd mates. These occlusions readily disrupt trajectory correspondences. The default policy inside ByteTrack dictates that tracks losing their matches get purged after a set delay, nominally 25 frames. In practice, however, the total occlusion span experienced by a calf will often stretch beyond 25 frames, and that policy can trigger unwarranted track deletions.

The trajectory removal module instead dynamically tunes the removal deadline for lost-match trajectories during occlusion events, taking cues from the trajectory of occlusion rates O_r recorded over the most recent five consecutive frames. As illustrated in **Figure 8**, it distinguishes among three characteristic conditions to forestall erroneous track removal, consequently improving overall tracking reliability.

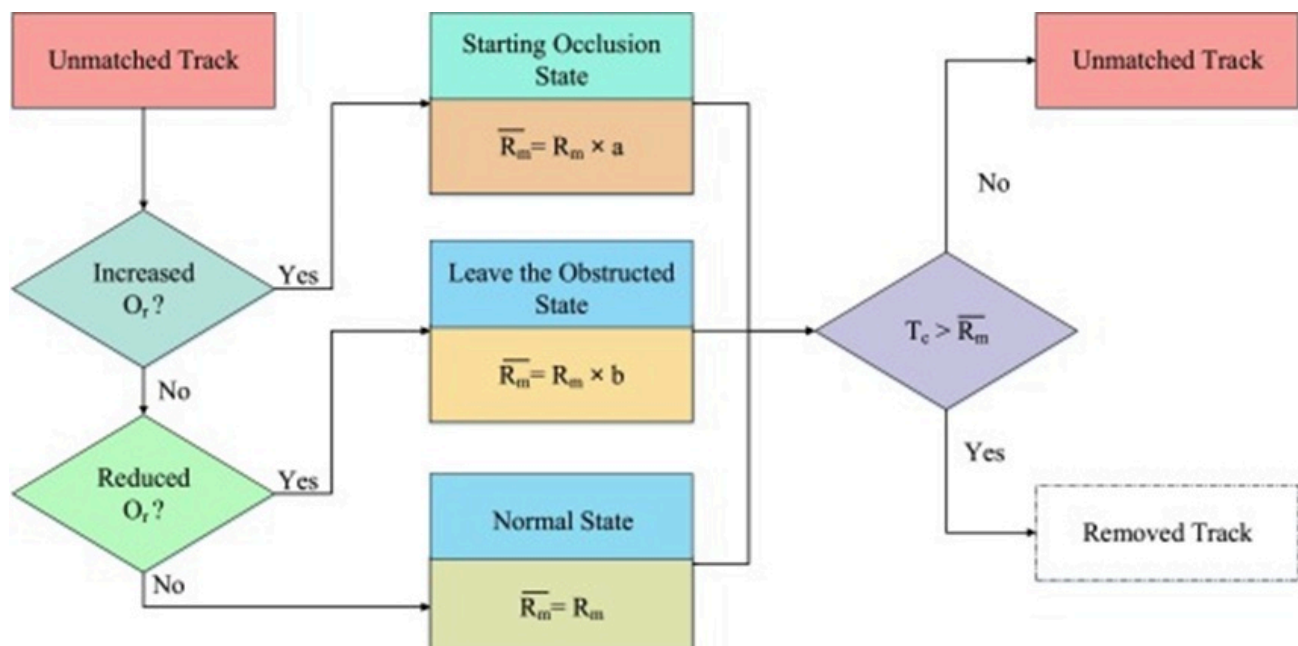


Figure 8. Flowchart of the Trajectory Removal module. Additionally, the dotted line indicates that this part has been removed.

To begin, each detected object D_i computes an occlusion rate O_r with respect to every other detected object D_j , and the peak value is assigned as the O_r of D_i , governed by Eq. 15.

$$O_r = \max_{j \neq i} \left(\frac{\text{Intersection}(D_i, D_j)}{\text{Area}(D_i)} \right) \quad (15)$$

Here, $\text{Intersection}(D_i, D_j)$ quantifies the region of overlap between D_i and D_j , whereas $\text{Area}(D_i)$ expresses the footprint of D_i .

This O_r score is subsequently bound to the track that D_i is matched with. Later, at the moment when a Matched Track flips into an Unmatched Track, the recent course of O_r across the preceding five frames is examined to determine the track's prevailing status. The assigned track removal timeout is then set according to that determined status; the reader may follow Figure 8 for the exact mapping.

Concretely, for an Unmatched Track, if O_r has displayed a rising trend over the past five consecutive frames, this is interpreted as the calf moving deeper into occlusion. Under this interpretation, the maximum removal period is broadened to $R_m^- = a \times R_m$, where R_m is the default maximum removal timeout; here, a is fixed at 8 and R_m at 25.

Conversely, if O_r has exhibited a declining trend over the past five consecutive frames, the calf is understood to be stepping out of occlusion. In this case, the removal period is set to $R_m^- = b \times R_m$, with b taken as 5.

If O_r has remained essentially flat, indicating a normal unobstructed state, the period stays at $R_m^- = R_m$.

Ultimately, should the cumulative duration of match loss T_c overflow R_m^- , the track is terminated. Otherwise, it is retained as an Unmatched track and forwarded for continued processing.

Evaluation metrics

To carry out a rigorous assessment of the YSD-BPTrack multi-object tracking framework we introduced for calves, a suite of distinct evaluation measures was drawn upon for benchmarking during both the detection and tracking phases.

Multi-object tracking evaluation metrics

Within the five indicators employed at the tracking stage, larger readings on HOTA (High Order Tracking Accuracy), MOTA (Multi-Object Tracking Accuracy), MOTP (Multi-Object Tracking Precision), and IDF1 (Identification F1), paired with a smaller IDSw (Identity Switches) figure, point toward a tracking system that is both more accurate and more dependable.

To unpack these, HOTA supplies an all-around measure of how well a multi-object tracking system performs.

$$HOTA = \sqrt{\frac{\sum_{c \in T} A(c)}{TP + FN + FP}} \quad (16)$$

Where TP stands for the volume of true positive matches, FN for the volume of false negative misses, and FP for the volume of false positive alarms.

Category-level accuracy $A(c)$ is derived as:

$$A(c) = \frac{TPA(c)}{TPA(c) + FNA(c) + FPA(c)} \quad (17)$$

Where $TPA(c)$ registers the true positive count linked to category c , $FNA(c)$ the false negative count for category c , and $FPA(c)$ the false positive count for category c .

MOTA captures the aggregate accuracy delivered by multi-object tracking systems throughout tracking sequences.

$$MOTA = 1 - \frac{FP + FN + IDSw}{GT} \quad (18)$$

Where FN corresponds to the number of targets the detector failed to flag, FP to the number of hallucinated detections where absent targets are wrongly reported, IDSw to the tally of identity confusions committed during tracking, and GT to the genuine number of targets present in the scene.

MOTP reflects the spatial precision with which a tracking system localizes targets.

$$MOTP = \frac{\sum_{t=1}^T \sum_{i=1}^{N_t} d_{i,t}}{\sum_{t=1}^T N_t} \quad (19)$$

Where T indexes the overall number of frames, N_t stands for the count of targets correctly tracked at frame t , and $d_{i,t}$ is the average positional discrepancy of the i -th correctly tracked target at frame t .

IDF1 evaluates the degree to which a tracking system consistently upholds target identities from start to finish.

$$IDF_1 = \frac{2IDTP}{2IDTP + IDFP + IDFN} \quad (20)$$

Where IDTP refers to identity assignments where the tracker output correctly corresponds to the real target, IDFP to identity assignments where the tracker output is erroneously linked to a different real target, IDFN to real target identities that received no tracker assignment, and IDTN to instances where real target identities were appropriately left unassigned, free of any incorrect matches.

Object detection evaluation metrics

Turning to the five indicators used during detection, higher figures for P (Precision), R (Recall), and $mAP_{0.5}$ (Mean Average Precision at an IoU threshold of 0.5) signal a detection model that is both more capable and more stable, whereas lower Params (Parameters) and FLOPs (Floating Point Operations) reflect superior computational thrift.

In more concrete terms, P measures how exact the model's detections are, whereas R measures how thorough they are.

$$P = \frac{TP}{TP + FP} \quad (21)$$

$$R = \frac{TP}{TP+FN} \quad (22)$$

mAP_{50}^{val} summarizes the detection model's global quality, capturing the mean Average Precision calculated under a 50% Intersection over Union (IoU) criterion.

$$mAP_{50}^{val} = \frac{1}{N} \sum_{i=1}^N AP_i \quad (23)$$

$$AP = \int_0^1 PRdR \quad (24)$$

Where N gives the overall number of classes involved in the detection problem, and AP_i represents the average precision value attained for the i -th class.

Results and Discussion

A series of experiments was executed to gauge the capability of the YSD-BPTrack tracking framework. This portion of the paper reports validation of the YOLOv8s-SD object detection model during detection, validation of the BytePro multi-object tracking approach during tracking, along with ablation analyses.

Verification of YOLOv8s-SD object detection model

Seeking to strengthen calf detection in occlusion-laden environments, we folded StarNet and DCNv4 into the YOLOv8s blueprint. The resulting architecture is termed YOLOv8s-SD for brevity. StarNet realizes implicit mapping into high-dimensional feature spaces solely through element-wise products. DCNv4 fortifies the sensing of occluded instances by appending learned offsets. Collectively, YOLOv8s-SD sharpens detection under occlusion while simultaneously compressing parameter counts and computational footprint.

Selected experimental figures are collated in **Table 2**. Within this table, StarNet is denoted as 'Star' and DCNv4 as 'DCN'. The variant that embeds StarNet into YOLOv8s is referred to as YOLOv8s-Star, the variant embedding DCN into YOLOv8s is referred to as YOLOv8s-DCN, and the variant fusing both StarNet and DCN into YOLOv8s is referred to as YOLOv8s-SD.

Table 2. Comparison of different detection models.

Model	Star	DCN	FLOPs (G)	Params (M)	mAP_{val50} (%)	R (%)	P (%)
YOLOv8s	×	×	28.6	11.1	98.9	97.9	98.1
YOLOv8s-Star	✓	×	17.4	6.5	99.0	97.7	98.1
YOLOv8s-DCN	×	✓	28.6	10.4	99.2	98.1	98.1
YOLOv8s-SD	✓	✓	17.4	5.9	99.2	98.0	98.2

Initially, StarNet was integrated into the backbone portion of YOLOv8s. Uniquely, where standard convolutional strategies expand channel dimensions to access high-dimensional feature spaces, StarNet accomplishes an implicit high-dimensional projection purely through the Star operation, avoiding any injection of extra parameters. In measurable terms, relative to the YOLOv8s starting point, YOLOv8s-Star held precision, recall, and mAP_{50}^{val} at closely

comparable levels while trimming parameters by 41.2% and computation by 39.2%. This outcome dramatically compresses the model's scale and runtime cost with no sacrifice to detection fidelity.

Occlusion of calves in the camera's field of view erodes detection quality, subsequently impairing tracking. DCNv4 was inserted into the detection head to amplify the model's spatial sampling flexibility and better capture occluded targets. Quantitatively, YOLOv8s-DCN edged ahead of YOLOv8s with a mAP_{50}^{val} of 99.2%, accompanied by a 6.3% parameter contraction.

Surveying the full comparison in **Table 2**, YOLOv8s-SD holds clear edges on several important fronts. Its parameter volume measures 5.9M, and its FLOPs stand at 17.4G, which translates into cuts of 46.8% and 39.2%, respectively, besides YOLOv8s. Beyond that, YOLOv8s-SD posts a mAP_{50}^{val} of 99.2%, equaling the peak score set by YOLOv8s-DCN and underscoring excellent detection prowess. As a result, the YOLOv8s-SD model not only sustains efficient detection throughput but markedly elevates the capacity to spot calves enduring occlusion, thereby furnishing a dependable platform for the tracking tasks that follow.

Verification of the BytePro multi-object tracking method

During the tracking stage of the YSD-BPTrack framework, BytePro contributes a reformulated bounding box prediction generation pipeline, a newly conceived rematching algorithm, and a re-engineered trajectory deletion strategy. Collectively, these advances significantly boost the YSD-BPTrack framework's competence at following multiple calves through occlusion episodes. The analysis below proceeds stepwise, spanning comparisons with competing tracking pipelines, evaluation of BytePro's tracking reliability across various detection backbones, and ablation testing.

Comparison of different tracking methods

Table 3 compiles quantitative benchmarking data for multiple tracking pipelines, wherein YSD-BPTrack pairs the YOLOv8s-SD detection model with the BytePro tracking technique.

Table 3. Performance comparison of different tracking methods.

Method	FPS	IDSw	IDF1 (%)	MOTP (%)	MOTA (%)	HOTA (%)
YOLOv8s+OcSort	36	6	89.1	92.8	86.2	83.1
YOLOv8s+DeepSort	5	7	84.5	92.8	86.2	81.2
YOLOv8s+BotSort	7	10	87.2	92.9	86.1	82.2
YOLOv8s+ByteTrack	39	9	85.2	90.9	87.0	78.6
YSD-BPTrack	41	1	98.3	92.5	99.1	92.4

Inspecting **Table 3**, YSD-BPTrack registers a HOTA equal to 91.6%, MOTA of 99.1%, MOTP of 92.5%, IDF1 of 97.1%, and an IDSw tally of merely 2. Placed alongside alternative tracking pipelines, YSD-BPTrack posts stronger HOTA, MOTA, and IDF1 readings, together with a lower IDSw count. Breaking this down, compared with YOLOv8s+OcSort, YSD-BPTrack pushes HOTA upward by 11.2%, MOTA by 15.0%, and IDF1 by 10.3%, while pulling IDSw down by 83.3%. Stacked against YOLOv8s+DeepSort, YSD-BPTrack pushes HOTA up by 13.8%, MOTA by 15.0%, IDF1 by 16.3%, and draws IDSw down by 85.7%. Measured beside YOLOv8s+BotSort, YSD-BPTrack pushes HOTA up by 12.4%, MOTA by 15.1%, IDF1 by 12.7%, and slashes IDSw by 90.0%. Finally, in relation to YOLOv8s+ByteTrack, YSD-BPTrack pushes HOTA up by 17.6%, MOTA by 14.0%, IDF1 by 15.4%, and drops IDSw by 88.9%.

Shifting to runtime speed, tracking pipelines that bundle a Re-ID component—DeepSort and BotSort among them—operate beneath 25 FPS, undershooting the real-time observation standards demanded in precision livestock contexts.

By contrast, our YSD-BPTrack attains the highest throughput via enhancements spanning both detection and tracking, clocking 41 FPS.

These experimental observations firmly establish that YSD-BPTrack delivers advantageous performance over several vital metrics, with MOTA and IDF1 standing out especially, revealing its adeptness at complex multi-object calf tracking.

A look at **Figure 9** confirms that our YSD-BPTrack tracking method exhibits stronger performance on both HOTA and IDF1 axes, indicative of excellent holistic multi-object calf tracking quality.

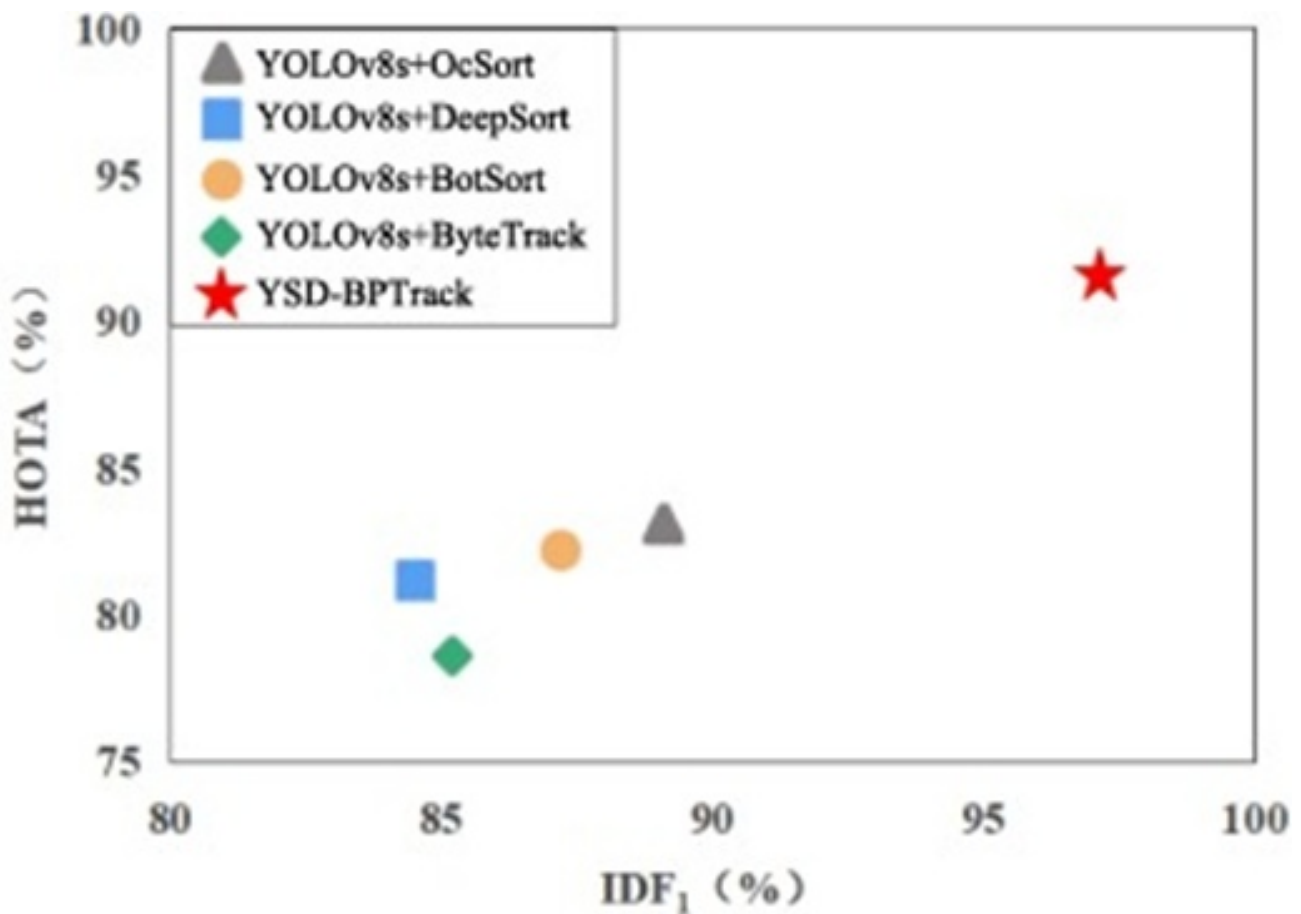


Figure 9. Performance comparison of different tracking methods on HOTA and IDF1.

Figure 10 makes the picture plain: as the calf situated on the right, originally bearing trajectory ID 1, comes back into view after occlusion at frames 105, 175, and 230, identity switches afflict the YOLOv8s+OcSort, YOLOv8s+DeepSort, YOLOv8s+BotSort, and YOLOv8s+ByteTrack systems. Our YSD-BPTrack, on the other hand, follows that very same calf without a single identity switch throughout. The empirical evidence therefore points to the YSD-BPTrack tracking framework handling calf tracking more robustly when occlusion is present.

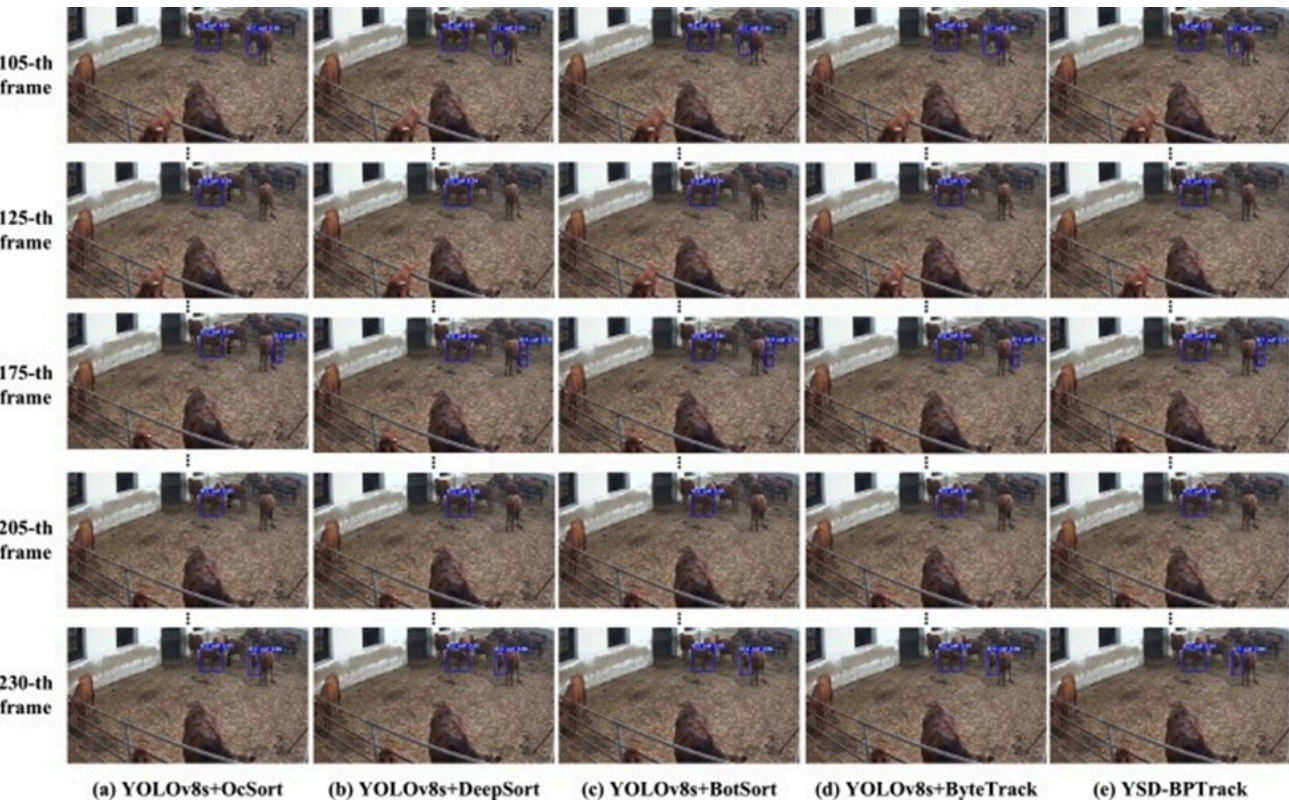


Figure 10. Qualitative tracking results of different tracking methods in video 1 of our calf tracking dataset.

Comparison of BytePro in different detection models

For this set of experiments, YOLOv5s, YOLOv7, and YOLOv8s—three mature, extensively deployed single-stage detection architectures—were first selected to serve as the detection-stage foundations. BytePro, which evolved from ByteTrack, was likewise chosen alongside ByteTrack to represent the tracking-stage options.

To more extensively verify the merits of our introduced tracking approach, calf tracking trials were conducted, coupling both ByteTrack and BytePro with the YOLOv5s, YOLOv7, and YOLOv8s detection backbones. The resulting measurements are organized in [Table 4](#).

Table 4. Performance comparison of the BytePro tracking method in different detection models.

Method	IDS _w	IDF1 (%)	MOTP (%)	MOTA (%)	HOTA (%)
YOLOv5s+ByteTrack	13	84.2	85.4	78.5	72.0
YOLOv5s+BytePro	6	85.7	86.6	81.8	72.4
YOLOv7+ByteTrack	11	85.7	90.2	85.4	78.0
YOLOv7+BytePro	6	88.0	92.4	86.4	80.6
YOLOv8s+ByteTrack	9	85.2	90.9	87.0	78.6
YOLOv8s+BytePro	5	88.2	92.2	88.5	84.6

Reading from the data in [Table 4](#), it is apparent that substituting BytePro for ByteTrack brings pronounced benefits irrespective of whether YOLOv5s, YOLOv7, or YOLOv8s is employed. For YOLOv5s, HOTA gained 0.6%, MOTA gained

4.2%, MOTP gained 1.4%, IDF1 gained 1.8%, while IDSw contracted by 53.4%. For YOLOv7, BytePro yielded a HOTA gain of 3.3%, a MOTA gain of 1.2%, a MOTP gain of 2.4%, an IDF1 gain of 2.7%, and a 45.5% drop in IDSw relative to ByteTrack. In the same vein, for YOLOv8s, BytePro produced a HOTA increase of 7.6%, a MOTA increase of 1.7%, a MOTP increase of 1.4%, an IDF1 increase of 3.5%, alongside a 44.4% shrinkage in IDSw.

In overview, the BytePro tracking technique unfailingly secures elevated tracking accuracy across YOLOv5s, YOLOv7, and YOLOv8s, while significantly suppressing IDSw counts. These findings collectively confirm BytePro's superiority and practical value in multi-object tracking scenarios.

Ablation studies

In the ablation experiments presented in this part, ByteTrack is taken as the foundational method for the tracking phase, and analyses are conducted under different configurations combining the KF, Rematching, and Trajectory Removal modules. YOLOv8s-SD is held fixed as the detection model in all cases.

To keep notation compact, we refer to the 'Improved KF module' as 'KF', the 'Rematching module' as 'RMA', and the 'Trajectory Removal module' as 'RMO'. The collected outcome data appear in [Table 5](#).

Table 5. Performance comparison of tracking methods with different module combinations.

Index	KF	RMA	RMO	IDSw	IDF1 (%)	MOTP (%)	MOTA (%)	HOTA (%)
1	×	×	×	6	94.2	90.9	90.1	88.6
2	✓	×	×	4	96.2	92.5	99.0	90.8
3	×	✓	×	4	96.2	92.1	99.0	90.3
4	×	×	✓	3	96.8	92.1	99.0	90.6
5	✓	✓	×	5	94.9	92.5	99.0	90.2
6	✓	×	✓	4	96.2	92.5	99.0	90.8
7	×	✓	✓	2	98.3	92.1	99.1	91.6
8	✓	✓	✓	1	98.3	92.5	99.1	92.4

Verification of the improved KF module

What the Improved KF module does is replace the third component of the Kalman filter state vector—previously holding the aspect ratio—with the width directly. This switch gives the Kalman filter a better ability to anticipate calf movement, and in particular, it alleviates the persistent trouble of tracking ID swaps inside occluded settings.

Looking at [Table 5](#), when the Improved KF module alone (Index 2) is incorporated, HOTA moves up from 88.6% to 90.8%, and MOTA rises from 90.1% to 99.0%. These gains confirm that tuning the Kalman filter state vector produces concrete advances in both tracking precision and robustness. Calves display movement styles quite unlike humans; limb motion can generate varying deformation patterns, making an aspect ratio state vector inappropriate for this multi-target tracking problem. Swapping that component for width permits the Kalman filter to produce more trustworthy estimates of calf location and velocity, which is especially useful for circumventing repetitive ID swaps during occlusion periods and delivers a clear jump in overall tracking quality.

[Figure 11](#) illustrates the effect: because of occlusion, the two calves followed with the baseline method (Index 1) experience an ID swap at frame 160. With the version that includes the Improved KF module (Index 2), however, the identities assigned to these two calves remain unchanged, and no swap is observed.

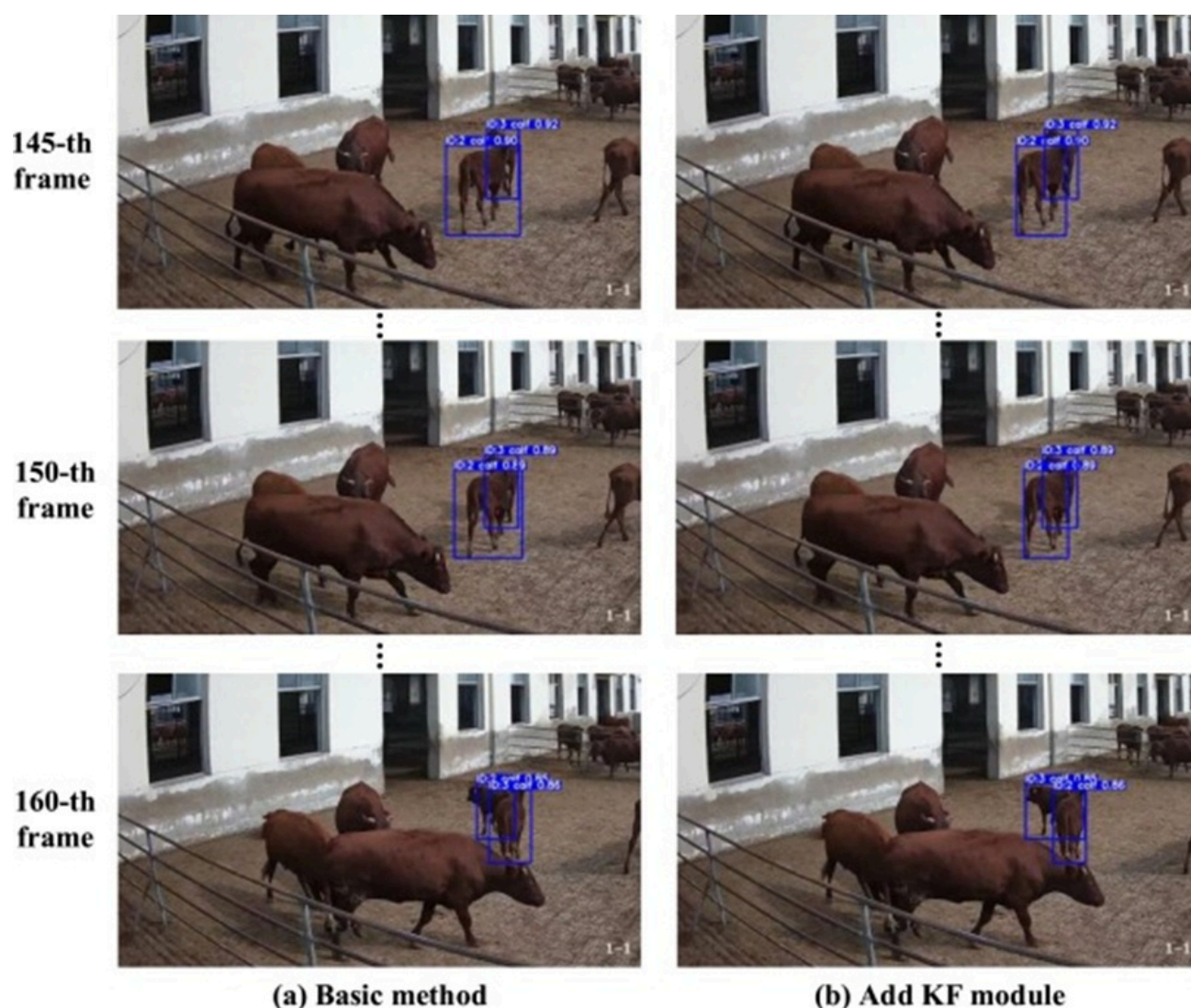


Figure 11. Qualitative tracking results of the basic tracking method and the added Improved KF module in video 7 of our calf tracking dataset.

The contribution of the Improved KF module—revising the Kalman filter state description by putting width in place of aspect ratio—substantially reinforces the steadiness and correctness of YSD-BPTrack when applied to calves. It also decreases both matching mistakes and the incidence of identity interchange.

Verification of the rematching module

The role played by the Rematching module is to shore up matching capability by deploying trajectory extension prediction boxes after the initial matching between tracks and detections fails. Full occlusion can cause some calves to move along nonlinear paths, and under such circumstances, Kalman filter prediction boxes are frequently unable to correctly re-associate with calves once they become visible again, leading straight to ID switching issues. Our Rematching module addresses these obstacles and measurably strengthens matching performance.

According to [Table 5](#), when the Rematching module alone (Index 3) is added, HOTA goes from 88.6% to 89.7%, and MOTA rises from 90.1% to 99.0%.

The situation captured in **Figure 12** shows the calf labelled ID 1 becoming entirely invisible at frame 130 due to occlusion and re-emerging at frame 155. Using the baseline method (Index 1), that calf is subject to an ID switch brought about by its nonlinear displacement while hidden. With the Rematching module method (Index 3) engaged, however, calf ID 1 stays correctly tracked from start to finish, free of any identity swap. Exploiting trajectory extension prediction bounding boxes to enhance matching works, in practice, to raise the tracking dependability for calves that come back into view after full occlusion.

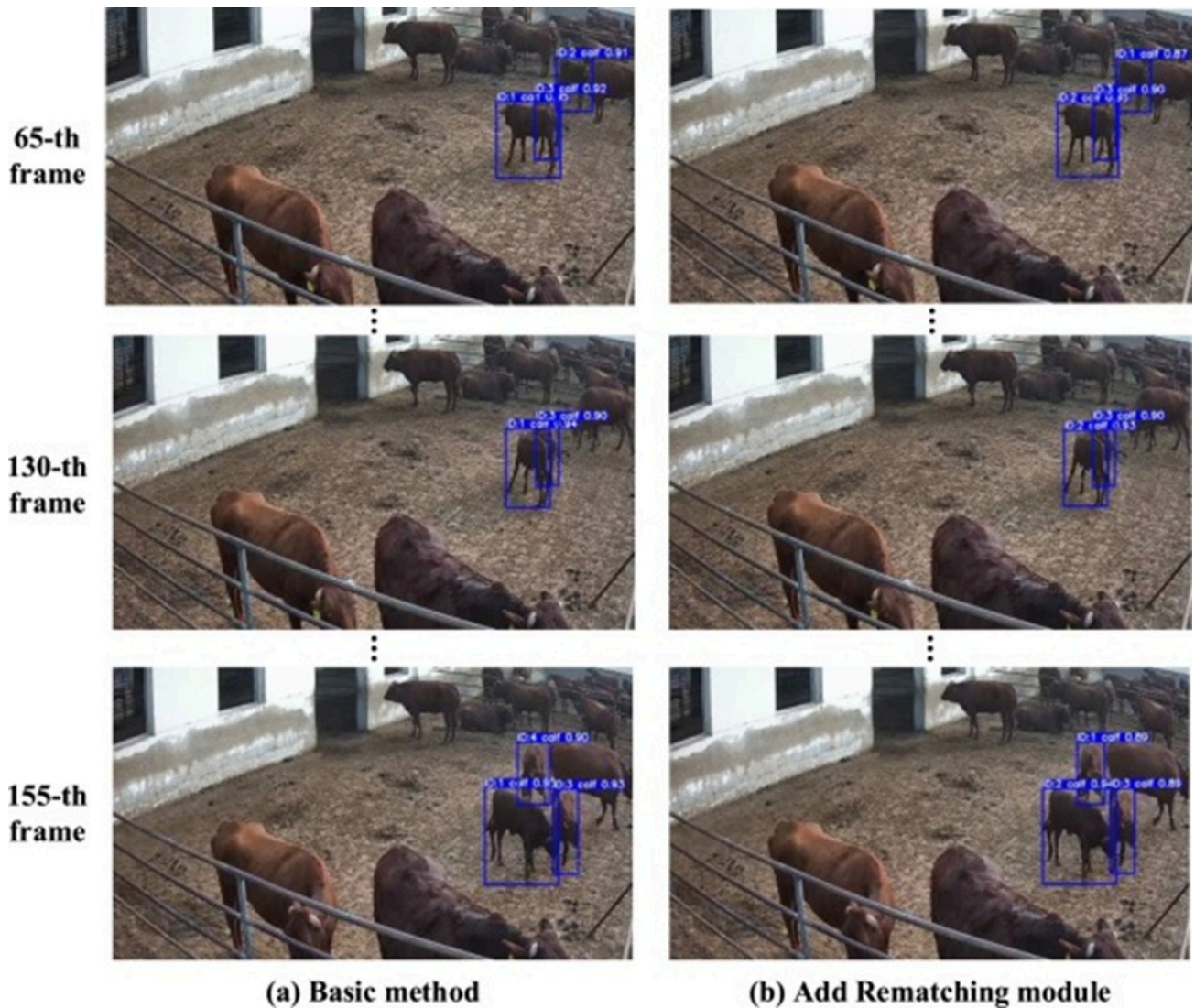


Figure 12. Qualitative tracking results of the basic tracking method and the added Rematching module in video 1 of our calf tracking dataset.

The focus of the Rematching module is squarely on tightening the link between unmatched tracks and detection targets that resurface once total occlusion ends, which translates into a meaningful lift in YSD-BPTrack tracking performance.

Verification of the trajectory removal module

The Trajectory Removal module dynamically alters the previously fixed removal interval for unmatched tracks by recognizing three separate phases: sliding into occlusion, emerging out of occlusion, and unobstructed presence. This flexible scheme better approximates the conditions encountered in actual cattle housing facilities, lowering the odds of mistakenly discarding unmatched tracks and limiting the number of tracks deleted purely due to occlusion.

As per **Table 5**, introducing the Trajectory Removal module on its own (Index 4) boosts HOTA from 88.6% to 90.6%. Alongside this, IDSw drops from 6 events to 3 events.

The case in **Figure 13** is instructive: the calf holding ID 1 is completely lost to occlusion at frame 70 and reappears at frame 115. The baseline method (Index 1), operating with a hard-coded 25-frame removal timeout for unmatched tracks, commits a mistaken ID swap. By comparison, when the Trajectory Removal module is active (Index 4), it adaptively purges unmatched tracks and does so in a manner that reliably preserves correct calf identification.

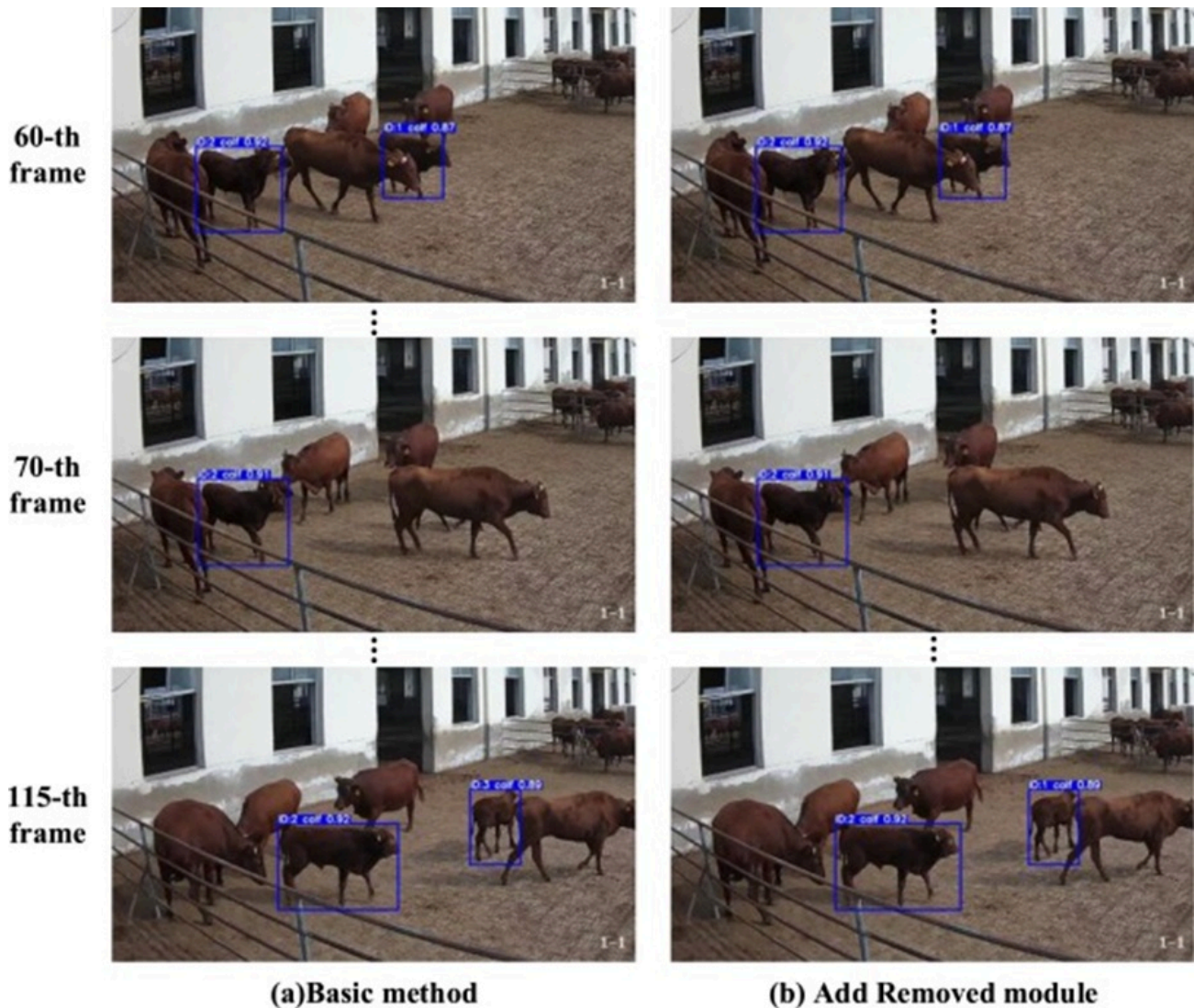


Figure 13. Qualitative tracking results of the basic tracking method and the added Trajectory Removal module in video 7 of our calf tracking dataset.

What this module brings to YSD-BPTrack is dynamic control over the expiry time of lost matching tracks, cutting back on the frequency of erroneous track termination, heading off ID swaps, and ultimately lifting tracking effectiveness.

Verification of multi-module combinations

Bringing these modules together into joint operation yields pronounced jumps in tracking quality, most obviously improving the overall continuity and correctness of multi-object calf tracking within occlusion-heavy settings.

Consulting **Table 5**, when the KF and Rematching modules are turned on together (Index 5), HOTA gains 1.8% over the baseline method (Index 1), IDF1 climbs by 0.8%, and IDSw shrinks by 16.7%. The tandem of KF and Rematching modules produces upticks on both HOTA and IDF1, reflecting better overall target tracking accuracy together with more robust identity continuity.

When the KF and Trajectory Removal modules are jointly activated (Index 6), HOTA is lifted by 2.5%, IDF1 by 2.1%, and IDSw contracts by 33.3% relative to the baseline method (Index 1). The cooperative effect of KF and Trajectory Removal modules stands out in the HOTA and IDF1 scores, highlighting the importance of discarding superfluous detections to bring down identity swaps and raise tracking precision. This combination not only sharpens identity tracking correctness but also meaningfully lowers swap frequency, thereby making the framework more stable and trustworthy.

When the Rematching and Trajectory Removal modules are applied side by side (Index 7), HOTA sees a 3.4% increase, MOTA a 10.0% increase, IDF1 a 4.4% increase, and IDSw a 66.7% decrease alongside the baseline method (Index 1). The fusion of Rematching and Trajectory Removal modules delivers solid progress on HOTA, MOTA, and IDF1, demonstrating that simultaneously drawing on rematching and adaptive removal strategies pushes tracking precision and identity stability even higher, most notably by cutting down identity swaps.

When all three—KF, Rematching, and Trajectory Removal—are engaged together (Index 8), the picture is one of a 4.3% HOTA rise, a 10.0% MOTA rise, a 4.4% IDF1 rise, and an 83.3% IDSw fall alongside the baseline method (Index 1). This three-way union of modules elevates the BytePro method's performance meaningfully on HOTA, MOTA, and IDF1.

Bringing the KF, Rematching, and Trajectory Removal modules into combined service makes their complementary and mutually reinforcing contributions across diverse performance dimensions clearly evident. Deploying any one module alone already confers sizeable benefits, especially to MOTA and IDF1. Pairing any two tends to further lift multi-target tracking correctness and identity maintenance stability. And activating all three simultaneously pushes overall object tracking attainment to its peak.

What YSD-BPTrack accomplishes is not only a reduction in how often calf identities get swapped within occluded environments but also a strengthening of the framework's overall steadiness and accuracy.

Stacked up against other established tracking frameworks, the system we present delivers cutting-edge tracking results, a performance leap made possible by the enhancements embedded in the Improved KF module, the Rematching module, and the Trajectory Removal module. In the opening stage of this project, we canvassed and synthesized existing knowledge on cattle tracking from research groups operating both within China and internationally [16, 17, 19, 23]. The collected literature across different laboratories indicates that most investigators construct multi-object tracking solutions upon the Tracking-by-detection blueprint [31, 32]. A number of these researchers have opted for tracking pipelines such as DeepSort [20] and BotSort [30], which fold in Re-ID convolutional blocks [33]. The downside is that the Re-ID block demands additional convolutional cycles to perform identity discrimination, translating into heavier computational loads and slower tracking operations.

In a different vein, the ByteTrack [24] pipeline cleverly makes use of both high- and low-confidence detection bounding boxes obtained during detection, side-stepping entirely the overhead of Re-ID convolutional blocks. This simplification is what gives ByteTrack its competitive edge on tracking benchmarks: it manages to raise throughput while keeping tracking fidelity at a respectable level. In its original form, ByteTrack populates the Kalman filter state vector with the target's center position, its aspect ratio, and its height to drive frame-to-frame prediction. In occlusion-heavy contexts, however, the silhouette of a calf can distort abruptly. As a remedy, encoding width instead of aspect ratio as the geometric cue within the state vector serves to raise tracking precision. Beyond this, applying a supplementary matching stage [34] aimed at calves hidden from view and adaptively regulating the time window after which lost trajectories are discarded both help suppress erratic identity alternations. On the strength of these architectural decisions and refinements, YSD-

BPTTrack not only lowers the frequency with which calf identities are abnormally swapped under occluded conditions but also markedly improves multi-object tracking accuracy overall.

While our tracking framework shows standout performance, it also has identifiable weak points, among them difficulty handling very long occlusions and highly congested scenes. Separately, we traced the computational bottlenecks of the tracking framework primarily to the footprint of the model in the detection phase and to the sophistication of the trajectory forecasting routine in the tracking phase. Even though trajectory forecasting pushes up the system's computational demand, we have already shrunk the detection model considerably. On balance, these complementary measures lead to noticeable gains in both computational economy and runtime speed.

For the experiments reported here, a single camera served as the sensing source, and our YSD-BPTTrack outperformed rival frameworks in tracking capability. Yet any single camera can only observe a fixed portion of the cattle roaming area, so tracking necessarily halts when an animal wanders outside that coverage. To move past this constraint, we aim to deploy a multi-camera tracking system in upcoming work. A related and equally pressing requirement is to safeguard the continuity of each animal's identity as it transitions between the fields of view of neighbouring cameras.

In the present study, we intentionally avoided incorporating a Re-ID (Re-Identification) module, as doing so would likely cut the tracking frame rate below what is needed for real-time performance. Still, when the setting involves handoffs across multiple cameras, the twin demands of correctly recognizing individual cattle from disparate viewpoints and maintaining high-speed tracking must both be satisfied. Solving this tandem challenge forms a core objective of our next investigations.

Beyond camera topology, we also intend to evaluate the tracking framework on additional datasets and to keep polishing it to confirm its transferability and sturdiness across other cattle breeds and assorted environmental conditions. Progress on these fronts will bring us nearer to offering more potent tracking frameworks for calves placed in heterogeneous surroundings, which in the end will contribute to sharper and more informed monitoring and husbandry decisions.

Conclusion

The present paper puts forward YSD-BPTTrack, a multi-object tracking framework purpose-designed for calves navigating environments with frequent occlusion. On the detection side, two modules—StarNet and DCNv4—are baked into the framework. StarNet trims computational budget without eating into detection quality. DCNv4, meanwhile, fortifies the framework's power to tease out spatial signatures of obstructed objects through the insertion of supplementary offsets. On the tracking side, we introduce a fresh rematching procedure together with a reformulated trajectory deletion protocol. The rematching procedure, more specifically, reconnects detection boxes to enlarged trajectory boxes, thereby cutting down on identity alternations whenever targets re-emerge after being hidden. The trajectory deletion protocol adaptively modifies the timeout governing unmatched trajectories to fit the realities of a cattle yard, lessening the chance of erroneously clearing trajectories. Set against the other models, YSD-BPTTrack registers dramatic jumps on several headline indicators: HOTA advances by 17.6%, MOTA by 13.9%, MOTP by 1.8%, IDF1 by 15.4%, and IDSw contracts by 88.9%. These experimental outcomes collectively position YSD-BPTTrack as a solid, dependable option for multi-object calf tracking.

Acknowledgements

None

Conflict of interest

None

Financial support

This work was supported by the GHfund D [ghfund202407042032]; the Shanxi Province Basic Research Program [202203021212444]; Shanxi Agricultural University Science and Technology Innovation Enhancement Project [CXGC2023045]; the Key Research and Development Program of Zhejiang Province under Grand [2024C01104, 2024001026]; Shanxi Postgraduate Education and Teaching Reform Project Fund [2022YJJG094]; Shanxi Agricultural University doctoral research start-up project [2021BQ88]; Shanxi Agricultural University Academic Restoration Research Project [2020xshf38]; Young and Middle-aged Top-notch Innovative Talent Cultivation Program of the Software College, Shanxi Agricultural University [SXAUKY2024005].

Ethics statement

None

Received: 19 Sep 2022

Revised: 07 Oct 2022

Accepted: 26 Oct 2022

Published online: 15 March 2023

Rights and permissions

Open Access The author(s) retain copyright. This article is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. It may be shared and adapted for non-commercial purposes with appropriate attribution, an indication of changes, and distribution of adaptations under the same license. Third-party material may be subject to separate terms identified in its credit line. View the license at <https://creativecommons.org/licenses/by-nc-sa/4.0/>.

References

- Aquilani C, Confessore A, Bozzi R, Sirtori F, Pugliese C. Review: precision livestock farming technologies in pasture-based livestock systems. *Animal*. 2022;16:100429.
- Velasquez A, Rao S, Roman N, Pinedo P. Early life behavior and disease dynamics of pair-housed Holstein heifer calves. *Appl Anim Behav Sci*. 2023;265:105982.
- Guo H, Ma X, Ma Q, Wang K, Su W, Zhu D. An interactive 3D point clouds analysis software for body measurement of livestock with similar forms of cows or pigs. *Comput Electron Agric*. 2017;138:60-8.
- Dutta D, Natta D, Mandal S, Ghosh N. Moonitor: an IoT based multi-sensory intelligent device for cattle activity monitoring. *Sens Actuators A Phys*. 2022;333:113271.
- Islam M, Lomax S, Doughty A, Islam M, Thomson P, Clark C. Revealing the diversity in cattle behavioural response to high environmental heat using accelerometer-based ear tag sensors. *Comput Electron Agric*. 2021;191:106511.
- Roberts C. Radio frequency identification (RFID). *Comput Secur*. 2006;25:18-26.
- Andrew W, Gao J, Mullan S, Campbell N, Dowsey AW, Burghardt T. Visual identification of individual Holstein-Friesian cattle via deep metric learning. *Comput Electron Agric*. 2021;185:106133.
- Xu B, Wang W, Falzon G, Kwan P, Schneider D. Automated cattle counting using Mask R-CNN in quadcopter vision system. *Comput Electron Agric*. 2020;171:105300.

Ren S, He K, Girshick R, Sun J. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans Pattern Anal Mach Intell.* 2017;39:1137-49.

He K, Gkioxari G, Dollár P, Girshick R. Mask R-CNN. *IEEE Trans Pattern Anal Mach Intell.* 2020;42:386-97.

Jiang B, Wu Q, Yin X, Wu D, Song H, He D. Flyolov3 deep learning for key parts of dairy cow body detection. *Comput Electron Agric.* 2019;166:104982.

Redmon J, Farhadi A. YOLOv3: an incremental improvement. *arXiv:1804.02767.* 2018.

Qiao Y, Guo Y, He D. Cattle body detection based on YOLOv5-ASFF for precision livestock farming. *Comput Electron Agric.* 2023;204:107579.

Wang CY, Bochkovskiy A, Liao HYM. YOLOv7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2023. p. 7464-75.

Awad AI. From classical methods to animal biometrics: a review on cattle identification and tracking. *Comput Electron Agric.* 2016;123:423-35.

Mar CC, Zin TT, Kobayashi I, Horii Y. A hybrid approach: image processing techniques and deep learning method for cow detection and tracking system. In: 2022 IEEE 4th Global Conference on Life Sciences and Technologies (LifeTech). 2022. p. 566-7.

Huang X, Hu Z, Qiao Y, Sukkarieh S. Deep learning-based cow tail detection and tracking for precision livestock farming. *IEEE/ASME Trans Mechatron.* 2023;28:1213-21.

Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, et al. SSD: single shot multibox detector. *arXiv:1512.02325.* 2015.

Aye C, Zin T, Kobayashi I. Black cow tracking by using deep learning-based algorithms. *ICIC Express Lett.* 2022;13:1313.

Wojke N, Bewley A, Paulus D. Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP). 2017. p. 3645-9.

Noe SM, Zin TT, Tin P, Kobayashi I. Comparing state-of-the-art deep learning algorithms for the automated detection and tracking of black cattle. *Sensors.* 2023;23:532.

Du Y, Zhao Z, Song Y, Zhao Y, Su F, Gong T, et al. StrongSORT: make DeepSORT great again. *IEEE Trans Multimed.* 2023;25:8725-37.

Zheng Z, Qin L. PrunedYOLO-Tracker: an efficient multi-cows basic behavior recognition and tracking technique. *Comput Electron Agric.* 2023;213:108172.

Zhang Y, Sun P, Jiang Y, Yu D, Weng F, Yuan Z, et al. ByteTrack: multi-object tracking by associating every detection box. In: Avidan S, Brostow G, Cissé M, Farinella GM, Hassner T, editors. *Computer Vision – ECCV 2022.* Cham: Springer; 2022. p. 1-12.

Luiten J, Ošep A, Dendorfer P, Torr P, Geiger A, Leal-Taixé L, et al. HOTA: a higher order metric for evaluating multi-object tracking. *Int J Comput Vis.* 2021;129:548-78.

Hussain M. YOLOv1 to v8: unveiling each variant—a comprehensive review of YOLO. *IEEE Access.* 2024;12:42816-33.

Ma X, Dai X, Bai Y, Wang Y, Fu Y. Rewrite the stars. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 2024.

Dai J, Qi H, Xiong Y, Li Y, Zhang G, Hu H, et al. Deformable convolutional networks. In: 2017 IEEE International Conference on Computer Vision (ICCV). 2017. p. 764-73.

Xiong Y, Li Z, Chen Y, Wang F, Zhu X. Efficient deformable convnets: rethinking dynamic and sparse operator for vision applications. *arXiv:2401.06197*. 2024.

Aharon N, Orfaig R, Bobrovsky B. BoT-SORT: robust associations multi-pedestrian tracking. *arXiv:2206.14651*. 2022.

Cho Mar C, Zin TT, Tin P, Kobayashi I, Horii Y. Cow detection and tracking system utilizing multi-feature tracking algorithm. *Sci Rep*. 2023;13:17423.

Luo W, Xing J, Milan A, Zhang X, Liu W, Kim TK. Multiple object tracking: a literature review. *arXiv:1909.03596*. 2021.

Li X, Zhang T, Zhao X, Sun X, Yi Z. Learning fused features with parallel training for person re-identification. *Knowl-Based Syst*. 2021;220:106941.

Nasseri M, Moradi H, Hosseini R, Babaei M. Simple online and real-time tracking with occlusion handling. *arXiv:2103.04147*. 2021.