FEATURE-BASED CALIBRATION OF DISTRIBUTED SMART STEREO CAMERA NETWORKS

Aaron Mavrinac, Xiang Chen, and Kemal Tepe

University of Windsor
Department of Electrical and Computer Engineering
401 Sunset Ave., Windsor, Ontario, Canada N9B 3P4

ABSTRACT

A distributed smart camera network is a collective of vision-capable devices with enough processing power to execute algorithms for collaborative vision tasks. A true 3D sensing network applies to a broad range of applications, and local stereo vision capabilities at each node offer the potential for a particularly robust implementation. A novel spatial calibration method for such a network is presented, which obtains pose estimates suitable for collaborative 3D vision in a distributed fashion using two stages of registration on robust 3D features. The method is initially described in a geometrical sense, then presented in a practical implementation using existing vision and registration algorithms. The method is designed independently of networking details, making only a few basic assumptions about the underlying network's capabilities. Experiments using both software simulations and physical devices are designed and executed to demonstrate performance.

Index Terms— camera network, calibration, collaborative, distributed, registration, 3D vision

1. INTRODUCTION

The relatively new concept of 3D visual sensor networks [2] is emerging within the area of distributed smart cameras. By collecting and processing true 3D information, such networks offer improvements in existing applications and promise entirely new possibilities.

The 3D sensing paradigm includes the use of passive 3D (stereo) vision, fusion of information from multiple views, and distributed collaborative processing. Many of the basic computer vision operations, including shape recognition, object tracking, motion analysis, and scene reconstruction, have been improved through one or two of these properties; we contend that all three in unison yield yet greater benefits. Thus, our work applies to *distributed smart stereo camera networks*, wherein each node consists of a device capable of passive 3D vision and the distributed algorithms operate primarily or exclusively on 3D data.

Funding for this research was provided in part by the Natural Sciences and Engineering Research Council of Canada.

In order to perform any useful collaborative processing of this information, there must exist some way to bring data from multiple nodes into a common reference frame. This is achieved through calibration, which includes spatial localization and orientation as well as temporal synchronization. While some distributed time synchronization methods from the sensor network literature are applicable, existing localization methods are insufficient.

This paper presents a novel scalable spatial calibration method for distributed smart stereo camera networks. It is designed independent of node architecture or network details, and makes few assumptions about node deployment or scene contents. The problem is reduced to a geometrical form in Section 3, from which the implementation in Section 4 follows. A more thorough treatment can be found in [1].

The majority of research in distributed smart cameras to date has focused on monocular vision at each node. A number of methods for distributed self-calibration have been proposed for this paradigm, and though the vision components are not readily applicable to 3D sensing nodes, the general localization and distribution concepts developed apply to any vision-based system.

From the perspective of traditional sensor networks, the primary challenges are the directionality of vision sensors, the higher degree of accuracy required by vision applications, and the large volume of raw sensor data. Conversely, from the perspective of traditional computer vision, the challenge is in the scalable distribution of processing among nodes and the related limitations of network bandwidth.

While traditional sensor network methods generally employ omnidirectional sensors and thus require only localization, vision-based networks also require orientation. To apply similar methods to directional vision sensors, the concept of the *vision graph* is introduced in [4], where an edge on the graph represents shared field of view rather than a communication link.

Functional calibration methods are presented for monocular distributed smart cameras in [4, 5]. These are based on wide-baseline stereo methods, which are generally not robust due to the matching problem [13], and require unwieldy initialization schemes or dictate deployment constraints. Some methods, such as [6], use motion of objects in the scene to calibrate, but these still suffer from the matching problem to a degree and require certain kinds of scene. Potentially more robust methods are presented in [7, 8]; however, these require the use of markers or beacons placed in the environment, which is infeasible in many cases and may constrain deployment or extension to dynamic calibration.

With the true 3D sensing network paradigm introduced in [2], advocating distributed smart stereo cameras, a calibration method called Lighthouse is presented in [3] which uses 3D features and geographic hash tables (GHTs) to localize and orient nodes. Our method employs the same basic concept, but is more complete and addresses some impracticalities in the former.

2. PRELIMINARIES

2.1. Definitions

2.1.1. Nodes and Groups

A *node* is the abstract or physical smart stereo camera device itself; nodes shall be denoted by sequential capital letters (A, B, and so forth). The set of all nodes in the network shall be denoted $\mathbb{N}$ (where $|\mathbb{N}|$ represents the total number of nodes). A *group* is a set of nodes which agree on a single *leader* node; a group led by node A shall be denoted G_A (where $|G_A|$ represents the number of nodes in the group). Every group is a subset of the full set of nodes ($G_A \subseteq \mathbb{N}$), and every node is a member of exactly one group (so, if G_A and G_B are two separate groups, $|G_A \cap G_B| = 0$).

2.1.2. Point Sets and Features

A *point set* is the full set of interest points detected locally at a node; the point set of node A shall be denoted S_A. The *overlap* between point sets S_A and S_B refers to the size of the intersection of the two sets $|S_A \cap S_B|$, said intersection occurring where a point in S_A corresponds to the same physical point as a point in S_B. The *percent overlap* is defined as follows:

$$\%O(S_A, S_B) = \frac{|S_A \cap S_B|}{\max(|S_A|, |S_B|)} \times 100\% \tag{1}$$

A *feature* is any subset of the point set of a certain size (determined by a parameter of the algorithm); when discussing a single arbitrary feature from node A, it shall be denoted F_A, where $F_A \subseteq S_A$. Two features F_A and F_B, from nodes A and B respectively, are considered to *match* (denoted $F_A \approx F_B$) if each point in F_A corresponds to the same physical point as a point in F_B. In the context of the algorithm, it is impossible to ascertain this correspondence, so the term *match* implies rather a presumed match based on a criterion of geometrical similarity.

2.2. Pose

Pose is a concept used here to describe the relative motion between two nodes in a distributed smart camera network, which is the basis of calibration. Each node is considered to have its own local coordinate system. The *relative pose* of node A with respect to node B is denoted P_{AB}, and is the rigid transformation in 3D Euclidean space from the coordinate system of A to that of B.

The transformation $P_{AB} : \mathbb{R}^3 \to \mathbb{R}^3$ consists of a rotation matrix (3×3 real orthogonal matrix) $\mathbf{R}_{AB}$ and a 3-element translation vector $\mathbf{T}_{AB}$. P_{AB} maps a point $x \in \mathbb{R}^3$ as follows:

$$P_{AB}(x) = \mathbf{R}_{AB}x + \mathbf{T}_{AB} \tag{2}$$

The *identity* pose is denoted P_I, and consists of the identity matrix $\mathbf{R}_I$ and the zero vector $\mathbf{T}_I$.

The inverse of pose P_{AB}, denoted P_{AB}^{-1}, reverses the pose transformation (so that $P_{AB}^{-1} = P_{BA}$). It can be determined as follows:

$$P_{AB}^{-1}(x) = \mathbf{R}_{AB}^{-1}x - \mathbf{R}_{AB}^{-1}\mathbf{T}_{AB} \tag{3}$$

A succession of pose transformations $P_{BC}(P_{AB}(x))$ can be composed into a single pose, denoted $(P_{AB} \circ P_{BC})(x)$, as follows:

$$(P_{AB} \circ P_{BC})(x) = \mathbf{R}_{BC}\mathbf{R}_{AB}x + (\mathbf{R}_{BC}\mathbf{T}_{AB} + \mathbf{T}_{BC}) \tag{4}$$

This transformation maps from the coordinate system of A to that of B, then from that of B to that of C; therefore, the transformation from A to C can be computed via composition as $P_{AC} = (P_{AB} \circ P_{BC})(x)$. This operation is transitive, so one node's pose relative to another can be computed indirectly over an arbitrary number of intermediate poses if they exist.

2.3. Graphs

Three types of undirected graphs are helpful in describing distributed smart camera calibration: the *communication graph*, the *vision graph*, and the *calibration graph* [4]. Graphs are described as *connected* if there exists a path connecting every pair of nodes, and *complete* if there exists an edge between each pair of nodes.

The communication graph describes the effective communication links between nodes in the network from the perspective of the layer presented to the application. A complete communication graph indicates that any node may communicate directly with any other node.

The vision graph describes which nodes share significant portions of their field of view. A pair of nodes have a connecting edge in this graph if the volume of space in the intersection of their fields of view is considered large enough that it might contain sufficient data for the operations required by the algorithm.

The calibration graph describes which nodes have a direct estimate of their pairwise pose. Obviously, it is desirable that this graph be connected, so that any two nodes X and Y may estimate their relative pose P_{XY} by composition of known pose estimates. Edges can only be established where there exist edges in the vision graph, and the most complete calibration graph possible is identical to the vision graph.

3. MAIN PROBLEM

3.1. Problem Statement

The overall objective is to spatially calibrate a series of homogeneous smart stereo camera nodes, with no *a priori* knowledge and using only the nodes' 3D visual data, in a distributed fashion. Assuming the visual data consists of a set of 3D points triangulated from stereo images of the environment, the problem may be reduced to geometrical terms:

> Given a set of nodes $\mathbb{N}$, each node $X \in \mathbb{N}$ having a point set S_X, estimate the pose P_{XY} for enough node pairs (X, Y) such that the calibration graph for $\mathbb{N}$ is connected.

The shared view assumption (3.2.2) and the repeatability criterion of interest point detection (4.2) imply a sufficient degree of overlap between a sufficient number of node pairs for convergence.

3.2. Assumptions

3.2.1. Pre-Deployment Offline Access

It is assumed that, prior to deployment of the network, there is a period during which each node may be accessed without restriction in a controlled environment, in order to perform certain essential modifications to software (such as assignment of a unique identifier, network configuration, and intrinsic/stereo calibration of the cameras).

3.2.2. Shared View

For full convergence, it is assumed that the vision graph is connected. This imposes a minimum qualitative constraint on node deployment that the shared field of view of the entire network be continuous and have substantial internal pairwise overlap.

3.2.3. Fixed Nodes

It is assumed that each node is fixed in its location and orientation relative to all other nodes. It is also assumed that, once internally calibrated for stereo vision, no node changes the relative motion between its cameras or the internal parameters (e.g. focal length) of either of its cameras.

3.2.4. Static Scene

It is assumed that the contents of the scene are fully static for the purposes of acquiring calibration point sets. This is solely for simplicity, and could easily be relaxed by employing background estimation techniques or accurate temporal synchronization.

3.2.5. Abstract Network

It is assumed that the nodes are capable of autonomously forming an ad-hoc network of some kind, wherein each node can be addressed by a unique identifier. From the algorithm's point of view, the network is assumed to be *fully connected* [17], or in other words, the communication graph is assumed to be complete. Additionally, it is assumed that arbitrary amounts of data can be sent with assured delivery.

3.3. Problem Analysis

3.3.1. Two-Stage Registration

Bringing the point sets, and thereby the node coordinate systems, into alignment with one another can be accomplished by *registration*. Registration algorithms may be divided into two types: coarse registration, which can align points without an initial estimate but are generally not very accurate; and fine registration, which require an initial estimate to align points but are very accurate [9].

For our purposes, no alignment estimate is initially available, yet high accuracy is desirable. The typical solution when presented with such a problem is a two-stage approach, using coarse registration to initialize fine registration. However, there is more to the problem in our case: it is not even known which point sets overlap or to what degree. We use a process of *feature matching* to determine how to proceed with registration between nodes.

3.3.2. Feature Matching

In order to find coarse pose estimates between nodes with no knowledge of their point set overlap in a distributed fashion, a pairwise feature matching process similar to that described in [3] can be employed.

The goal is to find pairwise matches between nodes' features, and then use those matches to calculate coarse relative pose estimates for the node pairs. Both are accomplished through the coarse registration algorithm; if the registration error falls below a certain threshold t_{ec}, the features are considered to match, and the registration result yields a coarse pose estimate between the source nodes.

Consider point sets from two nodes, S_A and S_B, from which, according to the coarse matching algorithm, each node randomly selects a feature of size $f \geq 3$, resulting in $F_A \subseteq S_A$ and $F_B \subseteq S_B$ where $|F_A| = |F_B| = f \leq |S_A \cap S_B|$. The performance of the matching scheme depends on the probability of a match between F_A and F_B, $P(F_A \approx F_B)$, which can be calculated as follows:

$$P(F_A \approx F_B) = \frac{|S_A \cap S_B|!f!(|S_A| - f)!(|S_B| - f)!}{|S_A|!|S_B|!(|S_A \cap S_B| - f)!} \quad (5)$$

It is therefore desirable to increase $|S_A \cap S_B|$ relative to $|S_A|$ and $|S_B|$ (i.e., increase the percent overlap), which translates into repeatability in interest point detection (4.2). There is a trade-off in the value of f between matching performance and false matches; generally, a low value such as $f = 4$ is adequate.

3.3.3. *Feature Categorization*

No details have yet been given about how to bring features together for matching in a distributed fashion. The idea of feature categorization is borrowed from the data-centric storage literature, used with reference to distributed smart camera networks in [2] and more specifically to their calibration in [3]. The goal is to evenly distribute the processing and storage of the data in a distributed system based on some quantitative or qualitative metric of the data itself. For this, a smooth, deterministic *geometric descriptor* function, denoted g, is used.

The solution space of this descriptor is then divided as evenly as possible among the nodes in the network, with some overlap (see below), and features detected locally at each node are sent to the appropriate node for matching to other geometrically similar features.

Ideally, the difference between the descriptors of two features F_A and F_B describes the degree of difference d between those features:

$$d(F_A, F_B) = |g(F_A) - g(F_B)| \quad (6)$$

Based on the measurement accuracy of a node and the specific coarse registration algorithm used, there is a similarity threshold t_d, such that it is necessary to compare two features F_A and F_B if $d(F_A, F_B) < t_d$, and unnecessary otherwise; this will be termed the *similarity condition*. The desirable overlap for categorization, then, is $t_d/2$ in all directions.

Note that categorization, and thus the nodes where features are matched, has no relation to the nodes where those features originated. When a match is found, the result is returned to one of the two source nodes, based on some deterministic selection function such that for a given pair of nodes the same node is always selected.

3.3.4. *Coarse Grouping*

In order to guarantee that all nodes with edges on the vision graph attempt pairwise pose refinement without the need for exhaustive feature matching, we introduce a grouping scheme wherein nodes are merged into ever-larger groups within the same coordinate system, albeit with only coarse estimates. Through pose composition, any node in a group can determine its coarse pose with respect to any other node. This is conceptually similar in some ways to the GHT scheme proposed in [3].

A node always knows its current group leader and the set of nodes comprising its group. Within a group (2.1.1), each node has a coarse pose estimate relative to the group leader, called its *group coarse pose estimate*, and denoted C_A for a node A. Relative coarse pose estimates (e.g. C_{AB} for node A relative to node B) can be computed from these, either directly or through one or more compositions. Initially, each node begins in a singleton group, of which it is the leader, with its group coarse pose estimate initialized to P_I.

A *merge* is initiated when two nodes have detected a certain minimum number t_m of consistent matches with each other. Consistency is enforced via a threshold t_c specifying the minimum Euclidean distance between the pose estimates' mappings of a given point (such as the centroid μ_S of the computing node's point set). Once a node has stored at least t_m matches with a particular other node, each time a new match is detected for that node, an average coarse pose estimate is computed for every combination M_i of matches containing the new match, and checked for consistency:

$$||C_m(\mu_S) - C_{avg}(\mu_S)|| \leq t_c, \forall m \in M_i \quad (7)$$

If a consistent average is found, it is considered a reliable relative coarse pose estimate, and is forwarded to the source nodes' group leaders and composed as necessary to merge the nodes' respective groups.

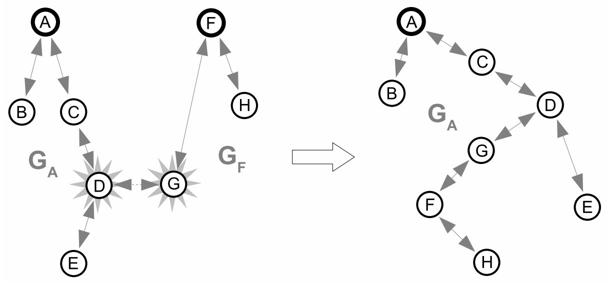


Fig. 1. Group Merging

Figure 1 illustrates a typical group merge. Node D, of group G_A, and node G, of group G_F, find a relative coarse pose estimate through feature matching, and initiate a merge. The nodes in group G_A do not modify their group coarse pose information. Node G's new group coarse pose estimate (C'_G) is the composition of its estimated pose relative to node D with node D's group coarse pose estimate:

$$C'_G = C_{GD} \circ C_D \quad (8)$$

The new group coarse pose estimates for the merging group's leader (C'_F) and any other nodes in the merging group (in this case, C'_H) can similarly be calculated as compositions of known pose estimates:

$$C'_F = C_G^{-1} \circ (C_{GD} \circ C_D) \tag{9}$$

$$C'_H = C_H \circ (C_G^{-1} \circ (C_{GD} \circ C_D)) \tag{10}$$

Since merging consists of composition operations, it is a transitive operation which can occur based on matches (and the resultant relative coarse pose estimates) between any pair of nodes in different groups. Figure 1 illustrates this by showing the actual history of merges leading to the groups as arrows between the node pairs; in reality, of course, every node in the group has a direct pose estimate to the leader (group coarse pose estimate).

3.3.5. Pairwise Pose Refinement

Once a given pair of nodes belong to a group via the feature matching process, those nodes can use their coarse relative pose estimate as a starting point for pose refinement. This is achieved by applying a fine registration algorithm to a large number of points initialized into coarse alignment.

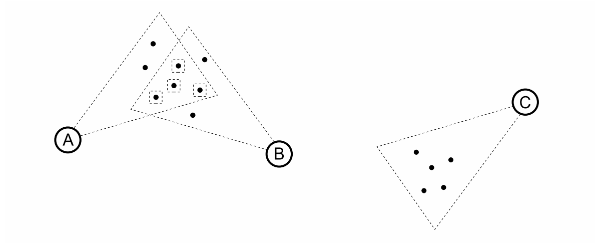


Fig. 2. Field of View Cone Approximation

As shown in Figure 2, the actual point sets used for fine registration are selected, at each node, as those falling within the intersection of the two nodes' fields of view, as approximated by a cone of a certain angle and length extending along the positive z-axis of each node's coordinate system (via the coarse pose estimate). If there are fewer than a specified minimum number of points, which includes the case where there is no intersection at all, the nodes do not attempt pose refinement.

3.3.6. Indirect Pose Estimation

A pair of nodes attempting to determine their relative pose can now communicate directly to find the shortest path along the existing pairwise fine pose estimates (calibration graph) and thus obtain a composition with a minimum of error. A node A may find such an estimate P_{AB} relative to a node B according to the following algorithm (suppose FP_A represents the set of fine pose estimates at node A):

1. If $P_{AB} \in FP_A$, select P_{AB} and end.
2. For each $P_{AX} \in FP_A$, request FP_X from node X. If $P_{XB} \in FP_X$, select $P_{AB} = P_{AX} \circ P_{XB}$ and end.
3. For each $P_{XY} \in FP_X$, request FP_Y from node Y. If $P_{YB} \in FP_Y$, select $P_{AB} = P_{AX} \circ P_{XY} \circ P_{YB}$ and end.
4. Continue until P_{AB} has been found.

As indirect fine pose estimates are found (even intermediate ones that were not requested), they should be added to FP to avoid unnecessary repetition of network requests and computations.

4. ALGORITHM DESIGN

4.1. Distributed Calibration Algorithm

The algorithm is split into ten distinct processes at each node; six for coarse grouping, and four for pairwise pose refinement. Each process acts upon receipt of a message, with the exception of the *feature selection process*, which executes periodically, and the *pose refinement initiator process*, which executes whenever the group composition is updated.

There are four parameters intrinsic to the algorithm itself, following from Section 3.3: the feature size f, the similarity threshold t_d, the match threshold t_m, and the consistency threshold t_c. Certain other implementation-specific parameters are also required, notably those for the coarse and fine registration algorithms; in particular, t_{ec} and t_{ef} are referenced here as generic error thresholds for the coarse and fine registration algorithms, respectively.

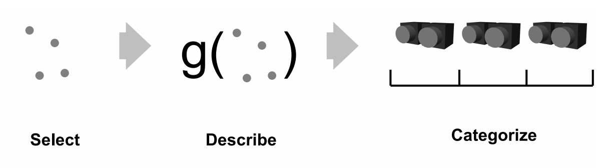


Fig. 3. Feature Selection Process

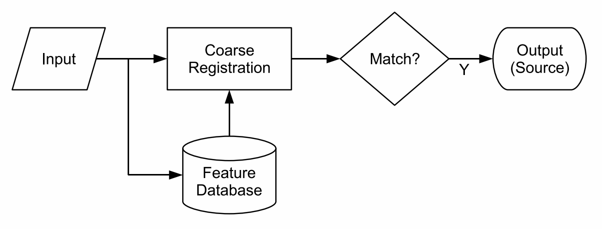


Fig. 4. Feature Matching Process

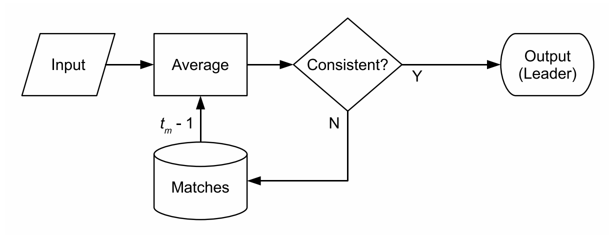


Fig. 5. Match Processing Process

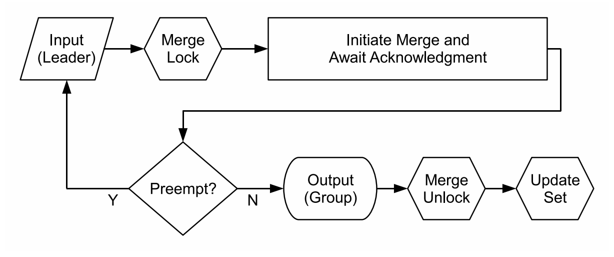


Fig. 6. Group Merge Initiator Process

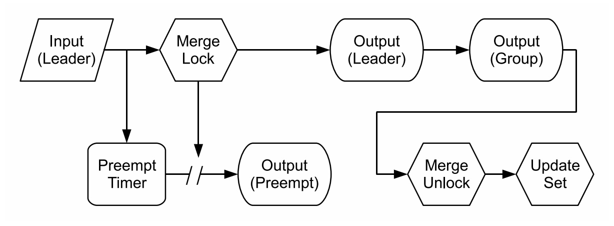


Fig. 7. Group Merge Responder Process

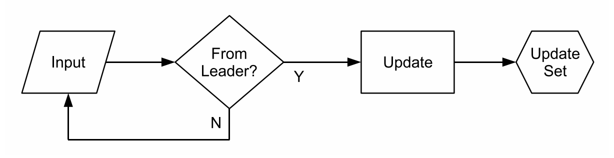


Fig. 8. Group Update Process

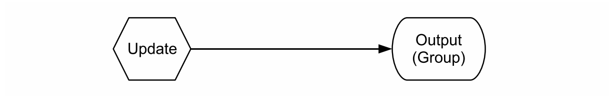


Fig. 9. Pose Refinement Initiator Process

4.2. Interest Point Detection

Three distinct parts of calibration are impacted directly by the interest point detection algorithm used: the correspondence algorithm, the coarse matching scheme, and the fine registration algorithm. In all three cases, it is the *repeatability* performance metric which is of interest; higher repeatability yields higher overlap in the point sets.

For practical purposes (including the availability of source code from the authors), the FAST interest point detector [14, 15] is selected for this implementation. Convergence can be encouraged by constraining nodes to share large portions of their fields of view or by calibrating on a scene with strong interest points.

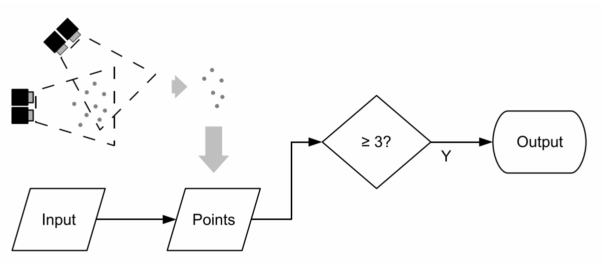


Fig. 10. Pose Refinement Responder Process

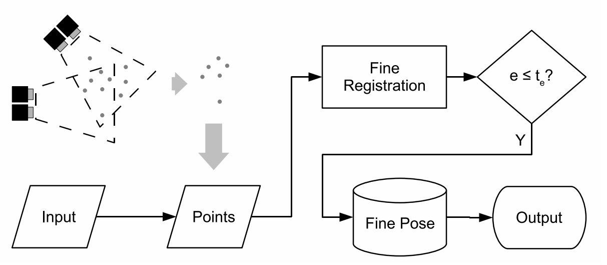


Fig. 11. Fine Registration Process

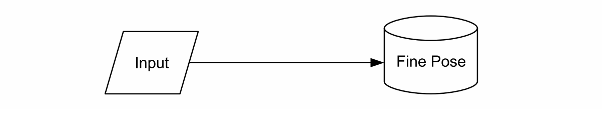


Fig. 12. Pose Update Process

4.3. Registration

Since matching features overlap fully, an excellent solution to the coarse registration problem is the fully-contained version of the DARCES algorithm [10], using three control points. DARCES without the RANSAC component is a relatively simple algorithm, allowing it to perform rapid matching on a large number of features.

The concept of the Iterative Closest Point (ICP) algorithm [11] lends itself well to the fine registration problem encountered in pairwise pose refinement. However, the difficulty of stable interest point detection, occlusion effects, and uncertainty about the overlap in field of view all contribute to poor overlap in the point sets used for pose refinement. The Trimmed Iterative Closest Point (TrICP) algorithm [12], used in this implementation, can be automatically tuned to any degree of overlap, and is applicable to overlaps under 50%.

5. EXPERIMENTS

5.1. Performance Metrics

5.1.1. Convergence

Convergence is the measure of the algorithm's ability to bring nodes into a common reference frame and its time performance in doing so. For our purposes, there are actually two distinct considerations:

1. The ability of coarse grouping to merge into a minimum number of groups.
2. The ability of pairwise pose refinement to establish a maximum number of pairwise estimates.

Calibration is considered successful in terms of convergence when coarse grouping merges the entire network into a single group and pairwise fine pose estimates are established such that the calibration graph is connected.

5.1.2. Accuracy

Accuracy is the measure of the error in the algorithm's resulting pose estimates. The mean error in a pose estimate can be determined by averaging the Euclidean distance between a number of points with ground-truth correspondence, detected and triangulated at the nodes separately from those used for calibration. Although error accumulates with the path length (number of pose compositions) in the calibration graph, it is more relevant to consider the path length in the vision graph, since the 3D reconstruction consistency among nodes observing the same part of the scene is the likely criterion.

5.1.3. Scalability

Scalability is the measure of the effect on the algorithm's performance of the number of nodes in the network. The three primary resources to consider are node-local computing resources (i.e. CPU and memory), node-local data storage, and network bandwidth.

In order to properly evaluate scalability, it is necessary to examine individual factors arising from the algorithm itself. The most significant of these can be summarized in terms of the number of nodes in the network $|\mathbb{N}|$ as follows:

- Feature dissemination requires bandwidth resources in $|\mathbb{N}|$ *per node*.
- Feature matching requires computing and storage resources in $|\mathbb{N}|$.

This assumes that each node maintains a more or less constant number of pairwise edges in the vision graph regardless of $|\mathbb{N}|$, as would be the case with most applications. In cases where this assumption does not hold, it is necessary to add a third factor:

- Pairwise pose refinement computation requires computing resources in $|\mathbb{N}|$.

Scalability in all three resources can be quantized experimentally in terms of the above factors.

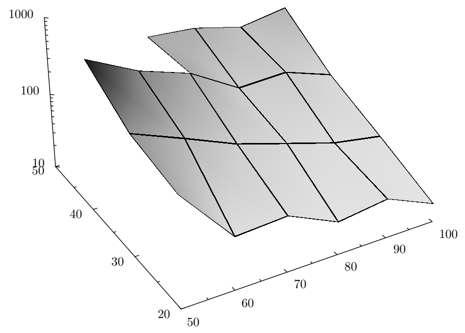
5.2. Manual Point Set

In order to test the capabilities of the calibration algorithm and tune its parameters under controlled conditions, the first experiment series is designed to operate on manually selected points with full correspondences across all four nodes. The primary purpose of this experiment type, once suitable parameters are found, is to test the effects of different point set sizes and overlap characteristics on convergence and accuracy.

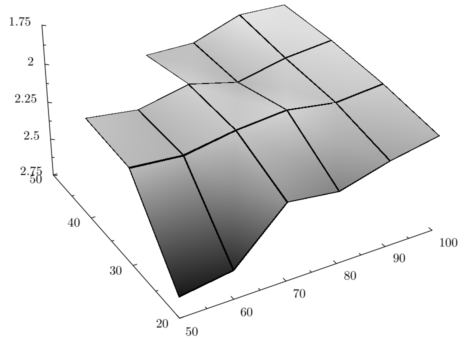
5.2.1. Procedure

A total of 22 point subsets are extracted from the data, and each is tested using the distributed calibration software, with all four nodes running locally on the same workstation. This procedure is repeated twice for each subset, and the average results for convergence time and mean error are calculated and recorded.

5.2.2. Results



(a) Convergence Time Trends in n and p



(b) Accuracy Trends in n and p

Fig. 13. Manual Experiment Results

5.3. Automatic Point Set

Having established some criteria for reasonably timely convergence in the manual point set experiments, the next step is to test real automatic calibration of the network. The purpose of these experiments is to test the convergence and accuracy performance of the algorithm in real conditions.

5.3.1. Procedure

Four instances of the local point detection software, configured to execute the distributed calibration software on completion, are run in automatic mode on the vision platform workstation. Convergence time and the final calibration graph are recorded. A ground truth point set is manually selected for each camera rig, and the mean error is calculated and recorded.

5.3.2. Results

The mean error and convergence time of a typical experiment from this series is shown below. Figure 14 shows the final calibration graph, Figure 15 shows the physical deployment of the nodes for the experiment, and Figure 16 shows the visualization of the resultant pose estimates (which can be seen to match the configuration in Figure 15).

- **Mean Error:** 2.7666 mm
- **Convergence Time:** 159 s

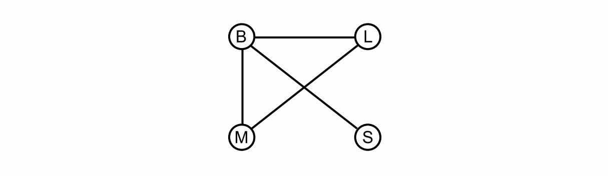


Fig. 14. Calibration Graph for Automatic Experiment

5.4. Virtual Point Set

Since only four physical camera rigs are available, testing scalability to larger networks is impossible in an automatic experiment and difficult to control using the manual methods. Instead, controlled virtual point sets are supplied to the same calibration algorithm implementation to test the scalability metric.

5.4.1. Procedure

Point sets are generated for 5, 10, 15, 20, and 25 nodes. The total outgoing bandwidth in kilobytes, final size of the matching database in features, and total number of coarse and fine registration executions are recorded.

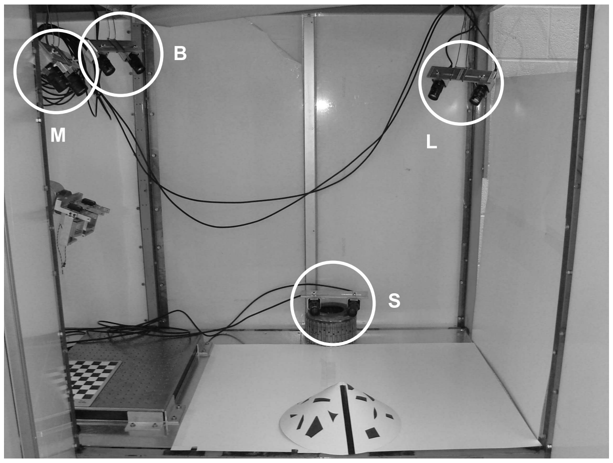


Fig. 15. Camera Deployment for Automatic Experiment

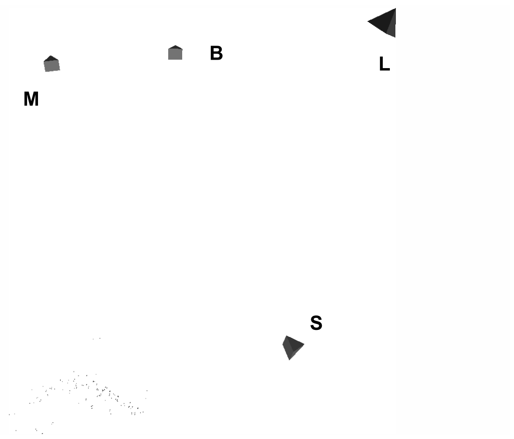


Fig. 16. Pose Visualization for Automatic Experiment

5.4.2. Results

As expected, total bandwidth usage per node increases approximately linearly in relation to the number of nodes in the network (Figure 17). This affects different networks in different ways. In a network where the physical medium is shared by all nodes – the worst-case scenario – the total network bandwidth usage is the relevant factor. In that case, the bandwidth usage increases non-linearly, potentially at up to $|\mathbb{N}|^3$. However, many routing methods used in sensor networks are much more efficient and therefore mitigate this effect.

The number of features stored at each node increases approximately linearly in relation to the number of nodes (Figure 18). Features are very small data (a series of f 3-tuples, an identifier, and a geometric descriptor value), but when scaling to extremely large networks it must be ensured that adequate storage is provided at each node for these features.

The number of coarse registration operations performed at each node increases approximately linearly in relation to the number of nodes (Figure 19); as expected, this is proportional

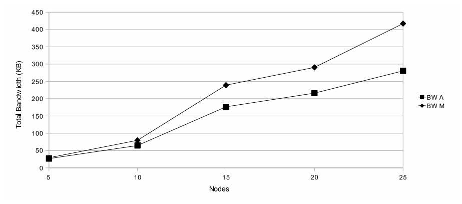


Fig. 17. Bandwidth Usage in $|\mathbb{N}|$ (Average and Maximum)

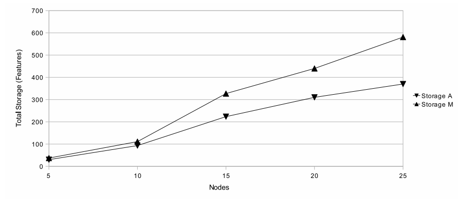


Fig. 18. Node-Local Storage in $|\mathbb{N}|$ (Average and Maximum)

to the number of features stored. If processing throughput is the limiting factor, this increase will cause the convergence time to increase linearly with the number of nodes.

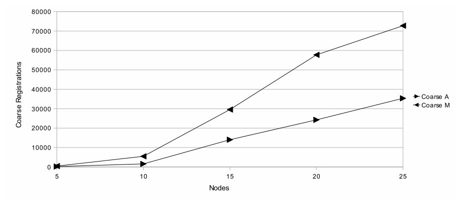


Fig. 19. Coarse Registration Processing in $|\mathbb{N}|$ (Average and Maximum)

Since, in this network, the number of vision graph edges per node does not generally increase as its total number of nodes increases, the number of fine registrations per node is approximately constant.

6. CONCLUSIONS

A calibration method for distributed smart stereo camera networks has been developed which converges well, provides accurate pairwise orientation, and scales well to large networks. This provides a base upon which to build a full 3D visual sensor network providing primitive data-centric queries, upon which in turn a variety of high-level applications can be developed.

Currently, the algorithm makes it possible for smart stereo camera devices to self-localize and self-orient relative to one another in a distributed fashion, allowing for various subsequent stages of realization for a variety of applications. The immediate opportunity is to provide a generalized framework for building these solutions, which would rest on the underlying assumption that the network is accurately calibrated and can perform 3D reconstruction across multiple views.

The major implementation drawback is the instability of interest point detection in the general case; at present, it is necessary to control the scene somewhat by adding one or more calibration targets for convergence to occur reliably. Improving this situation is an important avenue for future work.

7. REFERENCES

[1] A. Mavrinac, "Feature-Based Calibration of Distributed Smart Stereo Camera Networks," M.A.Sc. Thesis, University of Windsor, 2008.

[2] M. Akdere, U. Cetintemel, D. Crispell, J. Jannotti, J. Mao, and G. Taubin, "Data-Centric Visual Sensor Networks for 3D Sensing," in *Proc. 2nd Intl. Conf. on Geosensor Networks*, 2006.

[3] J. Jannotti and J. Mao, "Distributed Calibration of Smart Cameras," in *Proc. Intl. Workshop on Distributed Smart Cameras*, pp. 55–61, 2006.

[4] D. Devarajan and R. J. Radke, "Distributed Metric Calibration of Large Camera Networks," in *Proc. 1st Workshop on Broadband Advanced Sensor Networks*, 2004.

[5] W. E. Mantzel, H. Choi, and R. G. Baraniuk, "Distributed Camera Network Localization," in *Proc. 38th Asilomar Conf. on Signals, Systems and Computers*, 2004.

[6] S. Funiak, C. Guestrin, M. Paskin, and R. Sukthankar, "Distributed Localization of Networked Cameras," in *Proc. 5th Intl. Conf. on Information Processing in Sensor Networks*, pp. 34–42, 2006.

[7] C. Beall and H. Qi, "Distributed Self-Deployment in Visual Sensor Networks," in *Proc. Intl. Conf. on Control, Automation, Robotics and Vision*, pp. 1–6, 2006.

[8] C. J. Taylor and B. Shirmohammadi, "Self Localizing Smart Camera Networks and their Applications to 3D Modeling," in *Proc. Intl. Workshop on Distributed Smart Cameras*, pp. 46–50, 2006.

[9] J. Salvi, C. Matabosch, D. Fofi, and J. Forest, "A Review of Recent Range Image Registration Methods with Accuracy Evaluation," *Image and Vision Computing*, vol. 25, no. 5, pp. 578–596, 2007.

[10] C.-S. Chen, Y.-P. Hung, and J.-B. Cheng, "RANSAC-Based DARCES: A New Approach to Fast Automatic Registration of Partially Overlapping Range Images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 21, no. 11, pp. 1229–1234, 1999.

[11] P. J. Besl and N. D. McKay, "A Method for Registration of 3-D Shapes," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[12] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek, "The Trimmed Iterative Closest Point Algorithm," in *Proc. Intl. Conf. on Pattern Recognition*, pp. 545–548, 2002.

[13] A. Baumberg, "Reliable Feature Matching Across Widely Separated Views," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1774–1781, 2000.

[14] E. Rosten and T. Drummond, "Fusing Points and Lines for High Performance Tracking," in *Proc. 10th IEEE Intl. Conf. on Computer Vision*, pp. 1508–1511, 2005.

[15] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," in *Proc. 9th European Conf. on Computer Vision*, pp. 430–443, 2006.

[16] H. C. Longuet-Higgins, "A Computer Algorithm for Reconstructing a Scene from Two Projections," *Nature*, vol. 293, pp. 133–135, 1981.

[17] M. Raynal, *Distributed Algorithms and Protocols*, John Wiley & Sons, 1988.

[18] O. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint*, The MIT Press, 1993.

[19] Y. Ma, S. Soatto, J. Košecká, S. S. Sastry, *An Invitation to 3-D Vision: From Images to Geometric Models*. Springer-Verlag, 2004.