A Novel Decentralised System Architecture for Multi-Camera Target Tracking

Gaetano Di Caterina[1], Trushali Doshi[1], John J. Soraghan[1], and
Lykourgos Petropoulakis[1]

[1]Department of Electronic and Electrical Engineering, University of Strathclyde,
204 George Street, Glasgow, G1 1XW, UK

Abstract. Target tracking in a multi-camera system is an active and challenging research that in many systems requires video synchronisation and knowledge of the camera set-up and layout. In this paper a highly flexible, modular and decentralised system architecture is presented for multi-camera target tracking with relaxed synchronisation constraints among camera views. Moreover, the system does not rely on positional information to handle camera hand-off events. As a practical application, the system itself can, at any time, automatically select the best target view available, to implicitly solve occlusion. Further, to validate the proposed architecture, an extension to a multi-camera environment of the colour-based IMS-SWAD tracker is used. The experimental results show that the tracker can successfully track a chosen target in multiple views, in both indoor and outdoor environments, with non-overlapping and overlapping camera views.

Keywords: video analytics, multi-camera, decentralised, tracking.

1 Introduction

The need for automatic analysis of video data on computer systems has led to the development of image and video processing techniques usually referred to as Video Analytics (VA), to extract relevant information from surveillance camera feeds. Central to many smart surveillance systems is the detection and identification of a target object in consecutive frames, i.e. target tracking [1].

In the context of multi-camera systems, information extracted from a set of semantically clustered cameras can be fused together and exploited, to achieve a better understanding of the environment surrounding the cameras [2] and monitor areas wider than a single camera field of view (FOV). Each sensor can be associated with VA processing tasks [3,4], to distribute the surveillance workload among cameras and decentralise it towards the edges of the network. This approach produces a collaborative, or co-operative, network of smart surveillance sensors [5, 6]. To fully exploit the information gathered by a sensor network, both topological and geographical layouts of the network can be used. While the former defines which cameras have overlapping FOVs, the latter specifies the position in space of each sensor, with respect to a common coordinate frame.

From the network topology only, it is possible, at any time, to know which other cameras should be seeing a specific target in their FOV. If overlapping cameras can simultaneously detect the same target, information from multiple FOVs can be merged to obtain a better understanding of the target. Such a smart surveillance system can automatically select, from a set of views, the one that gives the best visualisation of the target, and camera overlapping can be exploited to overcome target occlusion [7]. Many algorithms have been proposed in literature for multi-camera system calibration [8, 9].

In this paper a complete, highly flexible, modular system architecture is proposed for decentralised multi-view target tracking, where synchronisation constraints among processes can be relaxed. Unlike the approaches in [10, 11], the presented novel architecture does not rely on geographical layout, to initialise the trackers or handle camera handoff events. Tracking in separate camera views is performed solely on the visible characteristics of the target, reducing the system setup phase to a minimum. For this purpose, only the knowledge of the topological layout of the network is required at initialisation step. This system architecture is complete, because it also includes initial target detection upon occurrence of events of interests; it is also modular because the specific algorithms implemented in its components can be changed and new features can easily be added to the system, without affecting how the proposed architecture works.

In the context of multi-view systems, colour as a discriminant feature is ideal for target tracking, as it requires minimal computation and it is very resilient against geometric transformations. To validate the effectiveness of the proposed architecture, a modified version of our colour-based IMS-SWAD tracker described in [12] is used as a tracking algorithm in each camera view. In practice any tracking algorithm could be adopted. However, the IMS-SWAD tracker has been chosen because it is easy to appreciate how its operation in each view does not rely on any positional information extracted from other views. Indeed, algorithmic parameters are automatically set in separate views, with respect to the colour characteristics of the target only.

The remainder of the paper is organised as follows. Section 2 describes in detail the proposed architecture and its components. Section 3 describes the improved IMS-SWAD tracker and its incorporation into the architecture. Experimental results are reported in section 4, while section 5 concludes the paper.

2 Decentralised Multi-Camera Tracking System

From a conceptual point of view, multi-camera target tracking system can be divided into four tasks:

1. initial target detection upon occurrence of an event of interest;
2. target status storage and broadcasting;
3. target tracking in each separate camera view;
4. data collection from all the trackers and collation.

Such tasks can be mapped into separate processes, which share information with each other for target tracking across multiple views. Therefore four types of processing entities are defined, (i) a detection agent (DA) symbolised as D, (ii) a tracking agent (TA) symbolised as T, (iii) a status server (SS) and (iv) a data server (DS). Fig.1a shows the layout of an implementation of the system.

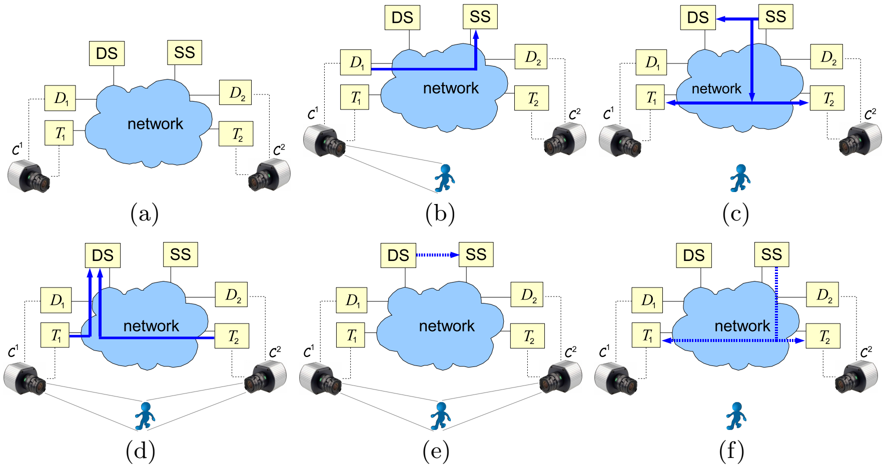


Fig. 1. System implementation with two cameras $\mathcal{C}^1$ and $\mathcal{C}^2$ (a). The DA D_1 detects a target and sends data to the SS (b). The SS broadcasts target information to both TAs T_1 and T_2 (c). The TAs track the target and send their results to the DS (d). The DS merges the TA results and sends the new target representation to the SS (e). The SS broadcasts the new target representation to all the TAs (f).

A separate TA and DA must be associated with each camera in the system, to track a target in a single camera view and to be able to detect events of interest in each camera view respectively. Conversely, it is reasonable to have a single SS acting as a central hub that receives detections from single DAs and broadcasts these to all the TAs. Concerning data collection, a single DS can act as a sink for the tracking information produced by all the TAs.

In general, for a multi-camera tracking system with N cameras, tracking agent T_n and detection agent D_n are associated with the n^{th} camera $\mathcal{C}^n$ with $n \in [1, N]$, and $\mathbf{F}_i^n$ is the i^{th} frame in the n^{th} camera view. For all the agents, a single SS and DS are available. As shown in Fig.1b, target information is first sent by a DA (D_1) to the SS, upon occurrence of an event of interest. The SS broadcasts this information to T_1 and T_2 (Fig.1c). The TAs send their tracking results to the DS (Fig.1d), which merges them in an attempt to resolve possible inconsistencies and produce a unique multi-camera track of the target. Also, the DS can produce a better representation of the target using information from different TAs; that can then be transmitted from the DS to the SS (Fig.1e), which can then broadcast it again to all the TAs (Fig.1f). This cycle allows for a refinement of the target model, by exploiting characteristics of the target in the different views available in the system.

All agents and servers are loosely coupled, as they share their information through simple messages over the network. The system architecture is therefore highly flexible and multiple configurations are possible to deploy the processing entities on physical processing units. For example: a) all entities can run on the same machine; b) SS and DS can run on the same machine, while each set of TA and DA associated with the same camera runs on a different machine; c) each entity runs on a dedicated machine.

2.1 DA – Target detection

Any DA in the system can select a target within its FOV, upon occurrence of a predefined event of interest. The detection algorithm implemented in the DA is application-dependent and one could either manually select the target, or apply the automatic event detection algorithms [13, 14]. In the current system, our adaptive algorithm for the detection of abandoned and removed objects presented in [15] has been implemented in the DAs. However, differently from [15], the algorithm in the DA D_n sends to the SS only the camera number n and the portion of frame $\mathbf{F}_i^n$ corresponding to the selected target (Fig. 2). The reason for this transmission is to further decouple the DAs from the TAs: the TAs can extract from the "image" of the target the features required for their tracking algorithm, without the DAs needing to know what these features and algorithms are. Moreover, separate TAs can implement different tracking algorithms, so they may need to extract different features, such as colour, texture, shape, and so on.

2.2 SS – Target information storage

The SS receives data from a DA, i.e. the portion of frame $\mathbf{F}_i^n$ representing the detected object (Fig. 2) and stores it along with a timestamp, the camera number n and a unique identification number ξ for the target.

The SS acts as a sink for all the DA detections, while it follows a publisher-subscriber pattern with respect to the TAs. More specifically, the SS is always running and the TAs subscribe to it at their setup time. When the SS receives target data from a DA, it broadcasts such information to all its subscribers, i.e. TAs, and also to the DS, so that the DS knows what target, with identification number ξ, is under tracking within the system. The TAs can then look for the new target in their FOVs, and track it if present. The role of the SS is to store and forward data between DAs and TAs, and between DAs and DS.

2.3 TA – Single view target tracking

The TAs perform target tracking in single views, independently of each other. After receiving target information from the SS, a TA is initialised in the current frame when the data from the SS is received. Then the TA performs a loop in which a single step of the selected tracking algorithm is executed on the next

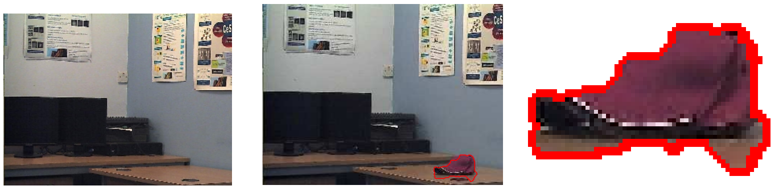

(a) (b) (c)

Fig. 2. DA target detection: (a) background image; (b) new frame $\mathbf{F}_i^n$ with the selected target in it; (c) close-up of the portion of $\mathbf{F}_i^n$ sent to the SS.

frame. If the target has been found, a message with relevant information is sent to the DS. Otherwise, the TA proceeds directly to the next frame. In the rest of the paper, the case where the target has been found by the tracker in the current frame is referred to as "tracking hit".

Any algorithm can be adopted in the TAs. However, the chosen algorithm must be able to extract discriminative features of the target from the data received from the SS, in order to overcome the problem of different target appearance across multiple FOVs in a multi-camera system. In the current implementation of the system, a modified version of the colour-based IMS-SWAD tracker introduced in [12] is adopted, as described in more detail in section 3.

2.4 DS – Data collation and multi-view tracking

The DS collects tracking results from all the TAs and collates them, to remove inconsistencies and create a unique coherent multi-view track of the target.

In a multi-camera tracking system it is possible to synchronise multiple machines using the Network Time Protocol (NTP), to be able to reliably collate target tracks from different views. However, it is difficult to ensure temporal synchronisation at a frame level among multiple trackers. Therefore the presented system adopts a notion of temporal synchronisation in terms of tracking hits falling in the same time interval defined by the DS. This means that the DS defines a time line of consecutive time slots t_s^{DS} of given temporal length Δt^{DS}. The TAs run at their own specific pace and regularly send their tracking results to the DS. Tracking hits flagged by different TAs and received by the DS within the same time slot t_s^{DS} are considered to be synchronous.

Similarly, each TA defines a time line of consecutive time slots t_s^{TA} of temporal length Δt^{TA}, and tracking hits in separate frames falling within the same time slot t_s^{TA} are accounted for as a unique tracking hit. At the end of each time slot t_s^{TA}, the TA sends its tracking results to the DS if a target was found in such a time interval; otherwise no transmission takes place. The information sent by the n^{th} TA includes the camera number n, the target number ξ, a timestamp and the value of the highest match λ_n^{ξ}, for all the tracking hits in time slot t_s^{TA}. The reason behind defining also a time slot t_s^{TA} for the TAs is to avoid the need to transmit possible tracking hits for every frame processed by a TA.

Having defined time slots for both DS and TAs, timing constraints for the overall system are relaxed and Δt^{DS} and Δt^{TA} can be set to accommodate the application requirements. As a guideline, assuming the worst case where the DS processes TA notifications sequentially, the lengths of Δt^{DS} and Δt^{TA} can be set according to:

$$\begin{aligned} \Delta t^{DS} &> N\Delta t^{TX} + \Delta t^{TA} \\ \Delta t^{TA} &> \Delta t^{TX} \\ \Delta t^{TA} &\geq \tfrac{1}{fps^{TA}} \end{aligned} \tag{1}$$

where N is the number of TAs sending information to the DS, Δt^{TX} is the transmission time from TA to DS and fps^{TA} is the TA processing rate. The "frame rate" of the DS is $1/\Delta t^{DS}$.

At the end of each time slot t_s^{DS}, the DS knows which TAs have found the target ξ in their FOV and which have not. Moreover it can use the match value λ_n^{ξ} of each tracking hit as a level of confidence for it. When any two TAs, T_n and T_m, send their tracking hits to the DS for the same target ξ, in the same time slot t_s^{DS}, the DS selects as best view for the time slot Δt^{DS} the view associated with the tracker that returned the highest match value; therefore the DS can generate a single continuous video stream made up of the portions of video feeds coming from the selected best views, at each Δt^{DS}. Such a strategy is also useful in overlapping cameras, to be able to select the best view of the target in case of occlusion.

Moreover, as the DS gathers all the tracking hits and collates them, it can be argued that it indirectly performs target tracking across multiple camera views as oppose to a single view in the TAs. Also, the DS can use the gathered data to update the current model of target ξ; the updated model can then be forwarded to the SS, for a new broadcast to all the TAs.

3 Multi-view IMS-SWAD tracking algorithm

The effectiveness of the proposed architecture for multi-camera target tracking is validated using the colour-based IMS-SWAD tracker described in [12]. This colour-based tracker is selected to highlight the fact that the proposed architecture does not require any position information of the target or of the cameras. The only required information is whether or not cameras have overlapping FOV. The only modification required in our IMS-SWAD [12] to be integrated within a multi-camera environment is in the initialisation step. After that, the algorithm running in each TA processes its own camera view independently from the others and therefore it is ideal for a parallel implementation in the context of a decentralised tracking system, as the one described in this paper.

For multi-view colour tracking, colour calibration among all the cameras is required. Therefore the Gray World Assumption [16] is used to colour-normalise all camera views. Such a normalisation is applied to both TAs and DAs. Further, to make the algorithm more resilient to different lighting conditions in separate views, the YCbCr colour space is adopted, but the luminance component Y is removed and only the red and blue chrominance Cb and Cr are used to compute colour distributions.

Knowledge of the camera topology is required for a correct setup of the instance of the algorithm in each view. For this purpose the tracking algorithm only needs to know which cameras have overlapping FOVs. Therefore, the required information can be easily encoded with a look-up table (LUT) stating whether two camera views overlap or not. Such an LUT can be provided to each tracker instance, i.e. TA, at setup time. In the context of the decentralised architecture (section 2), for a target ξ selected in camera $\mathcal{C}^n$, the initialisation of the tracker T_n is same as the one described in [12]. For the other $N-1$ trackers T_m with $m \neq n$, their initialisation depends on whether the FOVs of cameras $\mathcal{C}^n$ and $\mathcal{C}^m$ overlap or not.

3.1 Non-overlapping camera views

If cameras $\mathcal{C}^n$ and $\mathcal{C}^m$ do not overlap, target ξ selected at a given time instant in $\mathcal{C}^n$ is certainly not present in $\mathcal{C}^m$. Clearly the position in $\mathcal{C}^m$ of the best match for the given target model $\mathbf{Q}$ refers to an object which is not the target. Therefore the threshold τ_m for tracker T_m is computed as:

$$\tau_m = \arg\min_{\mathbf{o}_l^m} \left[d(\mathbf{o}_l^m) \right] \tag{2}$$

where $\mathbf{o}_l^m$ are the candidate points selected as in the tracker initialisation step in [12], but with no spatial constraints on their position in the initial frame $\mathbf{F}_0^m$.

After initialisation, tracker T_m proceeds to the failure recovery step, as by definition target ξ was not present in $\mathbf{F}_0^m$, so the initial target position in $\mathbf{F}_1^m$ is undefined. Here it is assumed that, when the target ξ enters the FOV of camera $\mathcal{C}^m$, its colour distribution will have a distance $d(\cdot)$ from the target model $\mathbf{Q}$ smaller than any other object in the frame $\mathbf{F}_0^m$ and therefore smaller than τ_m. So the tracker T_m can successfully start to track the correct target ξ in $\mathcal{C}^m$.

3.2 Overlapping camera views

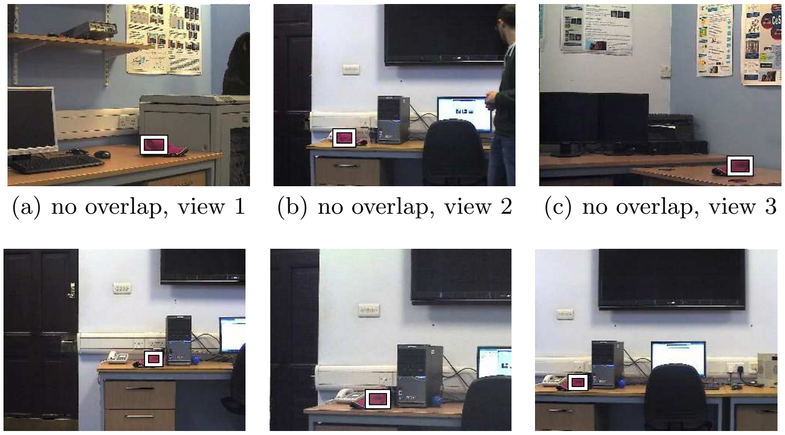
If cameras $\mathcal{C}^n$ and $\mathcal{C}^m$ have overlapping FOVs, it means that target ξ should be present in both $\mathbf{F}_0^m$ and $\mathbf{F}_0^n$. The failure recovery procedure applied to $\mathbf{F}_0^m$ gives the position of the best match for $\mathbf{Q}$ in the frame. This best match should be the correct target ξ, assuming that its distribution minimises the distance $d(\cdot)$ from $\mathbf{Q}$. So tracker T_m finds the correct target ξ in $\mathbf{F}_0^m$ and then proceeds to the next frame $\mathbf{F}_1^m$.

If the target ξ is hidden or not visible in $\mathbf{F}_0^m$, the best match found in the frame by the failure recovery procedure will refer to an incorrect target ξ'. This incorrect tracking hit is automatically corrected by the tracker as soon as target ξ is visible again in camera $\mathcal{C}^m$, as ξ gives a higher match than ξ' and therefore ξ is selected as target to track. However, an initial value for the threshold τ_m is computed anyway and it is updated by the tracker T_m in the next frames of $\mathcal{C}^m$.

This approach allows an initial value of the threshold for each camera view to be defined. Such threshold values are soon tuned by the trackers, so potential initial incorrect hits are confined to a small number of frames. Moreover possible tracking inconsistencies across different views can be resolved by the DS at data collation time as explained in section 2.4.

4 Experimental results

The purpose of the following experiments is to visually and numerically evaluate the proposed architecture and the IMS-SWAD tracker implemented in the TAs. In the current system software, the tracking block of the TA components is implemented in Matlab and the communication blocks, the SS and the DS are implemented in Java. The computer used to run the experiments is an Intel Core 2 Quad CPU at 3 GHz, with 3 GB of RAM. The cameras used to record the indoor sequences are three 1.3 megapixel IP cameras from Arecont Vision.



(a) no overlap, view 1 (b) no overlap, view 2 (c) no overlap, view 3

(d) overlap, view 1 (e) overlap, view 2 (f) overlap, view 3

Fig. 3. Indoor testing: non-overlapping (top row) and overlapping (bottom row) views.

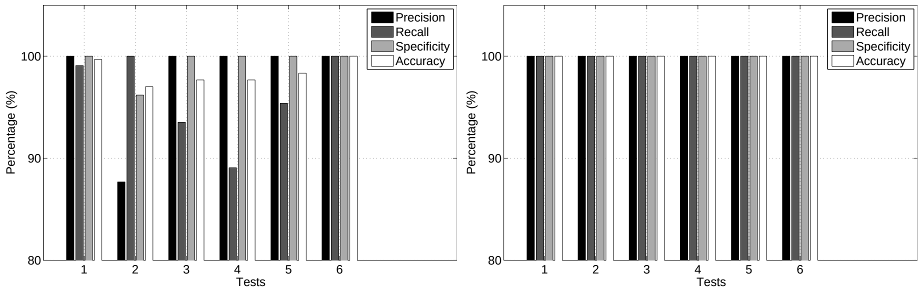
4.1 Tracking performance

The tracking performance of the system is numerically evaluated in terms of precision, recall, specificity and accuracy [1]. In all the experiments in this paper, the ground truth is manually annotated by putting a bounding box around the main colour of the chosen target.

As a first experiment, two multi-camera indoor video sequences have been recorded, respectively one with three non-overlapping views and one with three overlapping views (Fig.3). In this experiment, the target (a pink cap), which is manually moved from view to view, has been automatically detected by the algorithm implemented in the DAs, in one of the views and then tracked by the TAs in the other two views (Fig.3). This is repeated for all views to obtain a total of six tests (1-6) for both cases. After target detection in the DA, the portion of frame corresponding to the selected target is sent to the SS (section 2.1). As it is difficult to make sure that pre-recorded sequences are re-run synchronously during testing, the TA activity is simulated by launching one TA per sequence at the time. Each TA initially subscribes with the already running SS, which in turns sends to the TA the target information previously stored. The TA computes the target model needed and tries to track the target in its corresponding video sequence. In this way the target is then tracked in all six tests.

Numerical values of such tests compared with the manually labelled ground truth in Fig.4 show that the system has high precision, recall, specificity and accuracy, i.e. more than 85% in all 12 tests. In the overlapping case in particular, neither False Positives nor False Negatives have been detected (Fig.4b). These results indicate that the system and in particular its current implementation, has good tracking performance in an indoor environment.

As a second experiment, six entire video sequences (1-6) from the PETS2009 [17] dataset have been used. Since the focus of this experiment is not on the DA, the target (the woman in the red jacket (Fig. 5)), is manually selected in frame



(a) (b)

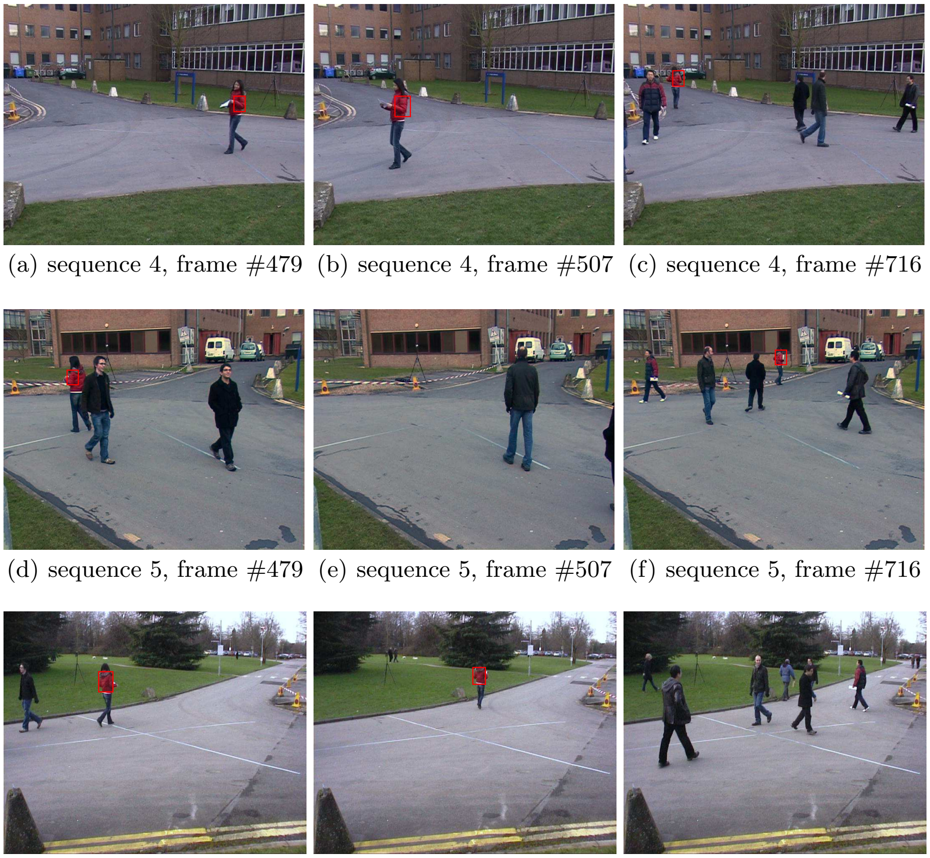
Fig. 4. Results for indoor testing with (a) non-overlapping views (b) overlapping views.

#72 of sequence 2. The target image is then sent to the SS and from there to the TA for each of the six PETS2009 sequences. The target is therefore tracked in all sequences as it moves across camera views. Of these video sequences, the sequences 1, 2 and 3 and sequences 4, 5 and 6 are synchronised with each other, which means that the n^{th} frame in each sequence refers to the same instant in time and it has a counterpart in the other two sequences (Fig. 5). It can be observed from Fig. 5 how the target may not be visible at times in some of the views (e.g. Fig. 5e), while it can still be tracked at the same time instant, i.e. same frame number, in other views (e.g. Fig. 5b and 5h). This aspect is fundamental for selecting the best view of a target, at any time.

Further, from numerical values in Fig. 6, it can be seen that the system has high precision and specificity also in an outdoor setup, with only 4 False Positives erroneously detected in all six sequences. In general, recall and accuracy are also high ($> 90\%$), apart from sequence 6, where the lower value of recall (83%) is due to the fact that in this sequence the target walks away from the camera and its size decreases to less than half of the original. In this case, the histogram of the target changes significantly, resulting in a poor match and therefore a high number of False Negatives. This second experiment indicates the effectiveness of the proposed architecture, in terms of communication between SS and TAs, also in an uncontrolled environment, such as outdoor sequences.

4.2 Best view selection and occlusion handling

In the proposed system, the DS receives information on tracking hits, i.e. camera number n, target number ξ and match value λ_n^ξ, from each TA which has found the target in its FOV, and from this data it can select the best view of the target. Therefore, it can produce a continuous video stream from multiple views, when the selected target appears in the different FOVs of non-overlapping cameras (Fig. 7). The current system implementation uses three IP cameras and for testing purposes only, the DS shows the best view of the target, at the top of its video output, and also the current feeds from the three cameras, at the bottom (Fig. 7). In the three sets of camera feed subimages in Fig. 7, it can be seen that the target (person in yellow) is sequentially detected in camera 1 (Fig. 7a), camera 3 (Fig. 7b) and camera 2 (Fig. 7c).



(a) sequence 4, frame #479 (b) sequence 4, frame #507 (c) sequence 4, frame #716

(d) sequence 5, frame #479 (e) sequence 5, frame #507 (f) sequence 5, frame #716

(g) sequence 6, frame #479 (h) sequence 6, frame #507 (i) sequence 6, frame #716

Fig. 5. Synchronised images from PETS2009 sequences 4 (S2-L1-12.34-05), 5 (S2-L1-12.34-06) and 6 (S2-L1-12.34-07) with correct target highlighted by tracking algorithm.

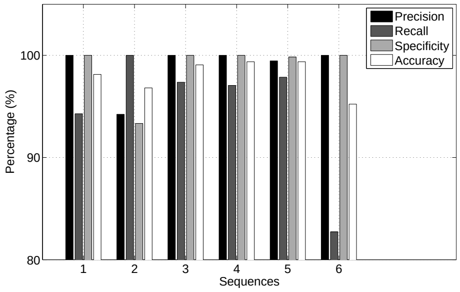
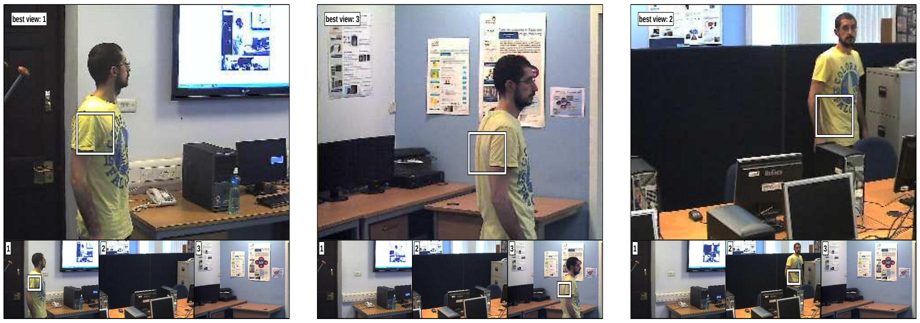


Fig. 6. Precision, recall, specificity and accuracy for six PETS2009 sequences.

Moreover, the DS can use the tracking results from the TAs associated with overlapping cameras, to select at any time the view with the highest TA match λ_n^ξ. Therefore the DS is able to easily solve the occlusion problem in overlapping



(a) (b) (c)

Fig. 7. Three images from the DS: the system is able to switch between camera views wherein the target is detected, to create a continuous video stream of the target.

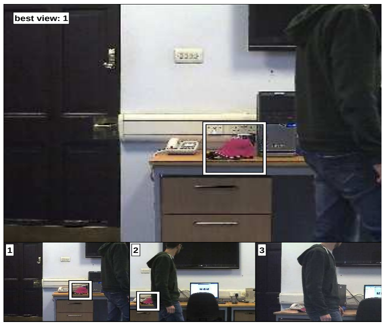


Fig. 8. DS with best view selection: the target is occluded in view 3; partially occluded in view 2; completely visible in view 1.

cameras, when the target is occluded in one of the views. As illustrated in Fig. 8, the target is correctly tracked in views 1 ($\lambda_1^\xi = 76.4\%$) and 2 ($\lambda_2^\xi = 44.5\%$). As view 1 gives higher λ_n^ξ value, it is selected as best view among the three available ones, and occlusion is implicitly solved.

5 Conclusion

In this paper a novel decentralised multi-camera tracking system architecture has been presented. The proposed architecture is highly flexible and synchronisation constraints on software and hardware can be relaxed. No positional information is used to localise the target in multiple views, while only the knowledge of occurrence of overlapping between views is required to initialise the trackers. The colour-based IMS-SWAD tracking algorithm adopted in the current implementation demonstrate the effectiveness of the overall system in both indoor and outdoor environments, in tracking a target over multiple overlapping and non-overlapping views. It is expected that the presence of multiple objects with colour similar to the chosen target may lead to wrong detection by the DA, and therefore possible tracking loss. However, the IMS-SWAD itself can be expanded to include other features, such as texture, which should help differentiate between correct target and such other objects with similar colour. Nontheless, the proposed multi-camera architecture would not require any modification. The system is ideal in situations where video synchronisation and detailed knowledge of cameras' setup is not available.

References

1. A. Smeulders, D. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah.: Visual Tracking: An Experimental Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence **36** (7) (2014) 1442-1468.
2. X. Wang.: Intelligent multi-camera video surveillance: A review. Pattern Recognition Letters **34** (1) (2013) 3-19.
3. R. T. Collins, A. J. Lipton, H. Fujiyoshi, and T. Kanade.: Algorithms for cooperative multisensor surveillance. Proceedings of the IEEE **89** (10) (2001) 1456-1477.
4. R. Tron and R. Vidal.: Distributed computer vision algorithms. IEEE Signal Processing Magazine **28** (3) (2011) 32-45.
5. P. Remagnino, A. I. Shihab, and G. A. Jones.: Distributed intelligence for multi-camera visual surveillance. Pattern Recognition, Elsevier **37** (4) (2004) 675-689.
6. M. Taj and A. Cavallaro.: Distributed and decentralized multicamera tracking. IEEE Signal Processing Magazine **28** (3) (2011) 46-58.
7. A. Ercan, A. Gamal, and L. Guibas.: Object Tracking in the Presence of Occlusions Using Multiple Cameras: A Sensor Network Approach. ACM Trans. Sen. Netw. **9** (2) (2013) 16:1-16:36.
8. O. Javed, K. Shafique, Z. Rasheed, and M. Shah.: Modeling intercamera spacetime and appearance relationships for tracking across nonoverlapping views. Computer Vision and Image Understanding **109** (2) (2008) 146-162.
9. E. Lobaton, R. Vasudevan, R. Bajcsy, and S. Sastry. A distributed topological camera network representation for tracking applications. IEEE Transactions on Image processing **19** (10) (2010) 2516-2529.
10. F. Porikli and A. Divakaran.: Multi-camera calibration, object tracking and query generation. in International Conference on Multimedia and Expo (2003) I-653-656.
11. B. Moller, T. Plotz, and G. A. Fink.: Calibration-free camera handover for fast and reliable person tracking in multi-camera setups. in International Conference on Pattern Recognition (2008) 1-4.
12. G. Di Caterina and J. Soraghan.: A robust, precise and flexible tracking algorithm based on IMS and SWAD. in IEEE International Conference on Acoustics, Speech and Signal Processing (2014) 6588-6592.
13. A. Adam, E. Rivlin, I. Shimshoni, and D. Reinitz.: Robust real-time unusual event detection using multiple fixed-location monitors. IEEE Transactions on Pattern Analysis and Machine Intelligence **30** (3) (2008) 555-560.
14. T. DOrazio and C. Guaragnella: A Survey of Automatic Event Detection in Multi-Camera Third Generation Surveillance Systems. Int. J. Patt. Recogn. Artif. Intell. **29** (1) (2015) 1555001–1–29.
15. G. Di Caterina and J. J. Soraghan.: An abandoned and removed object detection algorithm in a reactive smart surveillance system. in International Conference on Digital Signal Processing (2011) 1-6.
16. G. Buchsbaum. A spatial processor model for object colour perception. Journal of The Franklin Institute **310** (1) (1980) 1-26.
17. PETS2009.: Benchmark data. http://www.cvg.reading.ac.uk/PETS2009/a.html accessed: 14/05/2016.