Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

A structure-based approach for optimizing distributed reconstruction in Motion Capture systems

Andrea Masiero * **Angelo Cenedese** **

* *CIRGEO, Università di Padova, Viale dell'Università 16, 35020 Legnaro (PD), Italy (e-mail: masiero@dei.unipd.it).*
** *Dipartimento di Ingegneria dell'Informazione, Università di Padova, via Gradenigo 6/B, 35131 Padova, Italy (e-mail: angelo.cenedese@unipd.it)*

Abstract:
The diffusion of visual sensor networks, and in particular of smart camera networks, is motivating an increasing interest on the research of distributed solutions for several vision problems. Specifically, in this paper we propose a distributed solution to the problem of reconstructing target positions in large Motion Capture (MoCap) systems. Real time reconstruction by means of centralized procedures is practically unfeasible for very large systems, while the use of distributed computation allows to significantly reduce the computational time required for reconstruction, thus allowing the development of real time solutions.
Then the proposed distributed reconstruction procedure is optimized by exploiting information about the structure of the system: the *visibility matrix* states which objects in the scene are somehow measurable by a sensor (sensor-object matrix). Often, the typical localization of data from real application scenarios induces an underlying structure on the visibility matrix, that can be exploited to improve the performance of the system in understanding the surrounding environment. Unfortunately, usually these data are not properly organized in the visibility matrix: for instance, listing the sensors in a pseudo-random order can hide the underlying structure of the matrix. This paper considers the problem of recovering such underlying structure directly from the visibility matrix and designs an algorithm to perform this task.
Our simulations show that the distributed reconstruction algorithm optimized by means of the estimation of the structure of the visibility matrix achieves an important computational time reduction with respect to the standard (centralized) reconstruction algorithm.

1. INTRODUCTION

Recent advancement of mobile electronic devices and in particular of cheap mobile sensors are motivating an increasing interest on sensor networks for both research and industrial applications. In particular, several applications based on Visual Sensor Networks can be listed in the last decade (Rinner and Wolf [2008], Soro and Heinzelman [2009], Cucchiara et al. [2007]).

Among the possible types of Visual Sensor Networks, this paper considers a motion capture (MoCap) system (Wieber et al. [2006]) composed by a large number of cameras m (e.g. tens to hundreds) and of targets n (e.g. hundreds to thousands). The goal of the MoCap system is that of reconstructing the 3D positions of the targets in real time. This task is practically unfeasible when m and n are large and by using a centralized management of the information, where computations are performed by means of a single computational unit. Instead, as commonly done in recent works based on smart cameras (Rinner and Wolf [2008], Jovanovic et al. [2006], Liu et al. [2009], Klausner et al. [2008], Aghajan et al. [2008]), here each camera is assumed to be provided with computational and memory resources: the first goal of the paper is that of proposing an efficient strategy to distribute the computation among the available computational resources.

Several works in the literature have been proposed to merge information from different sensors and to parallelize such computation on a Visual Sensor Network (Falcou et al. [In ParCo, 2005], Tron et al. [2008], Franco et al. [2004], Goesele et al. [2007], Furukawa et al. [2010]). In Section 2, a slightly modified version of the classical 3D reconstruction procedure is considered (Hartley and Zisserman [2003], Hartley and Sturm [1997], Triggs et al. [1999]), and adapted to obtain fast reconstruction distributed on the network.

Furthermore, Section 3 aims at optimizing the reconstruction procedure by taking advantage of the system structure and sparseness: a wide range of applications involve large sparse (spatially distributed) systems (Chiuso et al. [2010], Narayanaswamy [2011]), ranging from sensor networks, automation (Bullo et al. [2009]), computer vision (Aghajan and Cavallaro [2009], Masiero and Cenedese [2012]), and so on. The very large size of such systems makes the use of full models practically unfeasible. Possible methods to manage such complexity typically exploit distributed strategies and the system sparsity. The mathematical rep-

resentation of large systems is usually given through sparse matrices (Tewarson [1973]) and graphical models that aim at capturing the structure underlying the system, since in these large sparse systems data are typically localized (i.e. position of sensors and the measured objects are correlated). A good knowledge of such structure can be exploited to improve the system performance (Section 4 and Chiuso et al. [2010]), but, unfortunately, most of the times data are not organized is such a way to make such structure evident.

In this paper a visibility matrix representation of the system is considered: such a matrix summarizes what are the objects measurable by each sensor (e.g.: in a camera network, the actual visibility relation between cameras and objects in the scene). The algorithm proposed in Section 3 aims at efficiently recovering the underlying structure from the visibility matrix U. Equivalently, this can be formulated as the problem of finding proper row and column permutation matrices P_R and P_C such that $P_R U P_C$ is a structured matrix. In this sense, the proposed approach is similar to the Cuthill-McKee algorithm (Cuthill and McKee [1969], George and Liu [1981], Golub and Loan [1989]) for reducing the bandwidth of a matrix. However, the proposed algorithm is more flexible and based on a robust similarity measure.

Our results, in Section 4, show that applying the reconstruction procedure to the system obtained by exploiting the structure information significantly reduces the required computational time.

2. CENTRALIZED AND DISTRIBUTED RECONSTRUCTION OF THE 3D POSITION OF TARGETS

Let a target i be placed at position ϕ_i in the 3D space and assume that camera j_1 has a local measurement of i on its image plane at position q_{ij_1}. Then, using this measurement it is possible to say that i is (approximately) on a point along the line r_{ij_1} passing through q_{ij_1} and the optical center of camera j. Let also r_{ij_2} be the ray associated to another camera $j_2 \neq j_1$ and the same target i, then the rationale is that the target position $\bar{\phi}_i$ can be estimated by intersecting r_{ij_1} and r_{ij_2} (geometric triangulation). The described procedure allows to pose the problem of the reconstruction of target positions as a geometric problem. It can also be adapted to deal with the multi-camera case and to take into account of the geometric errors on camera image planes (Hartley and Sturm [1997], Hartley and Zisserman [2003], Triggs et al. [1999]): in practical cases, to reduce the effect of noise one has to check the measurements with the best match.

In a classical reconstruction scheme, the matching of measurements is concurrently checked on all the cameras (Hartley and Sturm [1997], Hartley and Zisserman [2003]). However, here some modifications are introduced to make the problem manageable when dealing with a large number of cameras and targets. Similarly to (Goesele et al. [2007], Furukawa et al. [2010]) only small groups of cameras are used at the beginning (here we exploit pairs of cameras to compute candidate targets), while the obtained results are assessed and refined using all the available data. Given m cameras and n targets, the strategy can be summarized as follows:

- *matching*: For each pair of cameras (j_1, j_2), with $j_1 = 1, \ldots, m$ and $j_2 = j_1 + 1, \ldots, m$, the available 2D measurements from j_1 are compared with 2D measurements from j_2, searching for possible 3D real points through a geometric triangulation procedure;
- *back-projection*: When a pair of measurements from (j_1, j_2) is compatible, then the reconstructed point ϕ_i (potentially a real target position) is back-projected onto the other cameras image planes;
- *reconstruction*: Let k be the number of measurements from different cameras that are compatible with ϕ_i. If k is larger than a chosen threshold $\bar{k}$, ϕ_i is recomputed by using all the k measurements to produce the estimate of the real target position $\hat{\phi}_i \approx \bar{\phi}_i$. The used measurements are deleted from the list of measurements available for new target search.

This algorithm, considered as a *centralized* approach (i.e. executed on a single machine), shows a computational complexity increasing approximatively linearly with m and n, thus making this procedure not suitable for large scale scenarios. Conversely, if adapted to work in a *distributed* fashion on the computational grid formed by the camera network, a huge speed up of the whole reconstruction can be achieved.

The rationale is as follows: if all the cameras are provided with computational power, then (most of) the computations can be parallelized and distributed on the cameras. Furthermore, cameras that allow to reconstruct more targets has to be matched first, in such a way that most of the 2D measurements are deleted quickly (in the first step of the reconstruction procedure): this camera matching optimization, based on the estimation of the structure of the visibility matrix, will be considered in Section 3.

The method adopted in this work to distribute the computational load, namely the proposed *distributed reconstruction algorithm*, is formed by $m/2$ steps. At the sth step (for $s = 1, \ldots m/2$) the j_1th camera (for each j_1) execute the 3 reconstruction steps of the centralized procedure with camera $j_2 = j_1 + s$. The matching and the reconstruction step can be executed in parallel by all the couples of cameras. Instead, during the back-projection step couples of cameras have to be considered in sequential order.

3. ESTIMATION OF THE STRUCTURE OF THE VISIBILITY MATRIX

This section considers the problem of optimizing the camera matching order in the distributed reconstruction algorithm. The rationale is that the optimal camera matching order should depend on the visibility of targets from cameras and of the quality of reconstruction (i.e. let target i be visible by cameras j_1 and j_2, r_{ij_1} and r_{ij_2} be the rays associated by such two cameras to target i and let the distance between the target and cameras be fixed, then the maximal information about the target position is provided when the two cameras are such that r_{ij_1} and r_{ij_2} are orthogonal).

Let the visibility matrix U at a specific time t be defined as follows: $U(i,j) = u_{ij}$, $i = 1,\dots,n$, $m = 1,\dots,m$, and $u_{ij} = 1$ if target i is visible by sensor j, while $u_{ij} = 0$ otherwise.

Since the visibility matrix usually changes slowly with respect to the sampling rates of the systems, here the problem is considered from a static point of view, i.e. the visibility matrix U is considered at a fixed time instant, and assumed invariant during the process of interest [1].

In real applications, involving large distributed systems, data are typically localized, i.e. there is a correlation between sensor displacement and the measured data. However, since rows and columns in U typically appears in random (or non-optimized) order, such correlation cannot be see in the matrix U. Then the aim of this section is that of estimating proper permutation matrices P_R and P_C such that $\hat{U} = P_R U P_C$ is matrix as structured as possible, i.e. it "estimates" the underlying structure in the visibility matrix.

In particular, cameras that allows to reconstruct the position of a large number of targets will be close to each other in $\hat{U}$. Thus, applying the distributed reconstruction algorithm presented in Section 2 by considering the cameras ordered as in $\hat{U}$ aims at matching cameras that allows to reconstruct more targets first.

3.1 The algorithm

Algorithm 3.1 presents a procedure for estimating $\hat{U}_C$ (equivalently the algorithm can be slightly modified to compute the permutation matrix P_C, such that $\hat{U}_C = UP_C$), such that its columns are properly sorted to make evidence of structure in U. Then, applying the same algorithm to the rows of U_C it is possible to obtain the (typically quite structured) matrix $\hat{U}$.

Since the Algorithm 3.1 is applied in sequence to rows and columns of U, in the following we present it referring simply to nodes in U.

Let A be the set of available nodes, that is the set of nodes not already inserted in $\hat{U}$.

Algorithm 3.1 starts randomly picking a node j_1 in U and initializing $\hat{U} = U(:, j_1)$. Furthermore, it computes the set S containing the nodes candidate to be inserted at the next step. A node j is in S if it is available, $j \in A$, and j is the most similar to a node in S, or a node in S is the most similar to j.

Then, the algorithm iteratively inserts nodes in $\hat{U}$. At each step, the node to be inserted is chosen using the following criteria:

- If the candidate set S is not empty, then the node j in S with maximum similarity to the nodes at the left or at the right of $\hat{U}$ is inserted.

[1] In real applications the estimated visibility matrix should be periodically updated, however the results of our simulations show that the estimated visibility matrix can be considered as constant for several sampling periods without significantly affecting the system performance.

Alg. 3.1 Estimation of the structure in U

Initialize the set of available nodes: $A = \{1, 2, \dots, m\}$.
Randomly pick a column j_1 and set $\hat{U}_C(:,1) = U(:, j_1)$.
Update A: $A = A \setminus j_1$.
Compute $S_{j_1} = \{j | j \text{ is available and } j_1 = j_{\max}(j)\}$.
Initialize the set of candidate nodes: $S = j_{\max}(j_1) \cup S_{j_1}$.
for $j = 2 : m$
  if $S \neq \emptyset$
    Pick j_1 that maximizes the similarity
    $\max(w_L(j_1), w_R(j_1))$ among the candidate nodes in S.
  else
    Pick j_1 that maximizes the similarity
    $\max(w_L(j_1), w_R(j_1))$ among the available nodes A.
  end
  if $w_L(j_1) \geq w_R(j_1)$
    Insert the column j_1 to the left of $\hat{U}_C$.
  else
    Insert the column j_1 to the right of $\hat{U}_C$.
  end
end
Update A: $A = A \setminus j_1$.
Compute $S_{j_1} = \{j | j \text{ is available and } j_1 = j_{\max}(j)\}$.
Update S: $S = (S \cup j_{\max}(j_1) \cup S_{j_1}) \cap A$.

- If the candidate set S is empty, then the node j in A with maximum similarity to the nodes at the left or at the right of $\hat{U}$ is inserted.

It appears from this description how the similarity notion between nodes is fundamental for the correct working of the proposed algorithm. In this context, the similarity score between the columns j_1 and j_2 of matrix U is defined as follows:

$$w(j_1, j_2) = -\sum_{i=1}^{n} |U(i, j_1) - U(i, j_2)| \ , \qquad (1)$$

where the sum in $w(j_1, j_2)$ counts the dissimilarities between column j_1 and j_2: hence this score aims at minimizing the dissimilarities between adjacent nodes in $\hat{U}$.

Let $\bar{d}$ be a proper neighborhood size, then we define the similarity score of an available column j_1 with left (and right, respectively) columns in $\hat{U}$ at step t (i.e. $\hat{U}$ already contains $t-1$ columns) as:

$$w_L(j_1) = \sum_{j_2=1}^{\bar{d}} w(j_1, \phi(j_2)) \ ,$$

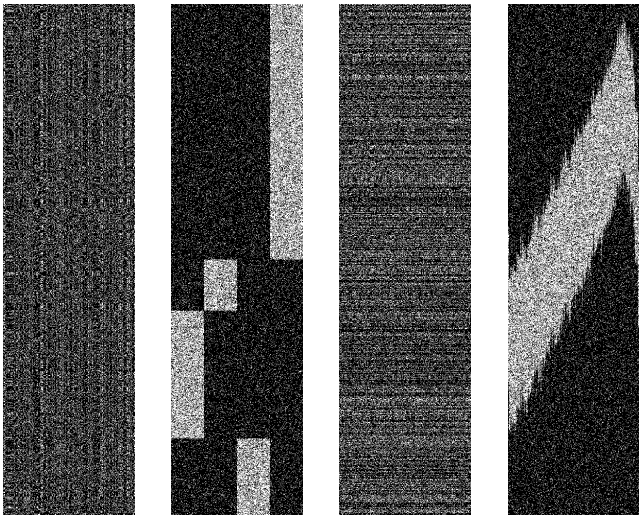
$$w_R(j_1) = \sum_{j_2=1}^{\bar{d}} w(j_1, \phi(t - j_2)) \ ,$$

where the function $\phi(j_2)$ is such that $\hat{U}(:, j_2)$ corresponds to the column $U(:, \phi(j_2))$. The net effect of averaging the similarity score over $\bar{d}$ columns is that of increasing the robustness of the algorithm, by reducing the influence of outliers.

Furthermore, the function $j_{\max}(j)$ indicates the node most similar to j:

$$j_{\max}(j) = \arg\max_{j_2}(w(j, j_2)) \ .$$

Algorithm 3.1 uses also the set S_j, that is closely related to $j_{\max}(j)$: S_j is the (possibly empty) set of



(a) (b) (c) (d)

Fig. 1. (a) Visibility matrix $\bar{U}$ taken as input by Algorithm 3.1 (columns and rows are randomly permuted). (b) $\hat{P}_R\hat{U}$, where $\hat{U}$ is the output of Algorithm 3.1. (c) Visibility matrix $\bar{U}$ taken as input by Algorithm 3.1 (columns and rows are randomly permuted). (d) $\hat{P}_R\hat{U}$, where $\hat{U}$ is the output of Algorithm 3.1.

columns that have j has most similar node: $S_j = \{\bar{j}|\bar{j}$ is available and $j = j_{\max}(\bar{j})\}$.

The similarity score (1), and the definitions of $w_L(\cdot)$, $w_R(\cdot)$, and $j_{\max}(j)$ can be easily modified to take into account of the similarities between columns of U.

Notice that Algorithm 3.1 is similar to the Cuthill–McKee algorithm for matrix bandwidth reduction Cuthill and McKee [1969], George and Liu [1981]. However, it considers a different similarity measure (1) with respect to the Cuthill–McKee algorithm. Furthermore, to improve the ability in reconstructing the proper structure of U, it allows to insert columns both at the left and at the right of $\hat{U}$.

Fig. 1 shows the results of the algorithm when applied to two sample visibility matrices. These matrices are visibility matrices from large network camera systems, where $U(i,j)$ is equal to 1 if the object or feature i is visible by camera j. In the case of Fig. 1(a)-(b) the data are mostly grouped in 4 clusters, while in Fig. 1(c)-(d) the cameras are approximately displaced along a circle (so the set of visible features have relatively few changes when considering close cameras, and viceversa).

Fig. 1(b) and (d) show the output of Algorithm 3.1 when it takes as input Fig. 1(a) and (c), respectively. As shown in Fig. 1(b), the proposed algorithm correctly estimates the structure of the block clustered matrix, while Fig. 1(d) shows the ability of approximately reconstructing the structure of U also when the data are not strictly clustered.

3.2 Analysis of the clustered visibility matrix case

In order to take into account of a quite clustered scenario, in this subsection we consider the case of visibility matrices that are reducible by means of row and column permutations to the sum of a block diagonal matrix with a sparse matrix. Actually, the sparse matrix takes into account of the off-block diagonal elements, which are assumed to be much fewer than the block diagonal elements.

Specifically, consider the following model for the U matrix: $U = P_R\bar{U}P_C$, where P_R, P_C are permutation matrices, while $\bar{U}$ is as follows,

$$\text{Prob}(\bar{U}(i,j)=1) = \begin{cases} \alpha \text{ if } (i,j) \text{ is in a block,} \\ \beta \text{ otherwise,} \end{cases} , \quad (2)$$

where $\alpha \gg \beta$. Furthermore, let $\bar{U}(i,j)$ be independent [2] on $\bar{U}(i_2,j_2)$, for $(i,j) \neq (i_2,j_2)$.

For simplicity of exposition let all the clusters have size $z \times z$, and let $n \geq m$: Then the worst case (i.e. that where off block diagonal elements have more influence) for the algorithm is when elaborating the columns of U.

Let j_1 and j_2 be two columns in the same block in $\bar{U}$, whereas j_3 is in a different block, and let ξ be defined as follows:

$$\xi = \sum_j \sum_{i=1}^{n} -|U(i,j_1)-U(i,j_3)|+|U(i,j_1)-U(i,j_2)| , \quad (3)$$

where j ranges among the neighborhood of j_1, and, for simplicity, j is assumed to be in the same block of j_1.

According with the similarity function (1) and with the use of w_L, w_R with neighborhood size $\bar{d}$, the probability that j_3 is more similar than j_2 to j_1 can be upper bounded by means of the one-sided Chebyshev inequality (Cantelli's inequality, Papoulis [1965]):

$$\text{Prob}(\xi > 0) = \text{Prob}(\xi - \bar{\xi} > -\bar{\xi}) \leq \frac{\sigma_\xi^2}{\sigma_\xi^2 + \bar{\xi}^2} , \quad (4)$$

where $\text{Prob}(\xi > 0)$ corresponds to the probability that j_3 is more similar than j_2 to j_1.

Since the variables in U are assumed to be independent, then mean and variance of ξ are:

$$\bar{\xi} = \bar{d}n \ \mathbf{E}[w_{i,j_1,j_2,j_3}] , \quad (5)$$

$$\sigma_\xi^2 = \bar{d}n \ \sigma^2_{w_{i,j_1,j_2,j_3}} , \quad (6)$$

where

$$w_{i,j_1,j_2,j_3} = -|U(i,j_1)-U(i,j_3)|+|U(i,j_1)-U(i,j_2)| . \quad (7)$$

Mean, $\bar{\xi}$, and variance, σ_ξ^2,, of w_{i,j_1,j_2,j_3} can be easily computed from the equations above and from the statistical model of the visibility matrix, (2).

By applying (4) it is possible to determine a numerical upper bound to the wrong matching of a column j_1. For instance, let $n = 1000$, $z = 60$, $\alpha = 0.95$, and $\beta = 0.01$, then $\text{Prob}(\xi > 0) \leq 0.0021$ for $\bar{d} = 5$.

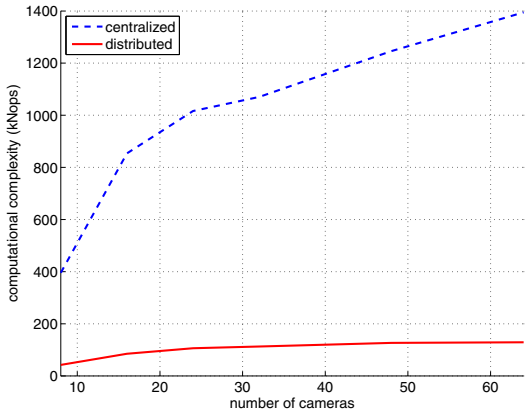
As a final consideration, the algorithm can also be used as a stand alone clustering algorithm: The similarity measure (1) can be computed on consecutive nodes of $\hat{U}$, then the links that obtain the minimum scores are selected to be the boundaries between clusters.

[2] This assumption is unrealistic for real systems, however it simplifies the statistical analysis of the model. Furthermore, since the considerations in this subsection are done for a generic unknown system, it is difficult to introduce a more realistic assumption.

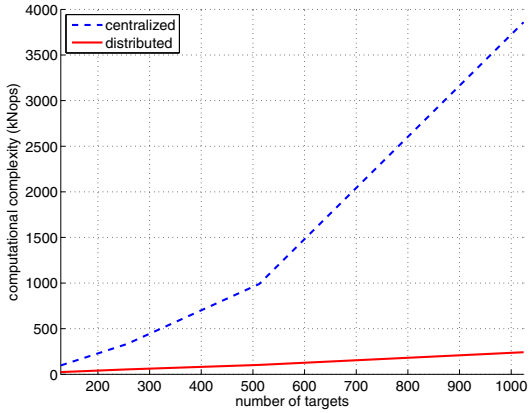
4. RESULTS

This Section compares the computational complexities of the centralized reconstruction algorithm, of the non-optimized distributed and of the optimized distributed algorithm in a case study. In order to make the comparison independent on the specific used devices, the results are reported in terms of number of elementary mathematical and communication operations. For simplicity, the cost of each elementary mathematical and communication operation is assumed to be unitary, however, similar results can be obtained also for different values of such costs. In all the considered comparisons the sample computational complexities are computed as the mean of 200 independent reconstructions.

First, in Fig. 2 we compare the computational complexity of the proposed centralized and (non-optimized) distributed reconstruction algorithm. The comparison is done in varying both the number of cameras (m ranges from 8 to 64, Fig. 2(a)) and the number targets (n ranges from 64 to 1024, Fig. 2(b)).



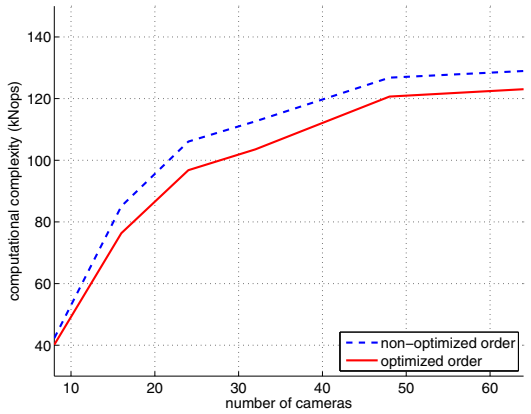
(a)



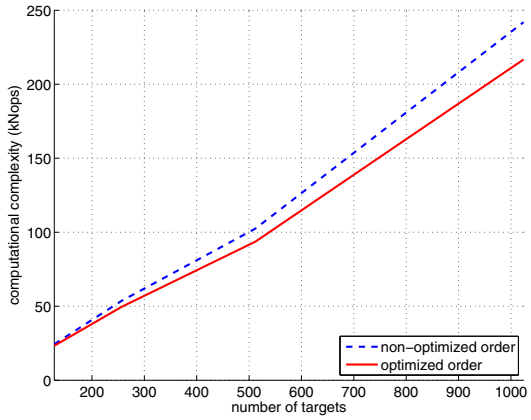
(b)

Fig. 2. Comparison of the computational burden for static reconstruction by means of the centralized (blue–dashed line) and of the distributed (red line) algorithm varying (a) the number of cameras, and (b) the number of targets.

Then, in Fig. 3 we compare the computational complexity of the optimized and non-optimized distributed reconstruction algorithm. The comparison is done in varying both the number of cameras (m ranges from 8 to 64, Fig. 3(a)) and the number targets (n ranges from 64 to 1024, Fig. 3(b)).



(a)



(b)

Fig. 3. Comparison of the computational burden for static reconstruction by means of the distributed reconstruction algorithm with non–optimized (blue–dashed line) and optimized (red line) visibility matrix structure, varying (a) the number of cameras, and (b) the number of targets. Structure of the visibility matrix is estimated as described in Section 3.

5. DISCUSSION AND CONCLUSIONS

As expected, Fig. 2 shows that the computational complexity of the distributed algorithm results to be much lower than that of the centralized one for all the considered values of the number of cameras (Fig. 2(a)) and of targets (Fig. 2(b)).

Interestingly, Fig. 2(a) and Fig. 3(a) show that the increasing rate of the computational complexity of the distributed algorithm decreases when considering a large number of cameras m. Then, the use of the distributed algorithm allows to obtain a computational time reduction that is approximately linear with the number of cameras m: the upper bound to such computational reduction is clearly m (i.e. when using m computational units, the parallelized computation cannot require less than $1/m$ times the computational time of the centralized one), however the distribution of the computational load among several units causes an increase in the time required for communication. Nevertheless, the observed approximately linear time reduction ensures the effectiveness of the method.

Furthermore, exploiting information about the system structure it is possible to further reduce the computational load. Several applications in automation, information technology, telecommunication consider large distributed systems. The network defined among the components of such systems is typically described by a sparse visibility matrix, where the components of the system are typically considered in a pseudo-random order. Thanks to the typical spatial localization of the data, the visibility matrix can usually be reduced by means of row and column permutations to a quite structured matrix. However, such structure, that can be exploited to improve the performance of the system, is hidden by the pseudo-random row and column order.

In Section 3 we have proposed an algorithm for estimating the structure of the visibility matrix, based on a procedure similar to the Cuthill-McKee algorithm Cuthill and McKee [1969], George and Liu [1981]. Such algorithm aims at grouping similar nodes. For this purpose, a proper similarity measure (1) has been introduced. As shown by the simulations of Section 3, the proposed algorithm effectively estimates the underlying structure of the visibility matrix. The case of a strictly defined structure (clustered matrix) has been also analyzed by means of a theoretical model in subsection 3.2. Then, the estimated (structured) visibility matrix can be used to determine the camera matching order in the distributed reconstruction procedure of Section 2. Since the matrix U can usually be considered as constant for several system sampling rates, the results of Algorithm 3.1 can be used in a number of consecutive reconstructions of target positions. Thus the time required for the execution of Algorithm 3.1 has low influence on the overall computational burden. Fig. 3 shows that the optimized reconstruction order allows to obtain a computational time reduction of 10% approximately.

To conclude, the combination of the distributed reconstruction algorithm with the optimized camera matching order allows to significantly reduce the computational time required for the reconstruction of target positions by the centralized algorithm.

REFERENCES

H. Aghajan and A. Cavallaro. *Multi-Camera Networks, Principles and Applications*. Academic Press, 2009.

H. Aghajan, R. Kleihorst, B. Rinner, and W. Wolf. Introduction to the issue on distributed processing in vision networks. *IEEE Journal of Selected Topics in Signal Processing*, 2:445–447, 2008.

F. Bullo, J. Cortés, and S. Martínez. *Distributed Control of Robotic Networks*. Applied Mathematics Series. Princeton University Press, 2009.

A. Chiuso, R. Muradore, and E. Aller-Carpentier. Sparse calibration of an extreme adaptive optics system. In *Proceedings of the 49th IEEE Conf. on Decision and Control, CDC 2010*, pages 1159–1164, December 2010.

R. Cucchiara, A. Prati, R. Vezzani, L. Benini, E. Farella, and P. Zappi. Using a wireless sensor network to enhance video surveillance. *Journal of Ubiquitous Computing and Intelligence*, 1(2):187–196, 2007.

E. Cuthill and J. McKee. Reducing the bandwidth of sparse symmetric matrices. In *Proc. 24th Nat. Conf. ACM*, pages 157–172, 1969.

J. Falcou, J. Sérot, T. Chateau, and F. Jurie. A parallel implementation of a 3D reconstruction algorithm for real-time vision. In *Parallel Computing: Current & Future Issues of High-End Computing, Proc. of the Int. Conf. ParCo 2005*, pages 663–670, In ParCo, 2005.

J.S. Franco, C. Ménier, E. Boyer, and B. Raffin. A distributed approach for real-time 3d modeling. In *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, 2004.

Y. Furukawa, B. Curless, S.M. Seitz, and R. Szeliski. Towards internet-scale multi-view stereo. In *Proceedings of the 23rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1434–1441, 2010.

J.A. George and J.W–H. Liu. *Computer Solution of Large Sparse Positive Definite Systems*. Prentice-Hall, 1981.

M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S.M. Seitz. Multi-view stereo for community photo collections. In *Proceedings of the 11th IEEE International Conference on Computer Vision (ICCV)*, 2007.

G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press: Baltimore, MD, 1989.

R.I. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157, 1997.

R.I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

M. Jovanovic, A. Klausner, M. Quaritsch, B. Rinner, and A. Tengg. Smart cameras for embedded vision. *Telematik*, 12(3):14–19, 2006.

A. Klausner, A. Tengg, and B. Rinner. Distributed multilevel data fusion for networked embedded systems. *J. on Sel. Topics in Signal Processing*, 2(3):538–555, 2008.

S. Liu, K. Kang, J.-P. Tarel, and D.B. Cooper. Distributed volumetric scene geometry reconstruction with a network of distributed smart cameras. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*, pages 2334–2341, 2009.

A. Masiero and A. Cenedese. On triangulation algorithms in large scale camera network systems. In *Proc. of the American Control Conf. (ACC)*, pages 4096–4101, 2012.

B. Narayanaswamy. *Sparse Measurement Systems: Applications, Analysis, Algorithms and Design, PhD Thesis*. Carnegie Mellon University, 2011.

A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, 1965.

B. Rinner and W. Wolf. An introduction to distributed smart cameras. *Proc. of the IEEE*, 96(10):1565–1575, 2008.

S. Soro and W. Heinzelman. A survey of visual sensor networks. *Advances in Multimedia*, 2009:1–22, 2009.

R.P. Tewarson. *Sparse Matrices*. Academic Press, 1973.

B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, pages 298–372, 1999.

R. Tron, R. Vidal, and A. Terzis. Distributed pose averaging in camera networks via consensus on SE(3). In *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conf. on*, pages 1–10, 2008.

P.-B. Wieber, F. Billet, L. Boissieux, and R. Pissard-Gibollet. The HuMAnS toolbox, a homogenous framework for motion capture, analysis and simulation. In *International Symposium on the 3D Analysis of Human Movement*, Valenciennes, France, 2006.