A Sparse Hybrid Map for Vision-Guided Mobile Robots

Feras Dayoub Grzegorz Cielniak Tom Duckett
Department of Computing and Informatics, University of Lincoln, Lincoln, UK
{fdayoub,gcielniak,tduckett}@lincoln.ac.uk

***Abstract*— This paper introduces a minimalistic approach to produce a visual hybrid map of a mobile robot's working environment. The proposed system uses omnidirectional images along with odometry information to build an initial dense pose-graph map. Then a two level hybrid map is extracted from the dense graph. The hybrid map consists of global and local levels. The global level contains a sparse topological map extracted from the initial graph using a dual clustering approach. The local level contains a spherical view stored at each node of the global level. The spherical views provide both an appearance signature for the nodes, which the robot uses to localize itself in the environment, and heading information when the robot uses the map for visual navigation. In order to show the usefulness of the map, an experiment was conducted where the map was used for multiple visual navigation tasks inside an office workplace.**

I. Introduction

Different methods have been introduced to tackle the problem of acquiring a map of a mobile robot's environment. The problem is called simultaneous localization and mapping (SLAM). Due to the importance of solving the SLAM problem before truly autonomous mobile robots can be built, the robotics community has given much attention to SLAM over the last decade. The result is a wide variety of methods, with each method having its own advantages and disadvantages.

Recently, the methods which deal with SLAM as a non-linear graph optimization problem, i.e. graph-based SLAM, have gained increasing attention in the literature. The output of these methods is a pose-graph where the nodes associate past poses of the robot with map features. The edges of the graph model spatial constraints between the nodes [1]. The optimization step aims to select the spatial configuration of the nodes, which best satisfies the constraints encoded in the edges. Generally, the output from the graph optimization algorithms is a pose-graph map, where the nodes of the graph are created at every step the robot has performed. Therefore, the graph is dense and it contains redundant information that can be removed leading to a more compact map, which is preferable in the case of mobile robots with limited resources.

In this paper we use a graph-based SLAM algorithm to produce an initial dense pose-graph map of the environment. Then the initial map is used to build a sparse hybrid map consisting of two levels, global and local. Fig. 1 illustrates the hybrid map. On the global level, the world is represented as a topological map. The topological map is extracted from the initial dense pose-graph using a dual clustering approach, introduced in this paper. The proposed approach selects nodes from the initial map which are located in areas such as doorways and corners, allowing the topological map to maintain full coverage of the environment while minimizing the required number of nodes. On the local level of the map, each node stores a spherical view representation of image features extracted from an omnidirectional image recorded at the position of the node. The spherical views contain the 3D location of the image features on a sphere. Thus, we only store the direction of the features (but not their distance or depth) from the centre of the sphere, which corresponds to the centre of that node. The spherical views are used for estimating the robot's heading in a visual navigation system where we use the map to perform a path following task.

The paper is constructed as follows. In Section II, we discuss related work in the field. Section III contains details of the method to build the initial map. In Section IV we present the dual clustering algorithm which selects the nodes of the global map. A visual navigation strategy is presented in Section V. The experiment and results are presented in Section VI. Finally, we draw conclusions in Section VII.

II. Background

Different vision-based mapping methods using graph optimization have been proposed [2], [3], [4]. These methods follow the same general approach where the map is built as a graph, with the nodes containing camera views from the environment and the graph edges are expressed as geometric constraints between these views. A loop closing mechanism is deployed to detect when previously mapped areas are revisited. When a loop is detected, a new constraint link is added to the graph and then a graph optimizer is invoked to correct the map. Although the work presented in this paper follows the same general approach, we differ by proposing a spherical view representation for the nodes in the local level of the hybrid map and also we introduce a dual clustering algorithm to reduce the number of nodes in the global level.

The loop detection mechanism in this paper uses a similarity measure between the views stored in the nodes of the map. The detection mechanism could simply be a direct one-to-one view matching procedure. However, more sophisticated methods for loop detection can be used to provide real time performance when the graph contains a large number of nodes. Such methods include the hierarchical vocabulary tree [5] and the FABMAP visual vocabulary [6].

A naive solution to reduce the number of nodes in the graph would be based on the time stamp of the nodes, where the graph is sampled using a fixed time step. This method would fail if the robot stands still for some time or has a changing speed while mapping the environment. Another simple solution would be based on the distance between the nodes [7]. However, this method does not take into

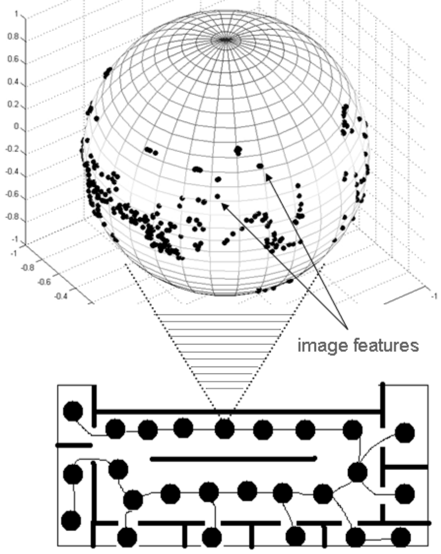

Fig. 1. Proposed hybrid map with two levels i.e. global and local. The environment is represented as an adjacency graph of nodes on the global level of the map and each node on the local level represents the 3D location of image features on a sphere. Our method represents the direction of the features (but not their distance or depth) from the centre of the sphere, which corresponds to the centre of that node.

account the rapid change in the appearance of the robot's surroundings which could occur when the robot crosses a door or goes round a corner. Successive images can differ considerably on the two sides of the doors or corners, which could become a challenge when the robot tries to actually use the map for navigation. This implies that image similarity should be considered in the process. In general, the methods which use image similarity to reduce the number of nodes in the map start by clustering the nodes based on image similarity and then choose key nodes as a representative for each image group [8]. These methods can be used when information about the geometric distances between the nodes is not available. However, in our case the distance between the nodes is provided by the graph optimization approach and we can use that as an additional source of information. Instead of choosing the key nodes in the graph based on the geometric distance alone or the similarity alone, we propose a method which selects the nodes in the graph based on a combination of the two metrics.

Recently, Ila et al. [9] introduced a pose-graph mapping method where they measure statistical content on links to enforce node sparsity and limited graph connectivity. The result is a compact map which contains non-redundant nodes and links. The main criteria with which the nodes are selected relates directly to reducing the uncertainty in the estimation of the robot position. This tends to produce densely distributed nodes when the robot performs curved paths and sparsely distributed nodes when the robot performs straight forward paths. This is because of the increasing uncertainty resulting from turning motions. The method presented in this paper uses different criteria to reduce the number of nodes in the map. Our main goal is to produce a map with sparse nodes located in places that are suitable to use the map for visual navigation tasks.

III. Building the Hybrid Map

We aim to produce a hybrid map extracted from an initial dense pose-graph map. The hybrid map consists of a global level contains a sparse topological map and a local level which stores a spherical view for each node (see Fig. 1). Each spherical view is generated from an omnidirectional image recorded from the position of the node.

The initial map building process is carried out based on a stop-sense-go strategy, where the robot is driven by a human operator. The driver follows the following routine. The robot starts a mapping step by capturing an omnidirectional image from a camera on-board, while it is static, along with the odometry reading from the wheel encoder. Then the robot moves a short distance forward and stops. If there is a need to perform a rotation, the robot rotates a certain angle and ends the current mapping step by stopping. The driver repeats as many mapping steps as required to cover the environment.

The constraints between each consecutive poses along the trajectory of the robot consist of two components, i.e. translation and rotation. These constraints can be extracted from the internal odometry of the robot. However, using the odometry alone is not always accurate enough to build the constraint network. First, the odometry measurements are affected by noise due to wheel drift and slippage, especially during rotation. Second, using odometry alone cannot provide any information about loop closure. To deal with these limitations, we use measurements from the omnidirectional vision sensor on-board as an input for a Bayesian filtering framework (extended Kalman filter) to reduce the error from the odometry measurements. In addition to that, the vision sensor is used to perform loop closure. A loop closure results in an edge in the constraint network which relates the current robot pose with a former robot pose. These loop closure edges contradict the pose estimates resulting from plain odometry. In order to correct the structure of the graph after a loop is detected, we use the graph optimization algorithm TORO [10] which considers each edge in the pose graph as a cost function for each two connected nodes and re-arranges the nodes in the graph such that the total costs associated with all edges are minimized.

A. *Relations Based on Odometry*

Let the robot's pose at any given time step t be represented as $\mathbf{p}_t = (x_t, y_t, \theta_t)$, where (x_t, y_t) are the coordinates of the robot and θ_t is the current heading. In our stop-sense-go strategy, robot motion between two consecutive poses is approximated by a translation d followed by a rotation δ, and the model which obtains the pose $\mathbf{p}_t$ from $\mathbf{p}_{t-1}$ is

$$\begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} = \begin{pmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \end{pmatrix} + \begin{pmatrix} \hat{d}_t \cos(\theta_{t-1}) \\ \hat{d}_t \sin(\theta_{t-1}) \\ \hat{\delta}_t \end{pmatrix}, \tag{1}$$

where $\hat{d}_t = d_t + \epsilon_d$ and $\hat{\delta}_t = \delta_t + \epsilon_{rot}$ are obtained from odometry measurement by adding independent Gaussian noise, where:

$$\epsilon_d \sim N(0, \omega_{range}|d|), \tag{2}$$

$$\epsilon_{rot} \sim N(0, \omega_{turn}|\delta_{turn}| + \omega_{drift}|d|). \tag{3}$$

ω_{range}, ω_{turn} and ω_{drift} represent the range error, the turn error and the drift error of the robot encoder respectively.

B. Tracking of the robot heading

In order to reduce the uncertainty in tracking the position of the robot as much as possible, we use an extended Kalman filter (EKF) [11] to track the heading component of the robot's pose as follows: at step t, we compute the relative orientation between the current pose of the robot $\mathbf{p}_t$ and the previous pose $\mathbf{p}_{t-1}$ using the two omnidirectional images which were recorded at each position. Then by adding this relative orientation to the heading θ_{t-1}, we obtain a vision based observation for the robot's heading at step t. Then by feeding this heading observation to the EKF filter with the robot motion model from Eq. 1 to produce a prediction step of the filter, we estimate the robot pose at step t. The relative orientation between two omnidirectional images is computed based on the epipolar geometry of spherical cameras [12].

C. Loop Closure Using Vision

Loop closing capability is an essential part of any mapping system. In our case, without this capability the robot can face the problem of assigning the same area in the environment to multiple nodes leading to a globally inconsistent map. Therefore, the robot uses its vision sensor to detect loops along the trajectory using place recognition based on image similarity. As mentioned earlier, each node in the map contains a group of image feature points extracted from an omnidirectional image recorded when the node was first created. Using these image features, the similarity between any two nodes in the map is measured using the number of matched feature points between the two groups of features stored in each node [13]. So in order to detect loop closures, the robot calculates the similarity between the current node and all the nodes which are located within a certain radius. When the ratio of the number of matched feature points to the number of features stored in the current node exceeds a certain threshold, a loop is considered detected. As a result, the robot adds a link between the two detected nodes in the graph with a distance of zero. Then TORO, the tree optimizer, is run on the graph to correct the pose estimates of the nodes in the map according to the newly added link.

There are two important parameters that appear in the above loop closure strategy. First, the distance in which the robot checks for a loop closure. The longer the radius the more nodes need to be matched with the current node. In our experiments we use a radius of 3m. Second, the matching threshold which the robot uses to decide that a loop has been closed. This threshold is affected by the texture of the mapped environment, which in turn affects the number of image features that can be extracted from the images. In our experiment a loop is considered closed if 35% of the features are matched (a parameter that was obtained experimentally for our set-up).

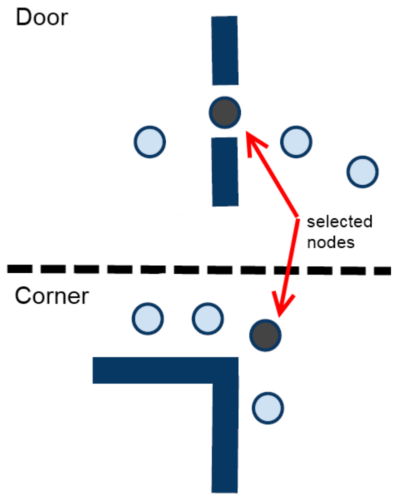

Fig. 2. Maximising the intra-cluster similarity aims to automate the selection of reference views from areas such the middle of the doorways and the edge of the corners.

IV. Graph Pruning by Dual Clustering

The resulting map from the above step is a dense graph containing nodes created at each step the robot has performed. In this section we extract a sparse topological map from the dense graph by deploying a graph pruning step.

In our case, the map is considered as a set of spatial objects with the nodes representing these objects. Each node has two attribute domains, a geometric domain in the XY plane and a non-geometric domain represented by image similarity. The aim is to select a sub-set of these nodes in a way that sufficiently covers the environment, allowing the robot to use the map for tasks such as autonomous visual navigation. In order to do that we use a clustering method called dual clustering [14]. Dual clustering is the process which partitions a set of spatial objects into different clusters in such a way that each cluster forms a compact region in the geometric domain while maximizing the similarity in the non-geometric domain.

In the geometric domain, the clustering algorithm produces compact clusters. This is a preferable effect as we do not want the selected nodes, which correspond to the centers of the clusters, to be very sparse creating gaps in the final map. In the non-geometric domain (i.e. image similarity), the clustering algorithm selects the centers of the clusters in a way which maximizes the intra-cluster similarity. The effect of this process is also preferable for our case because, in the cases where a cluster of nodes expands through a doorway or a corner, the center of the cluster will be selected from the middle of the doorway or the edge of the corner. This selection allows the node to cover both sides of the door and the corner preventing a discontinuity in covering the environment, which can be a problem when actually using the map for tasks such as visual navigation. Fig 2 illustrates this situation.

A. The clustering algorithm

Our clustering algorithm is inspired by a fast implementation of the dual clustering method presented in [14]. The algorithm performs clustering based on density in the geometric domain, while maximizing the similarity in the non-geometric domain. In order to achieve this, for each cluster center in the geometrical domain, the neighborhood of a given radius Ψ_d should contain a minimum number of points. And in the non-geometric domain, the similarity between the neighborhood points of each cluster and the center of that cluster should be above a certain threshold Ψ_s. The clustering process is performed incrementally as follows:

1) Initialize a cluster by starting from the first unclassified node in the graph N_p (based on the time stamp). If the number of nodes in the neighborhood of N_p is less than a predefined threshold κ the node is ignored. Otherwise, create a new cluster C.
2) Insert all nodes from the neighborhood of N_p, which have similarity with N_p greater than Ψ_s, in C.
3) Compute the center of the cluster N_{ref} by selecting the node which is most similar to all other nodes in the cluster, as
$$N_{ref} = \arg\max_{k \in C} (\sum_{j \in C} sim(N_j, N_k)),$$
where $sim(N_j, N_k)$, the similarity between the two views in the nodes N_j and N_k, is the number of matched image features.
4) Check an arbitrary node N_q, from the neighborhood of N_p. If the number of nodes in the neighborhood of N_q is at least κ, and the similarity between an unclassified node N_o in the neighborhood of N_q and the center of the cluster is above Ψ_s, then insert N_o to the cluster C and recompute the center of the cluster as in step 3. Repeat step 4 until the cluster C can not be extended any more.
5) Repeat all the steps until all the nodes in the graph are classified.

We discuss how to set the clustering parameters in our experiments in Section VI.

V. Using the Map for Navigation

Every map can be judged by its usefulness for practical purposes. In our case the map is used for a path following routine inside an indoor environment.

When robots work inside an indoor environment, their navigation generally is restricted to what the humans consider to be a path inside that environment, such as corridors and the areas between the furniture. These routes effectively simplify the task of navigation by limiting the robot to only one degree of freedom along the path. And by representing this path as a sequence of images, the following framework

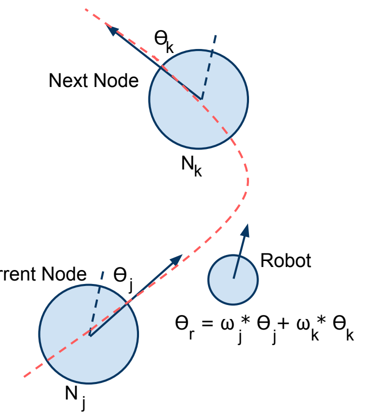

Fig. 3. The proposed visual navigation strategy. N_j is the current node in the path and N_k is the next node. The dashed line is the path, θ_j and θ_k are the relative orientations between the robot's heading and the reference orientation of the nodes N_j and N_k respectively. θ_r is the robot's desired heading.

of the appearance-based approach for visual navigation is used in the literature [15], [16], [17]:

- The path is first built during a learning phase where the robot is controlled by a human operator. During this phase the robot captures a sequence of images along the path.
- A subset of the captured images is selected to represent the reference images along the path.
- During the replay phase, the robot starts near the first position and is required to repeat the same path.
- The robot extracts the control commands of its motion by comparing the currently observed image with the reference images along the path.

In this work we adopted a similar framework for visual path following using a sequence of nodes from the map. Fig. 3 illustrates the navigation strategy. First the robot localizes itself to one of the nodes in the path. This is done by selecting the node which has the the highest similarity score with the currently observed view. Let S_j be the similarity score, i.e. the number of matched points. The similarity score is also computed between the current view and the next node in the sequence. Let S_k be the similarity score with the next node. Then the following ratio is computed:

$$\omega_j = \frac{S_j}{S_j + S_k}, \; \omega_k = \frac{S_k}{S_j + S_k}. \tag{4}$$

The heading angle θ_r is computed as a weighted sum:

$$\theta_r = \omega_j * \theta_j + \omega_k * \theta_k. \tag{5}$$

where θ_j and θ_k are the relative orientation between the current view and the nodes N_j and N_k respectively (see Fig. 3). By following this navigation strategy, the nodes in the path can be considered as directional signs which lead the robot toward its goal.

In order to estimate the relative orientation between two views, we use epipolar geometry. The method first estimates

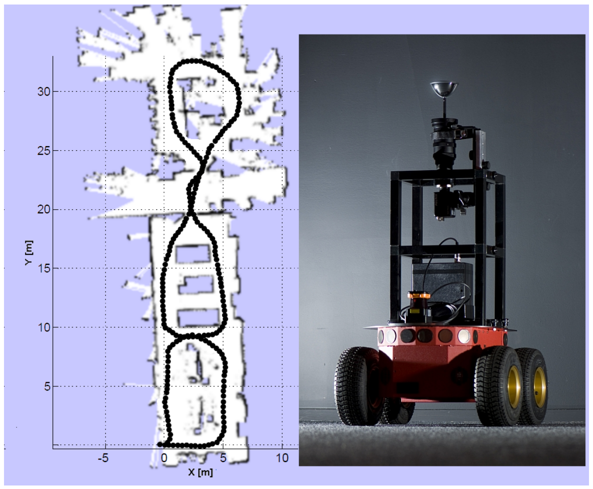

Fig. 4. Left: The true trajectory of the robot. Right: The experimental platform. An ActivMedia P3-AT robot equipped with an omnidirectional vision system.

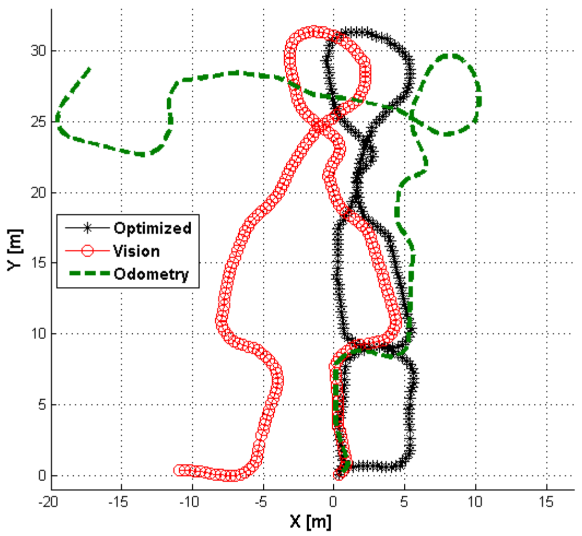

Fig. 5. The green dashed line represents the trajectory of the robot from the odometry. The red line is the result of using vision estimated relative orientation as an observation with an EKF filter. The black line is the final output after loop closing and graph relaxation.

the so-called essential matrix $\mathbf{E}$. Then, based on the method introduced by Hartley and Zisserman in [18], the essential matrix is factored to give Eq. 6 which contains the rotation matrix $\mathbf{R} \in SO(3)$ and the skew-symmetric matrix $[\mathbf{t}]_\times$ of the translation vector $\mathbf{t} \in \mathbb{R}^3$.

$$\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}. \tag{6}$$

After that, the the relative orientation is extracted from the rotation matrix $\mathbf{R}$.

VI. Experiments and Results

Our experimental platform is an ActivMedia P3-AT robot equipped with a GigE progressive camera (Jai TMC-4100GE, 4.2 megapixels) with a curved mirror from 0-360.com. The following experiment was carried out in an office floor at the University of Lincoln. We drove the robot on a tour between the offices while recording a set of omnidirectional images. The resulting database contains 222 images with approximately 35 cm between the consecutive images. Fig. 4 shows the positions of the recorded images obtained using the GMapping library [19]. The GMapping algorithm provides a SLAM solution using laser range-finder scans based on a Rao-Blackwellized particle filter. The output of the algorithm is an estimate of the robot trajectory along with an occupancy grid map of the environment. We would like to emphasize that our method does not use laser scan matching. We use this information for visualisation purposes only.

The robot starts from location $(x_0 = 0, y_0 = 0)$ and comes back to the same start point. The dashed line in Fig. 5 shows the trajectory of the robot based on the odometry alone where the drift effect is clear. Fig. 5 also shows the trajectory after we added the EKF as a filter for the observation of the robot heading. Although the robot does not have any method to detect loop closure, the resulting error in estimating the trajectory is smaller. The final output of the mapping step is shown as well in Fig. 5, where the graph optimization algorithm produced a map which is globally consistent.

In the next step, we applied our dual clustering algorithm which pruned the dense graph produced from the previous step and generated the final map. The number of nodes in the final map is affected by the initial neighborhood radius Ψ_d and the image similarity threshold Ψ_s. The parameter Ψ_d gives an initial radius for the node to cover and then the algorithm expands the node based on the image similarity threshold Ψ_s. In our experiments the map is intended to be used for autonomous visual navigation; therefore the sparsity of the map should not result in gaps where the robot cannot estimate its heading relative to one of the nodes in the map. In order to achieve that we assign Ψ_d to 1 m as a minimum distance between the nodes and Ψ_s to 35 feature points as the minimum number of matched points between any view in the node and the center of the node; and finally we assign κ, the minimum number of points required in the neighborhood of any node, to 1 m. Fig. 6 shows the result of the pruning step where a set of 23 nodes was selected.

In order to test the map for visual navigation, we performed five path following runs using the nodes of the map. The paths were chosen randomly, while covering all nodes in the map. At the start of each run the robot was given a sequence of nodes to follow. The robot then followed each path using the navigation strategy presented in Section. V.

In addition, an array of sonar sensors was used for obstacle avoidance. The same five runs were then re-executed using manual drive where a human driver steers the robot taking the best track where the robot was driven through the shortest distance and at the same time was kept away from obstacles. Table I shows the results for both the autonomous and the manual runs. In order to show the robustness of the navigation procedure, the 5 autonomous runs were executed

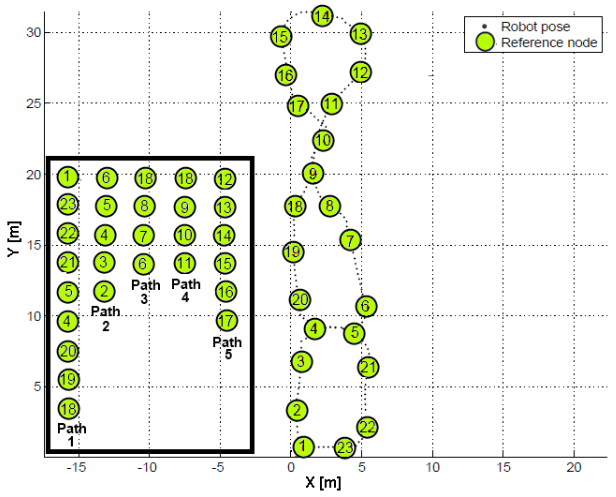

Fig. 6. The selected nodes from the dual clustering algorithm along with five sequences of nodes which the robot was given to follow.

twice. The mean and minimum sonar range distances to obstacles were calculated for each run along with the traveled distance. These values are used as an indication about the quality of the navigation performance. As the results show, although the robot takes a slightly longer distance to reach its goal, the autonomous routes were smooth and similar to the manual runs. The average mean distance to any obstacle was 1.00 m and for the autonomous runs was 0.88 m. The average value of the minimum range to obstacles was 0.49 m for manual driving and 0.44 m for the autonomous ones.

VII. Conclusion

This paper introduced a minimalistic mapping method using an omnidirectional vision sensor. The produced map is hybrid with two levels of representation, global and local. On the global level, the world is represented as a graph of adjacent nodes with each node containing a group of image features. On the local level, the features inside each node form a spherical view, which is used for estimating the robot's heading using multi-view geometry. The map is built using a sequence of images along with odometry information. The global consistency of the map is achieved using a graph optimization algorithm. In order to reduce the number of nodes in the map, a dual clustering algorithm for post-processing the initial map was developed. The map was used in an experiment where the robot performed multiple path following tasks.

TABLE I
Vision guided navigation results

		Distance [m]	Mean Range [m]	Minimum Range [m]
Path 1	Manual	22.87	0.85	0.43
	Auto1	24.07	0.82	0.42
	Auto2	24.28	0.82	0.44
Path 2	Manual	14.30	1.20	0.58
	Auto1	15.34	0.90	0.45
	Auto2	14.90	0.99	0.48
Path 3	Manual	9.81	0.94	0.56
	Auto1	10.66	0.85	0.41
	Auto2	10.70	0.87	0.51
Path 4	Manual	8.97	1.04	0.42
	Auto1	9.27	0.97	0.37
	Auto2	9.09	1.02	0.42
Path 5	Manual	15.52	0.98	0.48
	Auto1	16.42	0.85	0.46
	Auto2	16.69	0.78	0.47

References

[1] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4(4):333–349, 1997.

[2] F. Fraundorfer, C. Engels, and D. Nistr. Topological mapping, localization and navigation using image collections. In *Proc. IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 3872–3877, 2007.

[3] E. Eade and T. Drummond. Monocular slam as a graph of coalesced observations. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 1–8, 2007.

[4] K. Konolige, J. Bowman, J. D. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua. View-based maps. *The International Journal of Robotics Research*, 29(8):941, 2010.

[5] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[6] M. Cummins and P. Newman. Probabilistic appearance based navigation and loop closing. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages 2042–2048, 2007.

[7] M. Jogan and A. Leonardis. Robust localization using an omnidirectional appearance-based subspace model of environment. *Robotics and Autonomous Systems*, 45(1):51–72, 2003.

[8] O. Booij, Z. Zivkovic, and B. Krose. Sampling in image space for vision based SLAM. In *Proc. Workshop on the Inside Data Association, Robotics: Science and Systems Conference (RSS)*, 2008.

[9] V. Ila, J. M Porta, and J. Andrade-Cetto. Information-based compact pose SLAM. *IEEE Transactions on Robotics*, 26(1):78–93, 2010.

[10] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Proc. of Robotics: Science and Systems (RSS)*, 2007.

[11] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian bayesian state estimation. In *Radar and Signal Processing, IEE Proceedings F*, volume 140, pages 107–113, 2002.

[12] C. Geyer and K. Daniilidis. A unifying theory for central panoramic systems and practical implications. *Proc. European Conference on Computer Vision (ECCV)*, pages 445–461, 2000.

[13] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded Up Robust Features. In *Proc. European Conference on Computer Vision (ECCV)*, 2006.

[14] J. Zhou, F. Bian, J. Guan, and M. Zhang. Fast implementation of dual clustering algorithm for spatial data mining. In *Proc. International Conference on Fuzzy Systems and Knowledge Discovery*, volume 3, pages 568–572. IEEE Computer Society, 2007.

[15] G. Blanc, Y. Mezouar, and P. Martinet. Indoor navigation of a wheeled mobile robot along visual routes. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages 3354–3359, 2005.

[16] T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool. Omnidirectional Vision Based Topological Navigation. *International Journal of Computer Vision*, 74(3):219–236, 2007.

[17] O. Booij, B. Terwijn, Z. Zivkovic, and B. Krose. Navigation using an appearance based topological map. *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[18] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, March 2004.

[19] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with Rao-Blackwellized particle filters. *IEEE Transactions on Robotics*, 23(1):34–46, 2007.