Acknowledgements

I would like to thank my supervisor Dr. Aphrodite Galata for her commitment to the project and her generous advice on my progress throughput the year. Also, I would like to thank my parents for backing me up and help me cope with lots of things that I shall spend much time on.

Abstract

In this project I aimed to build a program for tracking players in broadcast football video. Due to the complexity of the topic, the project mainly focused on videos from static camera, but keeps the potential to extend to those from rotating camera. The whole program can be divided into 4 parts: player detection, player identification, tracking and label correction.

The program is based on a set of python scripts. A pretrained model produced by Mask-RCNN is used for player detection, SIFT features from MSERs of each detection are put into BOVW model and KNN is used for classification of BOVW features. Idea of tracking-by -detection is used in tracking part, and a tracklet is maintained for each player. Kalman filter is used here to give more robust tracking and do prediction of player locations where occlusions occur. A simple majority label assignment is done for label correction part.

1. Introduction

Figure tracking has been a widely investigated problem and there are many satisfying results for pedestrian tracking. However, research of tracking for sports players is not that well extended as it is more difficult due to unpredictable trajectories. Low quality of players also poses a challenge, as it limits number of features of each figure and brings difficulty to identification.

This report will describe current milestone and strengths & weaknesses of the system. Relevant work of previous students [1][2] provides some reference on methodology and further discoveries have been made. Evaluation will be done on player detection and identification. In the report, part 2 will describe previous works on the field of tracking sports players as a whole and relevant technologies on each part of system; part 3 states methods I worked on for each of 4 parts of the system, part 4 includes procedure of evaluations I mentioned above and gives their results, finally, strengths and weaknesses of my final work will be discussed in part5.

* 1. Aim

An ideal version of the project is a system that can generate tactics automatically, yet this includes translate players’ identities and trajectories into strategies, thus being way too complex for the given amount of time and will need participation of football professionals. For this project, the system is expected to generate a bounding box for every player in each frame and a correct and consistent identity should be given to each figure across the frames he/she appears in the video.

* 1. Challenges

A list of different challenges needed to be overcome is listed below.

1. Unlike pedestrian tracking, which give close and clear image of people, the camera for a broadcast sports video is usually placed relatively far from players and thus people in result videos usually have smaller size and keep less detail. This presents difficulty for both identification and identification, as limited information can be utilized. Apart from that, players in broadcast usually moves at a variant speed and unpredictable direction, posing challenge to detection and tracking. When players move at high speed, they usually look blurred in the video, making identification harder.
2. All players in a team wear the same kind of jersey, making individual player classification become difficult.
3. Distinguishability of players depends on many factors, like lighting condition of the scene, whether color of players’ jersey and background are similar, etc.
4. Using only one camera will cause problem of occlusion occurs frequently. Information of a player will get lost completely as long as he/she is occluded, causing difficulty to maintain continuity of tracking.
   1. Approaches

Going directly to a broadcast video of a real match is a step way too big, a reasonable baseline will be trying various kinds of methods on a simplified scene, like the dataset I am provided that includes only 4 players instead of 22. Satisfying results on such simple scenes will make it convincible that the same method has potential to work on a more complex video, otherwise it shall be abandoned.

(screenshot: occlusion, moving at high speed, dim background, jersey same color with background)

Python is used for the whole project, mainly for 3 reasons:

1. It provides various kinds of commonly used data structures like list and dictionary;
2. A large number of packages are available for python, getting me avoid of building many things from scratch.
3. Taking into the account of usage of deep learning framework, python is the best choice as almost all frameworks provides the best-written, most informative documentation only for python, python frameworks has been most wide used by other deep learning workers, with more bugs fixed, more problems discussed, being more robust, thus will cause less problem when I integrate a neural network model in the project.
4. Background

The topic of sportsperson tracking is still an ongoing one and a considerable part of work is about investigation into if tracking in low-quality sports video is achievable. In this section, previous works on this topic are introduced first, then for player detection, player identification and tracking, algorithms and ideas I tried on are briefly explained.

2.1 previous work

Work of Lu et al. [3] is where I get a majority of ideas to build my system. They used a Deformable Parts Model to detect players. SIFT features, MSER regions and color histogram are used represent players in classification task. Idea of tracking-by-detection is used in tracking part with Kalman Filter for player state estimation. Finally, Conditional Random Field is integrated into their work for performing joint classification, allowing them to borrow statistical strength from the reliable classifications to help ambiguous classifications(paraphase？）

Beetz et al. [4] built the system ASPOGAMO(Automated SPOrt Game Analysis MOdel) for football player tracking. It can give results of player detection and their traces and has the ability of working real-time. By estimating camera parameters, the system map location of players in a frame to their co-ordinations in a pitch. A color model is built for each type of players and template matching is used to localize location of players. Each model then forms a template and scan across the whole frame to detect and identify players at the same time. Classified detections are then put into Multiple Hypothesis Tracker for tracking task.

In work of Liu et al. [5] detection work is done by boosted cascade detector trained on Haar features, aided by background region filtering after the its color is learned. A player appearance model, built on GMM with bag of features representing players, is utilized to do classification. Tracking is regarded as a data association problem with the help of MCMC strategy.

Previous students [1] and [2] --basically work of [2] is based on [1] --tried background subtraction and linear SVM detector based on HOG feature for player detection, KNN based on BOVW features for classification and idea of tracking-by-detection with support of Kalman Filter for tracking. [2] also used CRF for label correction mentioned in [3], yet the author build a CRF per tracklet and did not utilize the mutex relation among tracklets, so it can be roughly regarded as using a majority label assignment.

He et al. [6] proposed Mask-RCNN network architecture for generic object detection, which has satisfying performance on detection of person and being one of the most state-of-the-art networks for object detection task.

Recently a survey [7] is published on various kinds of methods for sportsperson tracking, in which challenges and ways to mitigate them are briefly introduced. Unpredictability of players’ movement is seen as an obvious challenge, yet some researchers managed to mitigate its impact by predicting players’ locations in advance. Occlusion is another issue and () provides a possible solution of using multiple cameras.

Given that broadcast from single camera is used in the project, I first tried out the detection method used in [1] and [2], yet later migrate to using a pre-trained model from Mask-RCNN network. In identification phase, I utilized idea given in [3]. My approach for tracking is borrowed from the work of [3] and for label correction a method of majority label assignment is used after seeing the plausibility analysis of CRF on all tracklets in [5] and Actual effect of the simplified version of CRF used in [2].

2.2 Player detection

To do player tracking, we first need to know where they are. For videos from static cameras locating is relatively not that difficult, yet such kind of videos are too rare, especially in popular sports like football and basketball, so techniques suitable for moving scenes are later considered.

Machine learning with explicit features is likely to cope with the problem, deep learning with implicit features may also be a choice, as there are lot of successful experiences of applying both of them to general object detections.

2.2.1 Background subtraction

Background subtraction is a technique that removes all backgrounds in the image, leaving only the foreground. Here background means whatever we are not interested in.

By saying ‘subtraction’, we can easily think of the most direct and literal way of doing this: doing difference between images. Ideally, background won’t change across frames. So as long as we have an initial frame of background image and any other frame, we can simply isolate fore ground by subtracting pixels at the same position. Those represent background of course will not change and difference result will be 0, and for pixels occupied by foreground it shall be non-zero. Then simply applying a threshold to the difference image can give us the binary image, where zeros are backgrounds and ones are foregrounds.

However, we can seldom acquire a completely static background due to brightness change and wind disruptions to camera, meaning that background shall constantly be at slight change. This requires us to continuously update our background, or changed background will be misclassified as foreground, becomes noises in final binary image, and causing difficulty when we try to identify true foreground.

One way of doing this is using Gaussian Mixture Model(GMM), suggested by Friedman and Russell [3]. By adapting such a technique, we assume however the color of background model change, its value shall fall into a combination of Gaussian clusters that together represents the model. We try to do a binary classification on every pixel to determine whether it belongs to background or not.

Given a footage, the first frame is assumed to be the background image and used to build the GMM. As a new frame comes, every pixel will be classified and acquired background will be used to update the model, and this new background will then be subtracted from the next frame later. In this way, background is kept up-to-date and historical data older than a threshold will be discarded.

In the context of player detection, obviously players should be regarded as foreground and they should be extracted easily as they are always moving, unlike (almost) static background. However, it should be also noticed that there is also a moving ball in the frame, so additional methods should be used to filter out its blob.

2.2.1 detection by model based on HOG features.

A feature descriptor is some useful data extracted from an image that has actual meaning and can somehow represent the image. Among various kinds of feature descriptors, Histogram of oriented gradients (HOG), introduced by Dalal and Triggs [7], is widely applied to human detection.

For each cell that a picture is divided into, image gradients re calculated upon it. The gradients of adjacent cells (called blocks) are then grouped into HOGs and HOGs for each block of the picture eventually form the HOG feature of the picture. An SVM classifier can then be trained to do detection work according to the features of positive and negative samples, like the inventors of HOG did in [7]. For actual detection, a sliding window will go through the target picture and the cropped image captured by this window is put into SVM for detection.

2.2.2 detection by pre-trained model from Mask-RCNN

Instead of extracting features explicitly, a Convolutional Neural Network (CNN) picks information from pictures in implicit ways. Due to its convenience of adding non-linearity to models simply by adding activation functions, it gains extraordinary power of doing detection, segmentation and classification task and has become a popular, if not the first, candidate when dealing with images.

Through years of evolution CNN acquires larger power of instance segmentation (object detection & instance segmentation). By now the most state-of-the-art product is Mask-RCNN. For object detection it used a sub-architecture similar to Faster-RCNN, and for instance segmentation a Fully Convolutional Network (FCN) is utilized. Since only the detection part is what we want, we focus on the Faster-RCNN-like part.

Instead of getting a window swept across the whole picture, which is clearly non-efficient, the method of region proposal [8] is introduced. It first does image segmentation too see what could possibly be a meaningful object. For each segment, a blob is proposed and put into classifier to see if it is an object.

2.3 Player Classification

After detections are obtained, players picked out still needs to be classified to give their identification and to get their information updated correctly during tracking. In broadcast videos, face detection is not feasible due to low resolution and sometimes blurs during moving, so more salient and scale & rotation-resistant features should be chosen.

2.3.1 Features

2.3.1.1 RGB histogram

As human we mostly distinguish one person from another by different colors of clothes they wear, and computer may also borrow the knowledge of color for classification. Using raw color values may not be a plausible as this may cause our vectors to include similar features like a darker blue or a slightly brighter one. Similar feature does no help but adds to computational burden, so we use RGB histogram instead. For each color channel, a range of values are put into one bin, and a histogram represents occurrence frequency of colors in different value ranges. This histogram can then be used as (a part of) feature vector for classification.

2.3.1.2 SIFT

Players are always moving. They appear smaller in a frame when being far from camera and gets larger when getting close; directions they face are always changing and they are continuously in different poses. Despite that we still want to classify “detection of plater A” as “player A” however he/she looks in the video, so a feature that is invariant towards scale and rotation shall be used/integrated. Scale-invariant feature transform (SIFT), proposed by Lowe[9], meets this requirement. It has 4 major steps:

* + - 1. Scale-space extrema detection: The difference of Gaussians function (Figure 7) is used to efficiently search the image and find potential keypoints. Gaussian blur filters are applied across a range of scales, and the differences between adjacent images are computed to create difference images. Local minima and maxima in the difference images are noted as potential keypoints, found by comparing each pixel to the eight neighbors in the same image and nine neighbors in both adjacent images.
      2. Keypoint localization: Candidate keypoints are filtered to find those invariant to scale and orientation. Edge responses and keypoints with low contrast are removed. Locations are refined by using data from neighboring scales to find the most suitable scale.
      3. Orientation assignment: A histogram of oriented gradients is computed over the neighborhood of each keypoint, with 36 bins to cover 360. The bin with the highest peak is picked as the dominant orientation, and the patch is normalized by this so that all future calculations are done with respect to this orientation.
      4. Keypoint descriptors: The patch surrounding the keypoint is divided into 16 cells (4 by 4 sub-patches), and a histogram of oriented gradients with 8 bins is calculated for each cell (Figure 8). These are concatenated to form one vector of length 128 (16 cells ⇤ 8 bins) used to represent the patch.

2.3.1.3 MSER

Maximally stable extremal regions (MSERs) are regions that are connected components that keep its shape unchanged throughout appliance of a range of thresholds. Intuitively, these regions we get will have their color obviously different from their surroundings, thus should be considered regions of interest (ROI). In work of [3], SIFT features are extracted from MSERs of a detection.

2.3.2 BOVW

Bag Of Visual Words (BOVW) model originates from field of NLP, trying to represent an image with a set of ‘visual words’. In [3], SIFT from MSERs are not directly used for classification, but first grouped into visual words, then visual word histogram (occurrence frequency) is used as feature of vector for an image. This is done for 2 reasons:

1.number of SIFT extracted from every image is different. As we all know features should have the same length first to put into classifiers, so their length should be normalized in some way.

2. some SIFT in an image may represent similar features and it is unnecessary to see them as different ones; apart from that, one group of SIFTs from one image may also be similar to other groups of SIFTS and they should neither be seen differently. If we do so, we are just adding duplicate features to the final vector and increase computational burden.

2.3.3 Classifier

After feature vectors are gathered and normalized in length, they can then be put into classification task. Simple K Nearest Neighbor (KNN) classifier is used here.

Training the classifier is simple: just sow feature vectors of training examples into vector space together with their label. For classification phase, hyperparameter K is selected. After K is chosen, we then select K nearest neighbors, according to distance, of the example needs to be classified, do a majority vote, and the most common label is assigned as the label of the new example.

Here K can be neither too small, or classification procedure is really susceptive to noise; nor too large, else price of computational burden will overwhelm the benefit of mitigating over fitting and points near boundaries are likely to have equal number of classes from both classes, causing difficulty for label assignment.

2.4 Tracking

As we gather detections across frames, we shall also connect together detections for players to give their trajectories. In the project, this is done with idea of tracking-by-detection, assigning every detection in a new frame to one tracklet maintained for each player.

2.4.1 Kalman filter

Kalman filtering [10] provides a way of using observation and prediction of data comprehensively, considering the effect of system and environment noise. State estimation based on Kalman filter will be more accurate than relying solely on observation, as it cannot be fully precise.

In the circumstance of tracking, players’ locations are what we need to track, and historical information recorded for a player is saved in a tracklet maintained for him/her in temporal sequence. For each player, his/her location for current frame is first predicted based on historical information using the filter. The prediction is based on assumption that player moves almost linearly between fames, so formula for predicting can then be elicited:

(location updating formula)

After corresponding detection is found, it is then combined with the prediction made to give the final location of the player in current frame. Again, a prediction for next frame is made. This is done for every frame to keep position of players updated, their historical information saved in tracklets, hence tracking is achieved.

2.4.2 Temporal coherence

The method of determining which detection should be assigned to which tracklet is not simply assigning a detection to a tracklet in which previous detections have the same label as classification result is likely to be erroneous, instead, it is based on temporal coherence. People cannot run too fast, thus the distance they move between frames must be under a certain limit.

Base on this common sense, we can set a hyperparameter defining the maximal distance a player can move in each frame. After matching potential pairs of tracklets and new detections, we check if Euclidean distance between each pair of prediction made through Kalman filter and detection is smaller than this threshold. Qualified distance leads to a detection combining with paired prediction to give final location of corresponding player, otherwise the detection will be rejected by tracklet as they are potential errors.

2.5 Label correction

After a running through of target broadcast video, Tracklets will contain full trajectory information of a player, however, since detections in a tracklet may still be different due to the way we assign detections in tracking step. Here a majority label voting is done on each tracklet to give its final label.

1. Development

In this section, all method I tried for system implementation are described.

* 1. System workflow

Before the system starts to work, models for player detection and player classification are prepared. A target broadcast video will be gone through twice. In the first iteration, for each frame, detections are found, their labels are given by classification model, and assigned to tracklets. After this iteration finished, tracklets that only consists of false positives are filtered out and label correction is done on each remained tracklet. They are then sorted according to temporal sequence of the first frame. At last, tracklets are drawn on target video in the second iteration.

(flowchart)

* 1. Player detection

3.2.1 Heuristics

Some heuristics are used to eliminate false positives to some degree and speed up detection.One pre knowledge is that in a footage from static camera, the area representing pitch remain unchanged throughout frames. Also, players cannot run out of pitch. Consequently, for every frame a sub image that only contains pitch and players with in is cropped out for detection. This can accelerate detection process, since less areas need to be swept through, and reduce false positives, as any of them that will appear out of pitch region in the original region will no longer appear in the cropped one.Secondly, bounding box of a player mustn’t be too large or too small. Although scale of players may vary as they ran towards or away from camera, it should still between a certain range. So, boxes that do not have its size fall in the range should also be discarded.

3.2.2 Background subtraction

At first, I tried to use background subtraction for detection baseline. For a static video its background remains unchanged, thus making it plausible to capture a pure background frame and subtract it from every other frame to get foreground information.

The background subtraction algorithm implementation I used is from OpenCV [11], it is achieved base on Gaussian Mixture Model (GMM).

(pic of pure background)

(pic with players

As we can see, in the dataset the foreground (players & football) is easily distinguishable from background, thus we can expect their pixel values are clearly different. Heuristically, for each frame I get, we can simply do the subtraction and blobs we get marks location of player.

At first, background model will be subtracted from the frame and updated with background pixels. The difference image then goes through a threshold to filter out shadows captured.

The blobs too small represents background and noise points, so here a threshold is applied to filter them out. For rest of the blobs, findcontours [13] from OpenCV is used to draw a bounding box around each of them by using maximum and minimum information of x, y coordinates kept in obtained contour. It has an advantage that the rectangle is drawn stick with blob, thus marking precisely the player’s location and include little extra information.

A flow chart is given below to assist understanding of the subtracting workflow.

(flowchart)

Some detection examples are shown below.

(pic of occlusion) (pic of unconnected blob of one person) (pic of detection failure of static person)

From the examples above we can see that background subtraction have some shortcomings that may seriously hinder later steps.

First, it is too vulnerable to occlusion. When occlusion occurs, in the subtracted image we can only know there are something different from background, but we don’t know it represents 2 people as there are only one blob. Other information that may help us judge that here are two people are lot after subtraction process.

Second, it is vulnerable to static figures. Since the implementation uses background of new frames to update background, although this may give it some ability to cope with moving background, it also makes the algorithm difficult to find players that remain static for some time. When a player stops moving, more of its pixels will have similar value with pixels at the same place in the previous frame and these pixels will be seen as a part of background due to how the algorithm works. As a consequence, there will be no more blob for this static player that mean we completely lose track of him/her.

Third, blob for a player may not be always connected, causing the bounding box cannot cover the whole player.

I made an attempt to at least solve the third problem by adding several iterations of dilation then several iterations of erosion to connect blobs belong to the same person in order to give exactly 1 bounding box for 1 player with the correct size.

(pics of connected blob & exactly one box)

We can see that some how the third problem is solved, yet other 2 problems are still difficult to deal with using background subtraction

(pics of

3.2.3 SVM detector based on HOG

Then I migrate to use an SVM detector. An SVM detector should first be prepared. For each frame, a sliding window will sweep across the whole frame, and each window is then put into the classifier to see if it contains a person. I tried to used a pretrained classifier in OpenCV [14] and a self-trained one.

3.2.3.1 pretrained classifier

This classifier is trained on INRIA dataset, thus being a general purpose one. Given the common sense that boundaries of human beings are similar and HOG feature mainly describe edges, I propose a guess that sports players may also have similar HOG features with non-sportsperson. Thus, a general-purpose detector may have acceptable result on sportsperson detection.

(Results of applying the detector)

The results above actually denied my guess. We can see that the general-purpose detector gives too many false negatives, thus cannot provide enough detections to do continuous tracking. The reason may be that the dataset INRIA itself is not general enough – it only contains examples of pedestrians, yet HOG features of pedestrians are mutually similar but obviously different from those of sports players who often has more varied pose.

3.2.3.2 self-trained classifier

Then I decided to train a linear SVM detector based on HOG features using the given football dataset. OpenCV has implementation of HOG feature computation [15] and SVM classifier [16], so I only need to provide training data here.

After training examples are loaded, a 64\*128 sub image of each example is first cropped to fit sliding window size defined in the SVM implementation. Then HOG features are computed and put into the classifier for training. After the training completes, hard negative mining is used and obtained fale negatives are again put into the classifier for further training to improve its accuracy.

Using the self-trained classifier for detection should be in the same way as we use the pre-trained one, only this time load the self-trained SVM classifier

(a pic of many false negatives)

I observed that there are still too many false positives, so they are used to further train the classifier. I did this several times until false positives almost don’t appear.

(result of normal success on all players)

(result of occlusion at failure)

We can see that this time the classifier can give more continuous detection , yet another problem still exists: It is vulnerable to occlusion in a different way: the detector fails to detect players that are near each other. I guess there may be too few true positives describing occlusion, so I further added examples of occlusion into training examples.

(results of false negatives appears again)

However, false negatives began to appear again before problem at occlusion is solved. My guess is that after those examples are added, the SVM hyperparameter can no longer put all positive examples at one side and there are more negative examples at the positive side. This shall be a problem of hyperplane lacking non-linearity.

3.2.4 Detector from Mask-RCNN network

Speaking of adding non-linearity, a neural network is especially good at doing that. What’s more, compared to SVM’s limitation on types of kernel tricks, a neural network can achieve far more complex hyperplanes. The Mask-RCNN network implementation I used is from [17] and there are minor adjusting to the code to fit a newer version of Keras framework.

The model is trained on MS COCO dataset [18], it contains photos of 91 types of objects, including human,thus the model is also a general-purpose one. Instead of using only one type of feature, more comprehensive ones are extracted due to how convolution layers work.

Player detection work is like testing phase for a neural network. A frame is fed into the Mask-RCNN and output will be instance segmentation result for it, including a colored mask,a label and a bounding box for every recognized object. Here we only use bounding boxes produced.

3.2.4.1 Heuristic at occlusion

No matter how powerful the model is against occlusion, it still fails when occlusion is too severe. To somehow mitigate this, an extra heuristic is proposed.

Normally, the abounding box of a player cannot change suddenly in size when moving to the next frame. However, when occlusion start to hinder detection result, bounding box of one player will disappear and that of the other will suddenly gets larger. So when such a circumstance occurs and distance between the box that grow larger than a certain percentage and the one disappears is under a threshold, we know occlusion takes place, and the large, single box will be dismantled into two smaller ones according to its actual location and prediction of where the 2 smaller boxes should have been.

(pics of robustness at occlusion)

We can see that the result is more satisfying. It almost never gives false negatives (also due to the reason that there are no other objects fall into predefined classes of the dataset), and it is rather robust towards occlusion. Successful detection of all players under occlusion are correctly given until one almost get completely covered by the other.

3.3 Player identification

The classification model I utilized is from [1]. At first, I also tried to use hyperparameters recommended in its evaluation part, where only SIFT featuer is used, k=11 and size of bag of words is 200, yet this did not perform well in my system so further experiment is done, and finally my configuration is using SIFT based on MSERs, k=11 and size of bag of words is 75.

3.4 Tracking

Evaluation

Detection

Data collection

Some preprocessing is done on training and testing videos to shorten data collection time. For trainingVideo.mp4, data collection starts at 00:24 where the 2nd player enters the camera frame, for testingVideo.mp4 it starts at 00:22, where the first 2 players appeared.

Ground truth data is needed to do evaluation here. The method I used is basically the same as that in [1], except for the bounding boxes are got using model from Mask-RCNN network.

Evaluation standard

Ideally, a successful detection should include (approximately) the whole body of a player; in practice, detection of player entering the frame should also be considered; in the meanwhile, detections that only contains some body parts of a player or multiple players should be considered false positives.

Quality of detection is evaluated after basic heuristics are applied to filter out those obvious false detections, or to say, only evaluated on detections that are to be classified and put into tracklets later, as these are detections that will affect how the system perform as a whole. Detections filtered out at the beginning do not have influence on performance of the system.

Unlike the method of setting number of players(note as #oP) as a parameter of evaluation program in [1] to estimate total number of detections in a video, I calculate total number of player occurrences in a more fine-grained way:

Taking into consideration that players will enter and exit the camera frame from time to time, I first estimate frame intervals in which #oP is of a certain value. This can be nearly accurate by stopping at whenever number of players change and record the index of current frame. Between each of 2 continuous records #oP remain unchanged. By multiplying #oP and length of the interval and adding it up on all intervals, total number of #oP occurrences in a footage can be given more accurately than simply multiplying most common #oP with video length.

Detections in a footage are gathered automatically, and manual justification is done to distinguish true positives from false ones, as This justification of whether a detection is o the whole body of a player(sometimes including occluded parts) cannot be easily done.

Accuracy can then be given by:

(formula)

Evaluation & experiments

At first, the baseline accuracy of player detection on 2 videos are evaluated:

trainingVideo

testingVideo

Then I tried on some different values of scale change threshold th, controlling when a detection should be split into 2, as mentioned in 3.2.4.1:

th=1.3:

References

[1] A. Dunlop. Tracking Football Players, 2017.

[2] S. Akerman. Tracking Football Players, 2018.

[3] W. L. Lu, J. A. Ting, K. P. Murphy, and J. J. Little. Identifying players in broadcast sports videos using conditional random fields. In CVPR 2011, pages 3249–3256, June 2011.

[4] Michael Beetz, Suat Gedikli, Jan Bandouch, Bernhard Kirchlechner, Nico von Hoyningen-Huene, and Alexander Cli↵ord Perzylo. Visually tracking football games based on tv broadcasts. In IJCAI, pages 2066–2071, 2007.

[5] Jia Liu, Xiaofeng Tong, Wenlong Li, Tao Wang, Yimin Zhang, and Hongqi Wang. Automatic player detection, labeling and tracking in broadcast soccer video. Pattern Recogn. Lett., 30(2):103–113, January 2009.

[6]He, Kaiming & Gkioxari, Georgia & Dollár, Piotr & Girshick, Ross. (2017). Mask R-CNN.

[7] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 2005, pp. 886-893 vol. 1.

[8] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 580-587.

[9] David G. Lowe. Object recognition from local scale-invariant features. In Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2, ICCV ’99, pages 1150–, Washington, DC, USA, 1999. IEEE Computer Society.

[10] R. E. Kalman. A new approach to linear ﬁltering and prediction problems. Transactions of the ASME - Journal of Basic Engineering, 82:35–45, 1960.

[11] Itseez. Open source computer vision library. <https://github.com/opencv/opencv>

[12] [https://docs.opencv.org/3.4.2/d4/d86/group\_\_imgproc\_\_filter.html#gaabe8c836e97159a9193fb0b11ac52cf1](https://docs.opencv.org/3.4.2/d4/d86/group__imgproc__filter.html" \l "gaabe8c836e97159a9193fb0b11ac52cf1)

[13]