The role of homing in visual topological navigation

Ming Liu, Cédric Pradalier, François Pomerleau, Roland Siegwart
Autonomous Systems Lab, ETH Zurich, Switzerland
[firstname.lastname]@mavt.ethz.ch, rsiegwart@ethz.ch

***Abstract*—Visual homing has been widely studied in the past decade. It enables a mobile robot to move to a *Home* position using only information extracted from visual data. However, integration of homing algorithms into real applications is not widely studied and poses a number of significant challenges. Failures often occur due to moving people within the scene and variations in illumination. We present a novel integrated indoor topological navigation framework, which combines odometry motion with visual homing algorithms. We show robustness to scene variation and real-time performance through a series of tests conducted in four real apartments and several typical indoor scenes, including doorways, offices etc.**

I. Introduction

VISUAL navigation encompasses a range of techniques, including appearance-based navigation [1], [2] and feature map based navigation [3], [4]. Broadly speaking, it uses visual sensor information to map an unknown environment, localize the robot within a map and automatically drive the robot from waypoint to waypoint. Several mapping algorithms based on range finders have attracted attention, such as `gmapping`[1] and `karto`[2], but require considerable cost and weight budget. Reliable visual navigation algorithms are still necessary.

Concerning visual sensors, perspective cameras [4], stereo vision systems [5], or omnidirectional cameras [6] have been use in visual navigation systems. Among these sensors, omnidirectional camera has been widely used in visual navigation system, because of its 360° panoramic view.

There are several existing approaches in this field. An important difference among those approaches is perception. Initially regional template matching [1] [7], and texture based methods [8] were used. However, these approaches were not tested with large scale maps in general. The development of the visual keypoint descriptors, such as SIFT[9] and SURF [10], along with developments in computer vision techniques has significantly advanced the evolution of visual navigation. Plenty of visual navigation algorithms have been thus developed using these features [5], [11], [12]. Several practical descriptors for omnidirectional vision were also developed for topological mapping or navigation, such as fingerprint of the environment [13], vertical line based descriptors [14], FACT (fast adaptive color tags) [15], [16] and so on.

This work was supported by the EU project Robots@home (IST-6-045350) and EU FP7 project NIFTi (contract # 247870)

[1]http://www.ros.org/wiki/gmapping

[2]http://kartorobotics.com

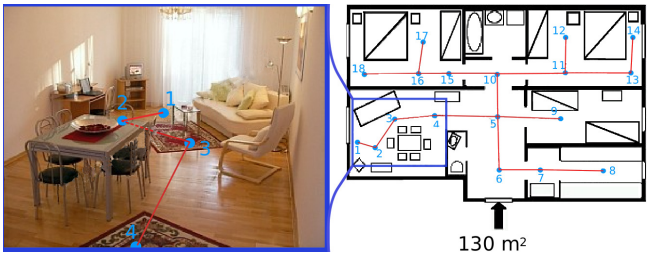


Fig. 1. An application instance in a 130m^2 apartment at Vienna Openring

There are two main frameworks of visual navigation algorithms amongst others. The first method requires to first track and then reconstruct [17], [4], [14], [5] the environment using a set of features. These methods are usually able to provide a global metric representation of the environment. However, there are risks that the constructed map may be lost due to error accumulation, leading to failure. The second framework is known as topological visual navigation. They are usually based on key-frame matching [2] to first localize the robot, before techniques such as visual homing [18], [19], [20], [21], [12] are used to reach predefined topological nodes. Topological visual navigation provides several advantages over the former framework including:

a) Sparse representation: Topological maps used for a topological visual navigation are created incrementally, where only feature changes are considered. A typical representation of the environment is a collection of visual features at certain poses. The computational and memory cost is usually low.

b) Independence from precise maps: Visual homing is less sensitive to error accumulation, commonly occurring in metric mapping. As such, a precise map of the environment is not required to ensure its success.

c) Lightweight planning: The path planning in metric maps can be computationally very expensive. In the contrary, visual topological navigation incurs a relatively low cost as planning is based on graph structure.

In our work, we adopt visual homing for visual topological navigation using an omnidirectional camera as the only sensor for navigation. A stereo vision system is also used for local obstacle avoidance.

A. Contributions

The goal of this work is to present a generalized lightweight visual navigation framework, which integrates

visual homing algorithms and odometry for mobile platforms. In our previous work [12], we introduced a state-of-art visual homing algorithm using image based visual servoing. We use this approach as a test case for the visual homing algorithm.

We apply this framework in order to learn and navigate a real apartment environment in real-time.

The proposed navigation system is built on a topological graph, with an example shown in the left of figure 1. Each node on the graph represents a pose (or navigation waypoint) in the environment. The system is modeled as a state machine structure, whereby the navigation and mapping processes can be switched. The transition between two neighbouring waypoints is performed by a two-phase algorithm using odometry and visual servoing. These two phases are denoted by *general positioning* and *pose stabilization*.

Regarding real-time applications in real environments, we found that it is important to assess how well-suited a particular position is to be considered as a waypoint. A poor choice of waypoint will easily lead to failure of the visual homing. Based on our study, the number of matched features is usually adequate, however the feature distribution of a certain position is the paramount. An isotropic distribution of features will generate more precise homing vectors than a poorly distributed one. In this work, we design and evaluate a validation criterion to assess the quality of candidate waypoints.

The paper is organized as follows. We start from a systematic level, introducing the structure of the navigation system in section II. In Section III, we will discuss the criteria to define the quality of a waypoint in a given scene. Experimental test results are presented in Section IV, followed by discussion and conclusions in section V.

II. Mapping and Navigation

A. System overview

At the system level, the navigation task is structured as shown in figure 2. Configuration layers such as sensors, data and low level processes are shown in different colors. The omnidirectional camera is used as the main sensor for navigation, while a 3D stereo vision system is used for local obstacle avoidance.

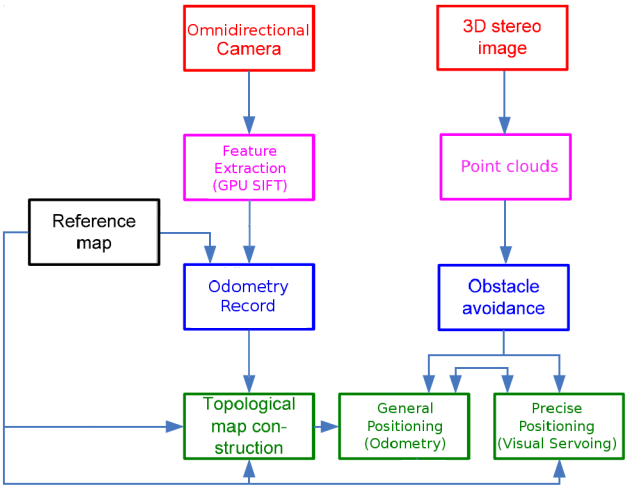


Fig. 2. The system structure

Due to the multi-layered system structure, an efficient management mechanism needs to be defined. A state-machine was built up as shown in figure 3, an can be divided into two major parts: mapping and navigation.

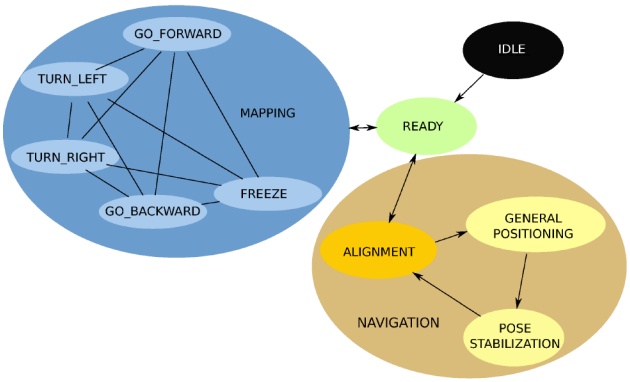


Fig. 3. A sketch of the state machine of the navigation system.

B. Mapping

The first issue regarding the mapping problem is how to represent the map and descriptors for topological nodes. Using a typical topological map, the high-level structure is a simple graph, which uses the vertices to represent the nodes and edges to represent the translation relationships between nodes. This is shown in figure 1. For each node, a structure containing keypoint features and odometry differences to neighbors is created. Specifically, the MAPPING state in figure 3 represents the mapping process, which requires basic states such as moving and freezing of the robot. When the robot arrives at the reference position, a node will be created upon the request from the user. The node structure will be registered locally and also saved in a remote database. The MAPPING state can always be triggered whenever the robot is in the READY state. It is a useful feature because it allows extending the topological map from any node, regardless whether or not the node was newly created, e.g. the mapping process can also start from any intermediate node occurring on the route. As shown in figure 1, a global map can be easily extended to several branches and dynamically managed.

C. Localization in the map

The localization in a topological map means that the robot can name its current location as the nearest node. The performance of the localization will greatly affect the navigation in the map. Therefore, we implemented a localization method that relies on visual feature matching.

We compare the current set of feature points F_c with each feature set $F_i, i \in [1,n]$ in the database. We look for the best F_{best} and second best F_{second} matching nodes, according to the matching ratio r_{best}, r_{second}. To ensure positive matching, we require

$$\frac{r_{second}}{r_{best}} \leqslant 70\%$$

When this constraint is met, the current node is exclusively located. Otherwise, the localization will return several possibilities which have to be verified at a later time. At extreme cases, the robot might be incorrectly localized, causing navigation process based on the localization to fail. The localization process can be triggered again at the failure poses.

D. Navigation

A global planner based on Dijkstra algorithm will perform path planning over all topological nodes in the graph. It results in a sequence of nodes which the robot needs to follow in order to navigate from the current position to the target position. As such, the topological navigation is decomposed into several node-to-node phases. Each node-to-node action is performed through the NAVIGATION state shown in figure 3.

The ALIGNMENT state means the robot attempts to align its orientation (heading direction) with the pose saved in the database. The translation between two nodes is handled by a two-phase method: *general positioning* (state=GENERAL POSITIONING) and *pose stabilization* (state=POSE STABILIZATION). The former is a combination of obstacle avoidance and odometry based position control. The latter is used to correct the accumulated odometry error for pose stabilization.

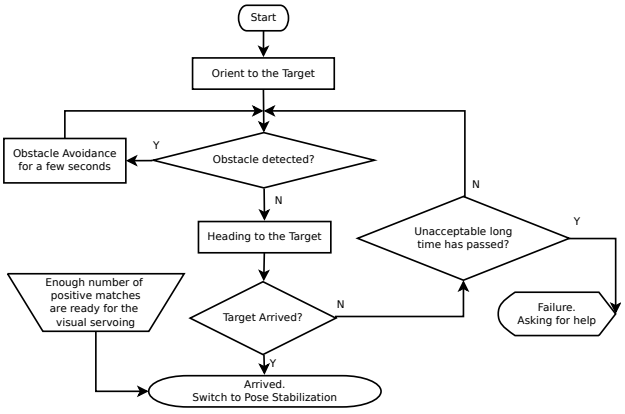


Fig. 4. The flow chart of the *general positioning* process

The procedure of *general positioning* is shown in figure 4. Two exceptions during the positioning process are "Target arrived" and "Obstacle detected". When obstacle(s) are detected, the system will switch to an exclusive obstacle avoidance control process. The mechanism ensures that when the robot's path is not blocked. In particular, the obstacle avoidance mode will take over the motion control using a naive obstacle contour following law. We set the time span for obstacle avoidance mode to 6 seconds for experiments in a typical apartment environment. After the obstacle avoidance process, the robot will try to reach the target again using the "Heading to the Target" process. The "heading to the Target" process is based on the odometry difference between current position and the target position.

As for the finishing conditions, there are two possible states during *general positioning*. They both can trigger the finish of this procedure. During the process of *general positioning*, once enough positively matched keypoints are observed, the working state will switch to *pose stabilization* automatically. This means the current position is near enough for a robust pose stabilization process. Second, when the odometry of the current position matches the target odometry within a bounded error region, the state-machine will also switch to the POSE STABILIZATION state.

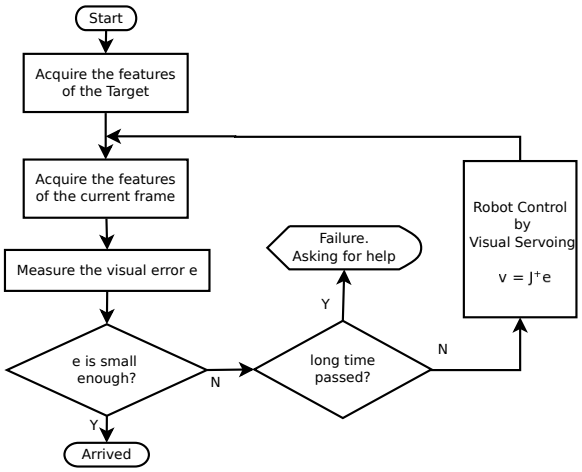


Fig. 5. The flow chart of the *pose stabilization* process

The *pose stabilization* is organized under the image based visual servoing (IBVS) framework as shown in figure 5. It uses an omnidirectional camera as the only sensor, and tries to correct the error in the image space by controlling the robot motion. It is able to perform a visual homing algorithm in real-time, by which the accumulated odometry error will be corrected. The algorithm retrieves the target image and features from the database and matches the current image with the target image. These images features are fed into the core visual servoing algorithm to generate homing vectors which directs to the target waypoint. For example, we could use the homing algorithm introduced in [12], where the homing vectors are calculated from the scale differences and the bearing angles. The algorithm shown in Eq. 1 and 2.

$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} = \sum_{i=1}^{n} \lambda_i (s_i - s_i^*) \begin{pmatrix} \cos\beta_i \\ \sin\beta_i \end{pmatrix} \tag{1}$$

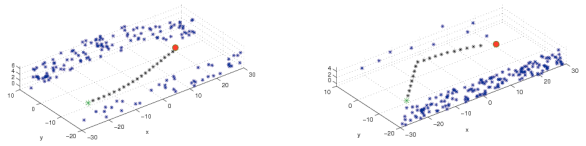
$$\begin{pmatrix} v_x \\ v_y \\ \omega \end{pmatrix} = \lambda \begin{pmatrix} -\frac{s_1^* l_1^*}{l_1^2}\cos\beta_1 & -\frac{s_1^* l_1^*}{l_1^2}\sin\beta_1 & 0 \\ \vdots & \vdots & \vdots \\ -\frac{s_n^* l_n^*}{l_n^2}\cos\beta_n & -\frac{s_n^* l_n^*}{l_n^2}\sin\beta_n & 0 \\ -\frac{1}{l_1}\sin\beta_1 & -\frac{1}{l_1}\cos\beta_1 & -1 \\ \vdots & \vdots & \vdots \\ -\frac{1}{l_n}\sin\beta_n & -\frac{1}{l_n}\cos\beta_n & -1 \end{pmatrix}^{+} \begin{pmatrix} s_1 - s_1^* \\ \vdots \\ s_n - s_n^* \\ \beta_1 - \beta_1^* \\ \vdots \\ \beta_n - \beta_n^* \end{pmatrix} \tag{2}$$

where v_x and v_y are the velocity components in the 2D navigation plane; s_i is the observed scale of a keypoint and s_i^* is the target scale of the corresponding reference keypoint; β_i is the bearing angle of the corresponding keypoint. The proof of stability based on Lyapunov theory and evaluation of the algorithm was given in [12].

III. Node validation

In order to enhance the robustness of the navigation system, we analysis the bottlenecks of the algorithm. One important fact is that sometimes the visual features are not stable enough for the navigation; in other words, the quality of references is poor. This problem is common to any visual navigation method, however an efficient metric to assess it is unavailable. To this end, we use entropy analysis to evaluate this quality.

To illustrate the problem, let us consider the two simulated situations shown in figure 6. In both cases, 160 features can



(a) Good Feature Distribution (b) Bad Feature Distribution

Fig. 6. Effect of different feature distribution

be observed, marked with blue '*'s. We show the green '*'s as the starting position and the red filled circles as the *home* position. The robot uses the same visual homing method in equation 1 to move from the starting position to the *home* position automatically. The paths marked by black '*'s are the simulated trajectories.

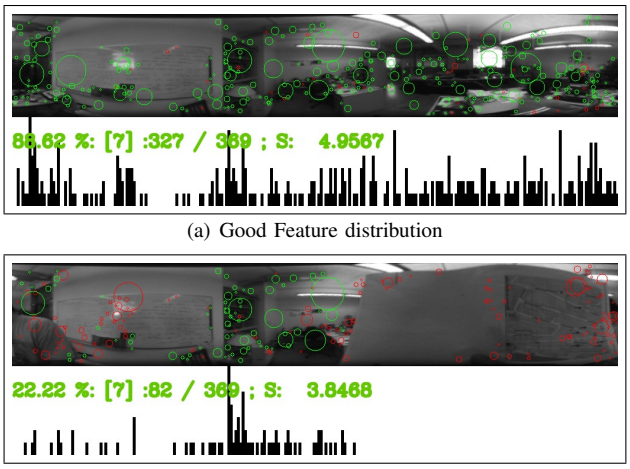
Figure 6(b) shows an extreme case of the feature distribution, i.e. most of the features are from one certain direction. Figure 6(a) depicts the case that 160 features are isotropically distributed on the simulated walls. Although in both cases the robot can observe the same amount of features and manages to move to the *home* position, the simulated trajectories are dramatically different. We could infer that a more isotropic distribution leads to a smoother trajectory. Moreover, it results in smoother control variables for the actuators.

We now consider the distributions of the features in a real situation. We took an image using an omnidirectional camera, and changed the feature distribution manually, using a plain paper to intentionally cover the field of view. A screen-shot of the angular feature distributions is shown in figure 7. The bottom histograms show how many features lie in a given vertical sector of the unwrapped omnidirectional image.

The number followed by the percent sign is the positive matching ratio, the integer in squared brackets is the maximum number of matching points within a single bin in the histogram. After that is the number of matching keypoints and at the end is the score "S", which should reflect the isotropicness of the distribution. The score "S" is calculated from the entropy of the distribution of the positive matches using

$$S = -\sum p_i \ln p_i \tag{3}$$

where p_i is the ratio of the matches in each bin over the whole histogram. The maximum score can be calculated from a uniformly distributed histogram. Empirically, if S is

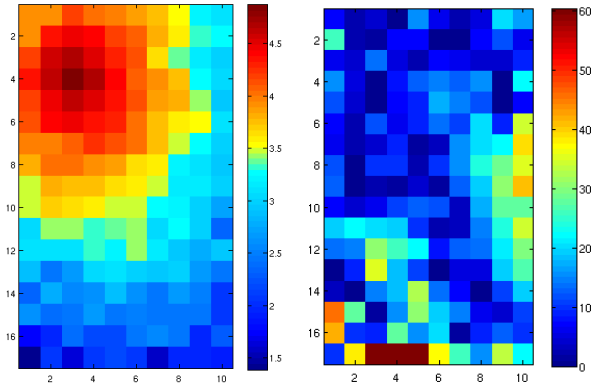


(a) Good Feature distribution

(b) Bad Feature distribution

Fig. 7. Histograms of the feature distribution. There are two kinds of circles that mark the matching keypoints : green (brighter) and red (darker). The green (brighter) circles mark the matching results after RANSAC; the red (darker) circles mark the rejected matches.

greater than 80% of this maximum score, the reference home position is acceptable. [3] It's worth noticing that the number of matched features (82) is still high in figure 7(b). Typically, in a less textured indoor environment, the homing method can work with a minimum number of positive matches around 40. This indicates that the number of positive matches can not fully indicate the quality of the description.



(a) Scores of matched features (b) AAE regarding to position (2,3)

Fig. 8. Relation between *AAE* and characteristics in features

In order to further valid this metric for node evaluation, we carried out a test on a widely cited indoor dataset [22]. Taking position (2,3) as reference, the average angular error (*AAE*) and the score are shown in figure 8. According to [23], *AAE* is obtained as follows:

$$AAE(\mathbf{ss}) = \frac{1}{mn}\sum_{x=1}^{m}\sum_{y=1}^{n} AE(\mathbf{ss}, \mathbf{cv}_{xy})$$

where AE is the absolute angular error between the calculated homing vector and ground truth. **ss** means saved scene, and

[3] In figure 7, there are 256 bins in the histogram. The maximum score is calculated by $S = \ln 256 = 5.55$, therefore the score threshold is 4.4

cv means current view. It shows that the shape of the error distribution is correlated to the shape of the score distribution, which means that our criterion can represent the quality of the description to a certain extent.

IV. Experiments and Results

The navigation system has been tested at several indoor environments with different setups. Figure 9 shows a collection of robot images and a typical sketch of robot structure.

(a) BIBA Robot (b) James Robot (c) Typical setup

Fig. 9. Robots collections.

As shown in figure 9(c), a pan-tilt unit is mounted to hold the stereo camera system. The motion of the pan-tilt unit will follow the control speed of the robot, in order to predict and observe potential obstacles. The detected local obstacles will be registered onto a local navigation cost-map [4]. The cost-map is then used to constrain the trajectory of the robot.

The experiment has been carried out separately both in day-night-time conditions, in order to test the robustness to light changes.

A. Functional test

An experiment of around 20min depicted by a plot of the visual servoing error is shown in figure 10. In order to get a more compact result, the logging for this plot will only be triggered when the system is switched to *pose stabilization* mode. It means that every time the error drops below the green bar (which marks the threshold of the accuracy), the system switches to *general positioning* state. It is important to note that there are time spans after each mode switching, which are not visible to the readers from the plot. The figure depicts that the fatal error can converge for every control cycle. An issue should be mentioned, regarding the areas marked with brown rectangles. In these areas, unstable convergences can be observed. After the analysis of the log and experiment process, the reason for this kind of behavior can be identified as follows. Our testing robot was a differential driven robot with non-omnidirectional dynamics. Therefore, the constraints in the motion model implies that the robot motion can reach singularities. This occurs in the cases when the extension of the straight line between the two active wheels passes through the target position. To solve this problem, we propose the following control law. The homing vector is translated into a linear speed v_L and a rotation speed ω, using (4). K_α and k_α are two parameters defining respectively the rotation speed gain and the trade-off between movement speed and heading alignment.

$$\begin{aligned} \alpha &= \arctan 2(v_y, v_x) \\ v_L &= e^{k_\alpha \alpha^2} \cdot \left\| \begin{matrix} v_x \\ v_y \end{matrix} \right\| \\ \omega &= K_\alpha \cdot \alpha \end{aligned} \tag{4}$$

Due limited space and its irrelevance to the proposed framework, we will not go into details of the analysis in this report.

B. Longer time roaming test

In order to test our visual navigation approach over a longer timeframe, we used a simple script to generate a sequence of random target nodes covering the whole transitional space. From a full battery status to empty, the robot drove around for 1 hour, successfully reaching more than 40 different targets. The total graph, shown in figure 1, contains 18 topological nodes in a 3 bed-room apartment including kitchen and living room ($130m^2$ in all). During the test, four people were randomly walking around the apartment. This confirms the reliability of our approach. For more intuitive result, please refer to the attached video [24] [5].

The existing test results in real apartment environment showed the robustness of our method against some illumination changes and occurrence of dynamic objects in the scene.

The results show that our method worked well at the nodes with richly textured scenes and unstable in low textured ones. It means that a memory mechanism combining working memory and long-term knowledge is necessary for a long-term navigation application. This part of work will be carried out in our further researches.

V. Conclusion

In this paper, we presented a visual topological navigation framework, representing a step towards a low-cost and robust solution to indoor navigation problem. Besides the mapping and localization parts, the navigation module consists of two phases: *general positioning* and *pose stabilization*. The system is organized as a state machine, which grants freedom to switch among these tasks easily. It uses omnidirectional camera as the only sensor for generating homing vectors and a stereo vision system to detect obstacles. It has been tested in several indoor environments, including apartments, office. The results show its reliability and robustness to dynamic scene changes, whilst managing collision free motion.

Supplementary Video

This video shows two experiments illustrating the performance of our visual homing approach in different setups. The first part of the video shows how the visual navigation can be implemented to move along a sequence of homing waypoints. Obstacles inserted on the path of the robot while homing show how alternating phases of dead-reckoning navigation and visual servoing help to integrate a reactive obstacle

[4]The implementation of the cost-map uses the ROS package on: http://www.ros.org/wiki/costmap_2d

[5]This link is a non-public link and never shown to public, which does not violate the copyright principles.

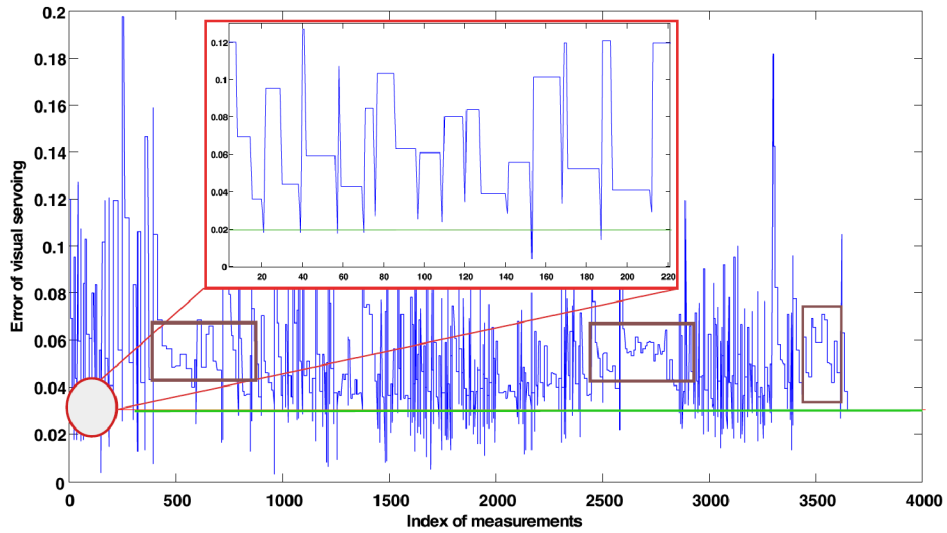


Fig. 10. Pose stabilization error vs measurement indices. The red rectangle is a magnified view of the errors in the corresponding region. The verticle axis is in the unit of visual servoing error.

avoidance mechanism. The second example hows a more complete navigation setup, with higher navigation speed, and more challenging lighting conditions.

REFERENCES

[1] C. Andersen, S. Jones, J. Crowley, C. Person, and C. Andersen, "Appearance based processes for visual navigation," in *IEEE International Conference on Intelligent Robots and Systems*. Citeseer, 1997, pp. 551–557.

[2] O. Booij, B. Terwijn, Z. Zivkovic, and B. Krose, "Navigation using an appearance based topological map," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 3927–3932.

[3] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid, "A Constant-Time Efficient Stereo SLAM System," in *Proceedings of the British Machine Vision Conference (BMVC)*. Citeseer, 2009.

[4] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[5] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid, "Rslam: A system for large-scale mapping in constant-time using stereo," *International journal of computer vision*, vol. 94, no. 2, pp. 198–214, 2011.

[6] N. Winters, J. Gaspar, G. Lacey, and J. Santos-Victor, "Omni-directional vision for robot navigation," in *Omnidirectional Vision, 2000. Proceedings. IEEE Workshop on*. IEEE, 2000, pp. 21–28.

[7] Y. Matsumoto, K. Ikeda, M. Inaba, and H. Inoue, "Visual navigation using omnidirectional view sequence," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'99)*, 1999, pp. 317–322.

[8] P. W. C. Maciel and P. Shirley, "Visual navigation of large environments using textured clusters," in *I3D '95: Proceedings of the 1995 symposium on Interactive 3D graphics*. New York, NY, USA: ACM, 1995, pp. 95–ff.

[9] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[10] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *Lecture notes in computer science*, vol. 3951, p. 404, 2006.

[11] M. Cummins and P. Newman, "Fab-map: Probabilistic localization and mapping in the space of appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[12] M. Liu, C. Pradalier, F. Pomerleau, and R. Siegwart, "Scale-only Visual Homing from an Omnidirectional camera," in *IEEE International Conference on Robotics and Automation*, 2012.

[13] A. Tapus and R. Siegwart, "Incremental robot mapping with fingerprints of places," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 2429–2434.

[14] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A robust descriptor for tracking vertical lines in omnidirectional images and its use in mobile robotics," *International Journal of Robotics Research*, 2009, special Issue on Field and Service Robotics.

[15] M. Liu, R. S. Davide Scaramuzza, Cédric Pradalier, and Q. Chen, "Scene Recognition with Omnidirectional Vision for Topological Map using Lightweight Adaptive Descriptors," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

[16] M. Liu and R. Siegwart, "DP-FACT: Towards topological mapping and scene recognition with color for omnidirectional camera," in *IEEE International Conference on Robotics and Automation*, 2012.

[17] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Real-time monocular SLAM: Why filter?" in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, Alaska, US, May 2010. [Online]. Available: http://wwwhomes.doc.ic.ac.uk/~strasdat/files/strasdat2010icra.pdf

[18] R. Basri, E. Rivlin, and I. Shimshoni, "Visual homing: Surfing on the epipoles," *International Journal of Computer Vision*, vol. 33, no. 2, pp. 117–137, 1999.

[19] C. Sagüés and J. Guerrero, "Visual correction for mobile robot homing," *Robotics and Autonomous Systems*, vol. 50, no. 1, pp. 41–49, 2005.

[20] K. Bekris, A. Argyros, and L. Kavraki, "Exploiting Panoramic Vision for Bearing-Only Robot Homing," *Computational Imaging and Vision*, vol. 33, p. 229, 2006.

[21] M. Liu, C. Pradalier, Q. Chen, and R. Siegwart, "A bearing-only 2D / 3D-homing method under a visual servoing framework," in *IEEE International Conference on Robotics and Automation*, Anchorage, 2010, pp. 4062–4067.

[22] A. Vardy, "Panoramic image database." [Online]. Available: http://www.ti.uni-bielefeld.de/html/research/avardy/index.html

[23] D. Churchill and A. Vardy, "Homing in scale space," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 1307–1312.

[24] M. Liu, "Video for iros2012." [Online]. Available: {http://www.youtube.com/watch?v=Dn0qX9q_X-4}