Object Tracking as Job-Scheduling Problem

Uwe Jaenen*, Henning Spiegelberg†, Lars Sommer†, Sebastian von Mammen*, Juergen Brehm† and Joerg Haehner*
* Chair for Organic Computing
Institute of Computer Science
Universität Augsburg, 86159 Augsburg, Germany
http://www.informatik.uni-augsburg.de/lehrstuehle/oc/
† Department for System- and Computer-Architecture
Institute for Systems Engineering
Leibniz Universität Hannover, 30167 Hannover, Germany
http://www.sra.uni-hannover.de

***Abstract*—This paper establishes a connection between object tracking from a systems point of view and the job-scheduling or job-shop problem. Often, surveillance areas cannot be fully monitored by a set of smart cameras at any given point in time. Decisions have to be made, which objects are to be tracked. The computer vision aspects of object tracking have made substantial strides which permits for elaborately planning the monitoring jobs. In this paper, object tracking is handled as a job-scheduling problem. As a result, tracked objects are considered as scheduling jobs that rely on smart cameras as resources that follow according tracking policies. The presented job-scheduling approach is based on proactive quotations advertised by the jobs. The main advantages of this algorithm are the avoidance of negotiation chains and the acceptance of local non-optimal solutions to benefit the overall performance.**

I. Introduction

This paper introduces a job-scheduling approach for planning distributed monitoring jobs in smart camera networks. Often, surveillance areas cannot be fully monitored by a set of smart cameras at any given point in time. Therefore, decisions have to be made, which objects are to be tracked. From the point of computer vision, object tracking is a combination of object detection and data association for assigning unique object-IDs. Various detection techniques for pan-tilt-capable smart cameras are known [1], [2]. Pre-evaluations have shown that the HOG-Detector [3] can be used with pan-tilt capable cameras as well. The correct assignment of unique object-IDs can be handled, e.g., relying on object positions in world coordinates if the object-density is low. More sophisticated strategies like using artificial neural networks are also known [4], [5]. As a consequence, our research hypothesis is: The computer vision aspects of object tracking have made substantial strides which permits for elaborately planning the monitoring jobs. In Figure 1 a distributed smart camera network is depicted alongside moving objects and groups of objects. In this paper, we consider these objects to be persons. At the bottom of the figure, a specific scene is shown in more detail. Person one and two diverge. Hence, smart camera (SC) one is only capable of tracking one person or the other. The obviously best solution would be, that SC1 tracks person one, SC2 tracks person two and the idle camera SC3 would track person three. The challenge is the distributed coordination of this object-camera association. Each object or object-group to be tracked can be considered as a job. In this context, the smart cameras represent a limited available resource. Object tracking can be considered as a general job-shop problem with an arbitrary number of jobs in an arbitrary shop with an arbitrary number of machines. In the following section, we will define two objectives for object tracking. In Sections III and IV object tracking is mapped to the job-shop problem which is handled by a proactive and quotation-based scheduling algorithm. The state of the art in Section VI gives an overview of already known scheduling algorithms within this field. In the evaluation in Section VII the performance of the introduced scheduling algorithm is examined with respect to the defined tracking objectives.

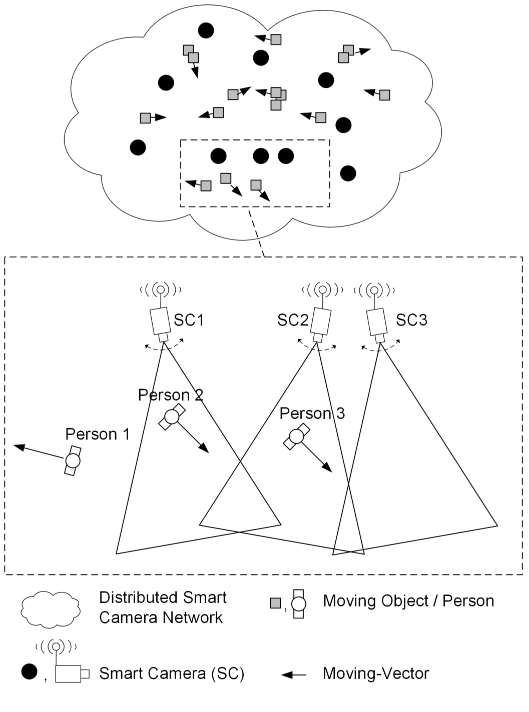

Fig. 1: Surveillance System consisting of several smart cameras.

II. Objectives of Object Tracking

The surveillance system is assumed to be one of the third generation according to the classification by Valera et al. [6]. This implies that the surveillance system performs autonomously. In particular, our system acts in a decentralized and self-organizing manner to execute its surveillance jobs. If the system detects a critical event, it will be automatically signaled to the user. The event could be, e.g., a conspicuous trajectory which has been detected by trajectory-analysis. The user is typically a member of the security staff working in a security-control-room. The whole surveillance area is displayed in form of pictograms on a huge multi-touch-table which is used to interact with a distributed smart camera network [7]. A simplified version of such an interaction-circle is depicted in Fig. 2. Engineering distributed systems often

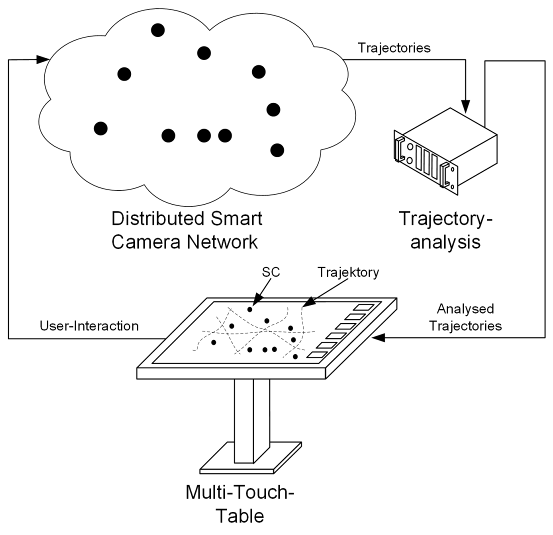

Fig. 2: Surveillance System

means optimizing the utilization of resources. However, our experience in projects like CamInSens [7] and QTrajectories [8] has taught us, that systems which do not reflect the expectations of the users will not be accepted in practice. This is why in this paper the objectives of tracking are specifically defined in terms of user expectations. In our experience, two tracking objectives are of major interest to the user. Firstly, it should be possible to gain an overview of the system across the entire monitoring area. In respect to object tracking, this requirement means maximizing the number of detected objects. Secondly, high-quality trajectories should be made available to the automated trajectory analysis. This requirement can be read as the maximization of the trajectory length of an arbitrarily large set of the longest trajectories. Note, the maximization of the average trajectory length is not an appropriate goal. It would promote a given longest trajectory and dismiss all shorter trajectories.
Formally, the objectives can be defined as follows, with M_{obj} as set of all detected objects, M_{traj} as set of all recorded trajectories and $max_e\{M_{taj}\}$ as set of the ϵ longest trajectories. In total, Equations 1, 2 formally describe our first and second objective, respectively.

$$max(|M_{obj}|) \tag{1}$$

$$max(max_\epsilon\{M_{traj}\}) \tag{2}$$

In Fig. 3 hypothetical results of object tracking routines are sketched. The y-axis represents the number of detected objects or partial-trajectories. The x-axis shows the longest monitored trajectory. An idealized algorithm for objective 1 captures the whole surveillance area. In this case, the number of partial-trajectories would equal the number of objects. An idealized algorithm for objective 2 would record the longest available trajectory and several partial-trajectories of other objects that pass through the cameras' fields of view as by-catch. Camera systems equipped with tracking algorithms which are not specialized on the described objectives will produce results in the blue area. This blue area is caused by the internal memory of computer vision techniques. If an object is detected in sufficient short continuous time periods, most computer vision algorithms will recognize the object as the object in the internal memory and interpolate the trajectory. Thus a high-speed panning camera head will record all objects and the longest trajectory in its surrounding. This causes the intersection of objective one and two, illustrated in Fig. 3 as the cone end of the blue area on the crossing of the black dashed and the red line.

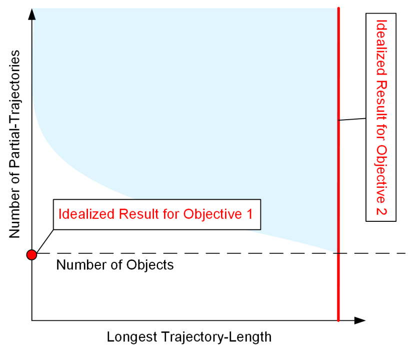

Fig. 3: Sketched results for objective 1 and 2

III. Mapping Object Tracking to the Job-Shop Problem

As mentioned in Section I each object or object-group can be understood as an individual job a_n. Each smart camera can be considered a resource r_i. If a job a_n has allocated a set of resources R_n to fulfill its tasks, it is denoted as $a_n \multimapdot R_n$. This set is a subset of all possible resources R that job a_n can use to fulfill its tasks, $a_n \multimap R$. To reduce the degree of resource utilization, an object is only tracked by a single camera $|R_n| = 1$. In order to map the object tracking problem to the job-scheduling problem a gantt-diagram has to be calculated for each camera. In this diagram, the predicted residence time of each object(-group) in its field of view is plotted, see Figure 4. A detailed introduction to the grouping of objects and the estimation of the residence time can be found in [8]. In Fig. 4, a case is presented in which person three could be tracked by SC1, SC2 and SC3, and it is actually tracked by SC2. Denoted as job-shop problem $a_3 \multimap \{r_1, r_2, r_3\}$ and $a_3 \multimapdot \{r_2\}$, with a_3 as tracking job for person three and r_2 as resource SC2. In a simplistic way, the tracking-performance of job a_n using smart camera r_i is defined by the trajectory-length for a specific object within a given time period. Formally it is described by $p_i^n(t)$. The accomplished work, or workload, of a job is defined by $P_i^n = \int_t p_i^n(t)$.

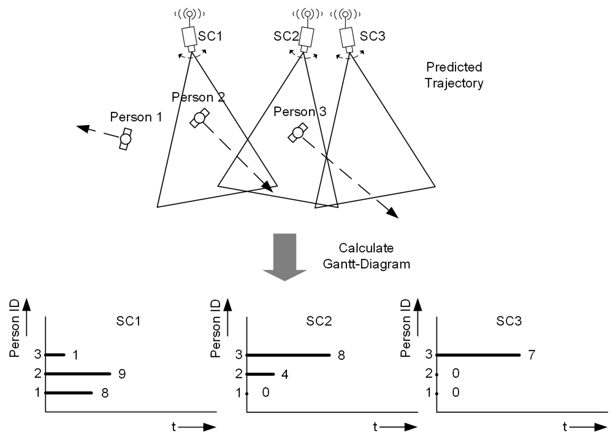

Fig. 4: Mapping trajectories to gantt-diagrams. The digits represent the predicted workload respectively trajectory length.

IV. PQB-Scheduling

This section describes our proactive and quotation-based scheduling approach. The algorithm is designed for self-organizing distributed systems. The main aspects are the compression of negotiation time and the mapping of enhancements of the overall system performance onto local jobs performance. Fig. 4 shows an example that elucidates these aspects. The depicted initial situation can be described as follows:
$a_1 \multimap \{r_1\}$ with $a_1 \rightarrow\bullet \{\}$, $a_2 \multimap \{r_1, r_2\}$ with $a_2 \rightarrow\bullet \{r_1\}$ and $a_3 \multimap \{r_2, r_3\}$ with $a_3 \rightarrow\bullet \{r_2\}$. The obviously best solution for assigning objects to cameras would be: $a_1 \rightarrow\bullet \{r_1\}$, $a_2 \rightarrow\bullet \{r_2\}$ and $a_3 \rightarrow\bullet \{r_3\}$.
The performance p_i^n of each job n on each camera i is assumed to be 1. Hence, the workload P_i^n of each job is increased by 1 at each discrete tracking time-step. A naive approach to object tracking could rely on negotiation chains. In this case, SC1 would have to hand over the tracking job concerning person two to SC2. As a consequence, SC2 would have to hand over the job of tracking person three to SC3. Due to the arising interlaced hand-over-negotiation-chains, this approach does not scale well for large networks. PQB-Scheduling avoids these complex and time-consuming chains. In the following, the compression of negotiation time and the mapping of enhancements of the overall system performance onto local jobs are described.

A. Proactive Broadcasting Quotations

The key idea for compressing the negotiation time is making information about the conditions for changing resources available to each job before it will be needed. At the same time, the local dependencies of these conditions have to be resolved automatically. This is achieved by proactively broadcasting quotations Q_i^n of those jobs a_n that occupy resources r_i. Quotations are constraints which have to fulfilled by another job $a_{n'}$ in order to receive the resource r_i from job a_n. In Fig. 5 the tracked persons and cameras are represented on the left-hand side of the figure. On the right, the resulting associations between smart cameras and jobs are depicted. We differentiate between jobs that occupy a smart camera and those that are only located on a smart camera. Only jobs that occupy smart cameras are allowed to broadcast constraints. If a job receives a constraint, it updates its quotation according to the local mapping described in the following section IV-B.

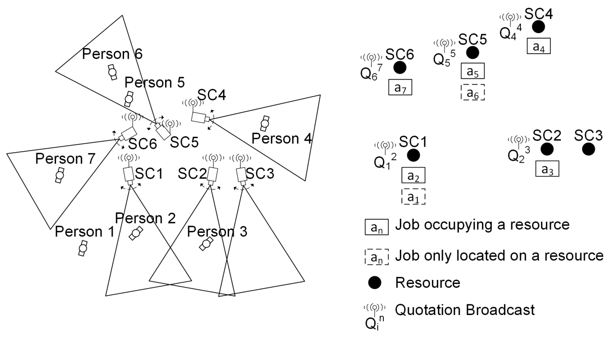

Fig. 5: Compression of negotiation time by proactive quotation broadcasts.

B. Local Mapping

Local mapping means the translation of possible global system performance enhancements onto a local performance enhancement. This local performance enhancement determines the behavior of the corresponding local job. It adapts itself accordingly resulting in an efficient coordination process without the need for a central planning instance. Consider a surveillance scenario that started in the initial state depicted in Fig. 4. For the ease of understanding, every workload-digit in the gantt-diagram in Fig. 4 is assumed to be 7, but 0 will stay 0. On the left-hand side of Fig. 6 the system workload over time is depicted, whereas the right-hand side shows the local workload evolution of job a_1 occupying resource r_1. If a person is tracked by a camera, the system workload is increased by one at each step. Time step (*) marks the divergence point of three different scenarios. The solid line depicts the initial person-to-smart camera assignment (Fig. 4). The black dashed line shows the optimal possible system performance, that is reached if $a_1 \rightarrow\bullet \{r_1\}$, $a_2 \rightarrow\bullet \{r_2\}$ and $a_3 \rightarrow\bullet \{r_3\}$. Thus, at time (**), the optimized workload offsets the initial workload (via person-to-smart camera assignment) by a positive increment Δ. On the right-hand side of Fig. 4, PQB-Scheduling maps the possible system workload improvement onto a local job-workload. For instance, let's assume that the performance of job a_3 on resource r_3 is reduced to $0,5$, this would automatically be mapped to P_1^1, see the red dashed line in the middle.

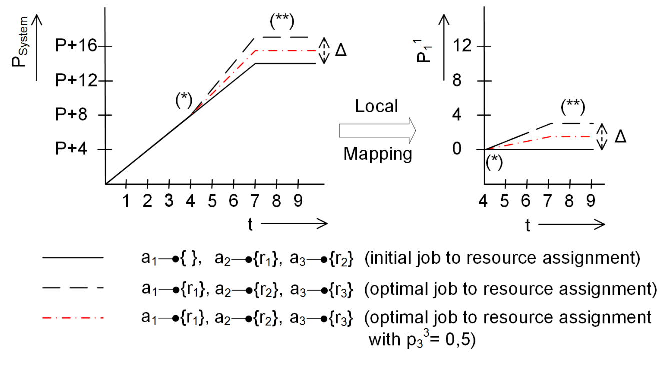

Fig. 6: Mapping of system workload onto local workload of a single job.

In the following a quotation Q_i^n is called sales quotation, because only a virtual price has to be paid by a job a_X to receive the resource r_i from job a_n. In this context, workload is equivalent to virtual money. In other use-cases than object tracking a job may need more than one camera, e.g. in multi-view scenarios. Then the quotation could contain a constraint to handover a dedicated smart camera to the offering job. However, a sales quotation solely implies that if job a_n releases its resource r_i, the beneficiary job a_X has to compensate for the loss, see Eq. 3:

$$Q_i^n = P_i^n = \int_t p_i^n(t) \tag{3}$$

If job a_n can migrate to an alternative resource $r_{i'}$, the quotation has to be reduced by the predicted success on that resource, see Eq. 4:

$$Q_i^n = P_i^n - \max_{i'}\{P_{i'}^n\} \tag{4}$$

If job a_n has to ransom the alternative resource r_i from an offering job $a_{n'}$ the possible success on resource $r_{i'}$ has to be reduced by these costs. The maximum predicted reduced workload of a_n on $r_{i'}$ is the mapping function (Eq. 5):

$$\max_{i'}\{P_{i'}^n - Q_{i'}^{n'}\} \tag{5}$$

The quotations to be broadcast by job a_n on resource r_i are given by:

$$Q_i^n = P_i^n - \max_{i'}\{P_{i'}^n - Q_{i'}^{n'}\} \tag{6}$$

The drawback of our approach is that it does not necessarily result in a concluded negotiation. This drawback emerges from the fact that each job assumes that it will be able to acquire certain resources, although their availability cannot be guaranteed—they might have been sold to another job in the meantime. As a result, the maximum loss $\int_t p_i^n(t)$ is caused by releasing the resource r_i in erroneous anticipation of allocating resource $r_{i'}$. To reduce this risk, each job may add a fee $f^n(i,i')$ to the quotation which quantifies the risk of an unsuccessful takeover. The broadcast quotation results to:

$$Q_i^n = P_i^n - \max_{i'}\{P_{i'}^n - Q_{i'}^{n'} - f^n(i,i')\} \tag{7}$$

C. Continuously Adapting the Tracking Objective

In Section II two tracking objectives, snapshots and automated analysis, were introduced. In this section, we assume that the user of the surveillance system changes his corresponding objective seamlessly. This is reflected by the pricing of quotations. Small costs (low constraints) on the retrieval of resources makes handovers more probable. A high price (high constraints) decreases the probability of a handover. A proper balance can, therefore, be realized by a weighting factor $\alpha(t)$. The term $\alpha(t)$ in Eq. 8 depends on the trajectory length T. If a snapshot is required, $\alpha(T)$ will decrease with the trajectory length, e.g. $\alpha(T) = 1/T$. To achieve long trajectories the price must increase with the length, e.g. $\alpha(T) = T$.

$$Q_i^n = \alpha(T)\left(P_i^n - \max_{i'}\{P_{i'}^n - Q_{i'}^{n'} - f^n(i,i')\}\right) \tag{8}$$

D. Job Behavior

Each job broadcasts its quotation and receives a set of quotations $\{Q_i^n, ..., Q_I^N\} = \{Q_{i'}^{n'}\}$. Based on these information, jobs try to find the best resource. To reduce the risk of deadlocks, circular relationships should be avoided. Therefore, each quotation logs a history composed of participating job-IDs n and resource-IDs i. A quotation is considered valid by a job, if its job-ID is not part of the history, no single resource-ID is included multiple times, and the current resource-ID is not included either. Invalid quotations will be erased by each job of his received set $\{Q_{i'}^{n'}\}$. In the next step job a_n will rate all alternative resources $r_{i'}$ to explore the best one. To rate an alternative resource, the costs $Q_{i'}^{n'}$ for the purchase of the resource $r_{i'}$ from job $a_{n'}$ must be subtracted from the predicted workload:

$$P_{i',red}^n = P_{i'}^n - Q_{i'}^{n'} \tag{9}$$

This maximum value from the set of reduced workloads $\{P_{i',red}^n\}$ quantifies the best achievable success on an alternative resource. The change to a different resource only makes sense if this value is greater than the workload on the current resource. If the change makes no sense, the job will send a quotation by itself if it currently occupies the resource r_i. A formal description of the job behavior is given by Alg. 1.

```
Data: $\{Q_{i'}^{n'}\}$, $P_i^n$, $P_{i'}^n$
Result: move to resource $r_{i'}$ {void}
receive $\{Q_{i'}^{n'}\}$ messages;
check validity of $\{Q_{i'}^{n'}\}$;
forall the $\{Q_{i'}^{n'}\}$ do
    calculate $P_{i',red}^n$;
end
if $max_{i'}\{P_{i',red}^n\} > P_i^n$ then
    initial transaction to resource $r_{i'}$;
else
    if $a_n \multimap \{r_i\}$ then
        send $Q_i^n$-message;
    end
end
```

Algorithm 1: Job decision process to stay on current resource or moving to an alternative resource.

For the ease of understanding we take a look at the example in Fig. 4. An assumed classical priority-based algorithm would assign the job with the highest performance to available resources in descending order. So the the initial, undesirable situation persists. PQB-Scheduling solves this problem using the local mapping function in Eq. 5: Job/Object a_3 broadcasts $Q_2^3 = P_2^3 - P_3^3 = 8 - 7 = 1$. Job a_2 broadcasts $Q_1^2 = P_1^2 - (P_2^2 - Q_2^3) = 9 - (4 - 1) = 6$. Job a_1 compares $P_{1,red}^1 = 8 - 6 = 2$ to $P_{idle}^1 = 0$ and initiates the resource handover.

V. Getting Reference Values for the Tracking Objectives

As described in Sec. II the tracking objectives comprise an overview of present objects in the surveillance area and the maximization of trajectory lengths for automatic analysis. In the following, two reference tracking approaches are presented, which are specialized on these respective objectives, formalized in Eq. 1 and 2. These reference approaches provide the foundation for the evaluation in Sec. VII.

A. *Snapshot of present objects*

In the context of the following theoretical considerations, we assume an idealized smart camera which executes pan-tilt-functionality infinitely fast. As a result, the camera would monitor its environment infinitely fast, yielding a closed-ring panoramic image. A monitoring plan would not be necessary in this case. For this reason, the number and location of all persons within a specific distance (r_{FoV}) to a camera are known. In Fig. 7 an according scenario is illustrated.

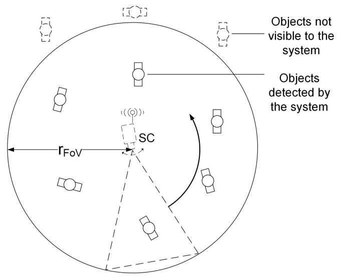
Objects not visible to the system
Objects detected by the system
SC
r_FoV

Fig. 7: Idealized smart camera with an infinite rotational speed.

B. *Long Trajectories*

To optimize the trajectory-length each camera follows these three rules:

§1 Objects will not be not discarded.

§2 Objects can be added to an existing tracking-job.

§3 When a track ends, the process with the longest continuous track, under consideration of §1 and §2, can be continued.

In order to determine the maximum trajectory length, the gantt-diagram presented in Section III is converted to a meshed graph. Each event in the gantt-diagram (the starting and ending of a track) is represented by a node. Initially, the graph is full-meshed along the time-line. Recursions are not intended. We assume the system is already tracking an object, e.g. object 1 in Fig. 8. Subsequently, all edges which violate rules §1, §2, §3 are withdrawn. The edges are weighted with the time period between two nodes and multiplied with the number of tracked objects. A longest path search on the graph yields the scheduling plan. In particular, we find the longest path, and thus the longest trajectory, relying on the well-known shortest path algorithm from Floyd-Warshall [9], simply negating the edge weights.

VI. State of the Art

Most scheduling algorithms for object tracking use greedy-search for priority-based scheduling policies. Objectives are often resource utilization, tracking quality (e.g. frames per second) and energy consumption. One of the most popular scheduling techniques is first-come-first-serve (FCFS). Objects will be tracked depending on their first detection time. FCFS often is used in state-based algorithms. DMCtrac [10] is a state-machine consisting of four states: master, slave, search and look. During the search-state the camera explores the surrounding for track-able objects. Therefore the coverage is optimized. The camera enters the look-state if a neighboring camera will handover an object. A camera is in the master-state, if it currently tracks an object. Entering slave-state, the camera is in reserve-mode which means all objects are tracked by at least one camera. The camera waits to be come active. Qianqian et al. [11] use a state-machine to reduce the energy consumption. The two-phase sleep scheduling protocol (TPSS) [11] mainly bases on nodes sleeping or discovering the surveillance area and nodes listening to the former nodes or tracking if they become notified by an awakening message. While this approach needs the location of a camera, a more sophisticated approach is given by Ren et al. [12] which does not require the position. The states are moving like a wave through the network which shall ensure the detection of each object. Qureshi et al. [13] and Costello et al. [14] use non-ptz-cameras for multi-object tracking and ptz-cameras to retrieve high-resolution images of single objects. In both papers, the presented scheduling algorithm only affects the ptz-cameras. Qureshi et al. use a weighted round robin schedule in a highly sophisticated simulation of a simulated railway-station. Costello et al. explore several techniques like FCFS and earliest-deadline-first (EDF) using a small setup of two cameras.
Apart from these classical algorithms some newly greedy-algorithms have been established for priority-based scheduling policies: objects with the most promising results will be assigned to the cameras. This correlates to the second objective defined in Sec. II. Rinner et al. [15] use vickrey auctions to assign objects to cameras. Here, the vickrey auction can be seen as a distributed variant of a greedy-search. The optimization criteria is the tracking-quality. Dieber et al. [16] use also learning approaches (ant colony optimization) to extract most frequented ways, to improve the correct object handover between cameras. Some algorithms focus on special aspects like [17], which take into account that also cameras are needed for recognition if an object disappears. On one hand, the PQB-Scheduling algorithm can be considered as an advancement of greedy-search. The jobs behave in a selfish, greedy manner by improving their local workload. But on the other hand,

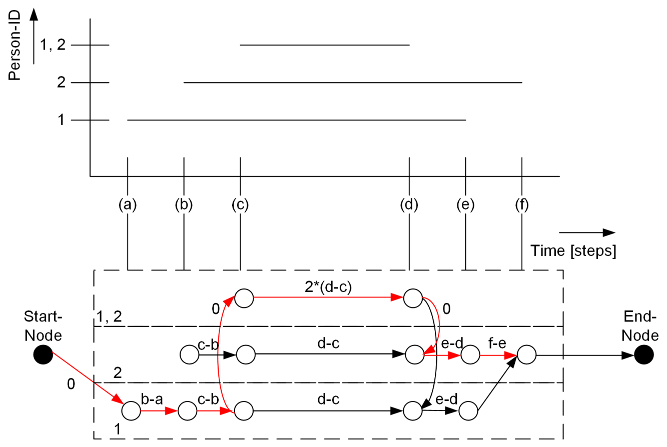

Fig. 8: Conversion of a gantt-diagram to a meshed graph. The red line shows the longest path.

the proactive broadcasting and local mapping enables jobs to restrain themselves (acceptance of local non-optimal solutions) to improve the system-performance. In future work learning approaches like Dieber et al. can be used to reduce failed object handover between cameras by learning the fee $f^n(i, i^{'})$.

VII. Evaluation

In this section, we present preliminary results regarding the performance of the PQB-Scheduling algorithm. As a first step, PQB-Scheduling will be compared to a classical priority scheduling algorithm (greedy search). As a second step, its capabilities with respect to the defined tracking objectives will be evaluated. For our evaluation, we simulated a smart camera network using the step-based simulation framework MASON (MASON: Multi-Agent Simulator Of Neighborhoods... or Networks... or something...).

A. Evaluation of PQB-Scheduling vs. Priority-Scheduling

In this section, we demonstrate the performance enhancement of PQB-Scheduling compared to a classical centralized priority scheduling algorithm. The performance is assumed to be constant over time, so performance- and workload-enhancements can be used synonymously in this special case. The priority scheduling algorithm maps jobs to resources, in descending order with respect to the workload. The simulation setup consists of six resources and six jobs. The performance of a specific job on a specific resource type was chosen randomly at the beginning of the simulation (performance-setting). Both, the priority scheduling and PQB-Scheduling, have been tested with the same 20 different performance-settings. A scheduling process was considered completed, if there has not been a change in the resources-job assignment for 50 simulation steps. Figure 9 shows the results of the comparison between priority-scheduling and PQB-Scheduling. For both algorithms the number of steps until termination and the relative fraction of the optimal performance are depicted. The optimal solution of assigning jobs to resources has been explored by brute force search a priori. The horizontal and vertical lines represent the standard deviation. The evaluation shows, that PQB-Scheduling reaches more than 99% of the possible workload in contrast to the centralized priority scheduling with more than 93%. On the one hand, PQB-Scheduling is a decentralized algorithm which is highly robust as it has no single point of failure. On the other hand, its communication overhead is higher compared to priority-scheduling. This is represented by the longer termination-time of PQB-Scheduling.

B. Evaluation of PQB-Scheduling With Respect to the Defined Tracking Objectives

In this section, we evaluate the performance of PQB-Scheduling with respect to the tracking objectives as defined in Section III. The snapshot described in Sec. V-A is used to determine the maximum number of traceable objects. The algorithm in Sec. V-B is used to get an impression of the maximum trajectory length.
In Fig. 10 the initial evaluation setup is depicted: The surveillance area is a grid of size 300 by 300 meter. It hosts 25 smart cameras (blue rectangles) and 30 moving objects (red points). The green circles represent the most effective tracking area

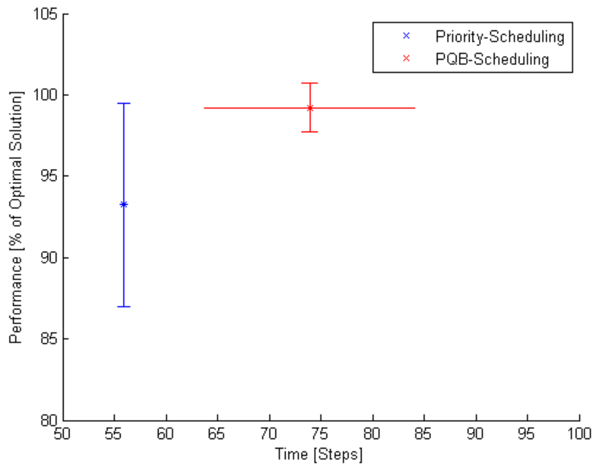

Fig. 9: Evaluation of Priority- and PQB-Scheduling

of each camera's field of view, see [8]. The smart camera's field of view has a length of 50 meter and an viewing angle of 30°. The distance between two cameras is 60 meters. Each smart camera actively tracks one moving object at any point in time. The moving objects follow the random waypoint model introduced by PalChaudhuri et al. [18]. They move 1.4 meters at each simulation step. Each simulation has a duration of 1000 steps and is repeated 10 times with different settings of object movements.
Figure 11 shows the results of the evaluation. The y-axis depicts the number of partial trajectories recorded throughout the simulation. The x-axis represents the longest measured trajectory. As expected, the Longest-Trajectory-Tracking algorithm delivers the longest trajectory. However, the number of partial trajectories captured by the Longest-Trajectory-Tracking algorithm is relatively low compared to the results by the PQB-Scheduling approach. This is consistent with the design focus on the longest predicted trajectory while ignoring of all other objects. The longest trajectory of PQB-Scheduling with $\alpha(T)$ = 1 (see Sec. IV-C) is about 14% shorter, but the number of partial trajectories is about 22% higher. To increase the trajectory length, $alpha$ has to be increased according to the recorded trajectory length. With $\alpha(T)$ = T the longest trajectory is only 10% shorter. To increase the number of detected objects, α has to be decreased with the recorded trajectory length. Therefore, it is set to $\alpha(T)$ = $1/T$. As a result, Fig. 11 shows fewer partial trajectories than using $\alpha(T)$ = 1. This is not contradictory to the previous statement that $\alpha(T)$ = $1/T$ increases the number of detected objects. As a detected object is not immediately discarded if it is temporarily not detected. It is kept in the memory. If it will be detected again within a sufficiently short time, it can be associated with its previous detection. As a consequence, increasing movement of the head of the camera reduces the number of partial trajectories (see also the blue highlighted area in Fig. 3).
Overall, our evaluations show that PQB-Scheduling is adaptable to the tracking-objectives without the calculation costs of the Longest-Trajectory-Tracking algorithm. It also induces a small increase of movement of the camera's head but that is negligible compared to the stress exercised on the pan-tilt-drive by a high-speed moving camera head.

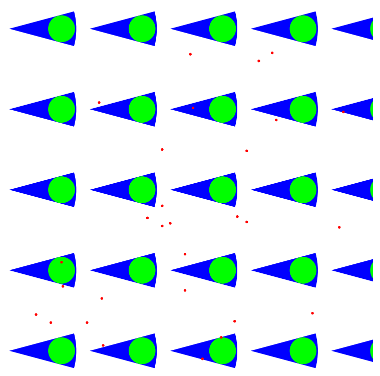

Fig. 10: Initial simulation setup.

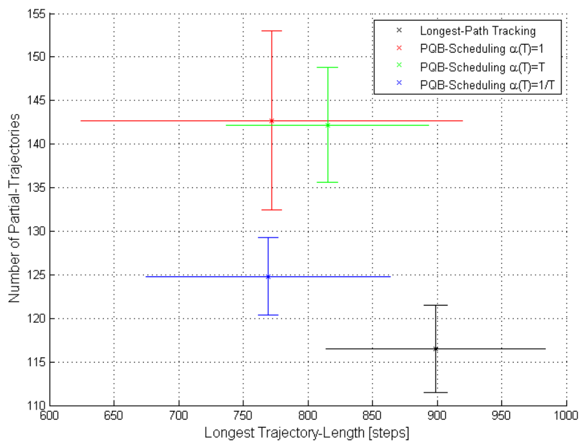

Fig. 11: Evaluation of PQB-Scheduling under constraints of objective 1 and 2.

VIII. Conclusion

This paper introduced a proactive, quotation-based job-scheduling algorithm (PQB-Scheduling) for object tracking in large smart camera networks. PQB-Scheduling reduces the negotiation time that arises during the organization of job-resource pairs by making information for changing resources available to each job before it will be needed. It resolves local dependencies by proactively broadcasting quotations that include information about the current job-resource assignment. In contrast to priority-based algorithms like auctions, local non-optimal solutions are accepted to enhance the system-wide performance. Two competitive tracking objectives are defined with regard to the users point of view. These are a snapshot-view of all persons in the surveillance area and long trajectories for automated analyses. In the evaluation it is shown, that PQB-Scheduling retrieves very good results with respect to the defined tracking objectives. Currently, we are planning to test this scheduling approach in a laboratory demonstrator.

References

[1] E. Monari and T. Pollok, "A real-time image-to-panorama registration approach for background subtraction using pan-tilt-cameras," in *Advanced Video and Signal-Based Surveillance (AVSS), 2011 8th IEEE International Conference on*, 2011, pp. 237–242.

[2] E. Monari, "Color constancy using shadow-based illumination maps for appearance-based person re-identification," in *Advanced Video and Signal-Based Surveillance (AVSS), 2012 IEEE Ninth International Conference on*, 2012, pp. 197–202.

[3] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, 2005, pp. 886–893 vol. 1.

[4] U. Jaenen, C. Paul, M. Wittke, and J. Haehner, "Multi-object tracking using feed-forward neural networks," in *Soft Computing and Pattern Recognition (SoCPaR), 2010 International Conference of*, 2010, pp. 176–181.

[5] U. Jaenen, C. Grenz, C. Paul, and J. Haehner, "Using feed-forward neural networks for data association on multi-object tracking tasks," in *computer Information Systems and Industrial Management Applications (IJCISIM), 2012 International Journal of*, 2012, vol. 4, pp. 383–391.

[6] M. Valera and S. Velastin, "Intelligent distributed surveillance systems: a review," *Vision, Image and Signal Processing, IEE Proceedings -*, vol. 152, no. 2, pp. 192–204, 2005.

[7] C. Grenz, U. Janen, J. Haehner, C. Kuntzsch, M. Menze, D. d'Angelo, M. Bogen, and E. Monari, "Caminsens - demonstration of a distributed smart camera system for in-situ threat detection," in *Distributed Smart Cameras (ICDSC), 2012 Sixth International Conference on*, 2012, pp. 1–2.

[8] U. Jaenen, U. Feuerhake, T. Klinger, D. Muhle, J. Haehner, M. Sester, and C. Heipke, "Qtrajectories: Improving the quality of object tracking using self-organizing camera networks," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. I-4, pp. 269–274, 2012. [Online]. Available: http://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/I-4/269/2012/

[9] R. W. Floyd, "Algorithm 97: Shortest path," *Commun. ACM*, vol. 5, no. 6, pp. 345–, Jun. 1962. [Online]. Available: http://doi.acm.org/10.1145/367766.368168

[10] M. Hoffmann, M. Wittke, Y. Bernard, R. Soleymani, and J. Haehner, "Dmctrac: Distributed multi camera tracking," in *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conference on*, 2008, pp. 1–10.

[11] Q. Ren, J. Li, and H. Gao, "Tpss: A two-phase sleep scheduling protocol for object tracking in sensor networks," in *Mobile Adhoc and Sensor Systems, 2009. MASS '09. IEEE 6th International Conference on*, 2009, pp. 458–465.

[12] S. Ren, Q. Li, H. Wang, and X. Zhang, "Design and analysis of wave sensing scheduling protocols for object-tracking applications," in *Distributed Computing in Sensor Systems*, ser. Lecture Notes in Computer Science, V. Prasanna, S. Iyengar, P. Spirakis, and M. Welsh, Eds., vol. 3560. Springer Berlin Heidelberg, 2005, pp. 228–243.

[13] F. Qureshi and D. Terzopoulos, "Surveillance camera scheduling: a virtual vision approach," *Multimedia Systems*, vol. 12, no. 3, pp. 269–283, 2006. [Online]. Available: http://dx.doi.org/10.1007/s00530-006-0059-4

[14] C. J. Costello, C. P. Diehl, A. Banerjee, and H. Fisher, "Scheduling an active camera to observe people," in *Proceedings of the ACM 2nd international workshop on Video surveillance & sensor networks*, ser. VSSN '04. New York, NY, USA: ACM, 2004, pp. 39–45. [Online]. Available: http://doi.acm.org/10.1145/1026799.1026808

[15] B. Rinner, B. Dieber, L. Esterle, P. Lewis, and X. Yao, "Resource-aware configuration in smart camera networks," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, 2012, pp. 58–65.

[16] B. Dieber, L. Esterle, and B. Rinner, "Distributed resource-aware task assignment for complex monitoring scenarios in visual sensor networks," in *Distributed Smart Cameras (ICDSC), 2012 Sixth International Conference on*, 2012, pp. 1–6.

[17] E. Monari and K. Kroschel, "A knowledge-based camera selection approach for object tracking in large sensor networks," in *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on*, 2009, pp. 1–8.

[18] S. PalChaudhuri, J.-Y. Le Boudec, and M. Vojnovic, "Perfect simulations for random trip mobility models," in *Simulation Symposium, 2005. Proceedings. 38th Annual*, 2005, pp. 72–79.