Politecnico di Milano

3D Trajectory in Sports - Table Tennis

July 17, 2018

Version 1.0

Authors:

- Matheus FIM (Mat. 876069)
- Reda AISSAOUI (Mat. xxxxxx)

Prof. Vincenzo CAGLIOTI

Contents

1 Problem Formulation **2**

2 State of the Art **2**

3 Solution **4**

3.1 Table detection 4

3.1.1 Lines detection 4

3.1.2 Lines grouping 5

3.2 Camera Calibration 6

3.3 Ball Tracking 6

3.4 2D Trajectory Estimation 6

3.5 Camera Synchronization 6

3.6 3D trajectory Estimation 6

4 Algorithm **6**

5 Experimental Results **6**

6 Conclusions **6**

6.1 Purpose and Scope 6

6.2 Definitions and Abbreviations 7

6.3 Reference Documents 7

7 Integration Strategy **9**

7.1 Entry Criteria 9

1 Problem Formulation

Table tennis is a demanding sport where the referees have to pay attention at a complex set of rules at all moments. For this reason, much can be obtained from having in-place a system capable of reconstructing the 3D trajectory of the ball during the course of a point. For instance, eventually unclear plays can be decided correctly or even coaches can better analyse the players performance. However this is not a simple task. Many of the techniques applied to other sports cannot be used for this context, as the table tennis ball can be hard to detect due to its reduced dimension and also cannot be relied to follow a parabolic trajectory as its small mass causes it to be heavily affected by the air resistance.

The proposed system takes as input real matches video sequences and is able to overcome a wide variety of problems, such as high object motion, blurring, uneven lighting, background merging, object confusion and occlusion. Thus generating as output an estimation of the 3D trajectory of the ball. The algorithm relies on videos recorded by two non synchronized RGB cameras and uses this information to detect the table, calibrate the cameras, track the ball in between frames, synchronize the cameras and finally combine the single trajectories for the production of the 3D trajectory.

The Experiments were conducted on 60fps and 120fps video samples recorded with GOPRO cameras in an indoor environment and produced satisfactory results. In all tests the trajectory was estimated with suitable accuracy, whereas the most satisfying results were experienced in the higher fps videos. Also, the adaptive "region of interest" allowed to reduce both the time of execution of the algorithm and the incidence of false positives.

2 State of the Art

Wong et. al.[**?**] described an algorithm capable of tracking table tennis balls in services and stated the accuracy was high enough for umpiring applications. Their work handled the following problems:

- **High object motion:** The high speed of the ball may cause it to be blurred, colour faded and distorted in shaped.

- **Multiple moving objects:** Apart from the ball, the players, the rackets and background elements in the scene exhibit motion.
- **Uneven lighting:** Light sources on the ceiling tends to cause the ball to be brighter at the top than at the bottom.
- **Occlusion:** The ball can be blocked by the player, by the rackets, or even disappear from the field of view in the upper part of the video.
- **Merging:** Low contrast from the ball and the background may cause difficulties in distinguishing both.
- **Object Confusion:** Objects in the background with similar color (e.g. human skin) and shapes (e.g. a bald person head) may be confused with the ball.
- **Small size:** Size of the ball is only a fraction of the frame, which randers histogram-based detection unsuitable.
- **Time constraint:** latency from detecting to tracking the ball must be minimised.

In order to overcome these issues it is proposed a method that executes an object segmentation followed by a object evaluation algorithm. The object segmentation is composed by a two-pass thresholding (TPT) algorithm, capable of adapting the threshold according to context. The first pass highlights only elements of the frame with a highly similar color to the one of the ball, and the second thresholding relax the constraints in order to reconstruct the shape of the object. The output is a set of candidate balls to be further evaluated by the object evaluation algorithm. This second algorithm is also executed in two steps. In the first it is controlled if the individuated object displays an upper rounded contour, its location is consistent with the predicted location and if it exhibits motion both at its centre and in the predicted location. And in the second, from the set of objects that were able to pass the first step, one is chosen as the ball based on spatial geometric measurements (area, maximum width, maximum height, perimeter, roundness, and error on the rounded upper contour).

These elements are going to be further discussed in the document as they were applied in the project.

Tamaki and Saito(20..) proposed a method of reconstructing 3D trajectories of table tennis balls. They consider both the problem reconstructing the trajectory from two rgb cameras and from one rgb-d

camera (i.e. capable of providing depth information). One of the novel approaches is the method that approximates that a ball is travelling on tilted planes, and thus it is able to accurately identify which of the possible candidates is the ball. Also they leverage on the work produced by **Zhang (CITE)** to compute the camera intrinsics ... (continue)

3 Solution

The proposed solution is divided in table tennis detection, camera calibration, ball tracking, 2D trajectory estimation, camera synchronization and 3D trajectory estimation.

3.1 Setup Phase

selection of the ball...

3.2 Table detection

In order to take detect the table, we have to take advantage of its features. The table edges are long straight lines. It is also blue and stands out from its environment. So our approach consists of finding the lines present in the input image and then filter them according to different criteria. After that, the lines are grouped depending on which side of the table they belong to. From this group of lines, the four sides of the table is calculated

3.2.1 Lines detection

Using Hough peaks and Hough lines, the algorithm start by detecting the lines that are present in the input image. We consider as a valid lines the ones that are on the edge of the table. The lines are therefore filtered on the following criteria:

Orientation

Given the point of view of the camera, we don't expect to have an edge of the table showing vertically on the image. Hough lines algorithm uses

a polar coordinates system, therefore we filter the lines directly from θ of the polar coordinates.

Color separation

A valid line will have from one side the blue table color and then from the other side any other color (floor green, wall white...). To decide which lines are valid, two sample pixels are taken from each side of the line. The midpoint of the line along with the orthogonal line going through the midpoint are calculated. The sample pixels are equally distant points from the centroid on the orthogonal lines. The HSV values of these points are considered as it is a more robust way to deduce the color.

Given two points $p_1 = (h_1, s_1, v_1)$ and $p_2 = (h_2, s_2, v_2)$, a color is considered blue if

$$blue_1 = (h_1 > BLUEMIN) \wedge (h_1 < BLUEMAX) \wedge (h_1 > VLIGHTMIN)$$

where the thresholds (BLUEMIN, BLUEMAX and VLIGHTMIN) are defined from the hue values of the blue color. A minimum value of brightness is also imposed as blacks may be considered as blue.

A line is considered valid if

$$line = blue_1 \veebar blue_2$$

3.2.2 Lines grouping

Still needs to be developed

After detecting the lines on the edge of the table we need to cluster them using kmeans ....

Because of the distortion caused by the wide angle lens, the lines detection algorithm sometimes return many lines for one table edge.

we have multiple lines per edge =¿ weighted average grouping

then grouping =¿ calculate intersects and then finished

3.3 Camera Calibration

3.4 Ball Tracking

The ball tracking is done using Wong et. al.[**?**] algorithm. The candidate objects are individuated in an object segmentation phase (two-pass-thresholding algorithm) and then selected based on an object evaluation step (ball selection algorithm).

3.4.1 Object Segmentation

The objective is to segment candidate balls from the other elements in the frame. Candidate balls are objects that exhibit similar properties to the object of interest (OOI). To perform this segmentation it is implemented a color thresholding algorithm. The basic idea is to create a binary differential image where pixels with a color similar to the one of the ball are turned white while other are turned black, the neighbouring white pixels are then merge together to form an object. However, to increase precision this is divided in two passes. In the first, a coarse threshold is applied so only pixels with a sufficiently similar color to the ball are filtered, the pixels are then grouped as described above. This first thresholding operation is expected to produce a distorted object, since the coarse threshold may cause small color variations due to lighting or movement to be ignored. In this matter, the second pass is applied only on the regions where candidate balls have been found and with a more relaxed threshold, allowing to recompose the original shape of the object without considering obviously wrong candidates.

Configuring the TPT thresholds is not an easy task, as it may widely varies depending on the specific conditions of the input video. Hence, it is implemented an automatic tuning of the TPT threshold. This can be done using a known reference location for the ball, selected in the setup phase described before.

-

3.5 2D Trajectory Estimation

3.6 Camera Synchronization

3.7 3D trajectory Estimation

4 Algorithm

5 Experimental Results

6 Conclusions

References

This section records all revisions to the Document.

Version	Date	Authors	Summary
1.1	15/01/16	Domenico Favaro, Caio Zuliani, Matheus Fim	Initial Release
1.2	17/01/16	Domenico Favaro, Caio Zuliani, Matheus Fim	Second Release

6.1 Purpose and Scope

The Integration Test Plan Document (ITPD) serves to present the integration sequence and testing for all Subsystems and Components that conform PowerEnjoy Car Sharing Service. This is a key part to guarantee the functioning and quality of the software. The Document will present the division of the System in Subsystems and Components that will endure individual testing as independent modules and then be subject to integration on the whole System.

6.2 Definitions and Abbreviations

- **RASD:** Reqirements And Specifications Document.
- **DD:** Design Document.
- **ITPD:** Integration Test Plan Document.

- **SDK:** Software Development Kit
- **App:** Application, refering to Web or Mobile App.
- **Subsystem:** Part of the system the generally encapsulates one or more features.
- **Component:** Self sustained part of the System that provides with functionalities and is part of one or more subsystems.
- **Bottom-up:** Referring to Bottom-up testing. Each component at lower hierarchy is tested individually and then the components that rely upon these components are tested.
- **Top-down:** Top-down integration testing is an integration testing technique used in order to mock or simulate the behaviour of the lower-level modules that are not yet integrated.
- **Mock:** Simulation that mimic the behavior of certain objects and fucntions in controlled ways, done to test the behavior of some other object.

For other concepts concerning the Service definition look in the **Glossary** section of the RASD and DD.

6.3 Reference Documents

- Specification Document: Assignments AA 2016-2017.pdf
- PowerEnjoy Requirements And Specifications Document (RASD)
- PowerEnjoy Design Document (DD)
- Example Document - Integration testing example document.pdf
- Testing Tools Documentation:
  - JUnit User Guide - `http://junit.org/junit5/docs/current/user-guide/`
  - Arquilian Guides - `http://arquillian.org/guides/`
  - JMeter User Manual - `http://jmeter.apache.org/usermanual/`

7 Integration Strategy

7.1 Entry Criteria

We define the criteria that must be met before integration testing of the system components. We consider Integration a part of the production development. In order for production to start all documentation must first be written and up to date, including RASD and DD, to have a clear and full scope of the system components functionalities and importance. Once in production, the integration of a singe component can be done when the following criteria is met:

- The Component feature must be 100% complete, that is all classes and functions must have been implemented.
- No tickets must be opened for the Component, no bugs or cosidered missing features must be present.
- Individual component testing must have been performed, using JUnit to test its classes and functions.
- All the interfaces the Component has to communicate to have to be present or at least mocked to be able to test its coupling.