TENNISSENSE: A PLATFORM FOR EXTRACTING SEMANTIC INFORMATION FROM MULTI-CAMERA TENNIS DATA

Ciarán Ó Conaire, Philip Kelly, Damien Connaghan and Noel E. O'Connor

CLARITY: Centre for Sensor Web Technologies,
Dublin City University, Dublin 9, Ireland
{oconaire, kellyp, connagha}@eeng.dcu.ie

ABSTRACT

In this paper, we introduce TennisSense, a technology platform for the digital capture, analysis and retrieval of tennis training and matches. Our algorithms for extracting useful metadata from the overhead court camera are described and evaluated. We track the tennis ball using motion images for ball candidate detection and then link ball candidates into locally linear tracks. From these tracks we can infer when serves and rallies take place. Using background subtraction and hysteresis-type blob tracking, we track the tennis players positions. The performance of both modules is evaluated using ground-truthed data. The extracted metadata provides valuable information for indexing and efficient browsing of hours of multi-camera tennis footage and we briefly illustrative how this data is used by our tennis-coach playback interface.

Index Terms— sports video analysis, semantic knowledge extraction, tennis, ball tracking, player tracking

1. INTRODUCTION

In this work, we are concerned with the automatic extraction of semantic information from tennis video in order to facilitate efficient browsing and retrieval of the video content by tennis coaches. In collaboration with Tennis Ireland [1], the national governing body for the sport of tennis in Ireland, we have developed the TennisSense system [2]. This is a technology platform which aims to provide their coaches with technological solutions that allow them to more effectively develop the next generation of elite tennis athletes. This platform includes a network of nine IP cameras positioned around the tennis court.

This paper describes the TennisSense system and how video analysis techniques are used to extract semantic information from tennis training matches recorded by the system. Specifically, we track the tennis ball and players, and detect when the ball is hit. We illustrate how this semantic information can be used in an interface for tennis coaches, allowing powerful navigation and searching of tennis video.

The remainder of this paper is organized as follows: Section 2 outlines previous work in the area of sports video analysis. We give an overview of the TennisSense platform in section 3. In section 4 and 5, we describe and evaluate the video analysis components we use for semantic information extraction, namely ball hit detection and player tracking, respectively. Section 6 provides illustrative examples of how the extracted semantic information is used in our tennis-coach playback interface. We give our conclusions and directions for future work in section 7.

2. PRIOR WORK

At present, there are a number of commercial technological solutions for sports video analysis. Both Protracker Tennis [3] and Dartfish [4] provide frameworks that simplify the manual annotation of sports video and give statistical feedback to the analyst. Dartfish also includes some semi-automated tracking and measurement features. ProZone [5] provide semi-automatic frameworks for indexing sports video but with high labour costs due to the manual intervention required to correct the errors of the automatic processing and to provide high quality annotation.

There is much work in sports video analysis focusing on the detection of important events in the video [6]. However, the majority of approaches use broadcast sports video and can exploit the editing style of the content, such as the close-up of a player after a soccer goal, for example. Player-tracking and ball-tracking are also common methods for extracting semantic knowledge and the majority of approaches employ background subtraction or frame differencing to obtain candidate blobs which are then classified, tracked or discarded [7, 8, 9, 10]. The main difficulty in such approaches is the problem of occlusion, but this can be overcome by using multiple cameras [10] or using an overhead camera, as we do in this work. Candidate detections can be filtered if they do not conform to certain constraints, such as knowledge of the ball's colour [9]. In our setup, the ball contains a relatively small

This work is supported by Science Foundation Ireland under grant 07/CE/I1147

number of pixels and is subject to compression artifacts, so we do not use such constraints.

3. TENNISSENSE SYSTEM

The TennisSense technology platform [2] is an indoor tennis-court that we have instrumented with extensive data-gathering infrastructure for use as a test-bed for sports and health research. The platform has been developed in collaboration with Tennis Ireland [1], the national governing body for the sport of tennis in Ireland, whose members regularly use the system to record training sessions and matches.

This infrastructure includes a UbiSense spatial localisation system [11] and nine IP cameras positioned around the court, with pan, tilt and zoom capability (see figure 1). The two cameras at either end of the court are AXIS 215 PTZ cameras and have pan, tilt and high zoom (PTZ) functionality. The high zoom is useful for obtaining a front view of the opposing side, or for focussing on the feet of the player from behind the court baseline. The other seven cameras are AXIS 212 PTZ cameras which have wide angle lenses (140°) and include fast digital PTZ functionality by subsampling from a high-resolution sensor. The video from all nine cameras is captured and synchronised at a single PC. In the future, we intend to use this framework to augment the video data with biomechanical and physiological data obtained a variety of other wearable sensors, for example: accelerometer-vests [12], heart rate monitors or sweat analysis [13].

An overhead camera was a specific requirement of the tennis coaches, in order to visualise tactical shots and movement during matches. Currently, we perform video analysis on this camera only (see figure 2(c)) and not on any of the other camera feeds that are available to us, but we plan to integrate data from these cameras in our future work. The overhead camera is an AXIS 212 PTZ camera and is positioned 13.8 metres above the court to overcome problems of occlusion. We work with a resolution of 640×480 and a framerate of 30Hz. In the following sections we describe the video processing components we use to automatically extract semantic information from tennis training matches and how we use this data to provide powerful semantic browsing and querying tools for tennis coaches.

4. BALL HIT DETECTION

The fundamental unit of play in a tennis match is the *tennis shot*; a tennis ball that is hit by one player in an attempt to land the ball on the other side of the court. In this section, we describe how we detect when a ball is hit by a player and how we track the trajectory of that ball. The hit detection and ball tracking algorithm was implemented in C++ using the libmpeg2 library and OpenCV [14], and it runs comfortably in realtime (although for simulation purposes we currently process the data offline).

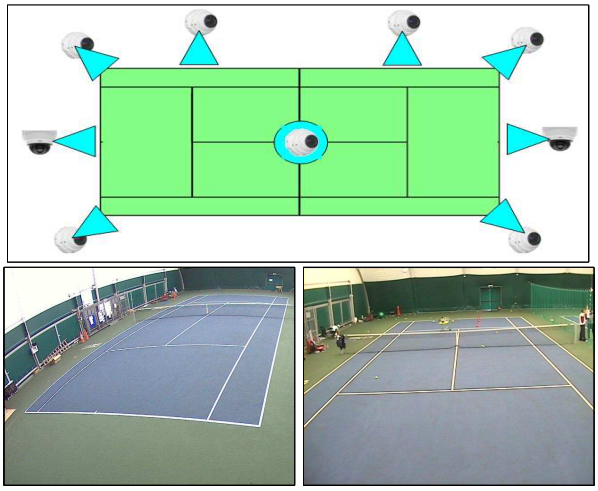


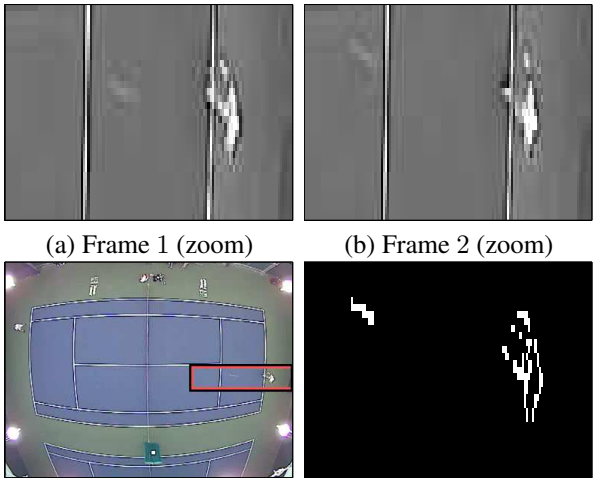
Fig. 1. A schematic layout of the nine camera surrounding our instrumented tennis court, along with sample images from two of the cameras. The central camera is positioned on the ceiling above the court.

4.1. Ball detection

Detecting a moving tennis ball is made difficult by compression artifacts, blurring of the ball due to its high speed and the camouflage effect of the white lines on the court when the ball is passing over them. Figure 2 illustrates our approach to this task. We first determine ball candidates using frame-differencing and thresholding. Instead of thresholding the absolute-difference between consecutive frames, we threshold the raw-difference, and therefore exploit the fact that the ball appears brighter than the background (which is generally the dark blue/green tennis court surface). After thresholding, we detect all connected component blobs and remove poor candidates based on their small size or their aspect ratio (ball blobs are usually blurred horizontally so that their bounding-boxes have a low height-to-width ratio). As shown in figure 2, player movement creates a number of additional false blobs. We use a heuristic method to remove most of these false blobs. Since the ball is usually far from other blobs, we remove any blob that is within a distance d_1 of 2 or more blobs. For all remaining blobs, we retain and store the position of their midpoint.

4.2. Ball tracks

We define a ball track as a series of blobs that follow a locally linear trajectory. We initialise a track if we find 3 blobs following a semi-linear path. That is, where the x coordinate of the points is descending (or ascending) in time, and the 2^{nd} point (in time) is within a distance d_2 of its predicted location based on a linear interpolation using the other 2 points. We also enforced speed constraints and only initialise the track



(a) Frame 1 (zoom) (b) Frame 2 (zoom)

(c) Camera view (d) detected blobs

Fig. 2. Blob detection: (a) and (b) show close-ups of the view highlighted in (c), the overhead view. Frame-differencing and thresholding of (a) and (b) results in image (d)

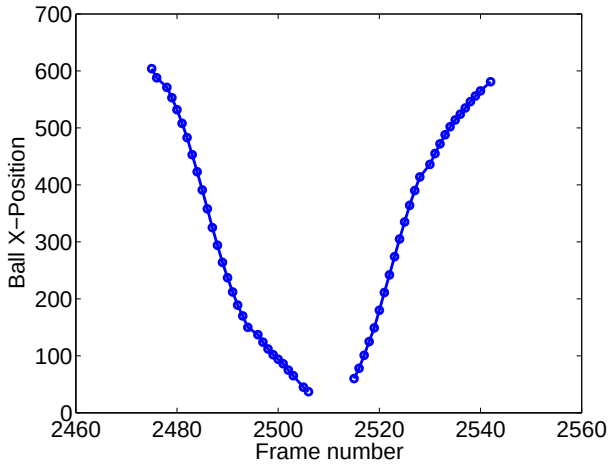


Fig. 3. A *ball track* is a series of detected blobs. Two ball tracks are shown here as a ball is hit from one side of the court and then returned.

if the speed of the blobs is within a minimum and maximum speed range.

Before initialising new tracks, all previous tracks are updated if possible. We sort tracks, longest to shortest number of blobs, and for each track we predict the location of the ball in the current frame. If there are blobs within a distance d_3 of the predicted location, the blob that is closest to the predicted location is assigned to this track. We enforce the constraint that any new blob must be consistent with the x velocity of previous blobs in the track (either ascending or descending). After all tracks are processed, any unassigned blobs are used to search for potential new tracks by an exhaustive search of the blob triplets in the previous 3 frames, as described above. If a track has not been assigned a blob for 5 frames, or if the point predicted by the track is outside the image, then this track is terminated.

4.3. Track post-processing and ball hit detection

We first remove any tracks that contain less than T blobs, as they are usually caused by noise or player movement. Occasionally, due to compression and blurring noise, the ball will be detected as two separate blobs. In those cases, a single ball trajectory can be detected as two tracks. We remove any tracks if at least 75% of their blobs are within 45 pixels of the blobs of another longer track.

Due to ball bounce, the actual location of the ball can sometimes be outside the predicted position and the correct trajectory will be split into two tracks. We merge any tracks that share any single blob. Merging is done by concatenating the list of blobs of both tracks, and taking the average position of any blobs that occur in the same frame.

Finally, we detect the exact time of each *ball hit* by computing the crossing time of the incoming and outgoing trajectories. We take all track pairs where the second track starts within half a second of the end of the first. Figure 3 shows an example of two consecutive tracks where the ball was hit at frame 2510, thereby reversing its direction. We determine the exact time of the ball-hit by fitting two lines, one to the end of the first track and one to the start of the second track. For simplicity, we use just two points to fit the lines. The crossing point on the (t, x) graph gives the time that the ball was hit. This hit is labelled as a *return* hit. All tracks that do not start with a return hit are labelled as *serves*. A serve, followed by a series of one or more return hits, is determined to be part of a *rally*.

4.4. Evaluation

To evaluate our ball-hit detection, we manually marked a ground truth of 26 minutes of video containing 243 instances of the tennis ball being hit from one player to the other. The players were the two top-seeded players in Ireland engaged in a competitive training match. Using this ground-truth and a list of detected hits, we use the maximum bipartite matching algorithm [15] to assign detected hits to ground-truth hits if they are within M frames of each other and hit from the same side of the court (left or right).

By varying the value of the minimum track length T (see previous section), the precision-recall plots shown in figure 4 were generated. On the graph, M is varied from 3 to 7 frames. Higher values of M mean a higher tolerance for small misalignment errors. Using $M = 6$ and $T = 9$, we have a precision of 0.9429 and a recall of 0.9506. False positives were mostly caused by tracks being split in two. Missed detections were caused by (i) the ball hitting the net and the track being too short to be detected, (ii) the ball being over the white line

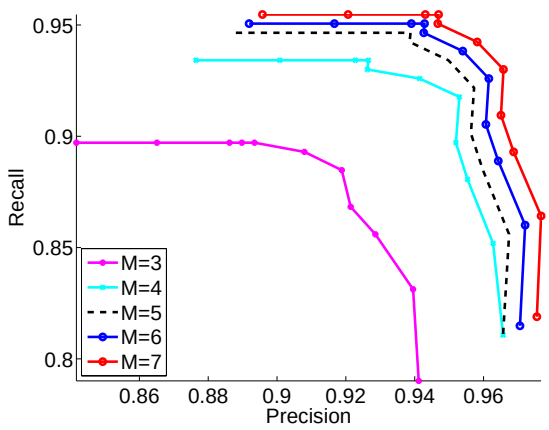
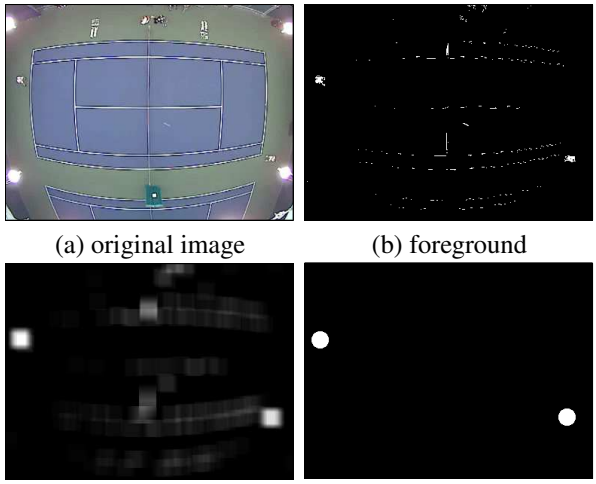


Fig. 4. Performance of ball hit detection



(a) original image (b) foreground

(c) filtered (d) detected people

Fig. 5. Player detection: Using the current camera image (a), foreground is detected (b) and filtered using a box filter, resulting in (c). The peaks of (c) are detected as players.

and therefore not being detected and (iii) the heuristic that we use (to remove the player's blobs) causing ball blobs close to a player to be removed.

5. PLAYER DETECTION AND TRACKING

5.1. Player detection

Since our camera is in a fixed overhead position, we can use background subtraction techniques to detect moving objects in the scene. Figure 5 illustrates our approach to player detection from an overhead camera view. Our background model is a layer based approach similar to the Stauffer and Grimson model [16] but using a fixed variance for each pixel layer. We also include a shadow-suppression module to reduce the effects of changes in lighting.

Since the camera is a fixed distance above the ground, we can assume that people will appear at a particular size, when viewed from above. We filter the foreground image with a box filter of size $P \times P$ ($P = 29$ in our system) and this reduces the effect of noise, highlighting likely locations of people. We use box-filtering instead of Gaussian filtering as it can be done efficiently using integral images [17]. We then employ a greedy algorithm to select the peaks of the filtered image and detect people.

We first detect all local maximum in the filtered image and discard any that are smaller than $T_1 = 128$. These are our *player candidates*. We then initialise an empty set of detected players and then process (in order of size) all candidates. If the candidate is not within a distance P pixels from a detected player, it is added to the list of detected players.

5.2. Player Tracking

To update player tracks, we process them from longest to shortest. If there is a peak in the filtered image greater than $T_1/2$ within a distance $P/2$ pixels of the last location of the player, then the track is updated with the highest peak within that range. Any track that has not been updated for 25 frames is terminated. To convert pixel coordinates to real-world coordinates (in metres), we correct the lens distortion and calibrate the cameras using the OpenCV camera calibration code [14].

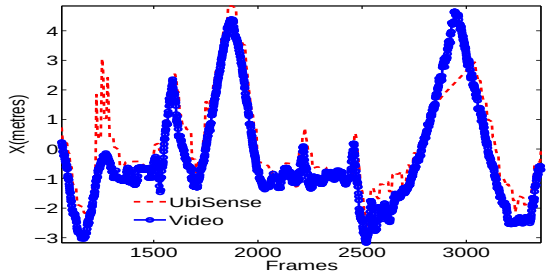
During matches, players are at opposite sides of the court, so their tracks do not get confused. However, during breaks in play, players can leave the court, or their tracks can come together and the identities of the players cannot be accurately maintained. Currently, if two players get within P pixels of each other, we split the tracks and assign a new ID to both tracks. We do this to ensure that each track (with a single ID) belongs to only one player. Player identification after an occlusion or a track merging is a non-trivial task from the overhead camera, but this task could be made easier by using other cameras and we plan to examine this in future work.

5.3. Evaluation

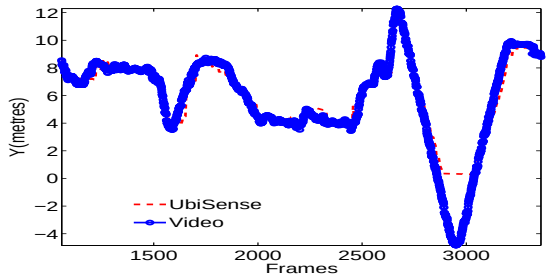
Since our installation has a UbiSense tag-tracking system [11], we used this system to create a player position ground-truth. Each player carried a tag in their pocket during the recording. The UbiSense system has a 3D localisation accuracy of $\pm 15cm$, but has a non-uniform sampling rate and can suffer from outliers or low accuracy near the border of the calibrated tennis court area. On the other hand, UbiSense does not have the same problems with player identification that video does, since each tag has a unique ID. Additionally, UbiSense is a low-bandwidth signal and does not require computationally intensive video processing.

We manually assigned the tracks we detected in video to one (or neither) of the UbiSense tags. Figure 6 shows a comparison of the two tracking systems during an 80 second period of the match. The two signals disagree at two points:

around frames 1250 and 2900. We observed the video at this point and verified that the Ubisense signal is noisy at this point and giving false readings as the player is near the border of the calibrated region. In the second case, the player leaves the court entirely and UbiSense does not give any readings for a time, then interpolates over the missing data. Results over the entire video are shown in figure 7. In both cases our video tracking gave the correct tracks. Therefore, while the UbiSense generally provides good accuracy, it was mainly a convenient way to obtain a comparison to our video tracking and we believe that the video tracking may be performing better than the median error suggests, though measured accuracy is still good.

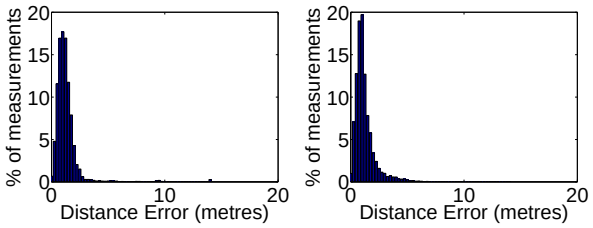


(a) X position (Player 1)



(b) Y position (Player 1)

Fig. 6. Player tracking using video versus UbiSense tracks: the two systems disagree on measurements at frame 1250 and frame 2900. From a visual inspection of the video, this is due to noisy and intermittent UbiSense readings, rather than a video tracking failure.



(a) Player 1 (b) Player 2

Fig. 7. Player tracking error histograms: The median errors were $1.10m$ (player 1) and $0.99m$ (player 2)

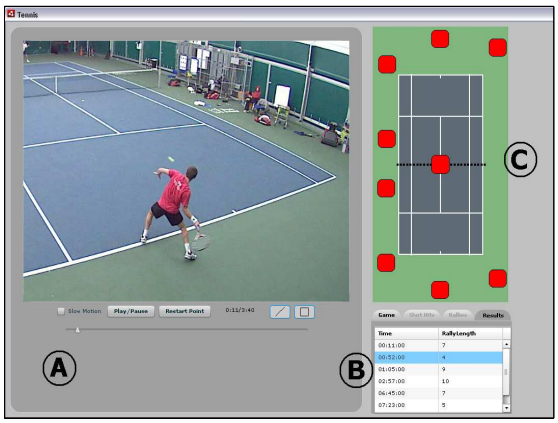
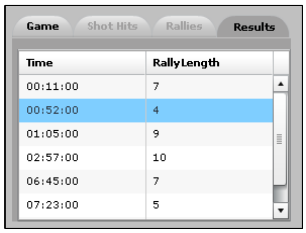
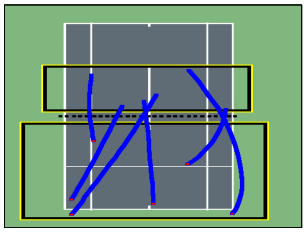


Fig. 8. Screenshot of the interactive coaching tool. Panels A, B and C are explained in the text.



(a) Section B (zoom)



(b) Section C (query mode)

Fig. 9. Interface components: (a) Rally selection list, (b) Returned shots from a query. Start-area and end-area (marked in yellow) define the shot query.

6. COACHING INTERFACE

The semantic information we extracted using the techniques described in sections 4 and 5 can be used to facilitate efficient browsing of the associated video, as well as advanced querying of the data. We now describe how this is done in our playback and querying interface.

6.1. Playback interface

Figure 8 shows the user interface for viewing and querying a selection of recorded tennis matches. It contains three main sections, labelled A, B and C. Using the *Game* tab in panel B, a game can be selected from a drop-down list. Video from the selected game can then be navigated using either the progress slider (panel A) or by selecting a rally from the results tab of panel B (see figure 9(a)). Panel C allows the user to view the game from any of the nine cameras by clicking on the red square for the appropriate camera. Video can be viewed in slow-motion also, by selecting the checkbox in panel A.

As well as providing efficient navigation of the video, using the search tabs in panel B, the user/coach can search

for particular types of information, such as: *find me all rallies lasting over* 8 *shots*. The user can also supply advanced queries by drawing on the court in panel C. Shown in figure 9(b) is an example where the user has requested all shots starting in the top bounding-box region and ending in the bottom bounding-box region.

6.2. Redundancy Removal

Often, a coach will want to watch all shots in a recorded match. In such cases, we can use the extracted information to remove redundant portions of the video, and therefore reduce the amount of time required to watch the video. Using 12 test videos (total length $=$ 5hrs 1mins 32sec) we can reduce the relevant video time by 42.46% on average using a simple approach. Using our hit detection algorithm, we simply take all frames within an N-second time window centred on the hit time of any shot. We used $N = 5$ so that in practice, most windows will be heavily overlapping since shots often occur in quick succession ($<$ 1s between shots in a rally). If we use the same approach, but only take those shots that are detected to be part of a rally, we reduce the time further, by 59.18% on average.

7. CONCLUSIONS

We have introduced the TennisSense platform that allows us to record videos of tennis matches from 9 configurable cameras and to augment this video with additional biomechanical and physiological data. Our video analysis modules were described for player and ball tracking and evaluated using ground-truthed data. The proposed algorithms demonstrate robust performance on challenging data and the resulting metadata is used in our playback-and-querying interface for providing tennis-coaches with powerful semantic browsing and querying tools to review hours of tennis footage.

We plan for future work to integrate the ball and player tracking modules to mutually improve their performance. Additionally, we are currently investigating the use of audio to detect the ball hits. As part of the broader TennisSense project, we also plan to capture the players' heart rate, respiration and physical motion [18] and to integrate these sources of data into our querying interface.

8. REFERENCES

[1] "Tennis ireland," http://www.tennisireland.ie/.

[2] "Tennissense," http://www.cdvp.dcu.ie/tennisireland/.

[3] "Protracker tennis," http://www.fieldtown.co.uk/.

[4] "Dartfish," http://www.dartfish.com/.

[5] "Prozone," http://www.prozonesports.com/.

[6] D. Sadlier and N. O'Connor., "Event detection in field sports video using audio-visual features and a support vector machine," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 10, pp. 1225–1233, 2005.

[7] H. Miyamori and S.I. Iisaku, "Video annotation for content-based retrieval using human behavior analysis and domain knowledge," in *IEEE International Conference on Automatic Face and Gesture Recognition*, 2000, pp. 320–325.

[8] F. Yan, W. Christmas, and J. Kittler, "A tennis ball tracking algorithm for automatic annotation of tennis match," in *British Machine Vision Conference*, 2005, pp. 619–628.

[9] G.S. Pingali, Y. Jean, and I. Carlbom, "Real time tracking for enhanced tennis broadcasts," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1998, pp. 260–265.

[10] G. Pingali, A. Opalach, and Y. Jean, "Ball tracking and virtual replays for innovative tennis broadcasts," in *International Conference on Pattern Recognition*, 2000, vol. 4, pp. 152–156.

[11] "Ubisense," http://www.ubisense.net/, 2008.

[12] R. Slyper and J. Hodgins, "Action capture with accelerometers," in *ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, July 2008.

[13] K.-T. Lau D. Kim S. Brady G. Wallace S. Coyle, Y. Wu and D. Diamond, "Design of a wearable sensing platform for sweat analysis," in *Personalised healthcare (pHealth)*, 2007.

[14] "Opencv," http://sourceforge.net/projects/opencvlibrary.

[15] Douglas Brent West, *Introduction to Graph Theory (2nd ed.)*, Prentice Hall, 1999.

[16] C. Stauffer and W.E.L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proceedings of CVPR99*, 1999, pp. II:246–252.

[17] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Dec 2001, vol. 1, pp. 511–518.

[18] G. Torre, M. Fernström, B. O'Flynn, and P. Angove, "Celeritas: wearable wireless system," in *NIME '07: Proceedings of the 7th international conference on New interfaces for musical expression*, New York, NY, USA, 2007, pp. 205–208, ACM.