Player tracking and automatic scored baskets detection

Roberto Mazzaro *DISI, University of Trento* roberto.mazzaro@studenti.unitn.it 229301
Sebastiano Dissegna *DISI, University of Trento* sebastiano.dissegna@studenti.unitn.it 232090
Mattia Carolo *DISI, University of Trento* mattia.carolo@studenti.unitn.it 232126

***Abstract*—**

In recent years AI technologies have been applied with success to different tasks related to the Computer Vision. One field in particular has seen in the last two decades a rise in popularity which is sport related tasks. Just by looking at recent applications it can be seen how those implementations have reached high enough performance allowing to be used even in official competitions like the football World Cup of 2022 where they used an algorithm for the automatic recognition of a player offside.

Speaking about basketball, different approaches have already been tried for different tasks from creating the highlights of a match to the real time tracking of the players of the two different teams. In this work we focused on further improving the framework delivered by two existing projects by implementing different architectures and strategies.

I. Introduction

Our work started from an already existing project [1] [2] that had as a focus the generation of some statistics of a basketball game. We were given two repositories that already tried to tackle the problem. As a basis, both projects use Mask R-CNN to detect the main actors in the scene by applying a mask in order to evaluate only the part of the video relative to the playing field. After the detection by using the K-Means algorithm the players and referees are grouped by comparing the color list found on the detection box. No tracking of the player bounding boxes and no team assignment check are done, so only the detection produced by the Mask R-CNN is used to know where players of each team are placed. The real difference between the two implementations consists in the detection of the playing ball. One work exploits Mask R-CNN [3] also for the tracking of the ball, while the other one leverages on YOLOv3 [4], an improved version of the more famous YOLO [5], which is developed with the main focus of doing real-time object detection. After the detection both works apply CSRT tracker and one of them also applies an interpolation based on the euclidean distance from the position of the detection and the tracker. The results given by the two precedent works shows that without a tracker the pure Mask RCNN implementations outperforms the YOLO one while, introducing a tracker, the overall performance of the second one becomes on par with the first with a much lower detection time. Moreover adding the interpolation slightly improves the performances. Given these information prior works are able to compute some statistics about the game, like in which part of the field the ball is, the direction of the attack, the ball possession and the pressing of a team over the other. Starting from these projects we implemented a tracking also for the players, we tried three different trackers, a custom one based on the CSRT tracker and two state of the art trackers, SORT and BYTE, where we introduced some custom parts. We try to apply those new trackers also to the ball detections but we obtained worse performances, so we just kept the already existing tracking based on the CSRT tracker with the interpolation procedure. Furthermore we also implemented the detection of the scored baskets and their values, to do that we tried two different methodologies, one based on the background subtraction and the other based on the ball tracking.

II. Players Tracking

Previous works rely only on the player detection performed with Mask R-CNN, without a tracker. Adding it is beneficial to our implementation because it stabilizes the players bounding boxes, avoiding continuous flickering and team changes due to error in the detection or team assignment. However player tracking is not so simple because we need a multi-object tracker able to deal with many occlusions. To solve this challenge we tried three different strategies, a customized solution based on the CSRT tracker and two solutions based on state of the art trackers, SORT and BYTE tracker, with some slight modifications.

A. Customized tracker

The idea here is to define a CSRT tracker, which is a single object tracker, for every player. To achieve that what we have done is:

1) For each player bounding box found in the detection phase, we search for the nearest already tracked bounding box. After we assign the new detection to the tracker that tracks this bounding box. In the case a new detection cannot be found a new tracker will be defined by initializing it with the unmatched detection.
2) The tracker is updated with the novel detection.
3) If the assigned detection has an euclidean distance greater than a predefined threshold or we don't have a detection for more than two consecutive frames the tracker will be reinitialized.
4) Previous steps may create duplicated trackers for the same player, so we look at each couple of tracked bounding boxes and if they are overlapping for more than the 80% we destroy one of the two tracker.

B. *SORT and BYTE trackers*

For both trackers we started from an official python implementation available online where later we applied a couple of changes. The most important are:

- We leverage only on the tracker part giving to it as input the bounding boxes founded by the Mask R-CNN. We avoided using the detection made available by the authors because together with the detection we also need the team assignment function.
- We added a parameter to the class where the trackers are defined that will be used to store the team number. Is important to store this information in the tracker, because as already explained, we compute it after the detection exploiting the found bounding box and the computed mask. This information moreover is then used to compute many statistics.

C. *Results*

With the customized solution we obtained satisfying results but it is very heavy from a computational point of view, moreover there are some problems to deal with, like heavy occlusions. This creates a problem for the system that deletes duplicated trackers since is not able to distinguish between occluded objects and other trackers initialized for the same object. Sometimes, as we can see in Fig. 1 arrow number 2, we still have problems of duplicated trackers for the same object when bounding boxes have very different size, while the problem indicated by arrow 1 with high probability is caused by a bad detection, however other tracker can deal better with this kind of problem. SORT and BYTE tracker are born as multi-object trackers, so the internal procedure is more efficient and the results are more coherent. In fact as expected both trackers require less time to compute the final result with respect to our customized solution. The quality of the tracked bounding boxes in the SORT tracker usually is slightly better with respect to our custom solution, in particular usually the bounding boxes are more stable and are followed better, on the other side with the current hyper-parameters it is possible that some player are not followed at all as shown in Fig. 2. BYTE tracker outperforms both solutions, we obtained far better results, less bounding boxes are lost and it is also able to better deal with occlusions even if we still have some imperfection as can be seen in Fig. 3. It is important to notice that problem in the team assignment are caused by errors in the already existing team number computation function.

III. Tracking Algorithms

A. *CSRT*

This algorithm was the first one used since it was already implemented in the existing framework for the tracking of the ball. The CSRT implementation offered by OpenCV is based on the work made by Lukezic at al. [6] which further improves Discriminative Correlation Filters introducing channels and spatial reliability. The algorithm uses a filter to learn the spatial and color (channel) characteristics of the object to be tracked, and then uses this learned filter to search for

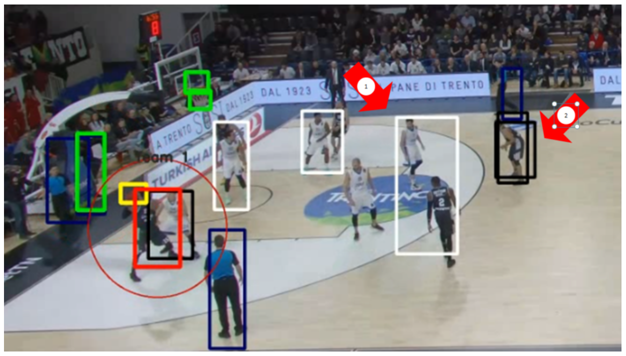
Fig. 1: Custom tracker example

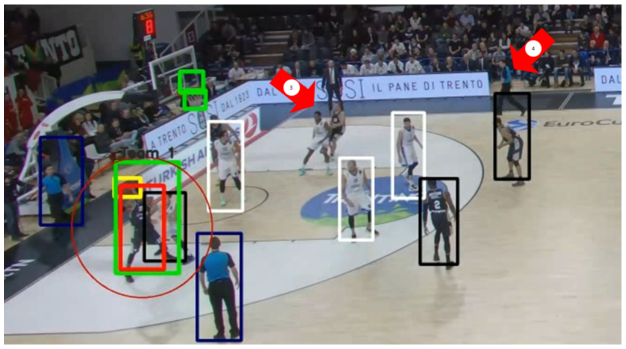
Fig. 2: SORT tracker example

the object in subsequent frames of the image stream. The algorithm begins by learning a correlation filter for the object in the first frame of the video stream, using the Discriminative Correlation Filters (DCF) method. Thanks to this learned filter the algorithm is capable of searching for the learned object in subsequent frames. The motives for which we maintained this tracker from the previous work is for its use of channel and spatial reliability measures so that we could have us a solid baseline. Channel reliability measures are used to weight the importance of different color channels in the tracking process, while spatial reliability measures are used to weight the importance of both different spatial locations within the frame and possible deformations of the interested object and both are critical for tracking an object like our ball .

B. *SORT*

The sort algorithm relies heavily on the detection quality for its performance since it simply uses a combination of familiar techniques such as the Kalman Filter and Hungarian algorithm for the tracking components. The version of the algorithm used is the one illustrated in the original paper [1]. The design philosophy is to keep the overall complexity low, so appearance features beyond the detection component are ignored in tracking and only the bounding box position and size are used for both motion estimation and data association. It's important to acknowledge that issues regarding short-term and long-term occlusion are also ignored since the main focus of this algorithm is an efficient and reliable handling of common frame-to-frame associations. Four are the main component of this proposed method: detection, propagating

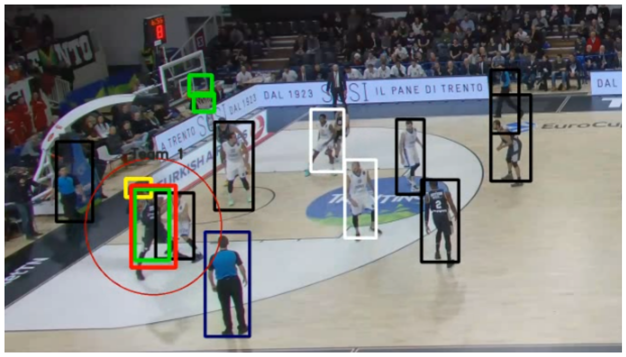

Fig. 3: BYTE tracker example

object states into future states, associating current detections with existing objects, and managing the lifespan of tracked objects. For the detection part different architectures can be used, in the original paper a Faster Region CNN is implemented, while in our case YOLO and a MaskRCNN are used. Now for the estimation model, the inter-frame displacements of each object is approximated with a linear constant velocity model, independent of other objects and camera motion, which in our case is not a problem since we have a fixed camera for all our videos. When a detection is associated with a target, the detected bounding box is used to update the target state where the velocity components are solved optimally via a Kalman filter framework. When assigning detections to existing targets, each target's bounding box geometry is estimated by predicting its new location in the current frame. The assignment cost matrix is then computed as the intersection-over-union distance between each detection and all predicted bounding boxes from the existing targets. The assignment is solved optimally using the Hungarian algorithm.

C. BYTE

The Bytetrack algorithm [7] was chosen for its quite impressive results in the MOT challenge. This method compared to all the others has one peculiarity, the fact that when associating, almost every detection box is used instead of only the high score ones. This is because low confidence detection boxes sometimes indicate the existence of objects, for example the occluded ones. Filtering out these objects causes irreversible errors for MOT and brings non-negligible missing detection and fragmented trajectories. The innovation of BYTE lies in this junction area of detection and association, where low score detection boxes are bridges to boost both of them. Now for the method itself, initially almost every detection box is kept, then they separated into high score ones and low score ones where high scores are associated with the tracklets. Some tracklets get unmatched because they do not match to an appropriate high score detection box, which usually happens when occlusion, motion blur or size changing occurs. After that the low score detection boxes and these unmatched tracklets are associated to recover the objects in the low score detection boxes and filter out background, simultaneously.

IV. Scored Baskets Detection

Our other contribution to the previous projects, is the function that understand when a basket is scored and its value. In this section we analyze only the first part of this function. To solve the problem of finding the scored basket we tried two different solutions both with pros. and cons.

A. Scored basket detection with ball tracking

First thing we tried was to exploit the ball tracking to reach this goal. The idea was very simple, define a bounding box over the basket net, and when this bounding box was overlapped for more than a certain threshold with the bounding box of the ball with high probability we have found a scored basket. To remove the false positives of when the ball passes closely over the rim but doesn't enter in the basket, we added an additional bounding box over it and added a constraint where we consider as a valid scored basket only the ones in which the ball enters first in the bounding box over the rim and then in the lower bounding box. The time that passes from an activation to the other must be smaller than a predefined threshold otherwise it won't be counted. We further improved this technique by adding also a delay after we detected a valid scored basket. In this way we don't search for scored points on the basket after the last point was signed removing the possibility of counting scored hoops multiple times. This problem rises because we can have bounding boxes sufficiently overlapped for more subsequent frames, furthermore is very likely that the detection of the ball gets lost after that the ball enters the basket resulting in a situation where the bbox of the ball remains stuck over the net. This methodology is simple and works well when the detection and the tracking of the ball work correctly, unfortunately since the ball moves fast and can be occluded many times, usually we have really few detections even for long clips. Nevertheless the ball tracker still shines in following the ball but sometimes it gets lost. This is due to the most common trajectory of a throw which unfortunately passes over the crowd. As we can see from Fig. 4 wrt. the normal background in the playing field, the crowd have a pretty heterogeneous set of colors giving a hard time to the detection part in finding the right candidate and locks the tracker in following the wrong objective. Furthermore the bounding box of the ball may change it's dimensions and this is a problem when we compute the interpolation between bounding boxes.

B. Scored basket detection with background subtraction

To solve the main issues with the previous methodology we tried another type of detection based on the background subtraction. The idea is that in order to understand if the ball enters a bounding box we do a background subtraction of the current content of the bounding box with a clean version of it, we count how many pixels have a difference value higher than a certain threshold and in case of an activation of a predetermined amount we consider it as if the ball has entered inside the bounding box. Like before we keep the structure with the delays and the two bounding boxes for each basket to

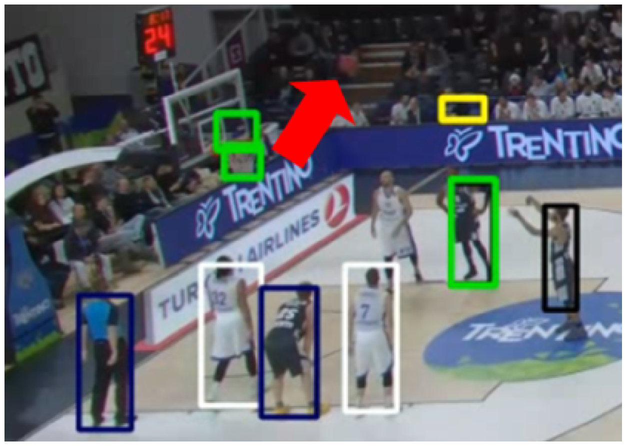

Fig. 4: Ball tracking issues

avoid false positives. In this case, this procedure is useful also even to avoid false positives due to the monitor that display advertisements, especially for the basket on the right where the net is overlapped with a monitor as can be seen in Fig. 5, without the two bounding boxes we count as scored basket the change of the displayed advertisements.

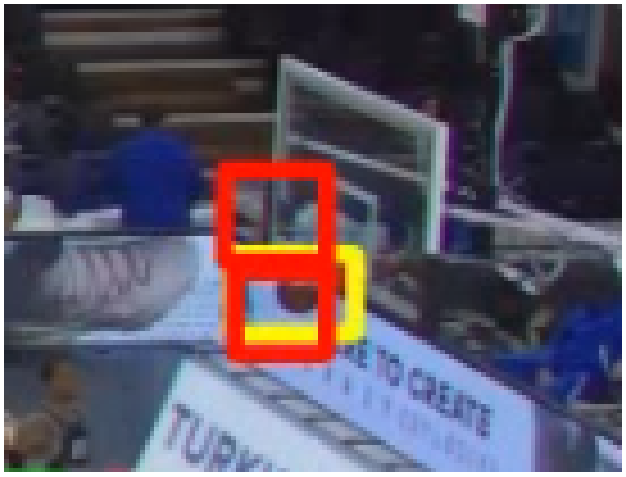

Fig. 5: Basket net overlapped with advertisement monitor

C. Results

Both strategies based on ball tracking and the one based on background subtraction are not able to distinguish cases where the ball enters in the bounding boxes following the correct order but it's not a made hoop, as shown in Fig. 6; these cases are the main reason for false positives detections together with movement in the background of the bounding boxes, that in the method based on background subtraction sometimes are able to confuse the system that thinks to have found a scored basket, however this last case is quite rare and thanks to the system with two bounding boxes we are able to avoid most of the false positive detections.

For the evaluation we focused on the detection of a correctly classified hoop to test our implemented methods. Both methodologies are sensitive to the bounding boxes positions,

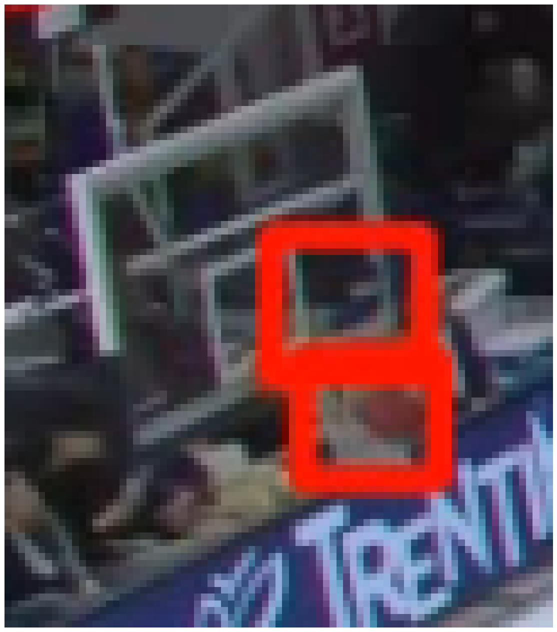

Fig. 6: Ball doesn't enter the basket

delays and thresholds that are selected, we tuned these parameters iteratively searching for the best compromise on average and after making sure we were getting good results on a different set of conditions we benchmarked our architecture against a video of a real match of basketball. The video represents the full first time of the match for a length of almost 25 minutes.

1) Quantitative analyses: For the quantitative part we treated the results of the classification of a hoop as:

- **True positives:** all the correctly classified scored hoops
- **False Positive:** all the not correct positive classifications of a hoop
- **False Negative:** all the times a scored hoop was not counted

In Tab. 1 we reported the results and some statistics for the two different methodologies of scored baskets detection. We did not take into account the true negatives because there are a lot of different situations to take into account so we reported only cases that implied an activation of the detector.

Methodology	TP	FP	FN	Prec.	Recall	F1
Ball Tracking	3	1	21	0.75	0.13	0.21
Background Subtraction	16	14	8	0.53	0.67	0.59

TABLE I: Scored Baskets Detection Results

As it can be seen the overall performance of the second method with background subtraction outperforms the one based only on the tracking of the ball. As already explained the main reason of this is because the second method is much more less dependant from the pure tracking of the ball which can be prone to errors given all the occlusions that happen during the full time of a match.

2) Qualitative analyses: By analyzing the results one of the things that can be seen almost immediately is how the detection is the main problem when applied to a long sequence. Since the ball can be left unseen due to various occlusions and start a series of wrong detections, more often than not just simple turns of the players are enough to sparkle this error. Objects that are usually tracked considering their ground truth that are the head, hands and shoes of the players. This detection problems are particularly tricky for every methodology relying purely on the tracking of the ball for counting the score since we have a prevalence of occlusions on the moment right before the shot making the detector pick a wrong element. Scenarios like these bring the tracker to follow wrong elements and without a detection that resets the tracker the suppose "ball" will be continuously followed ruining the score count.

V. Value of a Scored Basket

Once we have detected a made hoop we also try to assign a value to it. In a basket match we have to distinguish between 3 cases:

- Scored basket with a value of 1 point: this is the most rare case because only the baskets scored from a free throw assigned after a foul have this value. We try to recognize them exploiting the particular configuration of the players in a free throw that is characterized by three players for each team in the area of three meters and one of them is placed inside the bezel posed at two meters in front of the basket.
- Scored basket with a value of 2 points: the player who shoot has to be inside the three meters area near the basket.
- Scored basket with a value of 3 points: the player who shoot has to be outside the three meters area near the basket.

To distinguish between these three cases we follow the subsequent procedure, first we have some preliminary tasks that are executed at each frame:

1) Compute how many players for each team there are inside the three meters area.
2) Check if the player who has the ball is inside the bezel posed at two meters in front of the hoop.
3) Check if the player who have the ball is inside the three meters area.
4) Check if a scored basket has been detected in this frame.

After these steps, if a scored basket has been detected we do the following:

1) Check if in the last 5 frames there are more times where the player with the ball is inside the bezel than times we found it outside. If this condition is satisfied we compute the mean of the player of each team inside the three meters area and if for both teams this is between 2 and 3 we assign to the detected scored basket a value of 1 point, otherwise a value of 2 points.
2) If previous case is not verified, we check if in the last 5 frames we have found more times the throwing player inside the three meters area, than times we found it outside. If this condition is satisfied we assigned to the detected scored basket a value of 2 points.
3) If none of these cases is verified this means that the player who shoot was outside the three meters area so we assigned to the detected scored basket a value of 3 points.

Usually considering the last 5 frames it's more than enough to get clean results. There is just an exception which fortunately fires very rarely when we can consider only the first frame due to be analyzing the beginning of the video or when we have detection problems.

A. Results

This procedure is correct from a conceptual point of view, however due to some problems in the player detection, e.g. some missing player, or errors in the assignment of the team to a player, there are same difficulties, especially in recognizing a free throw from a more common throw inside the three meters area. This implies that most of the time some of the constraints are not satisfied assigning therefore a value of 2 points instead of 1 point. Moreover sometimes there are issues to compute the player who hold the ball. Previous projects address this issues in a simple way, they consider as the player who hold the ball the one that is nearest to the ball and that also has a distance from it lower than a certain threshold. This solution is good to compute statistics were average values are enough. This creates a tricky situation since the position of the bounding box of the shooting player is crucial to assign the points. To solve this issue we tried to implement a tracker of the player who hold the ball using a CSRT tracker initialized with the bounding box coordinates of the player who hold the ball found with the old strategy. This solution slightly improve the results but there are still some problems in the harder cases for example when the ball passes near to another player for some subsequent frames. In Fig. 7 we can see a correct attribution of the player who is shooting (the one highlighted with a green rectangle), after few frames, Fig. 8, the ball passes near another player for more than one frame and we have a wrong assignment.

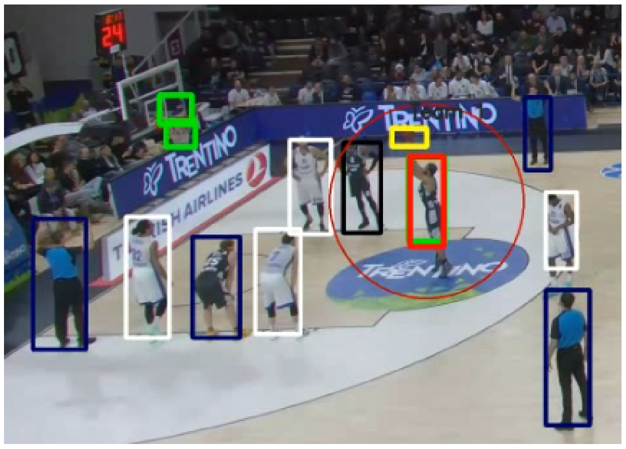

Fig. 7: Correct assignment of the player who shot

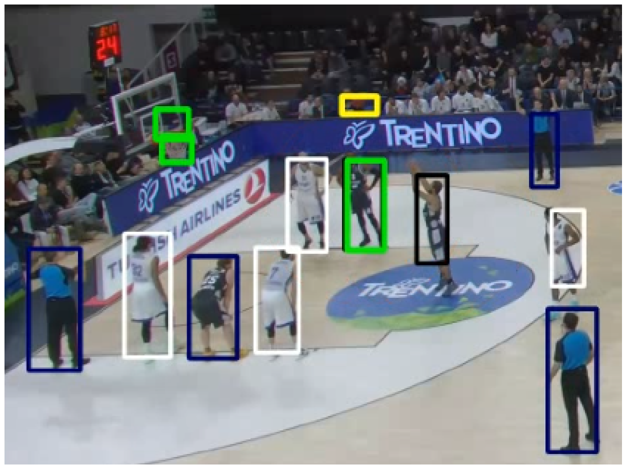

Fig. 8: Wrong assignment of the player who shot

Other times is the ball tracking to be wrong and also this could be a problem for the correct assignment of the points. In the example of Fig. 9 we can see how the tracking of the ball completely missplaced wrt. the ground truth even if in the end the final assigned value is correctly predicted.

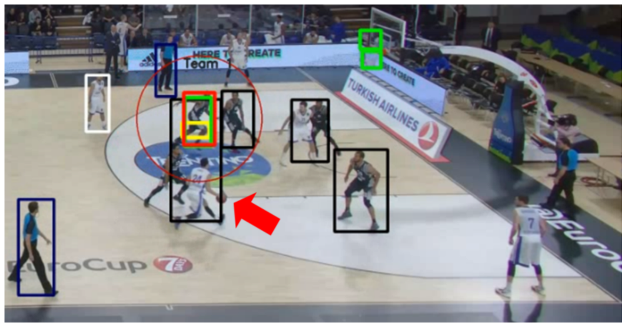

Fig. 9: Issue with ball tracking

VI. Conclusion

Starting from the previous works we improved them adding three main features:

- We tried three different solutions for implementing the players tracking, we obtained the best results with the BYTE tracker that considerably improve the quality of the player bounding boxes with respect to using only the player detection data and the other tested trackers.
- We implemented two different way for the detection of a scored basket, one based on the ball tracking and the other one on the background subtraction. The methodology based on the ball tracking is limited by the quality of the ball detection, while the other methodology based on background subtraction is more robust and obtain better results, although is still vulnerable to some particular cases that can not be solved without adding an extra camera.
- We implemented a function that tries to understand the value of a scored basket. Also here the performances are limited by the ball and players detection / tracking however we are satisfied with the obtained results.

References

[1] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, sep 2016. [Online]. Available: https://doi.org/10.1109%2Ficip.2016.7533003

[2] M. D. Ponte. [Online]. Available: https://github.com/MatteoDalponte/Basketball_statistics

[3] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," 2017. [Online]. Available: https://arxiv.org/abs/1703.06870

[4] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018. [Online]. Available: https://arxiv.org/abs/1804.02767

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2015. [Online]. Available: https://arxiv.org/abs/1506.02640

[6] A. Lukežič, T. Vojíř, L. Č. Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter tracker with channel and spatial reliability," *International Journal of Computer Vision*, vol. 126, no. 7, pp. 671–688, jan 2018. [Online]. Available: https://doi.org/10.1007%2Fs11263-017-1061-3

[7] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "Bytetrack: Multi-object tracking by associating every detection box," 2021. [Online]. Available: https://arxiv.org/abs/2110.06864