Feature based omnidirectional sparse visual path following

Toon Goedemé, Tinne Tuytelaars and Luc Van Gool

VISICS - PSI - ESAT
University of Leuven
Belgium
{*tgoedeme, tuytelaa, vangool*}*@esat.kuleuven.ac.be*

Gerolf Vanacker and Marnix Nuttin

PMA - Department of Mechanical Engineering
University of Leuven
Belgium
{*gerolf.vanacker, marnix.nuttin*}*@mech.kuleuven.ac.be*

Abstract— **Vision sensors are attractive for autonomous robots because they are a rich source of environment information. The main challenge in using images for mobile robots is managing this wealth of information. A relatively recent approach is the use of fast wide baseline local features, which we developed and used in the novel approach to sparse visual path following described in this paper.**

These local features have the great advantage that they can be recognized even if the viewpoint differs significantly. This opens the door to a memory efficient description of a path by descriptors of sparse images. We propose a method for re-execution of these paths by a series of visual homing operations which yield a navigation method with unique properties: it is accurate, robust, fast, and without odometry error build-up.

Index Terms— **Computer vision, robot navigation, path following, omnidirectional images.**

I. INTRODUCTION AND RELATED WORK

The goal of this research is to develop a system for mobile robot *sparse visual path following*, i.e. the re-execution of a path that is defined by images sparsely taken along that path. Fig. 1 illustrates this. Because of their rich environment information content, we use omnidirectional images taken with a catadioptric image sensor mounted on the mobile robot.

The most obvious approach to achieve this is a series of *visual homing* operations. First, the robot is steered towards the place where the first of the path images is taken. From there, the next path image is aimed at, and so forth. Each of these elementary visual homing operations consists of steering a mobile robot from a arbitrary place in the neighborhood towards a place that is defined by an image taken there. In our work we try to work with path images that are taken relatively far from each other. That greatly diminishes the required memory space to describe a certain path.

The essence of our method is a new approach to visual homing. Homing is a term borrowed from biology, where it is usually used to describe the ability of various living organisms, such as insects, to return to their nest or to a food source after having traveled a long distance.
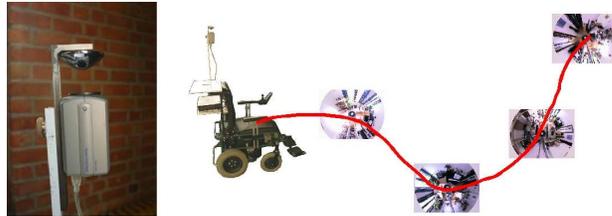
Fig. 1. Left: the omnidirectional camera. Right: A path is defined as a sequence of omnidirectional images relatively far from each other. Path following is performed as a sequence of visual homing operations.

Many researchers have tried to imitate this behavior in mobile robots. Because of the complexity that working with images brings along, there have been many efforts to solve the navigation problem using non-visual sensors. Robots are usually equipped with encoders or inertial sensors that continuously measure the location of the robot with respect to some coordinate system. Unfortunately, the errors of these sensors are considerable and, most importantly, cumulative. For this reason, they are classically supplemented by information provided by ultrasound or laser sensors, mostly through the creation of a grid-based map of the environment [1]. Although many of these methods are quite successful, grid-based approaches must inevitably find a compromise between high space and time complexity and high accuracy regarding the number of grid cells. Moreover, robots based solely on proximity sensors get easily confused because the sensors often return similar readings at different locations. This problem can be patched by using Monte Carlo localization techniques [2].

Vision is, in comparison with these sensors, much more informative. We observe that many biological species, in particular flying animals, use mainly their visual sensors for localization and homing.

Cartwright and Collett [3] proposed the so-called 'snapshot' model, inspired by the visual homing behavior of insects such as bees and ants. They suggest that insects fix the locations of landmarks surrounding a position by storing a snapshot image of the landmarks taken from that position. Their proposed algorithm consists of the construction of a home vector, computed as the average of landmark displacement vectors. Franz *et al.* [4] analyzed the computational foundations of this method and derived its error

and convergence properties. They conclude that every visual homing method based solely on bearing (azimuth) angles of landmarks, inevitably depends on basic assumptions such as equal landmark distances, isotropic landmark distribution or the availability of an external compass reference. Unfortunately, because none of these assumptions hold in our targeted application we search for an alternative approach. Moreover, these methods return no information about home distance, which is necessary information for a mobile robot to determine the appropriate speed and to be able to stop safely at the home position if needed.

Crucial in all visual homing methods is the selection of the landmarks. One must be able to find corresponding pixels between the present image and the target image. Franz [4] used a global optical-flow-like technique on the intensity string measured on the horizon circle. Also, Röfer *et al.* [5] computed a 1D optical flow using a modified Kohonen network. Much more descriptive and robust against occlusions is the technique of *local region matching*. In stead of looking at the image as a whole, local regions are defined around interest points in the images. The characterization of these local regions with descriptor vectors enables the regions to be compared across images. Because the built-in invariance against photometric and geometric changes, correspondences can be found between images with different lighting and different viewpoints.

Many researchers proposed algorithms for local region matching. The differences between approaches lie in the way in which interest points, local image regions, and descriptor vectors are extracted. An early example is the work of Schmid and Mohr [6], where geometric invariance was still under image rotations only. Scaling was handled by using circular regions of several sizes. Lowe *et al.* [7] extended these ideas to real scale-invariance. More general affine invariance has been achieved in the work of Baumberg [8], that uses an iterative scheme and the combination of multiple scales, and in the more direct, constructive methods of Tuytelaars & Van Gool [9], [10], Matas *et al.* [11], and Mikolajczyk & Schmid [12]. Although these methods are capable to find very qualitative correspondences, most of them are too slow for use in a real-time mobile robot algorithm. That is why we spent efforts to speed this up, as explained below.

Related to visual homing is *visual ego-motion estimation*, which has become possible lately due to increasing available processing power and powerful algorithms. Nistér *et al.* [13] presented a *visual odometry* system for real-time ego-motion estimation for a completely calibrated single camera (non-omnidirectional) or stereo rig. They track features and do a RANSAC-based robust estimation of the ego-motion. No 3D map is built. Although very fast, their method does not include loop-closing. Davison [14] developed a single camera SLAM method which estimates the ego-motion of the camera by building sparse probabilistic 3D maps with natural features. A motion model and loop-closing are implemented to reduce errors.

Argyros *et al.* [15] implemented a robot homing system that resembles our approach. The path from start to goal is defined by a sparse set of omnidirectional images. Corners are tracked and the homing vector is computed using vector averaging as in [3], silently assuming an isotropic landmark distribution. Their solution to find correspondences between to consecutive images is based on the tracking of visual features during a previous execution of that same path. This restricts the use to previously executed paths.

The application we envision for our method is a real-time automatic wheelchair [16] for indoor and outdoor use. During a training phase, all places are visited while recording omnidirectional images. Off-line, automatically a topological map is built from these sparse images, wherein nodes are crossroads and edges are paths, each path characterized by a sequence of images. When the patient in the wheelchair communicates a certain goal place to drive to, the wheelchair first localizes itself in the topological map (using a Bayesian localization scheme described in [17]), after which the path towards that goal is translated in a sequence of map images. This paper describes the way the wheelchair is steered along this sparse visual path.

The remainder of this paper is organized as follows. In section II, an overview of the proposed algorithm is presented. The main two steps of this algorithm are described in sections III and IV. Section V details the experiments we have performed and section VI draws a conclusion.

II. ALGORITHM OVERVIEW

Aim of the method is for a mobile robot to re-execute a path that is defined by omnidirectional images taken sparsely, say 2 to 4 metres in a typical indoor environment, from each other along that path. As sketched in fig. 1, following such a sparse visual path boils down to a succession of *visual homing* operations.

One of the advantages of this approach is the fact that no position errors build up during navigation. Each time the movement is relative to a new image position and previously made mapping and localization errors became irrelevant.

Each of these *visual homing* operations is performed in two phases, an initialization phase (section III) and an iterated update phase (section IV).

III. INITIALIZATION STEP

From each position within the reach of a target image (depending on the size and visual complexity of the environment), a visual homing procedure can be started. Our approach first establishes wide baseline local feature correspondences. That information is used to compute a robust estimate of the homing vector and to draft a local map, containing the feature world positions.

A. Wide baseline feature correspondences

Although wide baseline local features are common in computer vision, only recently, a class of *fast* wide baseline local features have appeared. We use the combination of two different kinds of these features, namely a rotation reduced and color enhanced form of Lowe's *SIFT features* [7], and the *invariant column segments* we developed [18].

1) Rotation reduced and color enhanced SIFT: David Lowe presented the *Scale Invariant Feature Transform* [7], which finds interest points around local peaks in a series of difference-of-Gaussian (DoG) images. A dominant gradient orientation and scale factor define an image patch around each interest point so that a local image descriptor can be found as a histogram of the gradient directions of the normalized image patch around the interest point. SIFT features are invariant to rotation and scaling, and robust to other transformations.

A reduced form of these SIFT features for use on mobile robots is proposed by Ledwich and Williams [19]. They used the fact that rotational invariance is not needed for a camera fixed on a mobile robot moving in a plane. Elimination of the rotational normalization and rotational part of the descriptor yields a somewhat less complex feature extraction and more robust feature matching performance.

Because the original SIFT algorithm works on grayscale images, some mismatches occur at similar objects in different colors. That is why we propose an outlier filtering stage based on a color descriptor of the feature patch based on global color moments, introduced by Mindru *et al.* [20]. We chose three color descriptors: C_{RB}, C_{RG} and C_{GB}, with

$$C_{PQ} = \frac{\int PQ \, dx \int dx}{\int P \, dx \int Q \, dx}, \qquad (1)$$

where R, G and B are the red, green, and blue color components along the column segment, centralized around their means. After matching, the correspondences with Euclidean distance between the color description vectors above a fixed threshold are discarded.

Between the image pair in fig. 2 the original SIFT algorithm finds 13 correct matches. Using our rotation reduced and color enhanced algorithm, the matching threshold can be raised so that up to 25 correct matches are found without including erroneous ones.

2) Invariant column segments: In earlier work [18], we developed wide baseline features which are specially suited for mobile robot navigation. Taking advantage of the movement constraints of a fixed camera on a robot moving in a plane, a very simple and fast algorithm can be carried out. The (dewarped) image is scanned columnwise and column segment features are defined between two local maxima of the image gradient. Each column segment is described by an 11-element vector containing geometrical, color and intensity information.
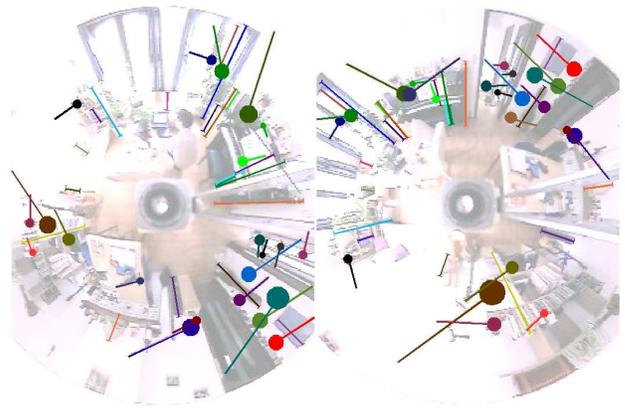

Fig. 2. A pair of omnidirectional images, superimposed with color-coded corresponding column segments (radial lines) and SIFT features (circles with tail).

Fig. 2 shows the matching results on a pair of omnidirectional images. As seen in these examples, the SIFT features and the column segments are complementary, which pleads for the combined use of the two. The computing time required to extract features in two 320×240 images and find correspondences between them is about 800 ms for the enhanced SIFT features and only 300 ms for the vertical column segments (on a 800 MHz laptop). Typically 30 to 50 correspondences are found.

Because only the local features are used and not the very pixel data itself, a path is described very memory efficient by solely the local feature data of the sparse path images.

B. Homing vector and local map initialization step

From a set of feature correspondences an initial homing vector can be estimated. In spite of our efforts, still feature mismatches can occur, for instance in the presence of multiple identical environment objects. Because the homing vector estimation is the basis for navigation and later corrections, a robust estimation technique must be used.

We found the solution in a *random sample consensus* (RANSAC) [21] estimation scheme. Out of a randomly chosen set of two correspondences, the target pose relative to the present pose can be computed using goniometrics. Fig. 3 illustrates this.

The angles labeled with α in the side view (top right) can be extracted from the image coordinates of the features. In order to do this, a 5 degree polynomial is fitted to the data of a calibration image like the one shown bottom right in fig. 3. It is clear that $d_1 \tan \alpha_1 = d_2 \tan \alpha_2$. In other words, we do not know the distances d themselves, but the ratio between them is known.

Now observe the top view, shown bottom left. Using the latter and the fact that the bearing difference angles β can be measured in the image, the quadrangle camera1–feature1–camera2–feature2 can be computed up to an unknown scale-factor.
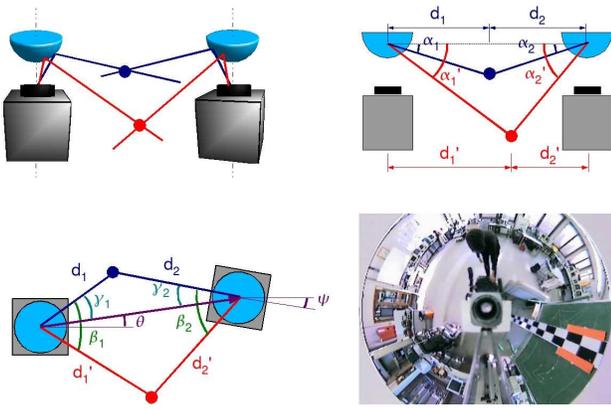
Fig. 3. Estimation of the homing vector between two camera poses given two corresponding visual features. The red and blue lines indicate the light rays towards the two features, respectively. Top left: perspective view, top right: side view (folded open so that the distances d are not shortened), bottom left: top view, bottom right: image used for calibration of angles α.

This yields a homing vector (R, θ, ψ), with (R, θ) the polar coordinates of the target position relative to the present position and ψ the robot orientation change between the two poses. Unfortunately, the vision scale ambiguity forces us to set R arbitrarily at 1.

A second step in the RANSAC algorithm is counting the inlier rate as a measure for the support of the estimated homing vector. For each correspondence the ratio $\frac{d_1}{d_2}$ is computed both as $\frac{\sin \gamma_2}{\sin \gamma_1}$ and $\frac{\tan \alpha_2}{\tan \alpha_1}$. If both outcomes differ too much, the correspondence is considered an outlier.

This calculation is repeated a sufficient number of times (200 in our experiments) on different random samples, and the homing vector with the highest support is kept.

C. Local map estimation

In order to start up the succession of map and location updating iterations, an estimate of the local map must be made. In our approach the local map is a 2D representation of the world, centered at the starting position of the visual homing operation. The world position of every corresponding visual feature used in the previous step must be represented in this map. Because the homing vector and the feature bearing angles seen from both viewpoints are known, the latter is computed easily by triangulation. Fig. 4 gives an example.

IV. UPDATE STEP

When estimates of the homing vector and local map are found, the robot is put into motion in the direction of that homing vector. We rely on a lower-level collision detection and obstacle avoidance algorithm to do this safely. During this drive, images are taken giving information to update the local map and the location of the robot in that local map, also known as *Simultaneous Localization And Mapping* or *SLAM*. Each time the target position can be updated too, so that an updated homing vector can be derived. We implemented these
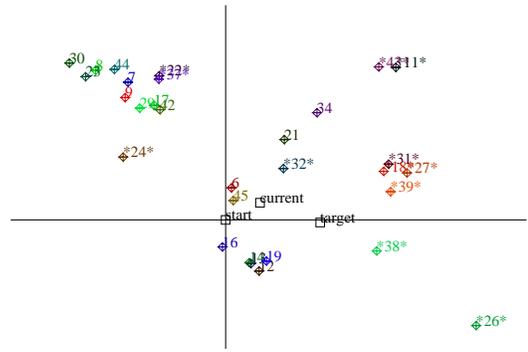

Fig. 4. Example of a feature map built up by our algorithm. Black squares show start, current and target robot positions. Colored diamonds show feature positions, labeled with their number. Stars in the feature label denote a feature that is lost during tracking.

updating operations as extended Kalman filters (EKF), using the image positions of the tracked visual features as inputs.

When close enough to one target, the movement towards the next target image is started. This yields a smooth trajectory along a sparsely defined visual path.

A. Feature tracking

The corresponding features found between the first image and the target image in the previous step, also have to be found in the incoming images during driving. This can be done very reliably performing every time wide baseline matching with first or target image, or both. Although recent methods are relatively fast (about $0.8s$ for a pair of 640×480 images, see [18]), this is still too time-consuming for a driving robot.

Because the incoming images are part of a smooth continuous sequence, a better solution is *tracking*. In the image sequence, visual features move only a little from one image to the next, which enables to find the new feature position in a small search space.

A widely used tracker is the KLT tracker of Kanade, Lucas, Shi, and Tomasi [22]. KLT starts by identifying interest points (corners), which then are tracked in a series of images. The basic principle of KLT is that the definition of corners to be tracked is exactly the one that guarantees optimal tracking. A point is selected if the matrix

$$\begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix}, \tag{2}$$

containing the partial derivatives g_x and g_y of the image intensity function over an NxN neighborhood, has large eigenvalues. Tracking is then based on a Newton-Raphson style minimization procedure using a purely translational model. This algorithm works surprisingly fast: we were able to track 100 feature points at 10 frames per second in 320x240 images on a 800 MHz laptop.

Because the well trackable points are not necessarily coinciding with the center points of the wide baseline features
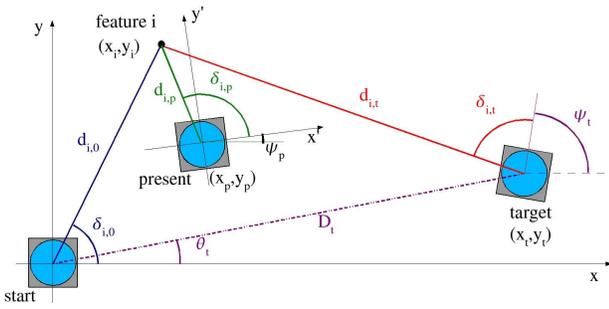
Fig. 5. Symbol definition for the Kalman filters.

to be tracked, the best trackable point in a small window around such a center point is selected. In the assumption of local planarity we can always locate the point in the relative reference system the wide baseline feature offers.

B. Local map updating

During movement, each time an image is taken new infomation of the environment is achieved and the local map of features can be corrected. Also, the position of the robot in that local map has changed and has to be corrected. For this SLAM-task, we implemented an extended Kalman Filter [23] (EKF), inspired on the work of Azarbayejani and Pentland [24] and similar to Kim and Chung [25], who adapted this algorithm for use with omnidirectional images.

Fig. 5 explains the various symbols used. The state vector we continuously update contains the present robot pose, the robot speed, and the polar feature coordinates:

$$\mathbf{x_p} = (x_p, y_p, \psi_p, v_p, d_{1,0}, \delta_{1,0}, d_{2,0}, \delta_{2,0}, \dots, d_{N,0}, \delta_{N,0}). \tag{3}$$

In the movement equation we express the expectation that the robot drives forward with his present velocity:

$$\hat{x}_p(k) = x_p(k-1) + v_p(k-1)\Delta t \cos \psi_p \tag{4}$$

$$\hat{y}_p(k) = y_p(k-1) + v_p(k-1)\Delta t \sin \psi_p. \tag{5}$$

Each iteration, a new measurement vector is acquired containing the feature bearings as seen from the new pose:

$$\mathbf{z_p}(k) = (\delta_{1,p}(k), \delta_{2,p}(k), \dots, \delta_{N,p}(k)). \tag{6}$$

The measurement equation $\mathbf{H_p}$ gives the estimate of these observations based on the present state vector. The i'th element of this vector is the function

$$\hat{z}_{p,i}(k) = H_{p,i}(\hat{x}_p(k)) = arctan\left(\frac{y'_i}{x'_i}\right), \tag{7}$$

with (x'_i, y'_i) the coordinates of the i'th feature relative to the present robot pose

$$\begin{bmatrix} x'_i \\ y'_i \end{bmatrix} = \begin{bmatrix} \cos \psi_p & -\sin \psi_p \\ \sin \psi_p & \cos \psi_p \end{bmatrix} \begin{bmatrix} d_{i,0} \cos \delta_{i,0} \\ d_{i,0} \sin \delta_{i,0} \end{bmatrix} - \begin{bmatrix} \hat{x}_p \\ \hat{y}_p \end{bmatrix}. \tag{8}$$

With this model an Extended Kalman Filter is set up. The choice for polar coordinates facilitates the computation of the necessary Jacobians of motion and measurement equation. The state vector is initialized according to the estimated local map of subsection III-B. We also set the initial variances on δ_i to zero to avoid instabilities, like in [24]. The variance on one of the d_i's is set to zero, to fix the scale. When during tracking features are lost, the corresponding elements of the state vector and the corresponding rows and columns of the covariance matrices are deleted, as detailed in [25].

C. Homing vector updating

The former EKF filter enables the construction of a local map with increasing accuracy and the present localization of the robot therein. However, only an estimate of the position where the target image is taken is known. A similar EKF procedure can be used to refine the target location as more accurate map information becomes available. The homing vector used to steer the robot is then the relative vector between the present location of the robot and that target position. As every distance, also this vector is only known up to a scale factor. But the relative length w.r.t. the original homing vector (with length 1) gives information for the robot how far to drive still.

We construct a second extended Kalman filter for the target localization. The state vector is the target pose: $\mathbf{x_t} = (D_t, \theta_t, \psi_t)$. We assume the target to be static, so the movement equation is simply $\hat{\mathbf{x}}_t(k) = \mathbf{I}.\mathbf{x_t}(k-1)$. The measurements are the feature bearing angles as seen from the target, $\mathbf{z_t} = (\delta_{1,t}, \delta_{2,t}, \dots, \delta_{N,t})$. These measurements are predicted using the measurement equation $\mathbf{H_t}$:

$$\hat{z}_{t,i}(k) = H_{t,i}(\hat{x}_t(k)) = arctan\left(\frac{y_i - y_t}{x_i - x_t}\right) - \psi_t, \tag{9}$$

Also in this case the computation of the Jacobians pose no problems. For the initialization of this EKF, the estimated data from the initialization step (section III) are used.

V. EXPERIMENTAL RESULTS

We have implemented the proposed algorithm, using a modified electric wheel chair as test platform, described in section V-A. The results of the tests are detailed in section V-B.

A. Test platform

Our wheelchair "Sharioto" (depicted in fig. 6) is a standard electric wheelchair that has been equipped with an omnidirectional vision sensor (consisting of a Sony firewire color camera and a hyperbolic mirror). The image processing is performed on a 800 MHz laptop. As additional sensors for obstacle detection, 16 ultrasound sensors and a Lidar are added. A second laptop with a 418 MHz processor reads these sensors, receives visual homing vector commands, computes the necessary manoeuvres, and drives the motors via a CAN-bus. More information can also be found in [26] and [27].
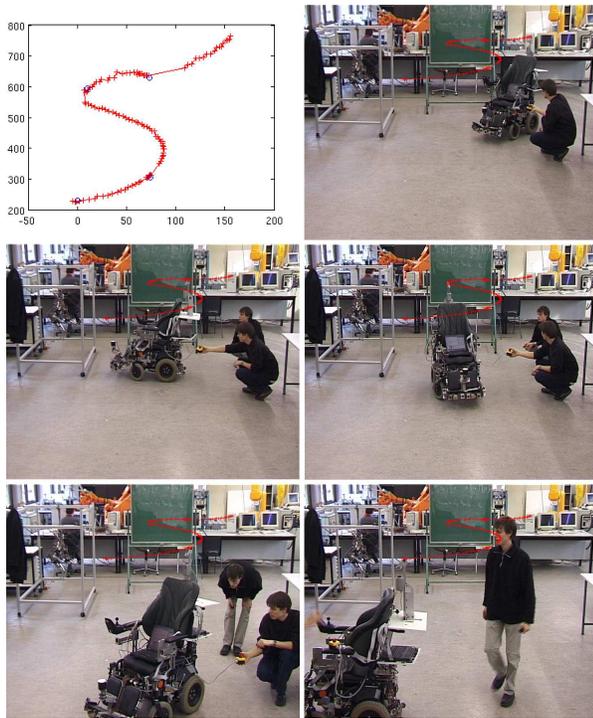
Fig. 6. Results of one sparse visual path following run. Top left: trajectory executed by the wheel chair. Blue circles denote target images. Top right: starting position. Thereunder: wheel chair at target positions along the path. All photos have the backprojected trajectory superimposed.

B. Results

Typical results of our algorithm are presented in fig. 6. A path was defined by four omnidirectional images taken at places about 2 metres apart along the path. From a starting position in the neighborhood of the first image, the visual path following algorithm was executed. During this motion, the top of the camera system was tracked in a video sequence taken from a fixed camera. The transformation to metric coordinates of this trajectory is shown top left in fig. 6.

The algorithm runs surprisingly fast on the rather slow hardware we used: the initialization for a new target lasts only 858 ms, while afterwards every 187 ms a new homing vector is computed.

VI. CONCLUSION

In this work, we developed a novel approach to sparse visual path following. Based on advanced fast wide baseline computer vision techniques a series of visual homing operations on path images taken far apart from each other is successfully executed.

ACKNOWLEDGMENT

T.G. thanks Eric Demeester and Dirk Vahooydonck.

REFERENCES

[1] S. Thrun, "Learning Metric-Topological Maps for Indoor Mobile Robot Navigation," AI Journal, 99(1), pp 21-71, 1999.
[2] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust Monte Carlo Localization for Mobile Robots," AI Journal, vol. 128, pp. 99-141, 2001.
[3] B. Cartwright, T. Collett, "Landmark Maps for Honeybees," Biol. Cybernetics, 57, pp. 85-93, 1987.
[4] M. Franz, B. Schölkopf, H. Mallot, and H. Bülthoff, "Where did I take that snapshot? Scene-based homing by image matching," Biological Cybernetics, 79, pp. 191-202, 1998.
[5] T. Röfer, "Controlling a wheelchair with image-based homing," In Spatial Reasoning in Mobile Robots and Animals, AISB-97 Workshop, Manchester University, 1997.
[6] C. Schmid, R. Mohr, C. Bauckhage, "Local Grey-value Invariants for Image Retrieval," International Journal on Pattern Analysis an Machine Intelligence, Vol. 19, no. 5, pp. 872-877, 1997.
[7] D. Lowe, "Object Recognition from Local Scale-Invariant Features," International Conference on Computer Vision, pp. 1150-1157, 1999.
[8] A. Baumberg, "Reliable feature matching across widely separated views," Computer Vision and Pattern Recognition, Hilton Head, South Carolina, pp. 774-781, 2000.
[9] T. Tuytelaars and L. Van Gool, "Wide baseline stereo based on local, affinely invariant regions," British Machine Vision Conference, Bristol, UK, pp. 412-422, 2000.
[10] T. Tuytelaars, L. Van Gool, L. D'haene, and R. Koch, "Matching of Affinely Invariant Regions for Visual Servoing," Intl. Conf. on Robotics and Automation, pp. 1601-1606, 1999.
[11] J. Matas, O. Chum, M. Urban and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," British Machine Vision Conference, Cardiff, Wales, pp. 384-396, 2002.
[12] K. Mikolajczyk and C. Schmid, "An affine invariant interest point detector," ECCV, vol. 1, 128–142, 2002.
[13] D. Nistér, O. Naroditsky, J. Bergen, "Visual Odometry," Conference on Computer Vision and Pattern Recognition, Washington, DC, 2004.
[14] A. Davison, "Real-time simultaneous localisation and mapping with a single camera," Intl. Conf. on Computer Vision, Nice, 2003.
[15] A. Argyros, K. Bekris, and S. Orphanoudakis, "Robot Homing based on Corner Tracking in a Sequence of Panoramic Images", Computer Vision and Pattern Recognition, vol. 2, p. 3, Kauai, Hawaii, 2001.
[16] T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool, "Vision Based Intell. Wheelchair Control: the role of vision and inertial sensing in topological navigation," J. of Robotic Systems, 21(2), pp. 85-94, 2004.
[17] T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool, "Markerless Computer Vision Based Localization using Automatically Generated Topological Maps," European Navigation Conf., Rotterdam, 2004.
[18] T. Goedemé, T. Tuytelaars, and L. Van Gool, "Fast Wide Baseline Matching with Constrained Camera Position," Computer Vision and Pattern Recognition, Washington, DC, pp. 24-29, 2004.
[19] L. Ledwich and S. Williams, "Reduced SIFT Features For Image Retrieval and Indoor Localisation," Australasian Conf. on Robotics and Automation, Canberra, 2004.
[20] F. Mindru, T. Moons, and L. Van Gool, "Recognizing color patters irrespective of viewpoint and illumination," Computer Vision and Pattern Recognition, vol. 1, pp. 368-373, 1999.
[21] M. Fischler, R. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis," Comm. of the ACM, Vol 24, pp 381-395, 1981.
[22] J. Shi and C. Tomasi, "Good Features to Track," Computer Vision and Pattern Recognition, Seattle, pp. 593-600, 1994.
[23] R. Kalman, "A New Approach to Linear Filtering and Prediction Problems," Trans. of the ASME–Journal of Basic Engineering, vol. 82, Series D, pp. 35-45, 1960.
[24] A. Azarbayejani and A. Pentland, "Recursive estimation of motion, structure, and focal length," IEEE Trans. Pattern Anal. Machine Intell., vol. 17, no. 6, pp. 562-575, June 1995.
[25] J.-H. Kim and M. Chung, "SLAM with omni-directional stereo vision sensor," Intelligent Robots and Systems, Las Vegas, Nevada, 2003.
[26] M. Nuttin, E. Demeester, D. Vanhooydonck, and H. Van Brussel, "Shared autonomy for wheel chair control: Attempts to assess the user's autonomy," in Autonome Mobile Systeme, pp. 127-133, 2001.
[27] E. Demeester, M. Nuttin, D. Vanhooydonck, and H. Van Brussel, "Fine Motion Planning for Shared Wheelchair Control: Req. and Prelim. Exper.," Intl. Conf. on Adv. Robotics, Coimbra, pp. 1278-1283, 2003.