Performance Evaluation of Multisensor Architectures for Tracking

Stefano Maludrottu, Alessio Dore, Hany Sallam, Carlo S. Regazzoni

Performance Evaluation of Multisensor Architectures for Tracking

Stefano Maludrottu, Alessio Dore, Hany Sallam and Carlo S. Regazzoni

Department of Biophysical and Electronic Engineering - DIBE
University of Genova

Abstract. In recent years the development of distributed fusion systems for real-world services (tracking, surveillance...) has remarked the importance of good design choices for both logical and physical architecture. The main objective of this paper is to implement a model of a distributed fusion architecture in order to evaluate the overall performance as well as the interactions of its subparts. The advantage of the proposed architecture analysis method are: 1) to allow quantitative comparison of different fusion structures through the evaluation of performance metrics and 2) to validate the algorithms involved as well as the physical structure (communication links, computational load...). Finally example structures are compared in order to find the best solution to some benchmark problems.

Keywords: Data Fusion Architecture, Tracking, Performance Evaluation, Multisensor

1 Introduction

Performance evaluation of data fusion systems for tracking is a key task for the correct design of effective and robust architecture. In fact in many applications, as videosurveillance or radar flight tracking, a precise target localization is required over time. In these domains multisensor approaches are successfully employed because of the capability of resolving problems caused by misleading observations, exploiting the redundancy introduced by the possibility of having multiple measurements for the same target. The architecture of these systems is typically complex and composed by many sensors that monitor wide areas and fusion modules providing a unique scene representation. Therefore in this domain several processing units and communications links are involved in this process and their behaviors affect performances.

In this paper a model of a data fusion system for tracking is proposed that takes into account different aspects of such system architectures to assess their performances for what concerns the algorithms accuracy and also to consider communication and computational complexity issues that can arise in complex multisensor systems.

1.1 Previous Works and Motivations

This paper presents a novel modeling method for distributed fusion architecture evaluation. In particular within this domain the problem of performing multi-camera tracking of moving objects using multiple of active sensors is addressed.

In the literature various approaches have been proposed for distributed fusion architecture modeling: in [1] a twofold architecture description is defined on physical and logical level that we take as an inspiration: the physical architecture described as a graph constituted of nodes (processing resources as sensors or fusion nodes) and links (heterogeneous communication channels) while logical architecture is defined outlining a task decomposition (i.e. tracking) into subtasks (i.e. data acquisition, filtering etc.); thus the distribution of intelligence in a fusion system architecture can be defined as a mapping of the subtasks in the various parts of the physical architecture. In [2], a hierarchical architecture is proposed for video surveillance and tracking; the camera network sends the data gathered to the upper level nodes in which tracking data are fused in a centralized manner. In some related works are introduced specific metrics in order to measure the reliability of each subpart involved in the data fusion. The importance of this topic is demonstrated by the relevant body of research evaluating reliability and accuracy in automatic tracking. The vast majority of the works found in literature defines metrics related to specific parts of the architecture missing a more generic analysis. However, during the design of complex data fusion based systems, it is important to consider the influence of the architectural aspects of the system on the overall performances.

The work described in [3] focuses on defining the main requirements for effective performance analysis for tracking methods. In [4], a number of metrics is defined for trajectory comparison, in order to determine how precisely a target is tracked by a computer vision system. In [5], an evaluation benchmark is presented to compare different visual surveillance algorithms developed for PETS (Performance Evaluation of Tracking and Surveillance) workshops. PETS metrics are specifically developed to provide an automatic mechanism to quantitatively compare similar purpose algorithms operating on the same data. In [6] a complex framework is proposed for direct comparison of event detection solutions based on any image analysis and understanding module. In the work presented in [7], the authors develop a tool to automatically generate accurate ground truth of realistic situations in multicamera videosurveillance systems based on a commercial game engine.

2 Data Fusion Frameworks for Tracking

2.1 Data Fusion architecture for Target Tracking

A general data fusion architecture of a system aiming at target tracking can be subdivided into several module in a hierarchic way. The lower level is constituted by sensors that gather information from the environment and try to detect the

targets position within their field of view (i.e the area observable by the sensor). Target tracking can be performed in several scenarios and the typologies of sensors that can be used may change from video cameras to radars, sonars or microphones. Despite the variability of the input data the sensor can be considered as a module that perceives the variations in the environment produced by the target and it converts resulting signals to position information. However, usually in a general situation the single sensor observation of the scene is affected by noise that can produce erroneous localization and tracking.

For example typical problems arising in video data flow are due to the superimposition of targets and elements of the scene occupying different 3D in the image plane. This issue, called *occlusion*, causes a momentary lack of observations with consequent difficulties in the position estimation procedure. Moreover, background elements can produce misleading observations (or clutter) that are to be excluded from the measurements-target association procedures. A possible and widely exploited approach to overcome these issues and to enhance the tracking robustness consists in using more sensors monitoring with complete or partial fields of view overlapping the same area. In this way it is possible to obtain multiple observations of the same entity whose joint processing likely improves tracking performance. The fusion module can be considered as a function that associates in a proper way the data produced by the sensors with overlapped fields of views and estimate an unique scene representation.

An example of the above mentioned architecture is shown in Figure 1(a). Sensors are considered as smart sensors, i.e. with an embedded processing unit and with the capability of performing their local localization and tracking tasks. Fusion nodes gather data from two or more sensors and process them jointly to generate an unique scene representation.

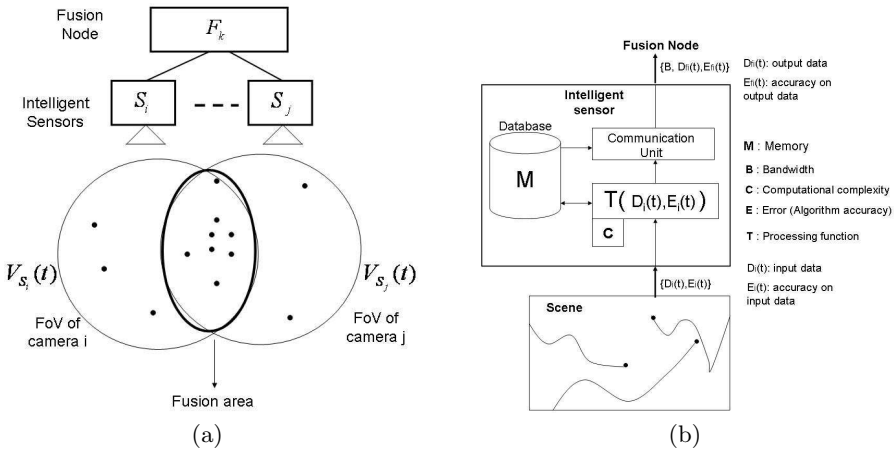

Fig. 1. (a) Example of a data fusion system for target tracking; (b) Representation of the context generator(scene) and intelligent sensor model

In the following a model for smart sensors S_i together with fusion nodes F_k that allows us to define a general architecture of a data fusion system for target tracking. In this work we will consider only a structure where sensors can exchange data only with the fusion node and vice-versa. This structure has been chosen since the focus of this work is on the hierarchical architecture where the fusion node overlooks the sensors and, eventually it modifies their parameters or it passes information gathered from other sensors. This model can be constituted by one or more fusion nodes according to the extension of the monitored area and the total number of sensors. Without loss of generality, we will consider the overall fields of view (FOVs) of the fusion nodes contiguous non-overlapped. In fact, if this is not the case it would be sufficient to introduce a higher fusion level performing the same tasks of the fusion nodes on their outputs.

Performance evaluation of a data fusion architecture does not pertain only on algorithms accuracy in localization but several other aspects must be considered as the data communication between sensors and fusion nodes, the computational complexity and the memory used. Then, in order to define a procedure to assess the performances of this kind of systems a general model of smart sensors and of fusion nodes has been studied that takes into account the elements involved in a multisensor tracking process.

2.2 Smart Sensor Model

Each smart sensor S_i observing a certain area $V_{Si}(t)$ is modeled as composed by three fundamental elements (see Fig.1(b):

- processing function T_i: it represents a tracking algorithm that produces a certain output data $\boldsymbol{D}_{fi}(t)$ given an input data $\boldsymbol{D}_i(t)$ referring to the scene observed in $V_{Si}(t)$. The input noise is modeled with $\boldsymbol{E}_i(t)$ and the noise on the processed data sent to the fusion node is $\boldsymbol{E}_{fi}(t)$. The variables $\boldsymbol{D}_i(t)$, $\boldsymbol{D}_{fi}(t)$, $\boldsymbol{E}_i(t)$ and $\boldsymbol{E}_{fi}(t)$ are vectors of dimension equal to the number of targets $N_i(t)$ detected by S_i at time t. The computational complexity C function of $\boldsymbol{D}_i(t)$ is associated to the processing function.
  The error vector in this tracking application coincides with the accuracy error that has to be evaluated through a suitable performance metric. In this paper, the considered metric is the distance between the estimated target position and the ground truth.
  Another error that must be taken into account in this scenario is the detecting error $E_i^D(t)$ caused by occlusions or clutter. The detection error implies that the estimated number of targets $N_i(t)$ is different to the real one $N_{GTi}(t)$ due to the false alarm or miss detection.
- database: it represents the memory M of the smart sensor. It has been decided to address it as database since in many tracking application algorithms require to compare actual measurements with past detected targets. The algorithm T exchanges information with the database that can also send additional information to the fusion node.

- communication unit: it is responsible for transmitting data to the fusion node. The communication unit has to satisfy the requirements of error free data transmission considering possible bandwidth B limitations of the link toward the fusion node.
- orientation: each smart sensor is considered to be placed in a certain known position; its FOV depends on its orientation. If a smart sensor has a fixed orientation its FOV $V_{Si}(t)$ becomes a constant V_{Si}.

2.3 Fusion Node Model

In this stage the data coming from each sensor are integrated to realize an accurate single model of target position. Its structure similar to the smart sensor model but some differences are present in the processing function and communication unit.

- genetic optimization: genetic algorithms are population based search algorithms [8] used as optimization tools in a lot of different applications. In this paper we implemented a GA-based optimization algorithm to find the best orientations of a group of smart sensors in a distributed multisensor architecture. The input to GA is the current position of moving objects in the environment as perceived by the fusion node and the output is the orientation angle of each camera. The fitness function to be optimized for each sensor separately is formulated as $\sum_i w_i d_i$, where d_i is the perpendicular distance from an object position to the central line of FOV of sensor i and w is the number of objects in a given position. The camera is considered to be best oriented minimizing the sum of distances.
- processing function T_F: the fusion node processing function T_F takes M data $\boldsymbol{D}_{fi}(t)$ (together with the respective error measure $\boldsymbol{E}_{fi}(t)$, $i = 1, \ldots, M$) as input, where M is the number of smart sensors connected to the fusion node F_k. Thus, the transfer function is: $T_F(\boldsymbol{D}_{f1}(t), \ldots, \boldsymbol{D}_{fM}(t), \boldsymbol{E}_{f1}(t), \ldots, \boldsymbol{E}_{fM}(t))$. The cardinality of $D_F(t)$ and $E_F(t)$ vectors is $N_F(t) = T_F(card(\boldsymbol{D}_{f1}(t) \cup \boldsymbol{D}_{f2}(t) \cup \ldots \boldsymbol{D}_{fM}(t)))$ i.e. the intersection of the targets detected by the sensors processed by the fusion algorithm T_F that manages the overlaps and handles the false alarm and miss detection problems. Due to the overlapping fields of view the $\boldsymbol{D}_{fi}(t)$ vectors have common elements, relative to the targets detected by multiple sensors). The output of $T(\boldsymbol{E}_{f1}(t), \ldots, \boldsymbol{E}_{fM}(t))$ produces an error vector $\boldsymbol{E}_F(t)$ in which each value is lower than the respective error value in $\boldsymbol{E}_i(t)$ detected from the i-th sensor. Worth of note is that the computational complexity C of the fusion node processing is related to the dimension of the overlapped fields of view and the number of targets present in that zone.
- database: the database performs the same tasks described for the smart sensor
- communication unit: it analyzes the transmission load for each communication link connecting the fusion node to the smart sensors. It can also send messages to the smart sensor communication unit in order to perform an

adaptive optimization of the data transmission. For example, if one link is overused the fusion node can ask to all the smart sensor connected to that link to reduce their transmission rate by a higher compression.

3 Simulation and Results

3.1 Multisensor Tracking System Simulator

On the basis of the architectural model described in Sect. 2 a tracking simulator has been realized to demonstrate the possibility to evaluate the system performances. A context generator has been designed in order to produce a set of realistic target trajectories simulating moving objects with stochastic dynamics. The simulated smart sensor S_i acquires data from the context generator only for what concerns his field of view $V_i(t)$ and processed them using a Kalman Filter. Multiple sensors with overlapped fields of view send their processed data to one fusion node F_k that processes the data using a modified Kalman Filter in addition to a Nearest Neighbor data association technique [9]. This experiment intends to show how it is possible to adapt a real tracking system into our proposed model.

Context Generator The context generator produces trajectories with linear autoregressive first order model with Gaussian noise. Trajectories lies in a two-dimensional space representing a map of a monitored environment. A Poissonian distribution is used to describe the probability of new target born and death. Occlusions are also simulated when two targets are aligned with respect to the camera line of sight and they are sufficiently close one to another. Clutter noise is also introduced at each scan as a Poisson process.

The context generator allows us to have a realistic but controllable environment by which simulating different situations automatically generating the correspondent ground truth.

Smart sensor

– Processing function: when a target trajectory enters in the field of view $V_i(t)$ of the i-th sensor a single-view tracking task is initialize to permit to know the position and to follow the movements of targets. For each target, in order to estimate the position of tracked objects a Kalman filter is used [10], whose equations are:

$$\boldsymbol{x}(k+1) = \boldsymbol{A}_w \boldsymbol{x}(k) + \boldsymbol{w}_w(k) \tag{1}$$

$$\boldsymbol{z}(k) = \boldsymbol{H}_w \boldsymbol{x}(k) + \boldsymbol{v}_w(k) \tag{2}$$

$\boldsymbol{w}_l$ and $\boldsymbol{v}_l$ are, respectively, the image plane and the measurement noise and they are independent, white Gaussian and zero mean. $\boldsymbol{A}_l$ and $\boldsymbol{H}_l$ respectively are the state transition matrix and the measurement matrix.

In this work the computational complexity is a linear function of the number of targets present in the FOV. When this function exceeds a threshold defined

to represent the unit processing capability a random lack of data to be sent to the fusion node is simulated.
- Communication unit: in this simulator the communication unit of the sensor is responsible to send data to the fusion node according to the fusion node indication, regarding the occupation of the link. A function has been implemented to simulate the optimization of data transmission
- Database: in this simulator the database is not modeled since the performed tasks don't need a data storage

Fusion node

- Communication unit: it handles the data stream coming from the smart sensors. A maximum bandwidth value B_{max} has been defined according to the physical constraints of the architecture. If $\sum_i B_i(k) > B_{max}$ (where B_i is the bandwith occupation of the sensor i at the time k) a sensor subset $S^*(k)$ is chosen in order to optimize the bandwidth usage and such that $\sum_i tB_i^*(k) <= B_{max}$. Various strategies can be implemented to select the sensor subset $S^*(k)$; first we assign to each sensor S_i a priority value p_i and we define the subset of selected sensors equal to an empty set $S^*(k) = \emptyset$; then we will add the sensors one by one, choosing at each step the sensor between the remaining ones with the highest priority value. After the sensor selection a feasibility check is performed; if the bandwidth constraint is satisfied the sensor is added to the selected sensor subset ($S^*(k) = S^*(k) + S_j$).
- Processing function: in the processing function initially the association of measurements with targets is performed; after that these targets are updated and for unmatched measurements a target initialization procedure is applied [9]. Each target $\boldsymbol{x}_t$ is represented by using $\boldsymbol{x}_t(k)$, state covariance $\boldsymbol{P}_t(k)$ and a category estimate $\boldsymbol{e}_t(k)$ for each frame k. The state vector is:

$$\boldsymbol{x} = \begin{bmatrix} x\ y\ \dot{x}\ \dot{y} \end{bmatrix}^T \tag{3}$$

where (x, y) are the ground plane coordinates and $(\dot{x}, \dot{y})$ is the velocity.
In the measurement m_t we include the ground plane position $\boldsymbol{z}_t(k)$ whose covariance is $\boldsymbol{R}_t(k)$, where: $\boldsymbol{z} = \begin{bmatrix} x\ y \end{bmatrix}^T$. Therefore we measure the state on the ground plane using a Kalman filter. The association between tracks is evaluated by using the Mahalanobis distance [11] e define:

$$\boldsymbol{S}_{jt}(k) = \boldsymbol{H}_w \boldsymbol{P}_t(k/k-1)\boldsymbol{H}_w^T + \boldsymbol{R}_j^{(i)}(k) \tag{4}$$

thus, defining $I(k) = \left[\boldsymbol{z}_j^{(i)} - \boldsymbol{H}_w \hat{\boldsymbol{x}}(k/k-1)\right]$:

$$d_{jt}^2 = I(k)^T \boldsymbol{S}_{jt}(k)^{-1} I(k) \tag{5}$$

where $\hat{\boldsymbol{x}}(k/k-1)$ is the target prediction. To determine the association this distance is compared against a threshold and the Nearest Neighbour Algorithm is applied in order to establish the association. We can make the

hypothesis that for each target, in one view, there is at most one measurement, i.e. $\sum_j \beta_{jt}^{(i)} \leq 1$; naturally if the measurement misses then $\sum_j \beta_{jt}^{(i)} = 0$. The overall measurement for each target is the result of single camera measurements weighted with their covariance $\boldsymbol{R}_j^{(i)}$ as follows:

$$\boldsymbol{R}_t = \left[\sum_i \sum_j \beta_{jt}^{(i)} \left(\boldsymbol{R}_j^{(i)} \right)^{-1} \right]^{-1} \tag{6}$$

$$\boldsymbol{z}_t = \boldsymbol{R}_t \left[\sum_i \sum_j \beta_{jt}^{(i)} \left(\boldsymbol{R}_j^{(i)} \right)^{-1} \boldsymbol{z}_t^{(i)} \right] \tag{7}$$

These equations express the accuracy improvement obtained thanks to the fusion of all the views; as a matter of fact Equation 6 implies that the overall uncertainty is lower than the one provided by a single camera. Moreover Equation 7 ensures that the fused measurement is biased to the most accurate measurement from one camera. The target updating is, then, attained in the following way:

$$\hat{\boldsymbol{x}}_t(k/k) = \hat{\boldsymbol{x}}_t(k/k-1) + \boldsymbol{K}_t(k) \left[\boldsymbol{z}_t(k) - \boldsymbol{H}_w \hat{\boldsymbol{x}}_t(k/k-1) \right] \tag{8}$$

$$\boldsymbol{K}_t(k) = \boldsymbol{P}_t(k/k-1) \boldsymbol{H}_w^T \left[\boldsymbol{H}_w \boldsymbol{P}_t(k/k-1) \boldsymbol{H}_w^T + \boldsymbol{R}_t(k) \right]^{-1} \tag{9}$$

its covariance is: $\boldsymbol{P}_t(k/k) = \left[\boldsymbol{I} - \boldsymbol{K}_t(k) \boldsymbol{H}_w \right] \boldsymbol{P}_t(k/k-1)$ with $0 < \eta < 1$.
For those measurements not matched to any existing targets a target initialization procedure has to be applied. Hence, every unmatched measurement from a camera must be checked against the other unmatched measurements from other cameras to find new targets.
We call $\boldsymbol{z}_{j^*}^{(i^*)}$ the unmatched measurement from the i^*th camera and x_n the new target associated with it. All the association matrices $\beta^{(i)}$ are extended by one column, $\beta_{jn}^{(i)}$, for the new target x_n; its elements represent the association between the jth measurement from ith camera and the new target. For the i^*th camera the elements of the columns nth, referring to the new target, are:

$$\beta_{jn}^{(i^*)} = \begin{cases} 1 & \text{if } j = j^* \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

Then the Mahalanobis distance between $\boldsymbol{z}_{j^*}^{(i^*)}$ and each unmatched measurement $\boldsymbol{z}_j^{(i)}$ from ith camera is calculated:

$$d_{i^*j}^2 = \left[\boldsymbol{z}_j^{(i)}(k) - \boldsymbol{z}_{j^*}^{(i^*)} \right]^T \left[\boldsymbol{R}_j^{(i)} - \boldsymbol{R}_{j^*}^{(i^*)} \right]^{-1} \left[\boldsymbol{z}_j^{(i)}(k) - \boldsymbol{z}_{j^*}^{(i^*)} \right] \tag{11}$$

where $\boldsymbol{R}_j^{(i)}$ and $\boldsymbol{R}_{j^*}^{(i^*)}$ are the measurement covariances. Then, the new target is initialised as follows:

$$\hat{\boldsymbol{x}}_n(k/k) = \left[\boldsymbol{z}_n(k)^T \; 0 \; 0 \right]^T \tag{12}$$

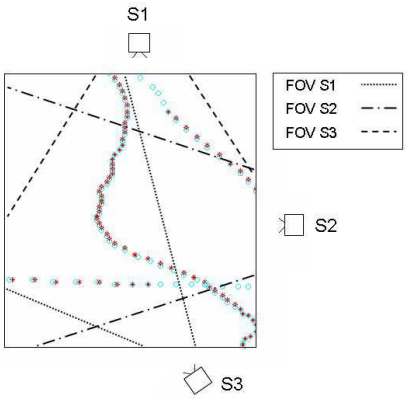

Fig. 2. Representation of the ground truth ('o') and fused target tracking ('*')

$$\boldsymbol{P}_n(k/k) = \begin{bmatrix} \boldsymbol{R}_n(k) & \boldsymbol{O}_2 \\ \boldsymbol{O}_2 & \sigma_v^2 \boldsymbol{I}_2 \end{bmatrix} \tag{13}$$

where $\boldsymbol{O}_2$ and $\boldsymbol{I}_2$ are 2×2 zero and identity matrices, respectively.
The processing capability of the fusion node is limited due to the hardware specific processing resources (CPU, memory...). Similarly to the communication unit we define a maximum CPU load C_{max}; each of the $S_i \subset S^*(k)$ sensors (output of the communication unit at a given time k) produces a CPU usage $C_i(k)$. If $\sum_i C_i(k) >= C_{max}$ we define a priority-based selection of the data streams in order to obtain a feasible subset $S^{'}(k)$ such that $C_i^{'}(S_i^{'}(k)) <= C_{max}$. This selection will cause a loss of input data in order to preserve the computational feasibility; thus the data fusion will be performed using only a subset of the whole input data to the fusion node.

– Database: in this simulator the database is not modeled since the performed tasks don't need a data storage

3.2 Experimental results

Three scenarios have been investigated in order to demonstrate the possibility of fusion architecture evaluation using the proposed method. The performance metric is defined as

$$perf = \frac{\sum_i |f_i - g_i| + w \cdot m_i}{TOT_i} \tag{14}$$

where $|f_i - g_i|$ is the euclidean distance between the target position $f_i$and its corresponding ground truth g_i at timestep i. The term w is a penalty value set equal to 10, m_i the number of missing associations between ground truth data and fused tracks at timestep i and TOT_i is the total number of timesteps of the generated context (see Figure 2).

First scenario Two alternative fusion architectures are defined to be evaluated using the proposed method in order to find which one has better performances. The overall structure (5 smart sensors, 1 fusion node and wireless

communications) is maintained fixed and only the sensors position has been changed (always ensuring the complete coverage of the context area) to find the best configuration. In the first structure the field of view of the smart sensors does not change over time, while in the second structure the field of view dynamically change according to the genetic-based optimization technique defined in Sect. 2.3. Ten different scenes with a fixed number of trajectories (10) have been provided by the context generator as input for the model. Parameters of GA are inizialized as follows: population size of 20, number of generations equal to 100 and crossover and mutation rate respectively 0.8 and 0.04. There are many parameters that influence the context generator algorithm behavior: the total number of trajectories, the generation rate, the Gaussian noise (i.e. measurement error in smart sensors), etc. Those parameters have been randomly modified for each context generation in order to provide a ground truth as diversified as possible. Table 3 presents the different results (obtained using the metric defined in Eq. 14)of the sample architectures and shows the connection between different camera placement strategies and performances: dynamic changing camera orientation outperforms a fixed orientation solution up to 20%.

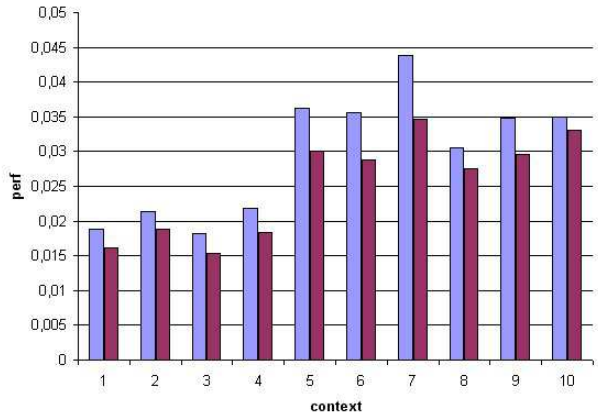

Fig. 3. Performance comparison of Architecture 1 (fixed camera orientation - in blue) and Architecture 2 (genetic-based algorithm for optimization of camera orientations - in violet)

Second scenario Focusing on the architectures defined in the previous test, the overall result of the data fusion has been compared with the tracks produced by each smart sensor in the architecture in order to point out the contribution of each one to the performance of the fusion architecture. As shown in Table 1 sensor 3 has the worst performance: it acts like a bottleneck for the entire structure; a better placement or perhaps a more efficient sensor could improve

Cont	Node	S1	S2	S3	S4	S5
1	0.0188	0.16	0.11	0.61	0.02	0.03
2	0.0214	0.11	0.19	0.65	0.02	0.03
3	0.0182	0.15	0.12	0.60	0.03	0.03
4	0.0218	0.14	0.22	0.69	0.03	0.02
5	0.0362	0.17	0.2	0.98	0.1	0.05
6	0.0356	0.17	0.28	1.04	0.08	0.06
7	0.0439	0.14	0.26	1.17	0.09	0.08
8	0.0306	0.13	0.15	0.75	0.04	0.07
9	0.0348	0.18	0.19	0.68	0.04	0.11
10	0.035	0.12	0.17	0.68	0.04	0.09
Avg	0.0296	0.15	0.19	0.78	0.05	0.06

Cont	Node	S1	S2	S3	S4	S5
1	0.0162	0.12	0.10	0.55	0.03	0.03
2	0.0189	0.11	0.17	0.58	0.02	0.04
3	0.0153	0.11	0.10	0.49	0.02	0.03
4	0.0184	0.11	0.18	0.58	0.03	0.04
5	0.03	0.13	0.14	0.67	0.08	0.05
6	0.0288	0.19	0.21	0.81	0.07	0.06
7	0.0347	0.14	0.23	0.8	0.06	0.05
8	0.0275	0.11	0.15	0.65	0.03	0.05
9	0.0296	0.13	0.15	0.52	0.04	0.10
10	0.033	0.11	0.21	0.55	0.04	0.11
Avg	0.0252	0.13	0.16	0.62	0.04	0.06

Table 1. (a) Performance comparison of subparts of Architecture 1(fixed camera orientation); (b) Performance comparison of subparts of Architecture 2 (genetic-based algorithm for optimization of camera orientations). Cont refers to the context used as input. Avg is the average performance over ten experiments. Performances are evaluated using the metric defined in Eq. 14.

the performance of the entire structure.

Third scenario Two different fusion structures have been compared: architecture A has five sensors, an average communication link and average computational resources while architecture B has only three sensors with wider fields of view (60° instead of 45°) and a more accurate target detection and tracking, a better communication link and better fusion node hardware. 10 different scenes with a fixed number of trajectories (10) have been provided as input for the model. In Table 2 is shown that architecture A, even with worse hardware obtains a better overall result thanks to the higher number of sensor placed on the area.

4 Conclusions and Future Works

In this paper a model of a system architecture for multisensor tracking has been proposed. The aim of this model is to represent basic tasks performed by such a system in order to address the problem of performance evaluation. This description allows one to validate algorithms and system design with respect to the monitored scenario. A simulator of a multisensor hierarchic tracking system has been presented to show the possibility of using the model as a tool to assess system architectures and algorithms.
Future works will be devoted to apply the model to decentralised fusion architectures and to realise a more accurate model of different tipologies of smart sensors.

Context	Arch. A	Arch. B
1	0.0254	0.0303
2	0.0198	0.023
3	0.0238	0.0316
4	0.0176	0.0201
5	0.0164	0.0271
6	0.0176	0.0.275
7	0.0258	0.0295
8	0.022	0.0278
9	0.0178	0.025
10	0.0312	0.0404
Average Perf.	0.0217	0.0282

Table 2. Performance comparison of different fusion architectures. Performances are evaluated using the metric defined in Eq. 14.

References

1. Marcenaro, L., Oberti, F., Foresti, G.L., Regazzoni, C.S.: Distributed architectures and logical-task decomposition in multimedia surveillance systems. Proceedings of the IEEE **89** (2001) 1419–1440
2. Micheloni, C., Foresti, G., Snidaro, L.: A network of cooperative cameras for visual-surveillance. IEE Visual, Image & Signal Processing **152** (2005) 205–212
3. Ellis, T.: Performance metrics and methods for tracking in surveillance. In: PETS 2002, Copenhagen. (2002)
4. Needham, C.J., Boyle, R.D.: Performance evaluation metrics and statistics for positional tracker evaluation. In: Computer Vision Systems, Third International Conference, ICVS 2003. (2003)
5. Young, D., Ferryman, J.: PETS metrics: On-line performance evaluation service. In: VS-PETS 2005. (2005)
6. Ziliani, F., Velastin, S., Porikli, F., Marcenaro, L., Kelliher, T., Cavallaro, A., Bruneaut, P.: Performance evaluation of event detection solutions: the creds experience. In: AVSS '05, Como, Italy (2005)
7. Taylor, G.R., Chosak, A.J., Brewer, P.C.: Ovvv: Using virtual worlds to design and evaluate surveillance systems. In: VS 2007, Minneapolis, MN, US (2007)
8. Holland, J.H.: Adaptation in natural and artificial systems. MIT Press (1992)
9. Xu, M., Orwell, J., Lowey, L., Thirde, D.: Architecture and algorithms for tracking football players with multiple cameras. IEE Proceedings - Vision, Image and Signal Processing **152** (2005) 232–241
10. Kalman, R.E.: A new approach to linear filtering and prediction problems. Transaction of the ASME - Journal of Basic Engineering (1960) 35–45
11. Mahalanobis, P.: On the generalized distance in statistics. Proceedings of the National Institute of Science of India **12** (1936) 49–55