Ball tracking in a Volleyball environment

Luca Cazzola, Alessandro Lorenzi

University of Trento, Signal Image and Video

Introduction

Goal

Ball tracking in volleyball

- without deep learning architectures

Challenges

- **Problem:** it's hard to discriminate the ball
  - **Shape**:
    - ball deforms into an elliptical shape
    - contours blends with the background
  - **Color & Texture**:
    - not always meaningful: ball is spinning
- **Solution:** use motion-related properties to identify a set of "**ball candidates**"
  - $\Rightarrow$ only factor constant in volleyball (if the ball is stopped, it is a foul)

Workflow

1. Identify **moving entities**
2. Construct bounding boxes around them (**detection step**)
3. Train a classifier to discriminate the regions that are more likely to contain a ball (**classification step**)

Assumption

The camera is fixed

Detection

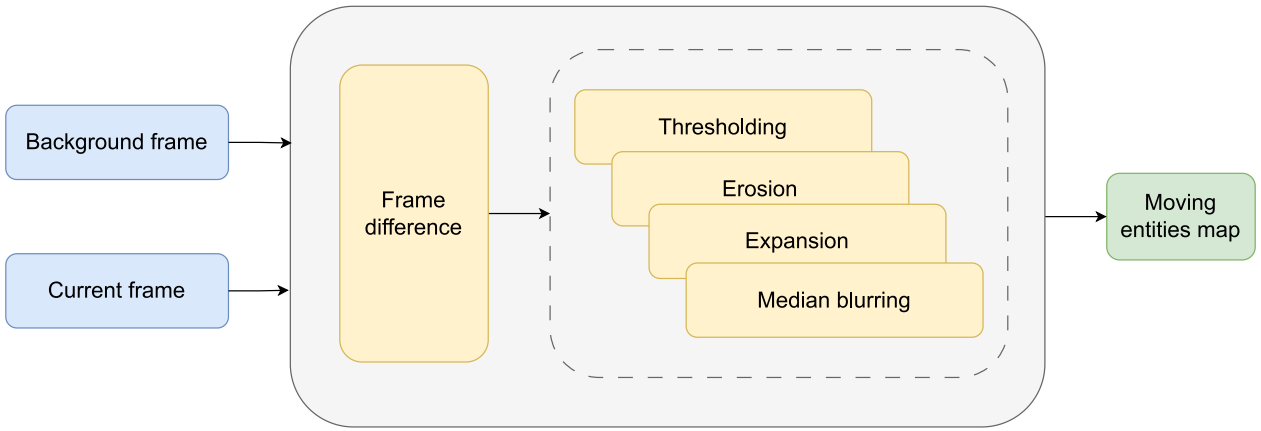
- How to extract the motion information?
  $\Rightarrow$ Use **background frame**
  - Problem: it's not available
  - **Solution:** estimate it by sampling a good amount of random frames from the video and compute the median among them

- Moving entities:
  - **players and ball**
  - **noise:**
    - net
    - referee
    - background

$\Rightarrow$ we have to **reduce the unwanted information**

Reduce the unwanted information

1. **Thresholding**: better distinction between actual moving entities and slight variations in the image
2. **Erosion**: eliminate isolated white spots
3. **Expansion**: restore the original size of remaining white regions and make them intersect with each other
4. **Median blurring**: fill more the gaps that might still exist



Detection: reduce the unwanted information

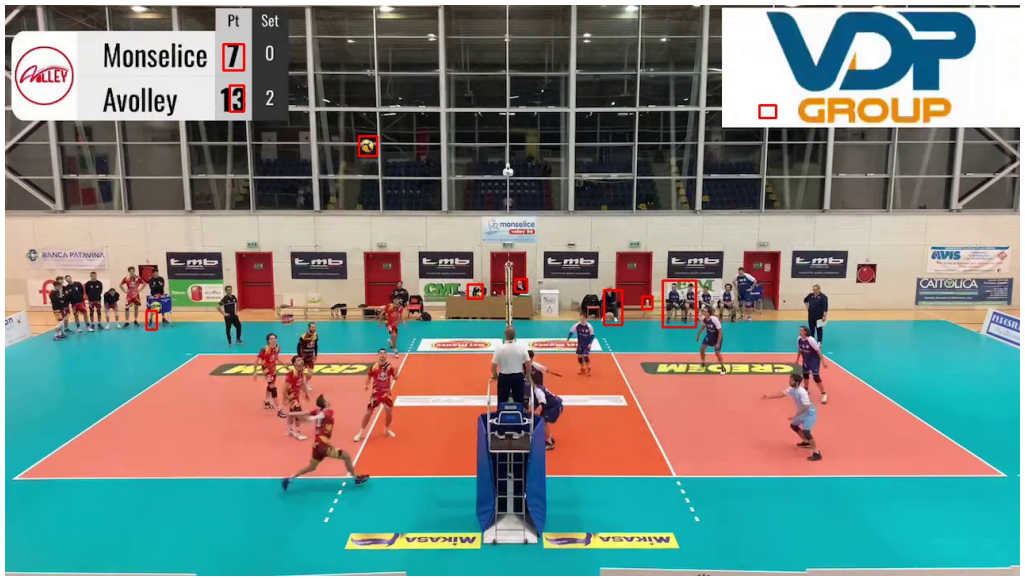
- Now: **extract bounding boxes** from the obtained map
  $\Rightarrow$ Compute the **contours** of the white regions

  We have a lot of bounding boxes. Need some criteria:
  - **Size**: average size of a bounding box containing a ball in the range [150, 3000] pixels
  - **Aspect ratio**: aspect ratio of the box generally is between [0.5, 2]
- $\Rightarrow$ Everything exceeding these limits can be really often safely discarded

Example

(a) Frame difference result

(b) Moving entities map



(c) Detected bounding boxes (in red)

Features

Reason on ball properties: shape, texture, color:

$\Rightarrow$ Use of **Histogram of Oriented Gradients (HOG)** features

Figure 2: HOG features visualization

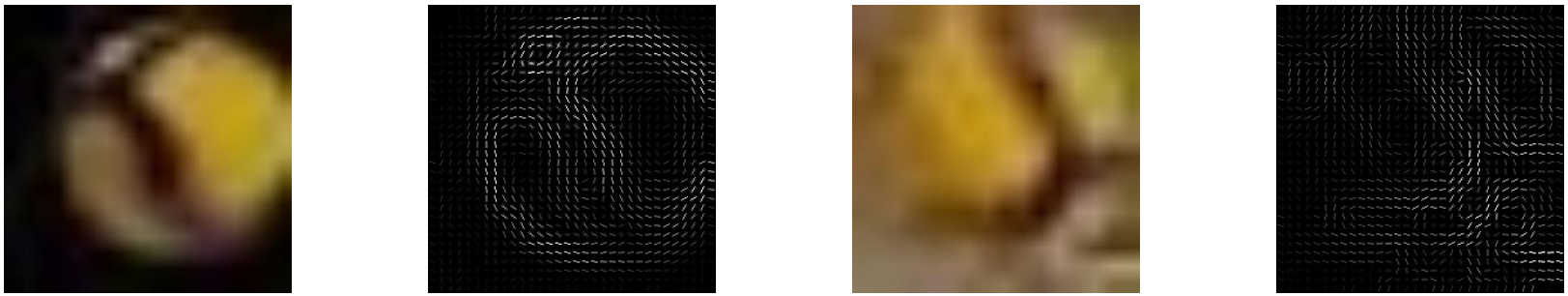


Figure 3: HOG features visualization - **Real case**

Training & dataset

- **random forest** to make classification
- we need a **negative set** (no ball) $\Rightarrow$ extract random regions of the frames
- $\sim$ 1200 positive samples (ball) and $\sim$ 1000 negative (no ball)
- number of features: reduce to 20 with **PCA (Principal Component Analysis)**

Training & dataset

Random forest training results:

	precision	recall	f1-score	support
0	0.97	0.91	0.94	160
1	0.94	0.98	0.96	221

Overall accuracy : 0.95

Conclusion

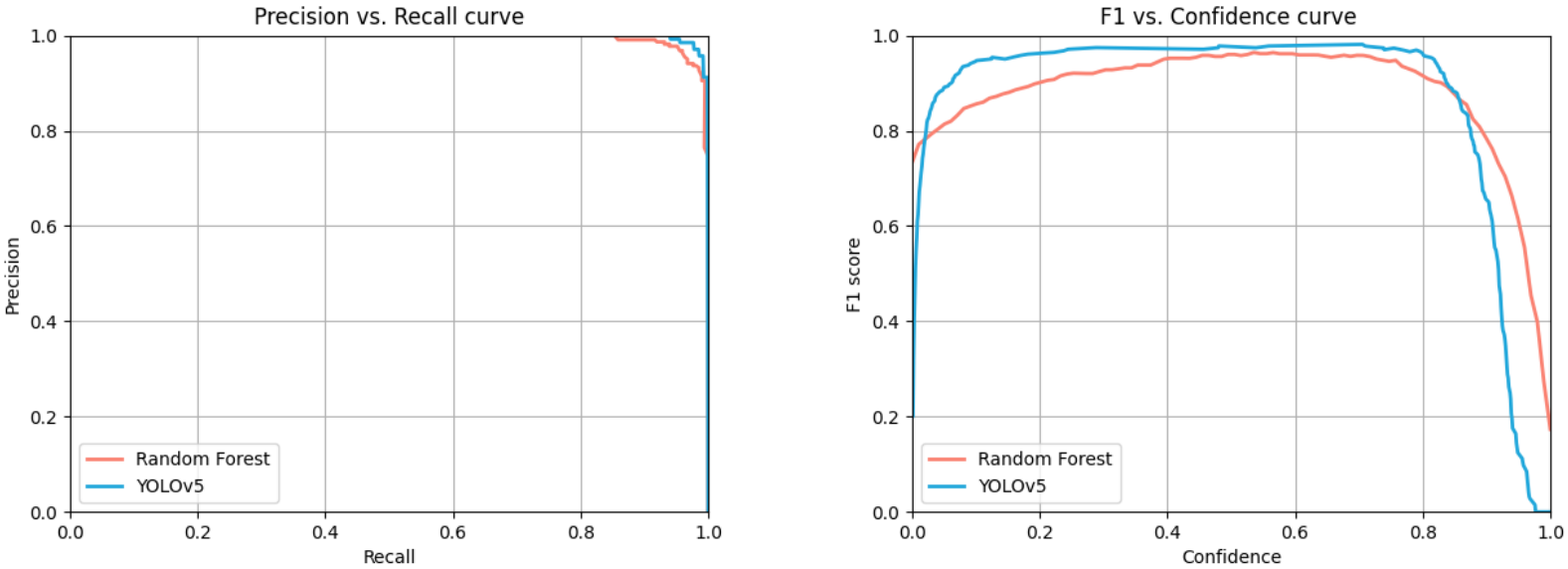


Figure 4: Random Forest - YOLOv5 comparison

- more good data ⇒ better performance for both models
- possible solution: combine the two (YOLO in feature extraction)

Thanks for the attention