AI

MDPI

Article

Real-Time Camera Operator Segmentation with YOLOv8 in Football Video Broadcasts

Serhii Postupaiev, Robertas Damaševičius * and Rytis Maskeliūnas

Center of Real Time Computer Systems, Kaunas University of Technology, 51423 Kaunas, Lithuania; serhii.postupaiev@ktu.edu (S.P.)

* Correspondence: robertas.damasevicius@ktu.lt

Abstract: Using instance segmentation and video inpainting provides a significant leap in real-time football video broadcast enhancements by removing potential visual distractions, such as an occasional person or another object accidentally occupying the frame. Despite its relevance and importance in the media industry, this area remains challenging and relatively understudied, thus offering potential for research. Specifically, the segmentation and inpainting of camera operator instances from video remains an underexplored research area. To address this challenge, this paper proposes a framework designed to accurately detect and remove camera operators while seamlessly hallucinating the background in real-time football broadcasts. The approach aims to enhance the quality of the broadcast by maintaining its consistency and level of engagement to retain and attract users during the game. To implement the inpainting task, firstly, the camera operators instance segmentation method should be developed. We used a YOLOv8 model for accurate real-time operator instance segmentation. The resulting model produces masked frames, which are used for further camera operator inpainting. Moreover, this paper presents an extensive "Cameramen Instances" dataset with more than 7500 samples, which serves as a solid foundation for future investigations in this area. The experimental results show that the YOLOv8 model performs better than other baseline algorithms in different scenarios. The precision of 95.5%, recall of 92.7%, mAP50-95 of 79.6, and a high FPS rate of 87 in low-volume environment prove the solution efficacy for real-time applications.

Keywords: image segmentation; video inpainting; deep learning; computer vision; sports informatics

Citation: Postupaiev, S.; Damaševičius, R.; Maskeliūnas, R. Real-Time Camera Operator Segmentation with YOLOv8 in Football Video Broadcasts. *AI* **2024**, *5*, 842–872. https://doi.org/10.3390/ai5020042

Academic Editor: Arslan Munir

Received: 4 April 2024
Revised: 24 May 2024
Accepted: 3 June 2024
Published: 6 June 2024

1. Introduction

The role of computer vision (CV) in the world of sports has been significantly increasing over the last couple of decades [1,2]. Employing automated video analysis in sports may be a possible solution to satisfy the demands of fans and professionals for diverse types of structured information. Sports video analysis has a wide range of applications, such as player tracking [3], extraction of ball trajectory [4], 3D-TV streaming and on-demand reconstruction [5], gameplay analysis [6], highlight extraction [7], player and referee action and gesture recognition [8], real-time video broadcast enhancements [9], and many others. The majority of viewers watch sports events via media streaming providers. Therefore, the quality of the broadcast, especially the camera angle and the consistency of the scenes captured, plays a crucial role in attracting and retaining the audience. However, professional camera control, effectively following the game flow, demands years of practice and expertise; therefore, it is not easily achievable for many levels of sports broadcasting.

Football is one of the most popular sports [10], being broadcast to millions of spectators around the world, varying from regional competitions to the World Cup, which is the most viewed football contest. Also, lower-tier leagues often have a limited budget to employ well-qualified broadcast teams. Consequently, their broadcast quality of service (QoS) may suffer, attracting fewer viewers, therefore bringing less revenue. However, even highly professional broadcast teams can run into these problems. One potential reason for the QoS

downgrade is visual distractions, such as unexpected intrusions of people or animals onto the field, spectators covering the camera view with their bodies or fan attributes, and others. Nevertheless, the most frequently overlooked and ignored issue in sports broadcasting is the "camera operators cameo" or "camera operator shots", when one camera operator accidentally captures another camera operator in their shot. This phenomenon regularly happens in live football broadcasts due to complex media standards and large broadcasting teams, which often make errors. These errors are sometimes seen as unintentional or unprofessional elements, which can detract from the viewing experience. This kind of issue places severe constraints on the camera operator team, as they constantly have to film the game while trying to avoid capturing other camera operators, which can lead to a loss of the context of some game moments or makes the broadcast less dynamic and immersive. In addition, camera operators sometimes have to intentionally include their colleagues in shots to ensure critical moments are captured and the resulting game scene is consistent. Such challenges create room for potential video broadcast enhancement by cutting out the camera operator, or unwanted object, from the video frames to increase the broadcast quality of experience (QoE) of the end users of the media system. Enhancements are already being implemented in sports like hockey by camouflaging camera operators against the white ice surface, so they can come out onto the ice rink and capture the scenes in detail during timeouts, score celebrations, and other actions without causing visual distractions [11]. However, football scenes are usually characterized by higher contrast, the larger scope of the game, and camera operators have much more complex types of equipment, which are hard to camouflage. Additionally, camera operators in football are often situated in close proximity to the action, making their continuous removal throughout the entire game necessary, rather than just during specific time intervals. Therefore, handcrafted means insufficiently eliminate distractions; hence, the application of CV enhancements is an essential solution for addressing such challenges effectively.

According to the media and broadcast recommendations [12] defined by the International Association Football Federation (FIFA), there should be multiple cameras strategically positioned across the stadium to provide optimal filming positions for the broadcaster while being seamlessly integrated into seating areas to avoid obstructing spectators' views. The number of cameras required may vary depending on the requirements of the competition. Figure 1 depicts the typical camera positions used during a football match. As it can be seen, the number of camera points in prestigious tournaments starts at nine. The dotted lines in the figure represent camera angles, indicating numerous overlapping camera angle views, which contribute to the visual distraction issues described above.

The purpose of this study was to design a system that is capable of seamlessly removing camera operators from live football broadcast, eliminating visual distractions. The core technology that lies in the process of cutting out camera operators from frames is called video inpainting. According to research, all the state-of-the-art (SOTA) video inpainting methods [13–17] are mask-aware, requiring masked frames with segmented camera operator instances along with video frames as the input to perform inpainting. Consequently, the instance segmentation of football camera operators is required. This research is placed as a subtask of the inpainting system, and its main objective was to develop a segmentation algorithm that is be capable of performing real-time instance camera operator segmentation when deployed in a low-volume environment.

The main contributions of this paper are as follows:

- A new "Cameramen Instances" dataset was created, which could be used by other researchers to refine their image segmentation models;
- A finetuned YOLOv8 instance segmentation model was developed to accurately segment camera operators during football broadcasts, producing masks that serve as the first step in an approach to enhance the quality of the football broadcast;
- The experimental results demonstrate the effectiveness of the YOLOv8 model in segmenting camera operators, providing the foundation for future integrations with a

video inpainting model to seamlessly remove camera operators from broadcasts and enhance overall broadcast quality.

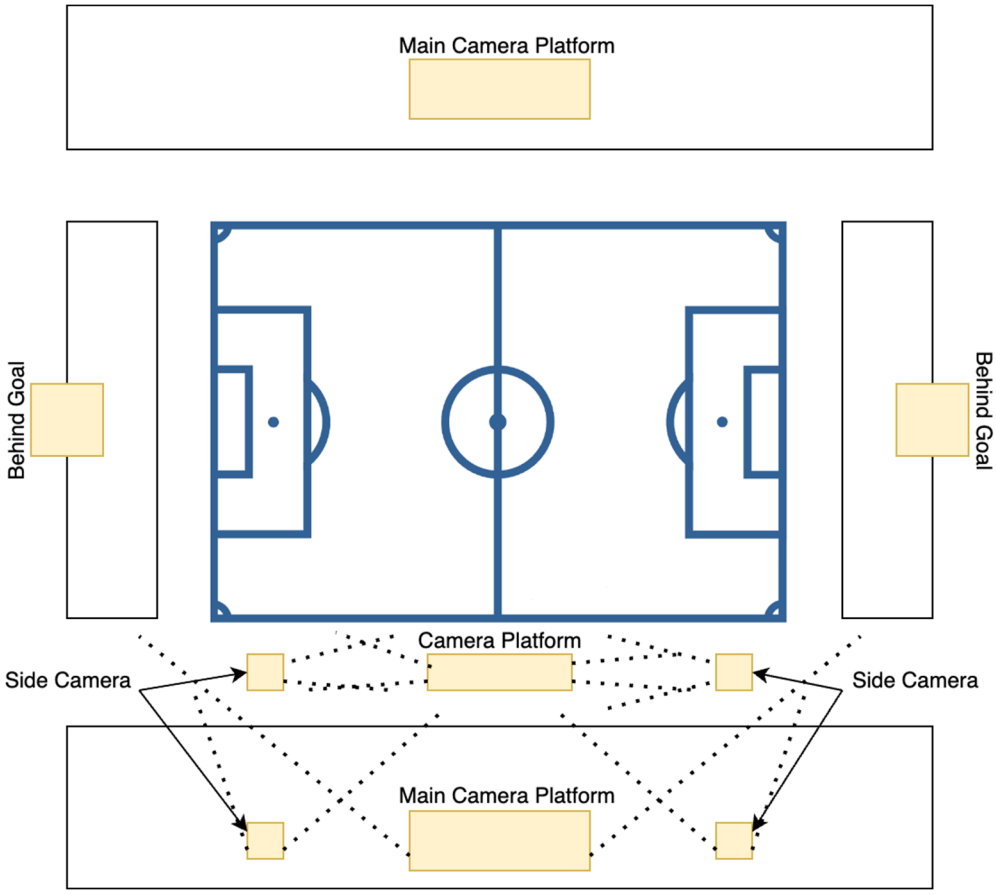

Figure 1. Strategic camera positions and overlapping angles of view (dotted lines) in a football stadium.

The subsequent sections of this paper are structured as follows: Section 2 provides a background on previous research in the field of instance segmentation methods; Section 3 presents the materials used and the applied methodology; Section 4 details the experiments and their results, providing insights into the effectiveness of the approach; Section 5 provides a discussion of the findings and marks potential prospects for future research. Finally, Section 6 represents the conclusions, summarizing the key findings of this paper.

2. Literature Review

In the rapidly evolving domains of artificial intelligence and deep learning, the field of CV has emerged with various outstanding approaches produced in the development of instance segmentation techniques. In contrast to conventional image analysis methods, instance segmentation provides detailed identification of individual-object instances down to the pixel level with improved resolution and accuracy.

In recent years, multiple segmentation algorithms have been introduced. These are FCN [18], Mask R-CNN [19], SegNet [20], DeepLab [21], PANet [22], PointRend [23], PSP-Net [24], SOLO [25], and YOLACT [26]. Among these, FCN is a fully convolutional network built upon existing classification networks such as AlexNet [27] and others and fine-tuned for segmentation tasks. Mask R-CNN is an extension of Faster R-CNN [28], having the capability to predict segmentation masks, with the RoIAlign module specifically designed to eliminate alignment inconsistencies. SOLO, on the other hand, uses a direct prediction method, eliminating the requirement for a region of interest (RoI) and, therefore, presenting an efficient solution. SegNet is a deep fully convolutional neural network developed for

pixel-wise segmentation, featuring a novel encoder–decoder module, where the decoder uses max-pooling indices for nonlinear upsampling, which provides competitive segmentation performance. The creators of DeepLab introduced novel techniques by applying convolution with upsampled filters, proposing atrous spatial pyramid pooling (ASPP), and combining deep convolutional neural networks (DCNNs) with probabilistic graphical models to enhance segmentation performance. PANet enhances feature information flow, providing robust support for small instances, while PointRed considers the segmentation challenge as a rendering task, outputting high-quality details along object boundaries. PSPNet exploits the features of global context information through a developed pyramid pooling module. YOLACT is a segmentation model that generates a set of prototype masks for the input image and linear combination coefficients for each detected object instead of predicting raw masks for each instance.

A separate note is required for architectures based on the transformer encoder–decoder structure, which have proved their effectiveness in instance segmentation since the introduction of DETR [29]. MaskFormer [30] demonstrates that using DETR-based mask classification performs well and achieves SOTA performance in panoptic and semantic segmentation. Mask2Former [31] also demonstrated the utility of such architectures for image segmentation using mask classification formula. The creators of SegFormer [32] implemented a powerful segmentation framework that unifies transformers with lightweight multilayer perceptron (MLP) decoders. Other researchers developed OneFormer [33], the first multitask universal image segmentation framework based on transformers and task-guided queries that combines semantic, instance, and panoptic segmentation into a single all-in-one architecture.

In addition, we overview the methods focused specifically on segmenting objects in videos. Researchers presented a new computer vision task, called video instance segmentation. To handle this task, they introduced Mask-Track R-CNN [34], a method that was built on top of Mask R-CNN with a novel tracking branch added. The network performs segmentation and tracking tasks simultaneously. DVIS [35] is a novel approach that is able to handle video instance, semantic, and panoptic segmentation tasks. The method decouples segmentation into tracking, refinement, and segmentation steps. The novel components of the model track objects frame-by-frame and produce spatiotemporal representations based on prealigned features. Near-online video instance segmentation, or NOVIS [36], is a transformer-based model that generates spatiotemporal masks for video clips and performs instance tracking between frames via overlap embeddings. GLEE [37] introduces a general object visual foundation method that is directly applicable to a wide range of object-level image and video tasks, including instance segmentation. IDOL [38] is one of the first successful frameworks for online video instance segmentation. The highlight of this method is that it uses the learned information prior to the embedding to reidentify missing instances due to occlusions to ensure the consistency of associations. Tube-Link [39] is a near-online instance segmentation framework that processes video clips to generate spatial–temporal tube masks. The method integrates tube-level linking with attention and employs temporal contrastive learning for feature discrimination.

Other model series like You Only Look Once (YOLO) have also significantly contributed to the development of instance segmentation methods. For example, YOLOv5-seg was introduced as an adaptation of YOLOv5 [40] to handle mask predictions. Likewise, YOLOv6-seg and YOLOv7-seg were released based on YOLOv6 [41] and YOLOv7 [42], respectively, to handle instance segmentation tasks.

In addition to the above, there are many domains where instance segmentation algorithms can be applied. In agriculture, it provides the possibility to automatically analyze farmland fields, obtain plant biomass characteristics [43], detect plant diseases [44], and perform other various optimization and automation tasks. In medicine, one of the critical applications of instance segmentation methods is the timely detection of human organ diseases, such as brain tumors [45] or polypoid lesions [46]. Instance segmentation is also widely applied in robotics and autonomous driving to ensure scene understanding and

road safety [47,48]. Furthermore, there has been rapid development of unmanned aerial vehicle (UAV) technology, which triggered fundamental changes in different industries, thus creating many possible applications for CV and instance segmentation. These include obstacle detection [49], object tracking [50], aerial image analysis [51], vehicle detection [52], and plenty of other areas where processes can be automated.

Another important realm of CV where the object should be localized is salient object detection (SOD) [53]. SOD is a vital task the CV that focuses on accurately detecting and segmenting visually distinctive image regions from the perspective of the human visual system (HVS). Zhao et al. [54] investigated the use of multicontext deep features for SOD. They employed two CNNs with similar structures to independently capture the global and local contexts of each image superpixel. Kalboussi et al. [55] explored the use of object proposals for segmenting salient objects in videos and presented a novel motion feature based on the optical flow structure tensor to enhance video SOD. Gao et al. [56] developed a method for cosaliency detection in IoT surveillance. The approach identifies common and salient foreground regions in multiple images. The authors implemented a multistage context perception scheme, fulfilled with two-path information propagation for analyzing interimage similarities and differences and a stage-wise refinement process to ensure semantic learning. Babahenini et al. [57] developed a new algorithm for saliency detection that calculates a weight map by normalizing saliency values from input images to distinguish between focused and defocused regions. Li et al. [58] employed multiscale deep features to predict saliency maps. The deep features from three distinct scales were combined and input into a stack of fully connected layers to calculate the saliency score.

Similarly, various studies have integrated CV into sports to enhance the quality of broadcast streams or to generate diverse types of information. For example, Rifai et al. adapted Mask R-CNN and video inpainting to identify badminton players to improve the game analysis process [59]. Other researchers developed a method for real-time human segmentation in sports videos, employing an online knowledge distillation process to adaptively create match-specific networks without manual annotation [60]. The resulting method exhibited exceptional effectiveness in football and basketball games. Notably, Gao et al. presented a novel strategy called simple dual refinement feature pyramid network (SDRFPN), which improved the abilities to handle athletes' features on different scales and fine details during basketball matches [61]. Another article [62] proposed the memory-efficient instance segmentation system (MISS) framework, which effectively integrates a visual prior for instance segmentation tasks, proving to be beneficial in settings with limited data and computational resources. An article [63] presented a unified DeepSportLab framework for player instance segmentation, ball detection, and other tasks in team games. For player instance segmentation, DeepSportLab employes the Panoptic-DeepLab [64] network, the first bottom-up and single-shot panoptic segmentation model that provides SOTA performance. Watanabe et al. proposed a new method that implements foreground silhouette extraction using dynamic object presence probability (DOPP) in sports scenes [65].

Remarkably, football has also seen a burst of research, which has produced advanced solutions to boost various aspects of the game, a few of which are worth mentioning in the context of instance segmentation. The authors of [66] applied the Two-Stage Mask R-CNN approach to track players in football videos. The use of a combination of YOLOv3 [67] and a simple online real-time (SORT) [68] player-tracking approach was developed [69] for accurately tracking the detected object in football videos. Other authors enhanced the performance for the player detection task [70]. They used the U-Net [71] architecture for generating a player probability map. U-Net consists of contracting and extracting paths, whose features are combined to obtain more precise segmentation.

Table 1 presents a comparison of the reviewed studies that further support the conducted analysis of SOTA solutions. The comparison includes YOLOv5 [40], YOLOv8 [72] and YOLOv9 [73] architectures.

Table 1. Comparison of studies based on architectural approach, used backbone, performance metrics, and segmentation tasks (IS, instance segmentation; SS, semantic segmentation; PS, panoptic segmentation; VIS, video instance segmentation; VSS, video semantic segmentation; and VPS, video panoptic segmentation).

Method Name	Architectural Approach	Backbone	Performance Metrics	Segmentation Task
FCN [18]	CNN-based	ResNet/VGG-16	mIoU	SS
Mask R-CNN [19]	CNN-based	ResNet	mAP, AP	IS
Mask-Track R-CNN [34]	CNN-based	ResNet-FPN	mAP, AP, IoU	VIS
SegNet [20]	CNN-based	VGG-16	mIoU, semantic contour measure	Pixel-Wise SS
DeepLab [21]	CNN-based	VGG-16/MobileNet/ResNet	mIoU	SS
PANet [22]	CNN-based	ResNet	mAP, AP	IS
PointRend [23]	CNN-based	ResNet-FPN	mAP, AP, mIoU	IS, SS
PSPNet [24]	CNN-based	ResNet	mIoU, pixel accuracy	SS
U-Net [71]	CNN-based	-	IoU, warp, pixel, and rand errors	Biomedical IS
SOLO [25]	CNN-based	ResNet-FPN	mAP, AP	IS
YOLACT [26]	CNN-based	ResNet-FPN	mAP, AP	IS
DETR [29]	Transformer-based	ResNet	mAP, AP	IS, SS, PS
MaskFormer [30]	Transformer-based	ResNet/Swin-(L, B, S)	mIoU	IS, SS, PS
Mask2Former [31]	Transformer-based	ResNet/Swin	mAP, AP, mIoU	IS, SS, PS
SegFormer [32]	Transformer-based	MiT-B0	mAP, AP	IS
OneFormer [33]	Transformer-based	Swin-L	mAP, AP, mIoU	IS, SS, PS
NOVIS [36]	Transformer-based	ResNet/Swin-L	mAP, AP	VIS
IDOL [38]	Transformer-based	Swin-L	mAP, AP	VIS
Tube-Link [39]	Transformer-based	Swin-L	AP	VIS
YOLOv5 [40]	YOLO-based	CSPDarknet53v1	mAP, AP	IS
YOLOv8 [72]	YOLO-based	CSPDarknet53v2	mAP, AP	IS, VIS
YOLOv9 [73]	YOLO-based	CSPNet	mAP, AP	IS, VIS
DVIS [35]	Complex	ResNet/Swin-L	mAP, AP	VIS, VSS, VPS
GLEE [37]	Complex	Swin-L/EVA02-L	mAP, AP	VIS, VSS, VPS

To complement the analysis provided in Table 1, Table 2 presents detailed performance and accuracy metrics for each method.

In recent years, the YOLO model series has gained significant attention among many researchers due to its good real-time detection capabilities. Particularly, the YOLOv8 models stand out, providing several advantages on reviewed instance segmentation models. The most significant strengths of YOLOv8 are its perfect balance between accuracy and speed, lightweight design, and minimal requirements for computational resources. These aspects place the given model as a suitable option for tasks that require real-time processing speed. Furthermore, all the architectural features and other benefits of the YOLOv8 are analyzed in the subsequent section.

Table 2. Performance and accuracy metrics of reviewed methods based on dataset, key accuracy metric, metric value, frames per second (FPS), and floating-point operations per second (FLOPS).

Method Name	Dataset	Key Accuracy Metric	Metric Value	FPS	FLOPS (G)
FCN-8s (VGG-16) [18]	PASCAL VOC	mIoU	62.2	8	136
Mask R-CNN (ResNet-50) [19]	COCO	mAP	35.4	5	275
Mask-Track R-CNN (RestNet-50) [34]	YouTube-VIS	mAP	30.3	2	320
SegNet (VGG-16) [20]	PASCAL VOC	mIoU	59.1	7	286
DeepLab (ResNet-50) [21]	PASCAL VOC	mIoU	85.7	16	162
PANet (ResNet-50) [22]	COCO	mAP	35.1	4	290
PointRend (ResNet-50) [23]	COCO	mAP	35.8	2.5	400
PSPNet (ResNet-50) [24]	PASCAL VOC	mIoU	85.4	10	225
U-Net [71]	PASCAL VOC	mIoU	79.1	12	180
SOLO (ResNet-50) [25]	COCO	AP	37.1	12	230
YOLACT (ResNet-50) [26]	COCO	AP	29.8	33.5	93.8
DETR (ResNet-50) [29]	COCO	AP	42	28	86
MaskFormer (ResNet-50) [30]	ADE20K	mIoU	45.8	14	180
MaskFormer (Swin-L) [30]	ADE20K	mIoU	53.5	8	268
Mask2Former (ResNet-50) [31]	COCO	AP	42.7	10	236
Mask2Former (Swin-L) [31]	COCO	AP	50.5	5	340
SegFormer (MiT-B0) [32]	ADE20K	mIoU	37.4	50.5	8.4
OneFormer (Swin-L) [33]	COCO	AP	53	9	260
NOVIS (ResNet-50) [36]	YouTube-VIS	AP	41.3	20	200
NOVIS (Swin-L) [36]	YouTube-VIS	AP	59.8	8	260
IDOL (ResNet-50) [38]	YouTube-VIS	AP	49.5	15	210
IDOL [38] (Swin-L)	YouTube-VIS	AP	64.3	9	270
Tube-Link (ResNet-50) [39]	YouTube-VIS	AP	49.5	18	210
Tube-Link (Swin-L) [39]	YouTube-VIS	AP	64.6	9	270
YOLOv5-nano [40]	COCO	mAP	36.1	36.1	7.5
YOLOv8-nano [72]	COCO	mAP	30.5	96.1	12.6
YOLOv9-compact [73]	COCO	mAP	42.2	15	26.4
DVIS (Swin-L) [35]	YouTube-VIS	AP	56.5	11	-
GLEE [37]	YouTube-VIS	AP	56.5	15	210

Notably, during the preparation of this research, the YOLOv9 [73] model was released; however, it is noteworthy that even its smallest model is not as lightweight as YOLOv8's: specifically 7.1 (M) params and 26.4 FLOPS (G) against 3.4 (M) and 12.6 FLOPS (G) for YOLOv9 and YOLOv8, respectively. Given the fact that the main objective of this research was to develop a segmentation algorithm that is capable of performing real-time instance camera operator segmentation within a resource-constrained setting, YOLOv8 was chosen for the baseline model despite the recent release of YOLOv9.

Also, there is the reasonable question as to why a video-segmentation-based method was not considered as a baseline. The field of video instance segmentation is still relatively new, and many of the existing methods require substantial computational resources or only perform offline segmentation. Conversely, YOLOv8 handles video segmentation at decent levels of speed and performance [74–76]. Furthermore, the aforementioned goal of this study was to design a real-time video inpainting system, where the SOTA methods like

that in [14] are quite demanding in terms of resources. Thus, it is critical for an instance segmentation model to consume minimal time and computing resources to allow the video inpainting component enough capacity to perform effectively in real time. Considering all these factors, YOLOv8 was selected as the baseline for this research.

The conducted review of existing solutions indicated a wide range of applications for CV, in particular, for instance segmentation across various domains [77–79], with a growing trend of employing CV in sports, especially in football. However, there is still a noticeable lack of research aimed at improving the quality of video broadcasts by eliminating visual distractions and providing a more immersive experience, pointing to promising directions for future exploration.

3. Materials and Methods

3.1. Cameramen Instances Dataset

Data collection is a crucial point in building accurate and robust solutions. Labeling of an immense number of data is necessary to maximize the potential of the recent and most advanced instance segmentation models. Therefore, aggregating a sufficient number of training images is essential for solving the camera operator segmentation problem. In this study, the data were collected from various online sources that distribute the records of football video matches, highlights, and backstage edits.

Over 100 videos were examined. Frames with a football camera operator were identified within these videos. The presence of the camera operator in these frames led to visual distractions and downgraded the QoS of the broadcast. Then, these frame segments were extracted into frames to serve as the input for the dataset. The camera operators were captured in diverse scenarios during different parts of football gameplay, half-time actions, pregame greetings, and postmatch discussions. To enhance the scope of the dataset, the images were collected under different weather and lighting conditions, camera angles, football game time episodes, places in the stadium, and filming equipment. Furthermore, camera operators were captured in different human poses, with different types of camera equipment setups. It worth noting that the camera operators in the frames were considered units that create visual distractions; thus, all their elements were annotated comprehensively. This means not only a human object was included in the final annotation but also any complex set of equipment they were handling, such as large camera units, protective stands, and others. Moreover, if a camera operator's assistant was present in the frame and overlapped them, this was annotated as a single unit to maintain integrity. Conversely, if the assistant was present separately from the camera operators, they were annotated independently. All these steps were taken to reflect the diverse and complex scenarios encountered in live football broadcasts and provide a robust training foundation for the segmentation algorithms.

The images were captured in JPG format with a resolution of 854 $\times$ 480 pixels. The choice of this resolution was guided by the end goal of designing a system that is capable of real-time camera operator inpainting during football broadcasts. Based on the literature review described in the previous section, it was found that SOTA video inpainting methods require significant processing power as their authors strove to present a generalized solution, often limiting their feasibility for real-time applications. Therefore, the resolution of 854 $\times$ 480 pixels was strategically selected to balance image clarity with segmentation and inpainting processing efficiency. This lower resolution allows faster processing; at the same time, the chosen resolution is sufficient to capture the necessary details to ensure high broadcast QoS. The dataset was split into training, validation, and test sets in an 8:1:1 ratio. As a result, 6542, 514, and 514 images were labeled for the training, validation, and test datasets, for 7570 images in total. Some of the examples are depicted in Figure 2.

Figure 2. Representative dataset samples from this camera operator segmentation study.

The annotations have one class—"camera operator"; the total number of annotations is 11240. The object segmentation dataset format follows YOLO. Detailed dataset metadata are available in Table 3.

Table 3. Dataset metadata.

Dataset	Ratio	Number	Camera Operator Annotations	Format	Resolution (px)
Train	8	6542	9947	YOLO	854 × 480
Valid	1	514	640		
Test	1	514	653		
Total	10	7570	11,240		

The photos were annotated manually by one annotator from the author team using the Roboflow platform [80]. During the annotation process, the polygon tool was utilized to mark the edges of the camera operators, which is required for instance segmentation. This tool allowed us to draw precise polygons around the perimeter of each camera operator, capturing the exact contours of the individual, their equipment, and assistants if present in the frame. To ensure the consistency and accuracy of the polygons across the dataset, a set of clear guidelines was defined for the annotator to follow. These guidelines specified how to handle the instance annotations under various conditions, such as partial visibility, overlapping with other objects, varying scales, lighting conditions, and camera angles. The incoming discrepancies were resolved through discussion with other team members. In addition to this, periodic consistency checks were conducted on a dataset to ensure ongoing fidelity. These processes ensured the high quality and reliability of the annotations, minimizing human error and variability in the dataset. Hereafter, the dataset is referenced as "Cameramen Instances" for simplicity's sake.

3.2. Image Preprocessing

Expanding the dataset size through data augmentation can help with mitigating overfitting during the training process and increase the model's robustness, which, in turn, improves the model's generalization. During the data preprocessing step, multiple techniques were employed to make the dataset more realistic. For example, adjusting the brightness by a fraction helps the model to perform well under various lighting conditions. Random rotation allows scenarios where objects appear in a variety of orientations. Mosaic enhancement is an essential method that outputs a random combination of four images. This procedure forces the model to learn to detect partially occluded and differently positioned objects. Enhancements such as cropping, mirroring, rotation by ±15%, grayscaling, scaling, HSV adjustments, and mosaic augmentation were adopted to increase the size of the dataset (Figure 3). Also, all the images were increased in width up to 864 pixels by applying zero padding, as the image width had to be a multiple of the maximum stride of 32 to be a valid YOLOv8 input.

Figure 3. Dataset augmentation. Mosaic, scaling, HSV adjustments, mirroring, and rotation augmentations are demonstrated. Bounding boxes show the instances of cameraman.

By applying data augmentation, the dataset became more similar to real-world situations and cases, which elevated the dataset's quality as well as the model's generalizability. The resulting dataset may serve as a resource for various specific tasks and facilitate better model training, validation, and testing, providing more relevant and robust outputs in the targeted domain of football camera operator segmentation.

3.3. YOLOv8 Network

To properly implement camera operator instance segmentation, the YOLOv8 [72] network was employed. The main aim in improving the segmentation approach revolves around strengthening the segmentation capabilities, with a particular focus on identifying camera operators in football video broadcasts. To achieve this objective, the capabilities of YOLOv8, an improved version of the original YOLO model, were utilized [81].

The YOLOv8 network is supported by Ultralytics [82], which has a convenient CLI for utilizing recent YOLO versions. Identically to YOLOv5, the authors released five pretrained models distinguished by different scales, featuring varying channel depths and filter numbers. The YOLOv8 framework performs various computer vision tasks, such as object detection, instance segmentation, and object tracking.

Built upon its predecessors, such as YOLOv3 and YOLOv5, YOLOv8 presents innovative features and refinements to further enhance its productivity and flexibility. In contrast to two-stage models, YOLOv8 facilitates the object detection process by directly predicting class boundaries and probabilities. Therefore, it removes the need for a separate region proposal network. Moreover, one of the highlights of YOLOv8 is its adoption of an anchor-free, center-based model for object detection with a decoupled head to independently handle object classification, and regression tasks. This, in turn, offers several benefits over conventional anchor-based methods such as YOLOv5.

YOLOv8 employs an identical backbone network to YOLOv5 but with some changes in the CSPLayers, which is now called the C2f module. It is worth noting that the original 6×6 convolution in the stem was replaced with a 3×3 convolution. For these modifications, refer to the design concept of YOLOv7 ELAN [42], which allows YOLOv8 to retain its lightweight characteristics while gaining more information about the gradient flow, which drastically improves model performance. The C2f module, or cross-stage partial bottleneck with two convolutions, fuses high-level features with contextual information to maximize detection accuracy. Consisting of two modules, ConvModules and "n" DarknetBottleNecks, the C2f module is linked together through Split and Concat operations, where "n" stands for the number of bottlenecks. The ConvModule is composed of the Conv2d-BatchNorm2d-SiLU activation sequence [83]. The C2f module incorporates the bottlenecks' outputs that embed two 3×3 convolutional layers interconnected with residual links (Figure 4).

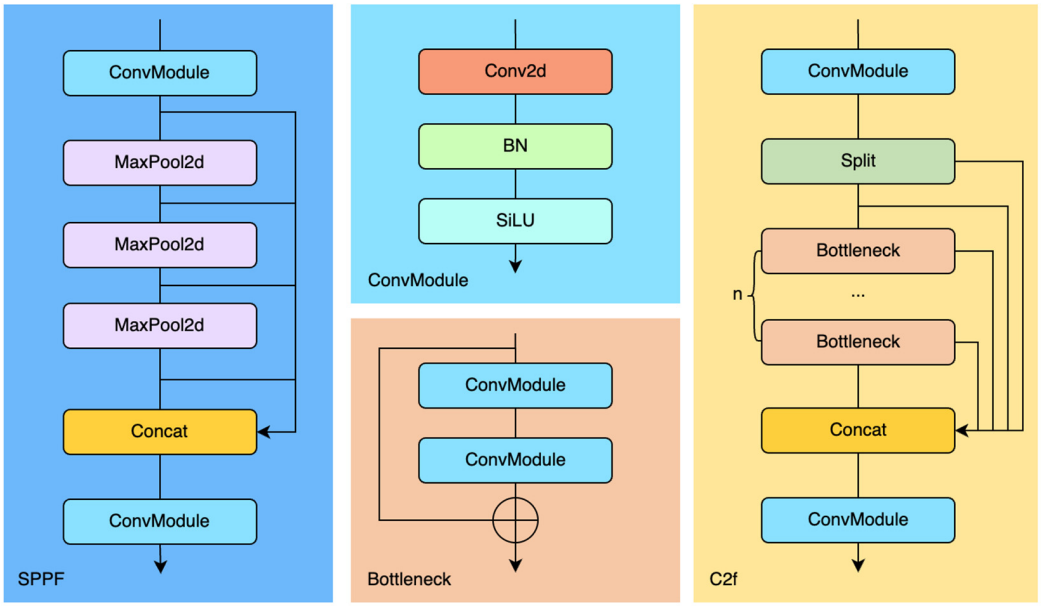

Figure 4. YOLOv8 core modules' architecture. The C2f module integrating ConvModules and DarknetBottleNecks, showcasing the shift to a 3×3 convolution and the advanced feature fusion for enhanced detection accuracy are illustrated. Stages of the architecture are shown in different colors.

YOLOv8 utilizes the spatial pyramid pooling fusion (SPPF). This module extracts the contextual information from images at diverse scales, therefore significantly improving the model's generalization capabilities. The SPPF block is placed at the end of the backbone and takes three max-pooling layers to collectively process features at different scales, which enhances the network's proficiency in feature abstraction (Figures 4 and 5).

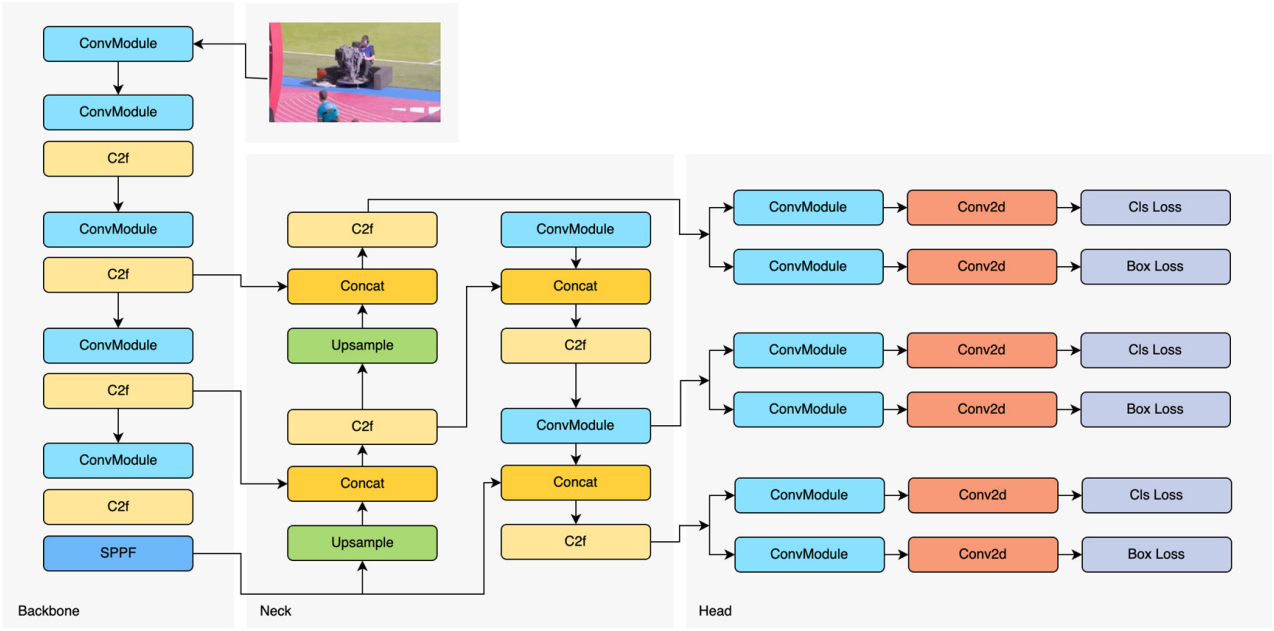

Figure 5. YOLOv8 architecture with backbone, neck, and head components.

In this study, YOLOv8 was chosen as a baseline model. It comprises three core elements: the backbone, the neck, and the prediction output head (Figure 4). The backbone network consists of a 3×3 convolution module, a C2f module, and an SPPF module. The neck network is placed between the backbone and the prediction output head and plays a pivotal role in feature integration at different scales. It consists of the path aggregation network (PAN) [84] and the feature pyramid network [85]. Contrary to previous YOLO versions, YOLOv8 eliminates the 1×1 convolution before upsampling, merges the feature maps obtained directly at separate stages of the backbone network, and then passes them to the head. As opposed to the YOLOv5, which uses the linked head part, the head part of YOLOv8 has a decoupled head structure, delineating the classification and segmentation branches, hence changing the model from anchor-based to anchor-free. This move discards the anchor concepts and employs an optimized method to define positive and negative samples while exceeding the accuracy and processing speed of the anchor-based model.

During the training phase of the model, the TaskAlignedAssigner, employed from task-aligned one-stage object detection (TOOD) [86], was used to assign positive samples relying on weighted classification and regression scores, as shown in Equation (1):

$$t = s^{\mu} \times u^{\beta}, \tag{1}$$

where s stands for the predicted score corresponding to the identified class; u is the intersection over union (IoU) between the predicted and the ground truth bounding box; μ and β are weight values. When the weights are multiplied, the degree of alignment can be evaluated.

The regression branch of the YOLOv8 employs complete intersection over union (CIoU) loss [87] and distribute focal loss (DFL) [88]. CIoU takes into account the constraints related to the center point distance and aspect ratio, and its equation as follows:

$$\begin{cases} L_{CIoU} = 1 - IoU + \frac{\rho^2\left(b, b^{gt}\right)}{c^2} + \alpha\nu \\ \nu = \frac{4}{\pi^2}\left(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w^{p}}{h^{p}}\right)^2, \\ \alpha = \frac{\nu}{(1-IoU)+\nu} \end{cases} \tag{2}$$

where α is used for trade-off balancing; v is used to evaluate the aspect ratio consistency; b and b^{gt} are for the centers of the predicted and ground truth boxes, respectively; w^{gt} and h^{gt} are the width and height of the ground truth container, respectively; w^{p} and h^{p} are the width and height of the prediction container, respectively; ρ signifies the Euclidean distance between the two center points; c denotes the length of the diagonal line of the area that contains both ground truth and prediction boxes at the same time.

DFL operates by letting the model quickly focus on locations close to the target. The distance within a point in a labeled box to the four edges of the predicted frame is assigned to the regression value. DFL is employed to widen the probability distribution around object y so that the network can quickly focus on pixels in the vicinity of the target location, and its equation is as follows:

$$\begin{cases} DFL(S_n, S_{n+1}) = -((y_{n+1} - y)\log(S_n) + (y - y_n)\log(S_{n+1})) \\ S_n = \frac{y_{n+1} - y}{y_{n+1} - y_n} \\ S_{n+1} = \frac{y - y_n}{y_{n+1} - y_n} \end{cases}, \tag{3}$$

where y_n and y_{n+1} are the left and right sides of predicted object y, respectively, $y_n \leq y \leq y_{n+1}$; S_n and S_{n+1} are the probabilities of the y_n and y_{n+1} values, respectively.

As for the classification branch, it leverages binary cross-entropy (BCE) loss, which can be expressed as the following equation:

$$L_{BCE} = -w[y_n \log(x_n) + (1 - y_n)\log(1 - x_n)], \tag{4}$$

where w stands for weight; y_n denotes the labeled value; x_n signifies the predicted value produced by the model.

YOLOv8 was chosen for multiple reasons. First of all, the given algorithm delivers superior performance in instance segmentation tasks, which was our main reason for choosing it for this particular camera operator segmentation challenge. Particularly, YOLOv8 provides real-time segmentation capabilities, which is crucial in the task of swift camera operator segmentation during live football broadcasts to ensure the highest-quality end-user experience. Secondly, the YOLO algorithms are easy to test and deploy. Finally, YOLOv8 is a proven technique with a self-driven community, which opens new avenues for research by providing accessible implementation resources.

3.4. System Model

Figure 6 illustrates the system model of the improved live football broadcast, which involves multiple cameras strategically positioned across the stadium; the cameras are operated by camera operators. The system is intended to improve the viewing experience by eliminating the visual distractions caused by operators shooting at overlapping angles.

The workflow of this approach involves several key phases: Firstly, during the live stream, there might be the case when "Camera Operator A" is captured in the camera angle of "Camera Operator B". This point highlights the importance of handling the overlapping shooting angles and eliminating unwanted visual distractions. Secondly, Camera Operator A is segmented from the video sequence using YOLOv8. By accurately detecting and further isolating the presence of operators in the footage, the system provides the foundation for their removal. Finally, the camera operator is inpainted from the video sequence based on the input video and masked camera operator frames. This phase seamlessly cuts out the camera operator from the video content, therefore improving the overall QoE level for the end users.

One of the most important modules in the system is a YOLOv8 instance segmentation algorithm. The model was meticulously trained to identify and segment camera operators in real time, ensuring accurate mask generation for subsequent inpainting. The training process exploited a robust dataset containing a variety of operator instances in different poses, sizes, and lighting conditions. YOLOv8 was fine-tuned to ensure it can distinguish

camera operators from other entities such as players and spectators, regardless of their shapes and positions.

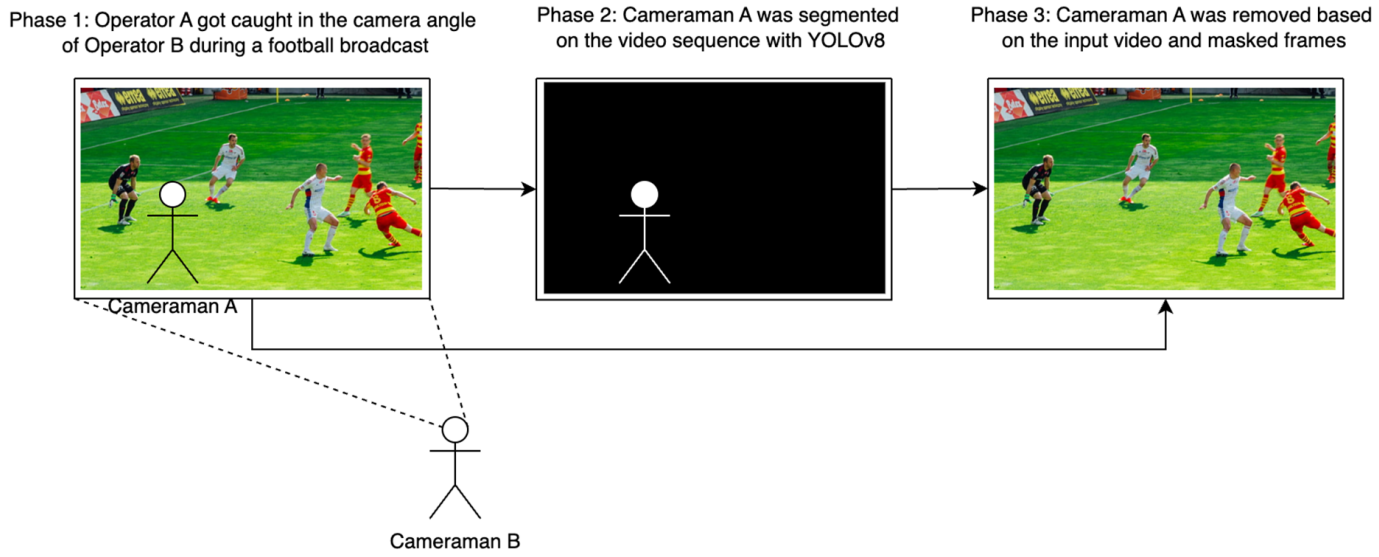

Figure 6. System model of the enhanced live football broadcast. The workflow in which cameras capture the action, the YOLOv8 model segments camera operators, and a video inpainting model removes them for a clearer broadcast is illustrated.

After the YOLOv8 model segments the operators, it outputs masks delineating their presence in each frame. These masks are then processed with a video inpainting model based on the End-to-End Framework for Flow-Guided Video Inpainting (E2FGVI) [14]. The authors of the framework managed to reach SOTA accuracy in removing the objects from the videos relying on masked frames. Video inpainting is a complex process that involves the reconstruction of the background in a way that seamlessly fills in the areas previously occupied by camera operators. E2FGVI performs well in maintaining temporal consistency between frames, ensuring that inpainted areas blend naturally with the surrounding video content. This is achieved by leveraging flow-guided information that helps with preserving the dynamic nature of live football broadcasts.

Comprehensive experiments were conducted to validate the effectiveness of the approach. The YOLOv8 model was thoroughly evaluated on a specially collected dataset consisting of football broadcast coverage with annotated operators. The evaluation metrics included segmentation accuracy, robustness, and real-time processing capability. The YOLOv8 model achieved competitive average precision in camera operator instance segmentation, demonstrating high accuracy in various scenarios, with detailed metrics and performance results described in the subsequent sections.

Furthermore, the robustness of the system was tested on additional video footage from football and other sports games to ensure its generalization capabilities. The system performed well, accurately segmenting and inpainting operators in diverse sports scenes. The quality of inpainting was assessed using subjective methods, such as viewer surveys, and objective metrics, including peak signal-to-noise ratio (PSNR) and the structural similarity index (SSIM). The results demonstrated that the inpainted frames maintained high visual fidelity with minimal artefacts, providing a clear and distraction-free viewing experience.

The entire system operates in real time, which is crucial for live broadcast scenarios. This live performance ensures that the processed frames are immediately available for streaming, allowing spectators to enjoy an uninterrupted experience of the game.

By removing the camera operators from the video sequence, the end users can have a more immersive and distraction-free viewing experience of the live stream. The critical point is that football broadcasting requires capturing dynamic and riveting footage;

therefore, cases of overlapping cannot be avoided by manually changing the camera angle, as meaningful video shots may be missed, which may lead to context loss of the game. Therefore, machine learning (ML)-based solutions should be applied to keep the spectators' QoE at an acceptable level.

The main focus of this study was segmenting camera operators from live football streams. By applying YOLOv8 in the given broadcast system, the ultimate goal was to provide spectators with a real-time, distraction-free playback that allows them to fully embrace the excitement of the game without any visual interference.

4. Experimental Results

4.1. Experimental Environment and Evaluation Metrics

The experimental environment used in this study can be seen in Table 4.

Table 4. Experimental environment setup.

Name	Version
CPU	Intel Core i7-11700 @ 2.50 GHz (Intel Corporation, Santa Clara, CA, USA)
GPU	Nvidia RTX 3080 Ti, 12 GB (Nvidia Corporation, Santa Clara, CA, USA)
Memory	32 GB
Operation System	Ubuntu 22.04, 64-bit
Deep Learning Framework	Python 3.8, Pytorch 2.2.1, CUDA 12.1

Evaluation metrics are crucial for the quantitative assessment of a model's performance. For instance segmentation tasks, metrics such as precision (P), recall (R), and mean average precision (mAP) are commonly applied . These metrics were utilized due to their relevance in assessing instance segmentation models. Moreover, the metrics are standard in evaluating YOLO-based solutions and allow consistent comparison with other models as they are widely used in SOTA solutions. The precision and recall measures offer insights into the model's accuracy and the ability to detect all salient instances, which are crucial for maintaining broadcast quality. The mAP gives a thorough understanding of the model's performance at different levels of segmentation difficulty. Also, this research proposes assessing metrics such as floating-point operations per second (FLOPS), model size, seconds per frame (SPF), and frames per second (FPS). These criteria were employed as they were primary standards in evaluating the real-time capabilities of our model, reflecting its computational efficiency, processing speed, and frame handling effectiveness in a live sports broadcasting context.

In combination, all the mentioned metrics ensured a comprehensive evaluation of the solutions, as they align with common practices in the instance segmentation field, enabling a meaningful comparison with other existing methods.

Precision and recall are calculated with Equations (5) and (6):

$$P = \frac{TP}{TP + FP}, \tag{5}$$

$$R = \frac{TP}{TP + FN}, \tag{6}$$

where TP stands for true positives, or the number of objects correctly identified as a camera operator, indicating accurate detection and labeling; FN, or false negatives, refers to the number of camera operator objects that were present but not accurately detected; FP (false positives) represents the number of objects that were incorrectly classified as a camera operator, highlighting the regions that were misidentified as football camera operators.

Precision measures the proportion of precisely labeled camera operators out of all the predicted cases of occurrence. It is a metric that assesses prediction accuracy. In real-time camera operator segmentation, precision denotes the level of confidence in correctly identifying positive instance segmentation. Greater precision reduces false detections, which results in a higher-quality football broadcast.

Recall signifies the proportion of objects detected. For camera operator segmentation, this metric is also significant as a higher value indicates that more episodes in the broadcast are delivered with maximum quality to the spectators. In other words, recall denotes the ratio between correctly identified targets out of the total number of targets.

The mAP metric is based on the precision–recall criterion, which operates on multiple object classes and determines positive predictions by employing the intersection over union (IoU). It picks a given IoU threshold and calculates the mean of the precision values observed at different recall thresholds. IoU is commonly used for computer vision and image processing tasks to quantitatively assess the degree of overlap between two regions, serving as a measure of similarity. IoU can be calculated as in Equation (7):

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|}, \tag{7}$$

The average precision (AP) is calculated as a weighted average of the precision for each threshold; weight is the increase in recall over the previous threshold. AP is expressed as follows in Equation (8):

$$AP = \sum_n (R_{n+1} + R_n) \cdot P_n, \tag{8}$$

$mAP0.5$ denotes the AP value when the IoU threshold is set to 0.5. To yield $mAP0.5 : 0.05 : 0.95$, the AP for each IoU threshold value ranging from 0.5 to 0.95 in an increment of 0.05 was calculated and averaged, as represented in Equation (9):

$$mAP0.5 : 0.05 : 0.95 = \frac{AP_{IoU=0.5} + AP_{IoU=0.55} + \ldots + AP_{IoU=0.95}}{n}, \tag{9}$$

With help of $mAP0.5$ and $mAP0.5 : 0.05 : 0.95$, the model segmentation capabilities in the football camera operator domain was evaluated at distinct IoU threshold values. For the sake of simplicity, the designation of mAP50-95 is used in the manuscript instead of $mAP0.5 : 0.05 : 0.95$; also, mAP50 is used instead of $mAP0.5$.

In addition, the performance of the suggested model was evaluated using its number of training parameters, which reflects the model size. Furthermore, the FLOPS, FPS, and SPF metrics were employed. FLOPS is a measure of model computational performance and a gauge of its complexity, while FPS signifies the speed at which the model can process each frame for camera operator instance segmentation in a football video stream, which can help with analyzing the model from the perspective of real-world applications. SPF was employed as it offers a direct measure of the time required to process a single frame. By expressing performance using SPF, it might be easier to observe and quantify small differences in model throughput during comparative analysis.

4.2. Experimentation Settings

The YOLOv8 algorithm was applied to the previously created camera operator dataset. All the experiments were conducted using the environment described in Table 2. A subset of the dataset (2000 images) was used to train the model for a limited number of epochs to analyze the model's behavior based on the experimental changes in hyperparameters and architecture while preserving time and computational resources. The model was trained for 15 epochs, using an 8/1.5/0.5 train/validation/test split of the camera operator subdataset. The experiments were divided into optimizer experiments, model size experiments, batch size and learning rate experiments, and layer freezing (Table 5).

Table 5. YOLOv8 experiment plan: architecture modifications and hyperparameter tuning for camera operator segmentation study.

Experiment No.	Batch Size	LR	Optimizer	Momentum	Epochs	Model	Frozen Layers
1	16	0.01	SGD	0.937	15	nano	--
2	16	0.001	Adam	0.9	15	nano	--
3	16	0.02	AdamW	0.9	15	nano	--
4	16	0.02	AdamW	0.9	15	small	--
5	8	0.0001	SGD	0.9	15	nano	--
6	28	0.1	SGD	0.99	15	nano	--
7	16	0.01	SGD	0.937	15	nano	15
8	16	0.01	SGD	0.937	15	nano	20

The rationale for hyperparameter selection was taken from [89], where the authors performed hyperparameter tuning for YOLOv5 for underwater detection. Thus, the objective of this batch of experiments was to investigate whether the model with alternative hyperparameters outperformed the one with the default settings, when applied to the Cameramen Instances dataset. The first experiment in Table 4 was assumed as the default YOLOv8 hyperparameter setting.

The YOLOv8 "nano" and "small" models were selected for experimentation due to their lightweight nature and the limitations of the experimental environment. The models were pretrained on the COCO [90] dataset; the full model configuration is described in Table 6.

Table 6. Characteristics of YOLOv8 iterations employed during the hyperparameter tuning experiments.

Model	mAP50-95	Params (M)	FLOPS (G)	Model Size (MB)
YOLOv8n-seg	30.5	3.4	12.6	6.7
YOLOv8s-seg	36.8	11.8	42.6	22.8

After completing the hyperparameter tuning experiments, the optimal parameters were selected for the YOLOv8 model. Then, the model was trained on the full dataset to create a solution capable of segmenting the camera operator instances. The outcomes of this training phase were extensively analyzed and compared with the quantitative results of the other existing SOTA approaches. Notably, all the models were pretrained on the COCO dataset and then further fine-tuned on the developed Cameramen Instances dataset for the sake of experimental integrity. The detailed plan of the experiments is outlined in Table 7.

Table 7. Comparison plan of segmentation performance of different models on the camera operator instances dataset.

Model	Pretraining Dataset	Training Dataset
YOLOv5n-seg [29] YOLOv7-seg [31] YOLACT [28] Mask R-CNN [19] YOLOv6n-seg [30] YOLOv8n-seg	COCO	Cameramen Instances dataset

The models taken for comparison were trained using the same set of hyperparameters as the models we used for our experiments.

4.3. YOLOv8 Experiments

To correctly analyze the hyperparameter tuning results for the YOLOv8 method, some additional metrics were considered to supplement the ones described at the beginning of the section. These were segmentation loss, classification loss, and training time. Table 8 lists the results of the experiments by providing the key metrics for each model.

Table 8. YOLOv8 hyperparameter tuning: evaluating performance and outcomes.

Experiment No.	P (%)	R (%)	mAP50	mAP50-95	Segmentation Loss	Classification Loss	Training Time (h)
1	94.2	85.4	92.7	68.9	0.873	0.4191	0.070
2	88.6	90.8	94.9	71.2	0.827	0.3451	0.071
3	79.2	82.5	89.9	64.5	1.288	0.6849	0.071
4	84.7	82.7	89.0	62.4	1.488	0.8898	0.128
5	69.9	63.9	72.5	51.7	1.492	2.258	0.083
6	87.1	76.7	89.0	57.7	1.522	0.7802	0.086
7	83.3	83.4	87.1	61.5	1.193	0.6159	0.057
8	68.4	66.2	72.8	47.7	1.494	0.9455	0.082

Experiment 1 and Experiment 2 exhibited commendable precision and recall values; however, the second experiment had in a higher mAP50-95 value, indicating its superior performance among all the experiments in this batch. The experiments with the full model size required dramatically longer training times, resulting in performance decreases, in comparison to those of previous experiments; however, this observation is in line with the expected behavior, as larger models may tend to converge slower due to their higher complexity and parameter count. During Experiment 7, it was agreed to freeze the first 15 layers of the YOLOv8 network to check if the model would benefit from preserving low-level features and allowing certain adaptations to mid-level features. The given experiment is noteworthy for the balanced precision and recall metrics, denoting this hyperparameter setup may be a good trade-off between generic and task-specific features. Moreover, the mAP value was comparable to that of the frontrunner experiments, and the resulting model had the shorter training time. This combination of results makes it an attractive choice in comparison to the other experiments. The rest of the experiments yielded suboptimal results: a higher number of frozen layers downgraded the performance (Experiment 8). Changing the batch size and learning rate to extreme values resulted in increased losses and reduced performance (Experiments 5 and 6).

After the analysis of the conducted experiments with hyperparameter tuning for YOLOv8 in the task of camera operator instance segmentation, we chose Experiment 2 as the best hyperparameter set due to its remarkable balance between precision, recall, and segmentation accuracy, supported by the highest mAP50-95 value of the candidates. The main hyperparameters used during Experiment 2 were as follows: a batch size of 16, a 0.001 learning rate, and the Adam optimizer with 0.9 momentum. Also, the model was trained for 15 epochs. Therefore, based on the analysis, the defined hyperparameter set is the most promising option for training the model, which enables real-time camera operator localization by providing accurate masks of the target object.

4.4. Comparison Experiments

In the experiments outlined in the previous section, the optimal hyperparameter settings were defined as follows: the batch size of 16, a 0.001 learning rate, and the Adam optimizer with 0.9 momentum. Therefore, this configuration was used to train the resulting model for 50 epochs, aiming to achieve maximal performance. The training graph is depicted in Figure 7, indicating a gradual mAP improvement during the first 30 epochs. The highest mAP50-95 was captured during the 42nd epoch.

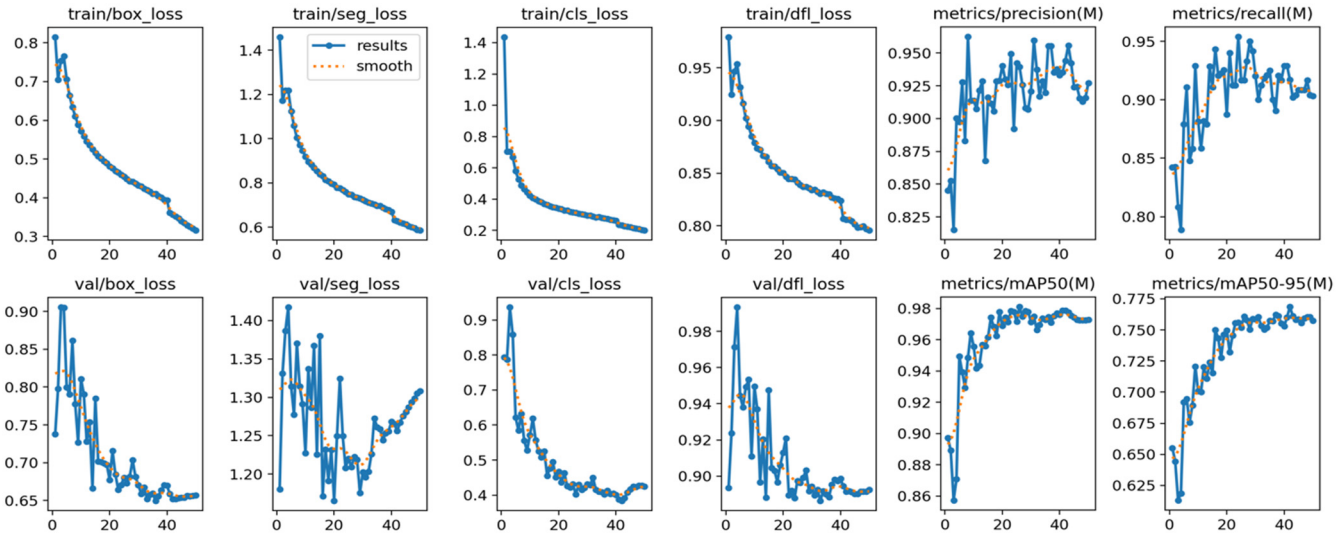

Figure 7. YOLOv8 training graph with 50 epochs. The evolution of box, classification, and DLF losses for training and validation are depicted, in addition to the discussed mAP, precision, and recall.

Afterward, the performance metrics of the resulting model were evaluated and compared to those of different algorithms such as YOLOv5, YOLOv6, YOLOv7, YOLACT, and Mask R-CNN (Table 9). Table 9 presents a detailed comparison of the several existing segmentation algorithms, providing their precision, recall, mean average precision, number of trainable parameters, FPS, SPF, FLOPS, and model size.

Table 9. Comparative analysis of different algorithms: performance testing of YOLOv5, YOLOv6, YOLOv7, YOLACT, Mask R-CNN, and proposed YOLOv8.

Name	P (%)	R (%)	mAP50	mAP50-95	Params (M)	FLOPS (G)	Model Size (MB)	FPS	SPF
YOLOv5n-seg	77.0	82.9	85.8	56.0	2.0	6.7	15.7	62	0.016
YOLOv6n-seg	69.2	91.5	92.6	64.5	4.9	25.46	64.6	30	0.033
YOLOv7	96.9	93.3	98.2	68.7	37.8	141.9	340.0	23.31	0.043
YOLACT	62.8	75.3	65.7	40.83	39.5	158.26	123	19.8	0.050
Mask R-CNN	73.4	89.2	90.5	68.5	44.4	134.3	176.2	24.6	0.040
YOLOv8n-seg	95.5	92.7	97.9	79.6	3.4	12	6.8	87	0.011

Based on the collected metrics, YOLOv8 produced superior performance compared with other models like YOLOv5, YOLOv6, and YOLACT. Notably, YOLOv7 and Mask R-CNN exhibited competitive results in terms of mAP, precision, and recall; particularly, YOLOv7 achieved the results that were most comparable to those of YOLOv8 in the task of camera operator segmentation. However, their relatively high FLOPS and SPF values and low FPS values (134.3, 0.040, and 24.6 for Mask R-CNN, and 141.9, 0.043, and 23.31 for YOLOv7) collectively indicate the poor ability of these models to perform in the described test environment in real time due to their computational complexity.

On the test dataset, YOLOv8 achieved exceptional performance with the following results: 95.5% precision, 92.7% recall, and 79.6 mAP50-95, demonstrating the ability to accurately segment camera operators in video frames. Moreover, YOLOv8 emerged as the optimal choice for real-time football broadcast systems requiring camera operator inpainting based on the FLOPS, SPF and FPS metrics, which were 12, 0.011, and 87, respectively. Specifically, the SPF value for YOLOv8 can be decomposed into multiple stages, such as preprocessing (0.1 ms), inference (1.0 ms), and postprocessing (1.4 ms). The postprocessing step takes the longest time compared to the other components, suggesting potential areas for future optimization research. In general, with an inference time of 1.0 ms, the model

very efficiently uses its neural network architecture. The YOLOv8 algorithm not only provides high-accuracy segmentation but is also employable in real time, making it the definitive approach for camera operator segmentation tasks.

4.5. Effect of Different Image Resolutions on Camera Operators Segmentation

To investigate the impact of input resolution on the segmentation results of the trained YOLOv8 network, several experiments were conducted with various input image sizes that are commonly used during broadcasts with an adaptive bitrate to cover as many users with different devices, as possible: 640 × 360 pixels, 854 × 480 pixels, 1280 × 720 pixels, and 1920 × 1080 pixels. A separate small test dataset containing about 100 images was collected for these experiments. The YOLOv8-based model tested in this experiment was trained on an 854 × 480 pixel dataset, described in Table 2. Table 10 provides a detailed overview of each image resolution's segment precision, recall, mAP, and inference time in milliseconds.

Table 10. Comparative analysis of network segmentation results for different input resolutions.

Resolution (px)	P (%)	R (%)	mAP50	mAP50-95	Inference Time (ms)
640 × 360	96.5	74.3	85.8	60.6	129.8
854 × 480	96.4	76.9	90.7	64.9	198.4
1280 × 720	83.1	82.9	91.8	68.9	461.4
1920 × 1080	70.7	72.4	80.7	60.0	1026.8

The results show a general decrease in precision as the resolutions increase, degrading from 96.4 at 854 × 480 to 70.7 at 1920 × 1080, indicating the model struggles to maintain accuracy with increasing resolution. However, the recall increased when the resolution was changed from 854 × 480 to 1280 × 720 and then dropped at the highest resolution, indicating that the model can detect more true positives at a slightly higher resolution, but fails as the resolution significantly exceeds the training conditions.

The pattern for mAP50-95 depicts peak performance at 1280 × 720 resolution, indicating the model's better generalization at this resolution before downgrading the performance level at the highest resolution tested. In addition, there was an expected increase in inference time with the growth in resolution, reflecting the substantial increase in time between 1280 × 720 and 1920 × 1080 resolutions.

This batch of experiments shows that the model demonstrates optimal performance at 854 × 480 and 1280 × 720 resolutions, where it reaches efficient precision, recall, and mAP levels. These resolutions offer satisfactory segmentation performance and a reasonable inference time.

4.6. Cross-Sport Evaluation of Camera Operator Segmentation

We extended the evaluation of the YOLOv8 camera operator instance segmentation to other sports. Table 11 summarizes the performance metrics, such as precision, recall, and mean average precision, for the analyzed sport types (rugby and lacrosse) for 100 movie sequences each, which were parameterized and processed in the same manner as described in Section 3.1.

Table 11. Performance metrics for camera operator segmentation in various sport broadcasts.

Sport Type	P (%)	R (%)	mAP50	mAP50-95	Resolution (px)
Rugby	96.3	83.3	90.5	69.6	854 × 480
Lacrosse	93.8	79.4	88.2	65.0	

The model achieved high precision and a notable recall rate for the both sports, resulting in competitive mAP50 and mAP50-95 values, suggesting that the model is effective

at correctly identifying and segmenting the camera operator with minimal false positives. The conducted experiment shows that the model generalizes well across different sports with varying visual contexts.

4.7. Statistical Tests

Statistical tests were performed to evaluate the conducted experiments to determine whether the observed differences in performance metrics like mAP50-95, precision, and recall were due to the specific adjustments made in the experiments rather than being a result of random variability in the data (all the details are available in the repository referenced in the Data Availability Section). The one-sample *t*-test method was applied to compare the target metric scores of the used YOLOv8 model against the average metric scores of the other experiments. The one-sample *t*-test was selected as it simplifies the process of comparing a target model against other experiments. Moreover, the test is particularly suited for small sample sizes and offers a straightforward analysis that provided inputs for the focused assessment in the conducted experiments. This approach allowed us to quantify how much better or worse the proposed model was compared to the other models. The t-statistic from the test measures the magnitude of the difference in units of standard error, while the *p*-value gives the probability that the observed difference in performance occurs by chance. Such quantification helps to define not just if the proposed model is different but also how significant that difference is in a statistical context.

Table 12 presents the results of the *t*-tests for the YOLOv8 hyperparameter tuning experiments. The null hypothesis was that any difference in the mAP50-95, precision, or recall score observed for Experiment 2 was due to random chance or variability in the data, rather than a significant effect of the hyperparameter tuning performed in that specific experiment.

Table 12. One-sample *t*-test results for hyperparameter tuning experiments: evaluating significance of mAP50-95, precision, and recall scores.

Metric	*p*-Value	t-Statistics Magnitude	*p*-Value Threshold
MAP50-95	0.0051	4.291	
Precision	0.0723	2.177	0.05
Recall	0.0064	4.082	

The *t*-test results show that Experiment 2 significantly outperformed the others in two out of three key evaluation metrics: mAP50-95 and recall, with *p*-values of 0.0051 and 0.0064, respectively, which are both below the 0.05 threshold, so the null hypothesis was rejected for these metrics. At the same time, the magnitude of the t-statistics was substantial enough to provide a high level of confidence in the results. Therefore, the difference in the mean mAP50-95 and precision between Experiment 2 and the other experiments implies that the changes made in Experiment 2 positively influenced the model's performance. Although the enhancements in precision did not reach statistical significance, with a *p*-value of 0.0723, it did exhibit a positive trend that may be useful in practical applications.

Table 13 presents the results of the *t*-tests for the comparative analysis of the YOLOv8 method against the other SOTA approaches. The null hypothesis was that any difference in the mAP50-95, precision, and recall score observed for the YOLOv8 model was due to random chance or variability in the data rather than a genuine effect or improvement brought about by the YOLOv8 model.

The *t*-test results in the SOTA comparison indicate that the performance metrics of the YOLOv8 model were significantly better than those of the other models. With a *p*-value of 0.0192 and a t-statistic magnitude of 3.789 for mAP50-95, and a *p*-value of 0.0271 and a t-statistic magnitude of 3.406 for precision, we rejected the null hypothesis for these metrics, confirming that the proposed enhancements were statistically significant. The recall metric, however, with a *p*-value of 0.1301, did not surpass the *p*-value threshold of 0.05, indicating a slightly less robust difference compared to the other models. Nevertheless, the recall value

of the YOLOv8 method was relatively high and comparable with that of the considered SOTA models.

Table 13. One-sample *t*-test results for SOTA comparison experiments: evaluating significance of mAP50-95, precision, and recall scores.

Metric	*p*-Value	t-Statistics Magnitude	*p*-Value Threshold
MAP50-95	0.0192	3.789	
Precision	0.0271	3.406	0.05
Recall	0.1301	1.901	

4.8. Visualization

Figure 8 visualizes some results of the applied method on the Cameramen Instances dataset. The selected images represent the common frames that are frequently captured during football broadcasts. These include the moments when the players enter the field to start the game and several moments when the primary camera operator captures side angles, which are on the edge of the football field. Notably, the model demonstrates great detection and segmentation capabilities, avoiding false positive outputs. The model effectively handles images containing multiple camera operator instances, which is a regular case in real-world scenarios.

Figure 9 illustrates a visual contrast of the different algorithms used for comparison with the YOLOv8 model. As mentioned before, YOLOv5, YOLOv6, YOLOv7, YOLACT, and Mask R-CNN were selected to qualitatively assess the visual outputs of the proposed method. Here, the YOLOv5 and YOLOv6 models suffered from numerous false positives in the segmentation results, mixing up instances like coaches, assistants, referees, and others. Moreover, they generated erratic mask outputs, which often contained redundant areas in the images. The YOLACT network aggregated the segmentation results relatively well; however, it produced severe mask expansion beyond the ground truth regions for most of the test dataset.

Mask R-CNN and YOLOv7 had the closest results to those of the YOLOv8 model. YOLOv7 had minimal trouble with accuracy in the segmentation masks and sometimes failed to fully encompass the objects. Mask R-CNN generated fairly accurate masks but had a sufficient number of false positive detections that were closely located to the ground truth objects. In contrast, YOLOv8 exhibited superior performance, generated relatively accurate masks, and avoided false detections, which were evidenced by both the qualitative (as illustrated in Figure 8) and quantitative results detailed in the previous subsection.

Additional visual analysis experiments were conducted to assess the model's adaptability and generalization by applying our model to other sport broadcasts where the problem addressed in this research may be relevant. Figure 10 illustrates a visual comparison using rugby and lacrosse as examples. These selected sports have similar field sizes and operator positions during the game. The results are promising and exhibit the model's effectiveness in camera operator segmentation in diverse sports environments. This extended evaluation provides insights into the model's potential for wider application in live sports broadcast environments, paving the way for future enhancements in the field of automated sports broadcasting.

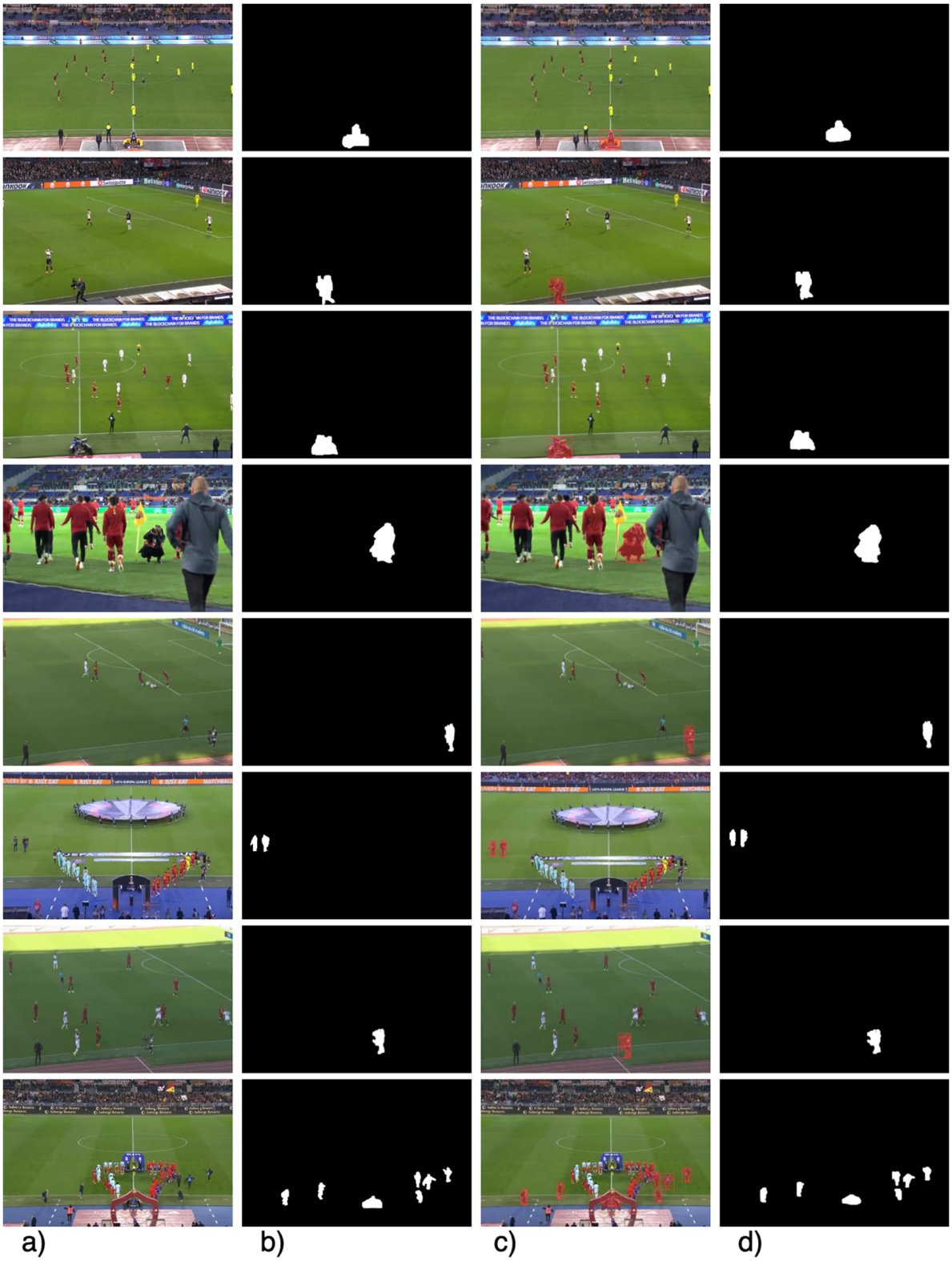

Figure 8. Segmentation results for the proposed YOLOv8 model for single- and multi-camera operator instances in the aggregated dataset's test images: (**a**) original images; (**b**) ground truth binary mask; (**c**) segmented camera operators outlined in red contour; and (**d**) segmented camera operator binary mask.

Figure 9. Segmentation results of the compared models for single- and multi-camera operator instances in the aggregated dataset's test images (outlined in red contour): (**a**) original images with ground truth segmentations; (**b**) YOLOv5 segmentations; (**c**) YOLOv6 segmentations; (**d**) YOLOv7 segmentations; (**e**) YOLACT segmentations; (**f**) Mask R-CNN segmentations; (**g**) YOLOv8 segmentations.

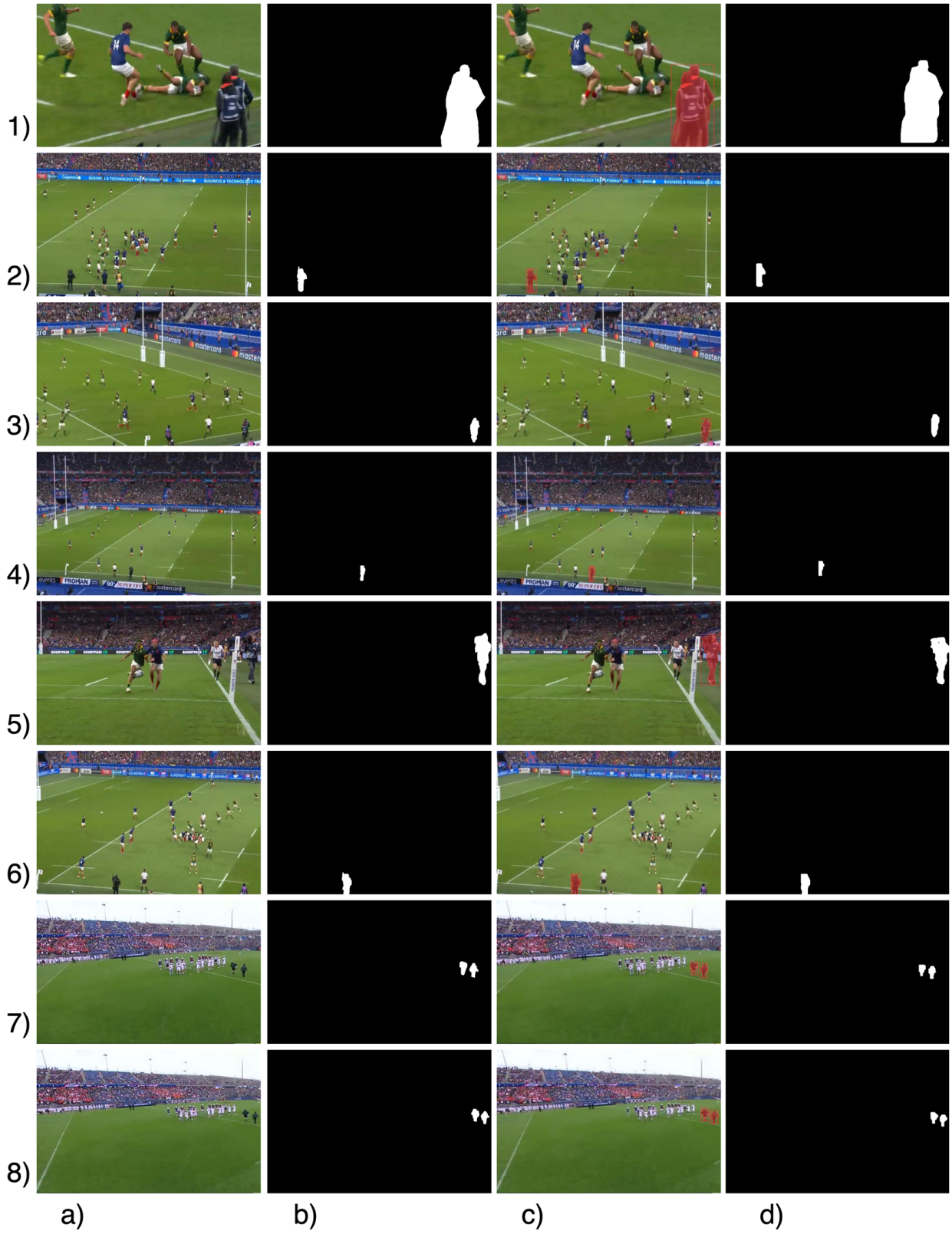

Figure 10. Segmentation results for the proposed YOLOv8 model for single- and multi-camera operator instances in the aggregated dataset's test images for other sport types: (**a**) original images; (**b**) ground truth binary mask; (**c**) segmented camera operators outlined in red contour; and (**d**) segmented camera operator binary mask. Images (**1**)–(**6**) represent rugby, and images (**7**,**8**) represent lacrosse.

5. Discussion and Future Research

5.1. Evaluation of Findings

Our integration of YOLOv8 into computer vision tasks achieved higher accuracy than previous iterations of YOLO models, highlighting it as a flexible and efficient approach for solving a variety of tasks such as object detection, instance segmentation, and image classification. A recent YOLO release introduced new features and enhancements to further optimize its performance and flexibility, including a new backbone network, an anchor-free detection head, and a revamped loss function. YOLOv8 exhibits superior accuracy, producing a competitive solution compared with other SOTA instance segmentation models. Also, the proposed model is exceptionally efficient and can be deployed in a low-volume environment, which makes it a perfect candidate for completing real-time segmentation tasks.

This research underscores the significance of timely camera operator detection in live football broadcasts to be able to remove them from video frames. The camera operator inpainting technique is capable of leading the revolution not only in football broadcasting but also in a broader domain of live sports streaming. By effectively cutting out unwanted objects from videos and hallucinating the background behind them, this technology can stimulate innovation in the realm of sports broadcasts for the development of new approaches and methodologies for shooting gameplay scenes. Further research in this area can introduce a new era in sports broadcasting where the emphasis is shifted from capturing the ongoing action to creating an immersive and uninterrupted viewing experience. The implementation of camera operator inpainting can lift the restrictions on broadcast companies so they can experiment with innovative angles, perspectives, and other effects to bring the game to life in previously unexplored ways.

Thus, these potential innovations can produce a more dynamic and thrilling playback experience for the end users. Camera operators would be able to capture more moments in the game in the smallest detail without being concerned about camera angles cluttered with unwanted objects, which would otherwise result in the loss of important shots and context. In addition, the application of camera operator inpainting can be further extended to processing not only sports gameplay but also pre- and postmatch action, offline processing of highlight reels, and postmatch analysis. Also, one important use of inpainting is restoring archival footage, which can revive outdated recordings of classic matches.

The YOLOv8 method demonstrates exceptional performance in camera operator segmentation compared with other methods. Through experimental studies, the optimal hyperparameters for this task were identified: a learning rate of 0.001, the Adam optimizer, a momentum of 0.9, and a batch size of 16, as delineated in Table 8. In a comparative analysis, YOLOv8 outperformed several SOTA segmentation algorithms, as mentioned in Table 9. In particular, the presented YOLOv8 model is characterized by high precision (95.5%), recall (92.7%), and mAP50-95 (79.6) values. These outcomes highlight the advantages of the YOLOv8 model in this domain and its potential to improve the quality and efficiency of camera operator segmentation in a variety of applications.

The YOLOv8 system was employed to ensure optimal data flow, proficiently capturing the relevant subtleties for segmenting camera operators and reducing redundant computational workload. Such a meticulously developed architecture enables smooth video frame processing without affecting accuracy. Furthermore, the use of the "Cameramen Instances" dataset in this study not only increased the validity of the obtained results but also highlighted the generalization capabilities of YOLOv8 when dealing with a wide range of datasets. By using cutting-edge training approaches and data augmentation techniques, YOLOv8 was designed to learn from various datasets, which is essential for segmenting camera operator instances, in which whose shape, size, appearance, pose, and illumination of the operator are highly variable. The adaptability of the model markedly improves its generalization ability and robustness, resulting in improved performance on previously unseen data. The YOLOv8 model represents a cutting-edge solution that was designed upon the previous YOLO iterations and improved by introducing structural

updates and modifications that eliminate old, resource-heavy modules of the predecessors and, as a result, refine its accuracy and performance, thus demonstrating its superiority in the segmentation domain.

5.2. Future Research

The presented approach highlights the significant potential of applying video inpainting to enhance broadcast QoS. This study sets the foundation for several promising avenues for further investigation. One such avenue of research involves increasing the number of samples in the created "Cameramen Instances" dataset, so the YOLOv8 system can capture and inpaint camera operators across a broader range of scenarios, for instance, across various lighting conditions or crowd densities.

Furthermore, the proposed system should not be considered limited to football" it could also be applied for other dynamic sports, where delivering immersive broadcasts is a key factor in retaining users on the stream. These sports are the ones that require complex media broadcasts, such as futsal, basketball, American football, rugby, and others. These games conventionally require multiple camera operators strategically placed all over the stadium to capture every scene. Thus, the use cases outlined in this research are similar, which indicates the designed system may be employed in related domains as well. Additionally, future advancements may include not only the inpainting of camera operators but also of other objects that create visual distractions during the stream.

These enhancements would involve hyperparameter tuning for YOLOv8, so, in our forthcoming research, there is a plan to design an evaluation framework to facilitate more comprehensive statistical analyses. The implementation of this framework will include the implementation of complex statistical tests to delve into the relationships between model characteristics and performance metrics, with an emphasis on comparing models such as YOLOv8 using progressive methodologies. This approach is intended to provide a more in-depth assessment of their effectiveness in diverse scenarios and domains.

6. Conclusions

In this study, YOLOv8 was employed for camera operator segmentation as a first step in their further inpainting from video frames during live football broadcasts. This research focused on camera operator instance segmentation. The resulting approach involves several key phases: Firstly, during the live stream, overlapping shooting angles may occur, creating the need to remove visual distractions, such as the camera operator instances. Secondly, the target camera operator is segmented from the video frames using YOLOv8, providing the foundation for its further inpainting. Finally, the camera operator is cut out from the broadcast based on masked frames with the captured target object. This research also highlights the motivation and significance of timely camera operator segmentation and removal from video streams, which may open prospects in the sports media broadcasting realm, as denoted in Section 5. In addition, during this study, a diverse dataset of "Cameramen Instances" was created, which includes more than 7500 samples and more than 11,000 annotations, which can serve as a foundation for further research.

The YOLOv8 model was chosen as the deep learning architecture due to its exceptional accuracy, performance, and low computational resource requirements. Also, various data augmentation techniques were employed that increased the diversity and robustness of the training dataset, which further improved the model's performance. During the process of optimizing the proposed approach, the optimal hyperparameter set for YOLOv8 camera operator segmentation was experimentally defined (Table 8). Metrics including precision, recall, mAP50-95, FLOPS, FPS, and others were employed for the evaluation of the compared models. The evaluation of the results shows that among the considered algorithms such as YOLOv5, YOLOv6, YOLOv7, YOLACT, and Mask R-CNN, YOLOv8 stands out with a precision of 95.5%, recall of 92.7%, mAP50-95 of 79.6, FLOPS of 12, SPF of 0.011, and FPS of 87 (Table 9), making it a promising choice in the task of real-time camera operator segmentation.

This study has the potential to serve as a groundwork for future research in unwanted object segmentation and inpainting during live sports broadcasts, not only in the football realm but also in related fields. The dataset constructed in this research may substantially accelerate the development of media streaming technologies.

Author Contributions: Conceptualization, R.M.; methodology, R.M.; software, S.P.; validation, S.P., R.D. and R.M.; formal analysis, S.P., R.D. and R.M.; investigation, S.P. and R.M.; resources, R.M.; data curation, S.P.; writing—original draft preparation, S.P. and R.M.; writing—review and editing, R.D.; visualization, S.P.; supervision, R.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: For the full implementation details and code, refer to the public GitHub repository at https://github.com/SerhiiPostupaiev/yolov8-research (accessed on 24 March 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Moeslund, T.B.; Thomas, G.; Hilton, A. (Eds.) *Computer Vision in Sports*; Springer International Publishing: Cham, Switzerland, 2014. [CrossRef]
2. Naik, B.T.; Hashmi, M.F.; Bokde, N.D. A Comprehensive Review of Computer Vision in Sports: Open Issues, Future Trends and Research Directions. *Appl. Sci.* **2022**, *12*, 4429. [CrossRef]
3. Manafifard, M.; Ebadi, H.; Abrishami Moghaddam, H. A Survey on Player Tracking in Soccer Videos. *Comput. Vis. Image Underst.* **2017**, *159*, 19–46. [CrossRef]
4. Kamble, P.R.; Keskar, A.G.; Bhurchandi, K.M. Ball Tracking in Sports: A Survey. *Artif. Intell. Rev.* **2017**, *52*, 1655–1705. [CrossRef]
5. Hilton, A.; Guillemaut, J.-Y.; Kilner, J.; Grau, O.; Thomas, G. 3D-TV Production from Conventional Cameras for Sports Broadcast. *IEEE Trans. Broadcast.* **2011**, *57*, 462–476. [CrossRef]
6. Yichen, W.; Yamashita, H. Lineup Optimization Model of Basketball Players Based on the Prediction of Recursive Neural Networks. *Int. J. Econ. Manag. Eng.* **2021**, *15*, 287–293.
7. Liu, W.; Yan, C.C.; Liu, J.; Ma, H. Deep Learning Based Basketball Video Analysis for Intelligent Arena Application. *Multimed. Tools Appl.* **2017**, *76*, 24983–25001. [CrossRef]
8. Wu, F.; Wang, Q.; Bian, J.; Ding, N.; Lu, F.; Cheng, J.; Dou, D.; Xiong, H. A Survey on Video Action Recognition in Sports: Datasets, Methods and Applications. *IEEE Trans. Multimed.* **2022**, *25*, 7943–7966. [CrossRef]
9. Thomas, G. Real-Time Camera Tracking Using Sports Pitch Markings. *J. Real-Time Image Process.* **2007**, *2*, 117–132. [CrossRef]
10. Baker, J.; Wilson, S.; Johnston, K.; Dehghansai, N.; Koenigsberg, A.; de Vegt, S.; Wattie, N. Talent Research in Sport 1990–2018: A Scoping Review. *Front. Psychol.* **2020**, *11*, 607710. [CrossRef]
11. CBC. Camouflaged Cameraman Blends in While Being Front and Centre at World Juniors. Available online: https://www.cbc.ca/news/canada/manitoba/camouflaged-cameraman-world-juniors-hockey-1.6701468 (accessed on 24 March 2024).
12. FIFA. Media and Broadcast. Available online: https://publications.fifa.com/en/football-stadiums-guidelines/technical-guideline/main-user-groups/media-and-broadcast (accessed on 24 March 2024).
13. Yu, Y.; Fan, H.; Zhang, L. Deficiency-Aware Masked Transformer for Video Inpainting. *arXiv* **2023**, arXiv:2307.08629.
14. Li, Z.; Lu, C.-Z.; Qin, J.; Guo, C.-L.; Cheng, M.-M. Towards an End-to-End Framework for Flow-Guided Video Inpainting. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; IEEE: New York, NY, USA, 2022. [CrossRef]
15. Liu, R.; Deng, H.; Huang, Y.; Shi, X.; Lu, L.; Sun, W.; Wang, X.; Dai, J.; Li, H. FuseFormer: Fusing Fine-Grained Information in Transformers for Video Inpainting. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021; IEEE: New York, NY, USA, 2021. [CrossRef]
16. Zeng, Y.; Fu, J.; Chao, H. Learning Joint Spatial-Temporal Transformations for Video Inpainting. In *Computer Vision—ECCV 2020*; Springer International Publishing: Cham, Switzerland, 2020; pp. 528–543. [CrossRef]
17. Gao, C.; Saraf, A.; Huang, J.-B.; Kopf, J. Flow-Edge Guided Video Completion. In *Computer Vision—ECCV 2020*; Springer International Publishing: Cham, Switzerland, 2020; pp. 713–729. [CrossRef]
18. Shelhamer, E.; Long, J.; Darrell, T. Fully Convolutional Networks for Semantic Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 640–651. [CrossRef] [PubMed]
19. He, K.; Gkioxari, G.; Dollar, P.; Girshick, R. Mask R-CNN. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 386–397. [CrossRef] [PubMed]

20. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495. [CrossRef] [PubMed]
21. Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 834–848. [CrossRef] [PubMed]
22. Wang, K.; Liew, J.H.; Zou, Y.; Zhou, D.; Feng, J. PANet: Few-Shot Image Semantic Segmentation with Prototype Alignment. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; IEEE: New York, NY, USA, 2019. [CrossRef]
23. Kirillov, A.; Wu, Y.; He, K.; Girshick, R. PointRend: Image Segmentation as Rendering. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; IEEE: New York, NY, USA, 2020. [CrossRef]
24. Zhao, H.; Shi, J.; Qi, X.; Wang, X.; Jia, J. Pyramid Scene Parsing Network. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; IEEE: New York, NY, USA, 2017. [CrossRef]
25. Wang, X.; Kong, T.; Shen, C.; Jiang, Y.; Li, L. SOLO: Segmenting Objects by Locations. In *Computer Vision—ECCV 2020*; Springer International Publishing: Cham, Switzerland, 2020; pp. 649–665. [CrossRef]
26. Bolya, D.; Zhou, C.; Xiao, F.; Lee, Y.J. YOLACT: Real-Time Instance Segmentation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; IEEE: New York, NY, USA, 2019. [CrossRef]
27. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
28. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; IEEE: New York, NY, USA, 2015. [CrossRef]
29. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-End Object Detection with Transformers. In *Computer Vision—ECCV 2020*; Springer International Publishing: Cham, Switzerland, 2020; pp. 213–229. [CrossRef]
30. Cheng, B.; Schwing, G.S.; Kirillow, A. Per-Pixel Classification is not All You Need for Semantic Segmentation. *arXiv* **2021**, arXiv:2107.06278.
31. Cheng, B.; Misra, I.; Schwing, A.G.; Kirillov, A.; Girdhar, R. Masked-Attention Mask Transformer for Universal Image Segmentation. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; IEEE: New York, NY, USA, 2022. [CrossRef]
32. Xie, E.; Wang, W.; Yu, Z.; Anandkumar, A.; Alvarez, J.M.; Luo, P. SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers. *arXiv* **2021**, arXiv:2105.15203.
33. Jain, J.; Li, J.; Chiu, M.; Hassani, A.; Orlov, N.; Shi, H. OneFormer: One Transformer to Rule Universal Image Segmentation. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023; IEEE: New York, NY, USA, 2023. [CrossRef]
34. Yang, L.; Fan, Y.; Xu, N. Video Instance Segmentation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; IEEE: New York, NY, USA, 2019. [CrossRef]
35. Zhang, T.; Tian, X.; Wu, Y.; Ji, S.; Wang, X.; Zhang, Y.; Wan, P. DVIS: Decoupled Video Instance Segmentation Framework. In Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 1–6 October 2023; IEEE: New York, NY, USA, 2023. [CrossRef]
36. Meinhardt, T.; Feiszli, T.; Fan, Y.; Leal-Taixe, L.; Ranjan, R. NOVIS: A Case for End-to-End Near-Online Video Instance Segmentation. *arXiv* **2023**, arXiv:2308.15266.
37. Wu, J.; Jiang, Y.; Lio, Q.; Yuan, Z.; Bai, X.; Bai, S. General Object Foundation Model for Images and Videos at Scale. *arXiv* **2023**, arXiv:2312.09158.
38. Wu, J.; Liu, Q.; Jiang, Y.; Bai, S.; Yuille, A.; Bai, X. In Defense of Online Models for Video Instance Segmentation. In *Lecture Notes in Computer Science*; Springer Nature Switzerland: Cham, Switzerland, 2022; pp. 588–605. [CrossRef]
39. Li, X.; Yuan, H.; Zhang, W.; Cheng, G.; Pang, J.; Loy, C.C. Tube-Link: A Flexible Cross Tube Framework for Universal Video Segmentation. In Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 1–6 October 2023; IEEE: New York, NY, USA, 2023. [CrossRef]
40. Ultralytics. Comprehensive Guide to Ultralytics YOLOv5. Available online: https://docs.ultralytics.com/yolov5 (accessed on 29 March 2024).
41. Li, C.; Li, L.; Jiang, H.; Weng, K.; Geng, Y.; Li, L.; Ke, Z.; Li, Q.; Cheng, M.; Nie, W.; et al. YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. *arXiv* **2022**, arXiv:2209.02976.
42. Wang, C.-Y.; Bochkovskiy, A.; Liao, H.-Y.M. YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023; IEEE: New York, NY, USA, 2023. [CrossRef]
43. Yang, T.; Zhou, S.; Xu, A.; Ye, J.; Yin, J. An Approach for Plant Leaf Image Segmentation Based on YOLOV8 and the Improved DEEPLABV3+. *Plants* **2023**, *12*, 3438. [CrossRef] [PubMed]
44. Yue, X.; Qi, K.; Na, X.; Zhang, Y.; Liu, Y.; Liu, C. Improved YOLOv8-Seg Network for Instance Segmentation of Healthy and Diseased Tomato Plants in the Growth Stage. *Agriculture* **2023**, *13*, 1643. [CrossRef]

45. Abdusalomov, A.B.; Mukhiddinov, M.; Whangbo, T.K. Brain Tumor Detection Based on Deep Learning Approaches and Magnetic Resonance Imaging. *Cancers* **2023**, *15*, 4172. [CrossRef] [PubMed]
46. Sahafi, A.; Koulaouzidis, A.; Lalinia, M. Polypoid Lesion Segmentation Using YOLO-V8 Network in Wireless Video Capsule Endoscopy Images. *Diagnostics* **2024**, *14*, 474. [CrossRef] [PubMed]
47. Li, X.; Yin, J.; Shi, B.; Li, Y.; Yang, R.; Shen, J. LWSIS: LiDAR-Guided Weakly Supervised Instance Segmentation for Autonomous Driving. In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 1433–1441. [CrossRef]
48. De Brabandere, B.; Neven, D.; Van Gool, L. Semantic Instance Segmentation for Autonomous Driving. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition: Workshops (CVPRW), Honolulu, HI, USA, 21–26 July 2017; IEEE: New York, NY, USA, 2017. [CrossRef]
49. Budiharto, W.; Gunawan, A.A.S.; Suroso, J.S.; Chowanda, A.; Patrik, A.; Utama, G. Fast Object Detection for Quadcopter Drone Using Deep Learning. In Proceedings of the 2018 3rd International Conference on Computer and Communication Systems (ICCCS), Nagoya, Japan, 27–30 April 2018; IEEE: New York, NY, USA, 2018. [CrossRef]
50. Lee, D.-H. CNN-based Single Object Detection and Tracking in Videos and Its Application to Drone Detection. *Multimed. Tools Appl.* **2020**, *80*, 34237–34248. [CrossRef]
51. Guo, J.; Liu, X.; Bi, L.; Liu, H.; Lou, H. UN-YOLOv5s: A UAV-Based Aerial Photography Detection Algorithm. *Sensors* **2023**, *23*, 5907. [CrossRef] [PubMed]
52. Valappil, N.K.; Memon, Q.A. Vehicle Detection in UAV Videos Using CNN-SVM. In *Advances in Intelligent Systems and Computing*; Springer International Publishing: Cham, Switzerland, 2021; pp. 221–232. [CrossRef]
53. Gupta, A.K.; Seal, A.; Prasad, M.; Khanna, P. Salient Object Detection Techniques in Computer Vision—A Survey. *Entropy* **2020**, *22*, 1174. [CrossRef] [PubMed]
54. Zhao, R.; Ouyang, W.; Li, H.; Wang, X. Saliency Detection by Multi-Context Deep Learning. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; IEEE: New York, NY, USA, 2015. [CrossRef]
55. Kalboussi, R.; Azaza, A.; van de Weijer, J.; Abdellaoui, M.; Douik, A. Object Proposals for Salient Object Segmentation in Videos. *Multimed. Tools Appl.* **2019**, *79*, 8677–8693. [CrossRef]
56. Gao, Z.; Xu, C.; Zhang, H.; Li, S.; de Albuquerque, V.H.C. Trustful Internet of Surveillance Things Based on Deeply Represented Visual Co-Saliency Detection. *IEEE Internet Things J.* **2020**, *7*, 4092–4100. [CrossRef]
57. Babahenini, S.; Charif, F.; Cherif, F.; Ahmed, A.T.; Ruichek, Y. Using Saliency Detection to Improve Multi-Focus Image Fusion. *Int. J. Signal Imaging Syst. Eng.* **2021**, *12*, 81. [CrossRef]
58. Li, G.; Yu, Y. Visual Saliency Detection Based on Multiscale Deep CNN Features. *IEEE Trans. Image Process.* **2016**, *25*, 5012–5024. [CrossRef] [PubMed]
59. Rifai, I.; Al Maki, W.F. Adapting Faster R-CNN and Video Inpainting for Badminton Player Detection. In Proceedings of the 2023 11th International Conference on Information and Communication Technology (ICoICT), Melaka, Malaysia, 23–24 August 2023; IEEE: New York, NY, USA, 2023. [CrossRef]
60. Cioppa, A.; Deliege, A.; Istasse, M.; De Vleeschouwer, C.; Van Droogenbroeck, M. ARTHuS: Adaptive Real-Time Human Segmentation in Sports Through Online Distillation. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Long Beach, CA, USA, 16–17 June 2019; IEEE: New York, NY, USA, 2019. [CrossRef]
61. Gao, F.; Wu, W.; Jin, Y.; Shi, L.; Ma, S. A Sparse Attention Pipeline for DeepSportRadar Basketball Player Instance Segmentation Challenge. In Proceedings of the MM '23: The 31st ACM International Conference on Multimedia, Ottawa, ON, Canada, 29 October–3 November 2023; ACM: New York, NY, USA, 2023. [CrossRef]
62. Hsu, C.-C.; Lee, C.-M. MISS: Memory-efficient Instance Segmentation Framework by Visual Inductive Priors Flow Propagation. *arXiv* **2024**, arXiv:2403.11576.
63. Ghasemzadeh, S.A.; Van Zandycke, G.; Istasse, M.; Sayez, N.; Moshtaghpour, A.; De Vleeschouwer, C. DeepSportLab: A Unified Framework for Ball Detection, Player Instance Segmentation and Pose Estimation in Team Sports Scenes. *arXiv* **2021**, arXiv:2112.00627.
64. Cheng, B.; Collins, M.D.; Zhu, Y.; Liu, T.; Huang, T.S.; Adam, H.; Chen, L.-C. Panoptic-DeepLab: A Simple, Strong, and Fast Baseline for Bottom-Up Panoptic Segmentation. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; IEEE: New York, NY, USA, 2020. [CrossRef]
65. Watanabe, R.; Chen, J.; Konno, T.; Naito, S. Accurate Background Subtraction Using Dynamic Object Presence Probability in Sports Scenes. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; IEEE: New York, NY, USA, 2021. [CrossRef]
66. Husein, A.M.; Chalvin; Ciptady, K.C.; Suryadi, R.; Harahap, M. Detecting and Tracking Player in Football Videos Using Two-Stage Mask R-CNN Approach. *IAIC Int. Conf. Ser.* **2023**, *4*, 132–138. [CrossRef]
67. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
68. Bewley, A.; Ge, Z.; Ott, L.; Ramos, F.; Upcroft, B. Simple Online and Realtime Tracking. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; IEEE: New York, NY, USA, 2016. [CrossRef]

69. Naik, B.T.; Hashmi, M.F. YOLOv3-SORT: Detection and Tracking Player/Ball in Soccer Sport. *J. Electron. Imaging* **2022**, *32*, 011003. [CrossRef]
70. Biliskov, I.; Saric, M.; Russo, M.; Stella, M. Players Detection Using U-Net Based Fully Convolutional Network. In Proceedings of the 2021 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), Split, Hvar, Croatia, 23–25 September 2021; IEEE: New York, NY, USA, 2021. [CrossRef]
71. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Lecture Notes in Computer Science*; Springer International Publishing: Cham, Switzerland, 2015; pp. 234–241. [CrossRef]
72. Ultralytics. Ultralytics YOLOv8 Docs. Available online: https://docs.ultralytics.com/ (accessed on 29 March 2024).
73. Wang, C.-Y.; Yeh, I.H.; Liao, H.-Y.M. YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. *arXiv* **2024**, arXiv:2402.13616.
74. Zhai, X.; Huang, Z.; Li, T.; Liu, H.; Wang, S. YOLO-Drone: An Optimized YOLOv8 Network for Tiny UAV Object Detection. *Electronics* **2023**, *12*, 3664. [CrossRef]
75. Safaldin, M.; Zaghden, N.; Mejdoub, M. An Improved YOLOv8 to Detect Moving Objects. *IEEE Access* **2024**, *12*, 59782–59806. [CrossRef]
76. Vats, A.; Anastasiu, D.C. Enhancing Retail Checkout Through Video Inpainting, YOLOv8 Detection, and DeepSort Tracking. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Vancouver, BC, Canada, 17–24 June 2023; IEEE: New York, NY, USA, 2023. [CrossRef]
77. Liu, Y.; Zhou, N. Jumping Action Recognition for Figure Skating Video in IoT Using Improved Deep Reinforcement Learning. *Inf. Technol. Control* **2023**, *52*, 309–321. [CrossRef]
78. Žemgulys, J.; Raudonis, V.; Maskeliūnas, R.; Damaševičius, R. Recognition of basketball referee signals from real-time videos. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *11*, 979–991. [CrossRef]
79. Žemgulys, J.; Raudonis, V.; Maskeliunas, R.; Damaševičius, R. Recognition of basketball referee signals from videos using Histogram of Oriented Gradients (HOG) and Support Vector Machine (SVM). *Procedia Comput. Sci.* **2018**, *130*, 953–960. [CrossRef]
80. Roboflow. 2020. Available online: https://roboflow.com (accessed on 11 March 2024).
81. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; IEEE: New York, NY, USA, 2016. [CrossRef]
82. Ultralytics. 2013. Available online: https://github.com/ultralytics/ultralytics (accessed on 11 March 2024).
83. Terven, J.; Córdova-Esparza, D.-M.; Romero-González, J.-A. A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 1680–1716. [CrossRef]
84. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; IEEE: New York, NY, USA, 2018. [CrossRef]
85. Lin, T.-Y.; Dollar, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; IEEE: New York, NY, USA, 2017. [CrossRef]
86. Feng, C.; Zhong, Y.; Gao, Y.; Scott, M.R.; Huang, W. TOOD: Task-Aligned One-Stage Object Detection. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021; IEEE: New York, NY, USA, 2021. [CrossRef]
87. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12993–13000. [CrossRef]
88. Li, X.; Lv, C.; Wang, W.; Li, G.; Yang, L.; Yang, J. Generalized Focal Loss: Towards Efficient Representation Learning for Dense Object Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 3139–3153. [CrossRef]
89. Isa, I.S.; Rosli, M.S.A.; Yusof, U.K.; Maruzuki, M.I.F.; Sulaiman, S.N. Optimizing the Hyperparameter Tuning of YOLOv5 for Underwater Detection. *IEEE Access* **2022**, *10*, 52818–52831. [CrossRef]
90. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In *Computer Vision—ECCV 2014*; Springer International Publishing: Cham, Switzerland, 2014; pp. 740–755. [CrossRef]