A comparative study on multi-person tracking using overlapping cameras

M. C. Liem and D. M. Gavrila

Intelligent Systems Laboratory, University of Amsterdam, The Netherlands
{m.c.liem,d.m.gavrila}@uva.nl

Abstract. We present a comparative study for tracking multiple persons using cameras with overlapping views. The evaluated methods consist of two batch mode trackers (Berclaz et al, 2011, Ben-Shitrit et al, 2011) and one recursive tracker (Liem and Gavrila, 2011), which integrate appearance cues and temporal information differently. We also added our own improved version of the recursive tracker. Furthermore, we investigate the effect of the type of background estimation (static vs. adaptive) on tracking performance. Experiments are performed on two novel and challenging multi-person surveillance data sets (indoor, outdoor), made public to facilitate benchmarking. We show that our adaptation of the recursive method outperforms the other stand-alone trackers.

1 Introduction

Tracking multiple persons in dynamic, uncontrolled environments using cameras with overlapping views has important applications in areas such as surveillance, sports and behavioral sciences. We are interested in scenes covered by as few as 3-4 surrounding cameras with diagonal viewing directions, maximizing overlap area. This set-up makes establishing individual feature correspondences across camera views difficult, while inter-person occlusion can be considerable.

Various methods have been proposed recently for such a multi-person tracking setting using overlapping cameras, but few quantitative comparisons have been made. In order to improve visibility regarding performance characteristics, we present an experimental comparison among representative state-of-the-art methods.[1] We selected one recursive method [15] and two batch methods [2, 1] for this comparison. Furthermore, we made some performance improving adaptations to [15]. The trackers were combined with the static background estimation method from [18] and the adaptive background estimation method from [24].

2 Related Work

In recent years, various methods performing multi-person detection and tracking using overlapping cameras have been presented. In [16], person positions

[1] This research has received funding from the EC's Seventh Framework Programme under grant agreement number 218197, the ADABTS project.

are found by matching colors along epipolar lines in all cameras. Foreground images are projected onto horizontal planes in the 3D space in [13], detecting objects at ground plane locations where multiple foreground regions intersect in multiple planes. Similarly, [20] uses images containing the number of foreground pixels above each pixel to create 3D detections at positions with the highest accumulated score. In [5], people's principal axis are matched across cameras. In [8], a Probabilistic Occupancy Map (POM) is presented for person detection. A generative model using a discretized ground plane and fixed size regions of interest approximates the marginal probability of occupancy by accumulating all evidence received from foreground images from every camera. Detections in [15] are generated using a volume carving [22] based 3D scene reconstruction, projected onto the ground plane. Similar to [15], [11] proposes a model in which multiple volume carving based scene configuration hypothesis are evaluated. Instead of solving hypothesis selection in 3D, the graph cut algorithm is used to label the pixels of each camera image as background or one of the people in the scene. In [10], an iterative model is presented labeling individual voxels of a volume reconstruction as either part of an object, background or static occluder.

Combining detections into long-term tracks can be approached in several ways. *Recursive* trackers perform on-line tracking on a frame-to-frame basis, often using well known algorithms like Mean-Shift [6], Kalman filtering [11] or particle filtering [4]. When tracking multiple objects simultaneously, the issue of consistently assigning tracks to detections should be solved. Well known solutions are the Joint Probabilistic Data Association Filter [9, 12] and Multiple Hypothesis Tracking [19]. In [15], matching tracks to detections is approached as an assignment problem in a bipartite graph, but evaluation is focused on track assignment instead of tracking performance like this paper. Particle filters have also been extended for multi-target tracking [14, 7].

Batch mode trackers optimize assignment of detections to tracks over a set of multiple frames at once. Tracking is often modeled as a linear or integer programming problem or as an equivalent graph traversal problem. Flow optimization is used for tracking in [23], finding disjoint paths in a cost flow network defined using observation likelihoods and transition probabilities. In [2], flow optimization is combined with POM. This is extended with an appearance model in [1].

3 Methods

This section gives an overview of the tracking methods and background models to be compared. We will refer to the method from [15] as Recursive Combinatorial Track Assignment (RCTA), while the batch methods from [2] and [1] will be referred to as K-shortest paths (KSP) and K-shortest paths with appearance (KSP-App), respectively. Our adaptation of RCTA will be referred to as RCTA$^+$.

3.1 Recursive Combinatorial Track Assignment

RCTA [15] uses volume carving to create a 3D reconstruction of the scene. This reconstruction is projected vertically onto the ground plane and segmented using

EM clustering such that N person location hypotheses $\mathcal{P}^n$ (i.e. detections) with sufficient size and vertical mass for a hypothetical person remain. Using the M tracks $\mathcal{T}^m$ from the previous frame, an assignment hypothesis $\mathcal{A}^i$ consists of the assignment of ν person hypotheses to a track, with $0 \leq \nu \leq min(N, M)$. $\mathcal{A}^i$'s likelihood uses the positions of $\{\mathcal{P}^n, \mathcal{T}^m\} \in \mathcal{A}^i$, appearances (separately computed for head, torso, legs) of $\mathcal{P}^n \in \mathcal{A}^i$ and foreground segmentation. Selection and tracking of actual persons is solved jointly by finding the most likely assignment hypothesis $\mathcal{A}^*$. Occlusions between different $\mathcal{P}^n$ make features like the appearance of $\mathcal{P}^n$ depend on all other $\mathcal{P}^n \in \mathcal{A}^i$, making finding $\mathcal{A}^*$ intractable. A two-step approach solves this problem.

In the *preselection* step, Munkres' algorithm [17] computes the top K candidates for $\mathcal{A}^*$ using an approximation of the likelihood based on a subset of features independent of occlusion. In the *verification* step, $\mathcal{A}^*$ is selected by evaluating all K candidates using all features. More details can be found in [15].

We propose a number of improvements to the algorithm. The most important changes are related to way the foreground observation likelihood (the likelihood of the segmented foreground given a person hypothesis) is computed. This likelihood is based on the overlap between the binary foreground segmentation B for each camera and synthetic binary foreground images S created by drawing 1.8×0.5 m rectangles in each camera at the locations of all $\mathcal{P}^n \in \mathcal{A}^i$. The overlap score is quantified as $\sum S \oplus B$, with $\oplus$ the per-pixel *XOR* operator. In [15], this score is normalized by $\sum S$. We choose not to normalize, allowing computation of the score for empty S. This lets creation of new tracks in an empty scene be guided by the foreground segmentation instead of a default value for the score. We also change the way S is constructed in the *preselection* step, where all $\mathcal{P}^n$ are evaluated independently. In [15], the observation likelihood for $\mathcal{A}^i$ is based on S containing only the $\mathcal{P}^n$ being evaluated. When the scene has multiple people, such S do not match B well, resulting in low observation likelihoods. Instead, we use the Kalman filtered person predictions of all tracks other than the one $\mathcal{P}^n$ is currently being assigned to as the basis of S. For track to detection assignment and person creation we add a rectangle at the corresponding $\mathcal{P}^n$ location, while for track deletion the rectangle corresponding to that track is removed. This makes the observation likelihood of $\mathcal{A}^i$ computed in the *preselection* step more similar to the one computed in the *verification* step. Incorrect $\mathcal{A}^i$ will also be pruned more frequently since they increase $\sum S \oplus B$. In [15], any track assigned to the same detection has the same observation likelihood.

Furthermore, an extra term was added to the object creation likelihood, requiring a hypothesis to explain a minimum number of extra foreground pixels when adding a person. This reduces the chance of accepting detections generated by foreground noise as new persons. The value depends on the expected size of a person entering the scene per camera. The 'train station data' (see sec. 4) with cameras relatively close to the scene, uses 10% of the number of pixels in the image. The 'hall data', with cameras further away from the scene, uses 8%.

When computing appearances of $\mathcal{P}^n$ in the *preselection* step, we not only use appearance cues for $\mathcal{P}^n$ guaranteed to be fully visible under any $\mathcal{A}^i$ as in

[15], but relax this constraint and compute the appearance of the visible part of $\mathcal{P}^n$ visible for at least 25%. Furthermore, instead of using a fixed distribution for the likelihood term based on the distance between a track and a detection [15], we use each track's Kalman filter's predictive distribution to evaluate the likelihood of assigning a detection to that track. This increases tracker flexibility and improves the chance of re-establishing a lost track at a later point in time.

Finally, to reason about $\mathcal{P}^n$ occluded by static objects, we opted to use manually created foreground masks to reconstruct a volume space containing static objects (see [10] for an automatic method). The foreground segmented images at each timestep are augmented with the static object foregrounds before volume reconstruction. After reconstruction, the static object volume is subtracted.

3.2 K-Shortest Paths

KSP [2] does tracking by minimizing the flow trough a graph constructed by stacking POMs [8] from a batch of sequential frames. Each POM location is a graph node, connected to its 9 neighbors in the next frame. Tracks are modeled as flows trough this graph with costs defined by the POM probabilities at locations connected by the tracks. Finding the optimal set of disjoint tracks with minimum cost is formulated as a linear programming problem. Like [2], we use consecutive batches of 100 frames. Track consistency between batches is created by adding the last frame of the previous batch in front of the current batch, forcing flows to start at the track locations from the last frame of the previous batch.

KSP-App [1] extends KSP, incorporating appearance information into KSP's linear programing formulation. For this purpose, the KSP graph is stacked L times, creating L 'groups'. Each of these groups is assigned a predefined appearance template and the number of objects that can have a path in each group is limited. Each appearance consists of one color histogram per camera. Using a KSP iteration, the graph is pruned and appearances are extracted at locations along the tracks and at locations where tracks are separated by at most 3 nodes. The extracted appearance information is compared to the templates using KL divergence and the graph's edges are reweighed using these values. KSP is run a second time using the new graph to determine the final paths. More detailed descriptions of KSP and KSP-App are found in [2] and [1].

3.3 Background Estimation

The datasets used in our experiments contain significant amounts of lighting changes and background clutter. An *adaptive* background estimation method, compensating for changes in the scene over time by learning and adapting the background model on-line, would be preferred in this case. The method presented in [24] uses a Mixture of Gaussians per pixel to model the color distribution of the background and is to some extent robust with respect to illumination. In our scenarios however, where people tend to stand still for some time (up to a minute), preliminary experiments have shown that the adaptive nature of the method causes them to dissipate into the background, creating false negatives.

RCTA solves this by adding tracker feedback into the learning process, updating the background model only at locations without tracks. For the KSP methods, this type of feedback is not straightforward since tracking results are only available after processing the full batch, preventing frame-to-frame reinforcement of the learned background model. Therefore, we use the foreground segmentation method from [18], implemented by [21], as a second, *static* background estimation method. It models the empty scene using eigenbackgrounds constructed from images of the empty scene under different lighting conditions. Nevertheless, foreground segmentations created by this method show more noise than foregrounds generated by RCTA$^+$'s adaptive method. For comparison we used the static background model for both KSP and RCTA methods. Furthermore, we used the foreground segmentations from RCTA$^+$'s adaptive background model as input for the KSP methods, effectively cascading KSP and RCTA$^+$.

4 Experiments

Datasets. Experiments were done on two datasets (fig. 1)[2]. The outdoor 'train station data' has 14 sequences of in total 8529 frames, recorded on a train platform. Between two and five actors enact various situations ranging from persons waiting for a train to fighting hooligans. The scenes have dynamic backgrounds with trains passing by and people walking on the train platform. Lighting conditions vary significantly over time. The area of interest (a.o.i.) is 7.6×12 m and is viewed by 3 overlapping, frame synchronized cameras recording 752×560 pixel images at 20 fps. Ground truth (GT) person locations are obtained at each frame by labeling torso positions, annotating the shoulder and pelvis locations of all persons in all cameras and projecting these onto the ground plane.

The indoor 'hall data' is one 9080 frame sequence recorded in a large central hall. During the first half, actors move in and out of the scene in small groups. After this, two groups of about 8 people each enter one by one and start arguing and fighting. Fig. 2(a) shows the number of people in the scene over time. The 12×12 m a.o.i. is viewed by 4 overlapping, frame synchronized cameras recording 1024×768 pixel images at 20 fps. GT positions are generated every 20^{th} frame by annotating every person's head location in every camera, triangulating these points in 3D and projecting them onto the ground plane. This data is considerably more difficult than the previous dataset, since it contains more, similarly clothed people forming denser groups, and the cameras are placed further away from the scene. Furthermore, many people wear dark clothing, which combined with the dark floor of the hall and multiple badly lit regions complicates foreground segmentation.

Evaluation Measures. Tracking performance is evaluated using the same metrics as in [1]. A missed detection (*miss*) is generated when no track is found within 0.5 m of a GT location, while a false positive (*fp*) is a track without a GT

[2] The data set is made available for non-commercial research purposes. Please follow the links from `http://isla.science.uva.nl/` or contact the second author.

Fig. 1. All viewpoints of the train station data (top) and the hall data (bottom).

location within 0.5 m. The mismatch error (*mme*) counts the number of identity switches within a track and is increased when a track switches between persons. The global mismatch error (*gmme*) is increased for every frame a track follows a different person than the one it was created on. The number of GT persons (*gt*) is the total number of annotated person positions within the area covered by all cameras. Annotations outside this area and cases where a track is on one side of the area boundary and the ground truth on the other are not counted. This results in small differences between *gt* for different experiments.

Multi Object Tracking Precision (MOTP) and Multi Object Tracking Accuracy (MOTA) [3] summarize performance. MOTP describes the average distance between tracks and GT locations, while MOTA is defined as $1 - \frac{fp+miss+mme}{gt}$.

Implementation Details. For the train station data, POM settings are taken from [8], using 20 cm grid cells and person ROI of 175×50 cm. For the hall data, 40 cm grid cells are used. Smaller cells cause POM to detect too few people in the dense second half of the scenario. POM's person prior was set to 0.002 in all experiments. RCTA parameters were taken from [15], using voxels of $7 \times 7 \times 7$ cm. Appearance templates for KSP-App are sampled at manually selected POM locations for the train station dataset. For the hall dataset, running KSP-App turned out to be problematic. The density of people, combined with the 40 cm grid cells enlarging the spatial neighborhood of each cell, limit graph pruning, increasing the problem complexity. Even when using a reduced set of 5 instead of 23 templates, we were only able to process a small part of the scenario after a day. Therefore, we were unable to get KSP-App results on the hall dataset.

Most scenarios in the train station dataset start with persons in the scene. Because RCTA has a low probability of creating new tracks in the middle of the scene, tracking is bootstrapped using GT detections. For a fair comparison, the batch mode methods use ground truth initialization for the first frame as well.

All methods are implemented in C^{++}.[3] Experiments were performed on a 2.1 GHz CPU and 4 GB RAM. Computation time was measured on a 960 frame

[3] KSP and RCTA implementations were kindly provided by the authors. For KSP-app we used our own implementation as it could not be made available.

(a) Results on the train station data

method	background	MOTA	MOTP	miss	fp	mme	gmme	gt
RCTA$^+$	adaptive	0.89	15	655	2296	53	2749	28348
KSP-App	static	0.84	17	2478	1907	43	5173	28410
KSP	static	0.84	17	2515	1945	57	9940	28409
RCTA$^+$	static	0.74	14	5779	1467	44	3851	28348
RCTA	adaptive	0.72	16	3203	4559	76	9689	28348
RCTA	static	0.67	15	3872	5321	120	9176	28348
KSP-App	RCTA$^+$ adaptive	0.92	16	1070	1248	29	4983	28429
KSP	RCTA$^+$ adaptive	0.91	16	1182	1358	46	8997	28429

(b) Results on the hall data

method	background	MOTA	MOTP	miss	fp	mme	gmme	gt
RCTA$^+$	adaptive	0.54	18	1203	747	97	1371	4419
RCTA$^+$	static, low thr.	0.38	20	1537	1030	186	1839	4419
KSP	static, low thr.	-0.16	29	2702	2286	126	1364	4414
RCTA$^+$	static, high thr.	0.12	18	3839	55	15	371	4419
KSP	static, high thr.	0.28	28	1996	917	252	1966	4411
RCTA	adaptive	0.30	17	2654	361	62	1536	4419
RCTA	static, low thr.	0.24	16	3085	250	27	1085	4419
RCTA	static, high thr.	0.24	17	3084	199	58	1248	4419
KSP	RCTA$^+$ adaptive	0.22	28	1902	1188	344	2148	4415

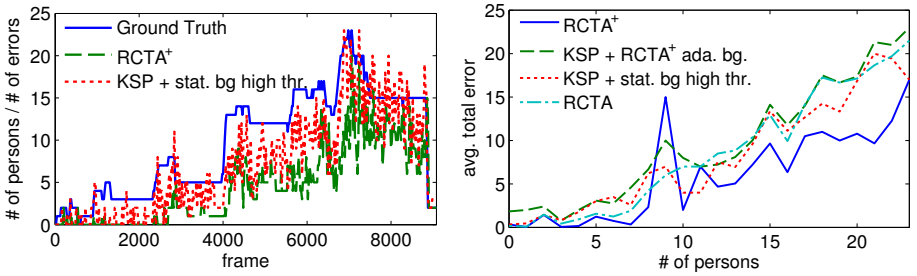
Table 1. Performance of all methods and background models, on both datasets. MOTA: accuracy, higher is better. MOTP: precision (cm), lower is better. *miss*: misses. *fp*: false positives. *mme*: mismatch error. *gmme*: global *mme*. *gt*: ground truth persons.

sequence with 4 persons. KSP took ±8.9 seconds per frame (s/f) while KSP-App needed ±9.7 s/f. Of these times, ±8.8 s/f is used by POM (this is longer than stated in [8] because we use more ground plane locations and higher resolution images). RCTA and RCTA$^+$ perform detection and tracking at ±6.5 s/f.

Tracking Results. Table 1 shows the results per dataset. For the train station data, scores are accumulated over all scenarios. Table 1(a)'s top 6 rows show results of the stand-alone trackers combined with each background estimation method. RCTA$^+$ shows overall improvement over RCTA. Combining RCTA$^+$ and the adaptive background model gives the highest MOTA and lowest *gmme*. Static background experiments suffer from foreground segmentation errors. For the train station data, the static background model was configured to minimize false positives from strong illumination changes and shadows, classifying as few people as possible as background. This trade off results in higher *miss* rates for methods using static backgrounds. Because both RCTA methods assume a person to be well segmented in all cameras for reliable volume carving, foreground segmentation errors have most effect here. The POM detector has no such assumption, making it more robust to these artifacts.

MOTP is worse for the KSP methods since volume carving, allowing higher spatial resolutions than POM, offers more positional flexibility. KSP-App's main improvement over plain KSP is in the *mme* and *gmme*. This is to be expected, since KSP-App performs extra processing of the KSP tracks to correct id switches.

RCTA$^+$'s slightly higher *mme* reduces the number of *gmme*. Part of the extra id changes switch the tracker back to its original target. This also accounts for



(a) # of errors/persons over time. (b) Average total error vs. # of persons

Fig. 2. People and error statistics over time for the hall dataset.

some of RCTA$^+$'s *fp*. When a detection disappears, RCTA$^+$ often switches to an untracked hypothesis with similar appearance instead of removing the track, especially when the track is at some distance from the scene's boundaries. When the original target re-emerges some time later, RCTA$^+$ switches back.

The last two rows of table 1(a) show that using adaptive backgrounds generated by RCTA$^+$ as input to the KSP marginally improves performance over RCTA$^+$. KSP benefits from the cleaner foregrounds, while KSP-App improves on this result by providing more stable tracking results. The *gmme* for this last method is still higher than for pure RCTA$^+$, as a side-effect of the reduced *mme*.

Table 1(b) shows the results on the hall dataset. The large number of people and their close proximity in the second half of the scenario results in lower performance compared to the train station data. Standard RCTA's failure creating tracks is seen in the high *miss* rate but lower number of *fp*. This can to a large extent be blamed on the overlap computation as discussed in sec. 3.1. RCTA$^+$ using its adaptive background model again outperforms the other methods, this time including the RCTA$^+$-KSP cascade. This shows a more fundamental issue of KSP and the POM detector with crowded scenarios. When persons' foreground segmentations are not separated in any view, POM will detect too few persons, assuming the rest of the foreground regions are noise. Enlarging the POM grid to 40 cm cells partially compensates this, but causes missed detections when people are very close together and lowers detection precision. RCTA$^+$'s volume carving and clustering approach has less problems splitting dense groups, but also creates incorrect detections when the volume space is incorrectly clustered. KSP's lack of appearance model makes it is more prone to track switches as well.

Because of the challenging conditions of the hall dataset described earlier, using the same configuration for the static background model as for the train station data results in many missing foreground segments. Therefore, we did additional experiments using a lower segmentation threshold, detecting more people but increasing the foreground noise from illumination and shadows. Results using the 'high' and 'low' threshold settings are marked as resp. 'high thr.' and 'low thr.' in table 1(b). Again, RCTA$^+$ shows sensitivity to missing detections, resulting in a lower MOTA for the high threshold static backgrounds.

Fig. 3. Examples of tracking results. (top) Train station data: RCTA$^+$, KSP-App, KSP and KSP-App/RCTA$^+$ cascade. (bottom) Hall data: RCTA$^+$, KSP with low background threshold, KSP with high background threshold and KSP-RCTA$^+$ cascade.

KSP shows bad performance when using the low threshold however, producing more errors than the *gt*, resulting in a negative MOTA. When using the high threshold, KSP shows better results, but also suffers from the missing detections.

Fig. 3 shows some examples of tracking results from one of each dataset's viewpoints. In fig. 2(b) the average total error ($fp + miss + mme$) per frame containing a certain number of people is shown for the hall data for the best performing versions of each method. The figure shows a relatively constant error up to 7 people, after which it starts to increase linear with the number of people in the scene. The RCTA$^+$ error shows an outlier at 9 people because the dataset has only 1 annotated frame containing 9 people, at the end of the scenario. At that point, multiple *fp* of people who just exited the scene still linger. Fig. 2(a) shows the evolution of both the number of people during the scene, and the error per frame for RCTA$^+$ and KSP with static background and high threshold.

5 Conclusion

In this paper, three state-of-the-art tracking methods and our adaptation of one have been compared in combination with two types of background estimation. RCTA$^+$ with adaptive backgrounds consistently outperforms the other stand-alone methods, including RCTA. For lower person-density scenarios, KSP-based methods give competitive results. For higher person-density scenarios, KSP-based methods suffer from the limitations of the POM detector when persons overlap in many cameras. RCTA$^+$ with adaptive backgrounds outperforms the latter by a MOTA score of 0.26.

References

1. Ben Shitrit, H., et al.: Tracking multiple people under global appearance constraints. In: Proc. of the ICCV. pp. 137–144 (2011)

2. Berclaz, J., others.: Multiple object tracking using k-shortest paths optimization. IEEE Trans. on PAMI 33(9), 1806–1819 (2011)
3. Bernardin, K., Stiefelhagen, R.: Evaluating multiple object tracking performance: the clear mot metrics. JIVP 2008, 1–10 (2008)
4. Breitenstein, M.D., et al.: Online Multi-Person Tracking-by-Detection from a Single, Uncalibrated Camera. IEEE Trans. on PAMI 33(9), 1820–1833 (2011)
5. Calderara, S., Cucchiara, R., Prati, A.: Bayesian-Competitive consistent labeling for people surveillance. IEEE Trans. on PAMI 30(2), 354–360 (2008)
6. Collins, R.: Mean-shift blob tracking through scale space. In: Proc. of the IEEE CVPR. vol. 2, pp. II–234 (2003)
7. Du, W., Piater, J.: Multi-camera people tracking by collaborative particle filters and principal axis-based integration. In: Proc. of the ACCV, vol. 4843, pp. 365–374 (2007)
8. Fleuret, F., et al.: Multicamera people tracking with a probabilistic occupancy map. IEEE Trans. on PAMI 30(2), 267–282 (2008)
9. Fortmann, T., Bar-Shalom, Y., Scheffe, M.: Sonar tracking of multiple targets using joint probabilistic data association. IEEE JOE 8(3), 173–184 (1983)
10. Guan, L., Franco, J.S., Pollefeys, M.: Multi-view occlusion reasoning forprobabilistic silhouette-based dynamic scene reconstruction. IJCV 90(3), 283–303 (2010)
11. Huang, C.C., Wang, S.J.: A bayesian hierarchical framework for multitarget labeling and correspondence with ghost suppression over multicamera surveillance system. IEEE Trans. on ASE 9(1), 16 –30 (2012)
12. Kang, J., Cohen, I., Medioni, G.: Tracking people in crowded scenes across multiple cameras. In: ACCV. vol. 7, p. 15 (2004)
13. Khan, S., Shah, M.: Tracking multiple occluding people by localizing on multiple scene planes. IEEE Trans. on PAMI 31(3), 505–519 (2009)
14. Kim, K., Davis, L.: Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering. In: Proc. of the ECCV, vol. 3953, pp. 98–109 (2006)
15. Liem, M., Gavrila, D.M.: Multi-person localization and track assignment in overlapping camera views. In: Pattern Recognition, vol. 6835, pp. 173–183 (2011)
16. Mittal, A., Davis, L.: M 2 Tracker: a multi-view approach to segmenting and tracking people in a cluttered scene. IJCV 51(3), 189–203 (2003)
17. Munkres, J.: Algorithms for the assignment and transportation problems. Journal of the Society for Industrial and Applied Mathematics pp. 32–38 (1957)
18. Oliver, N.M., Rosario, B., Pentland, A.P.: A bayesian computer vision system for modeling human interactions. IEEE Trans. on PAMI 22(8), 831–843 (2000)
19. Reid, D.: An algorithm for tracking multiple targets. IEEE Trans. on Automatic Control 24(6), 843–854 (1979)
20. Santos, T.T., Morimoto, C.H.: Multiple camera people detection and tracking using support integration. PRL 32(1), 47–55 (2011)
21. Sobral, A.C.: BGSLibrary: A opencv c++ background subtraction library (2012), software available at `http://code.google.com/p/bgslibrary/`
22. Szeliski, R.: Rapid octree construction from image sequences. CVGIP 58(1), 23 – 32 (1993)
23. Zhang, L., Li, Y., Nevatia, R.: Global data association for multi-object tracking using network flows. In: Proc. of the IEEE CVPR. pp. 1–8 (2008)
24. Zivkovic, Z., van der Heijden, F.: Efficient adaptive density estimation per image pixel for the task of background subtraction. PRL 27(7), 773–780 (2006)