Proceedings of the 2003 IEEE/RSJ
Intl. Conference on Intelligent Robots and Systems
Las Vegas, Nevada · October 2003

V-GPS – Image-Based Control for 3D Guidance Systems

Darius Burschka and Gregory D. Hager
Computational Interaction and Robotics Laboratory
The Johns Hopkins University
Baltimore, MD 21218
{burschka|hager}@cs.jhu.edu

***Abstract*— We present our approach for pose verification with monocular cameras in 3-dimensional space based on the image-based control paradigm. We describe the extensions to our previous control system for mobile navigation that allow us to estimate the complete set of pose parameters in space. The major contribution of this approach is a sensor-independent formulation of the image formation that allows a flexible configuration with a variety of sensor systems including standard cameras, omnidirectional cameras and laser systems. Our second contribution is a way to re-initialize the tracked landmarks during a multi-segment navigation in applications with significant deviations from the pre-taught trajectory as it is the case for handheld systems and flying robots.**

The presented system can be used as a guidance system for visitors. The localization is based on known landmarks that are in our case natural landmarks in the environment. These landmarks correspond to the satellites of a GPS system. We call it V-GPS (vision-based GPS) because of this similarity in the concept. A camera carried by a person allows to navigate along pre-specified paths through environments, like galleries, hospitals, parks, and other public places.

I. Motivation

Localization is an essential task in navigation systems. It can be subdivided into two categories of the initial localization and the relative localization. While the former requires an identification of known reference structures in the camera image to find the current pose relative to a known, a-priori map [14], [3], often it is merely necessary to register correctly the relative position changes in consecutive image frames.

Since the projection in a monocular camera results in a loss of one dimension, the estimation of the three-dimensional parameters in space requires a metric reference to the surrounding world, a 3D model, that is used to scale the result of the image processing back to Cartesian coordinates. Assuming that the projective geometry of the camera is modeled by perspective projection [12], a point, ${}^cP = (x, y, z)^T$, whose coordinates are expressed with respect to the camera coordinate frame c, will project onto the image plane with coordinates $p = (u, \nu)^T$, given by

$$\pi({}^cP) = \begin{pmatrix} u \\ v \end{pmatrix} = \frac{f}{z} \begin{pmatrix} x \\ y \end{pmatrix} \tag{1}$$

Points in the image correspond to an internal model of the environment that is stored in 3D coordinates ${}^mP$.

Localization with a monocular camera system is in this case formulated as estimation of the transformation matrix ${}^cx_m$ such that an image point $p = (u, \nu)^T$ corresponds to an actual observation in the camera image for any visible model point ${}^mP$.

$$p = \pi({}^cx_m({}^mP)) \tag{2}$$

This method allows an initial localization at any position in the world where a sufficient number of points of the reference model can be matched to the points in the image. The processing requires a generation and maintenance of a 3D model of the scene together with a computational intensive and error-prone correspondence search between the model points and their projections in the image.

In many cases the initial position of the system is known or not relevant, since the initial frame may define the starting position. We propose a navigation system that operates in the image space of the camera for this domain of applications. The system maintains the correspondences between the "model" and the world based on a simple landmark tracking since both the model and the perception are defined in the same coordinate frame of the camera projection $p = (u, \nu)^T$.

We propose a navigation system similar to a GPS system. It is based on a monocular camera that can be carried by a person or mounted on a flying system. This system operates on a set of known landmarks in the world, similar to the satellites of the GPS system, whose positions are a-priori known or learned in a teaching phase as described in this paper. The goal is to build a system that can be used as a 3D guidance system in public places, where a desired path is presented to the system once and, afterwards, the system can repeat it indefinitely. Our method allows navigation in a local area, which is not necessarily restricted to indoor environments. It helps the operator to stay close to a pre-specified path.

In contrast to most image-based control systems our approach needs to cope with significant deviations form the pre-specified path. The generated signals are supposed to be used as hints for a human to walk through a building and they do not control the motion directly. The

translational and especially orientation errors can become significantly large. They may exceed the convergence areas of typical image-based control systems based on Image Jacobian matrices.

Our system is motivated by the same idea as the system presented in [13], where a tracking approach for "2.5D space" was proposed. The system is supposed to compensate the drawbacks of classical position-based visual servoing. In the approach presented in [13], 8 landmarks are necessary to estimate the pose of an object in space. A reduction to 4 points is only possible in case that 4 coplanar points can be identified. The co-planarity constraint is a special case that is difficult to enforce in all situations. Additionally, a robust tracking of 8 landmarks in the image is contradictory to our goal to build a compact system running on standard hardware, like for example laptop PCs.

Our pose estimation is based on a model-based approach that compares the 3D coordinates of the model with the current image perception. In [6] a recursive model-based object pose estimation is presented that is based on orthographic projection of points onto camera image. This approach is limited to configurations that can be projected onto a planar image. In our case, we propose a pose estimation method allowing robust pose verification from 3 tracked landmarks that can be placed anywhere on the sphere encapsulating the sensor (Section II-C). Our approach operates in image coordinates of the camera using a novel representation for the 3D model that does not require any knowledge about the three-dimensional position in the world.

The paper is structured as follows, in the following section we present the algorithms used in the *teaching* and *replay* step that are necessary for a correct function of the system. We describe the way the system processes the image data in the *teaching step* to build an internal representation, a "model", of the landmark positions that is later used in the *replay step* to calculate a relative pose error for the navigation system. In Section III we evaluate the accuracy of the system for different landmark configurations and its convergence speed to reduce an initial error between the expected and the actual pose. We conclude with an evaluation of the system performance and our future plans.

II. Approach

We discussed already in [5] the properties of an *ideal generic sensor* for navigation purposes. Images from different sensor systems, like conventional perspective cameras, omnidirectional systems and laser range finders can be transformed into a unified representation of this ideal sensor that samples the world in spherical coordinates (Fig. 1).

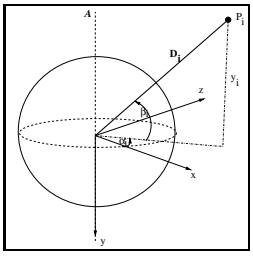

Fig. 1. The position of an arbitrary point P_i is represented in the spherical coordinate system (α_i, β_i, D_i)

The position of an observed point P_i in space can be described as

$$P_i^*(\alpha_i, \beta_i, D_i) = D_i \cdot \begin{pmatrix} \cos\beta_i \cos\alpha_i \\ \sin\beta_i \\ \cos\beta_i \sin\alpha_i \end{pmatrix} \quad (3)$$

assuming that the camera system is in the center of the sphere (Fig. 1).

Camera-based implementations of the *ideal sensor* are merely capable of measuring the two angles in spherical coordinates for the landmark observations $p_i = (\alpha_i, \beta_i)$. It is not possible to estimate the value for D_i directly from the camera image.

In case that the camera motion is restricted to the horizontal x-z plane (Fig. 1) we can write the Equation (3) in a form that is independent of the camera position.

$$\forall \beta_i \neq 0: \quad P_i(\alpha_i, \beta_i) = y_i \cdot \begin{pmatrix} \frac{\cos\alpha_i}{\tan\beta_i} \\ 1 \\ \frac{\sin\alpha_i}{\tan\beta_i} \end{pmatrix} \quad (4)$$

$$\text{using} \qquad D_i = \frac{y_i}{\sin\beta_i}$$

The value y_i is constant for a static landmark P_i observed from any point within the x-z plane. The observed landmark must not be part of the x-z plane ($\beta_i \neq 0$). This form allows to express the 3D position of the tracked landmark from any position within the x-z plane of operation using the current observation $(\alpha_i, \beta_i)^T$, if the value y_i is known.

As we will show later, the restriction to the motion in the x-z plane can be generalized to any motion, but the presented formulation allows a simple and robust generation of the reference model in the *teaching step* that does not require any knowledge of the pose in space.

In [4] we presented a system based on the Image Jacobian matrix that generates the navigation signals from the error between the expected and the observed landmark position for this kind of motion in the x-z plane. The current approach extends the motion in the *replay step* to an arbitrary motion in all 6 dimensions.

A. Sensor Model

As we mentioned already in the previous section our approach uses the abstraction of an *ideal generic sensor*

that can be mapped onto a variety of physical sensor configurations. In the following subsections we list the required transformations for the individual sensor types.

1) ***Standard Perspective Camera:*** We compute landmark observations p_i^* by first normalizing for pixel spread and focal length (that is, converting to a unit focal length camera with the feature image coordinates (u_i, ν_i)), and then computing *observed* angles in the camera image as

$$\begin{aligned} \alpha_i &= \arctan\, u_i = \arctan\left(\tfrac{x_i}{z_i}\right) \\ \beta_i &= \arctan \tfrac{\nu_i}{\sqrt{1+u_i^2}} = \arctan \tfrac{y_i}{\sqrt{x_i^2+z_i^2}} \\ &\text{with } u_i = \tfrac{x_i}{z_i} \quad \wedge \nu_i = \tfrac{y_i}{z_i} \end{aligned} \tag{5}$$

2) ***Omnidirectional Camera:*** Our algorithm uses the angular representation of the imaged points p_i^*. This representation is not necessarily bound to the planar image of a conventional camera, but it can be used with any geometry of the image plane as long as a single viewpoint F in the projection is ensured. This can be achieved with a hyperboloid mirror and a standard perspective camera in an omnidirectional camera system [1].

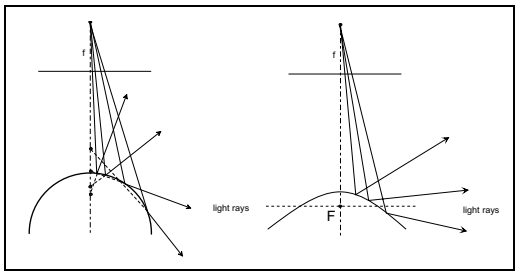

Fig. 2. Spherical mirror does not have a single viewpoint.

In [2] a design of a hyperbolic mirror with the shape

$$y = f(r) = \sqrt{\frac{b}{1-b} r^2 + b} \tag{6}$$

was suggested to ensure the single viewpoint imaging property. We calculate the following dependencies between the angles β_i and γ_i, that represent the angle observed by the perspective camera and the actual elevation angle, from the shape function (6) of the mirror to be

$$\cos\gamma_i = -\frac{2\sqrt{b} - (1+b)\sin\beta_i}{(1+b) - 2\sqrt{b}\sin\beta_i}. \tag{7}$$

The value $\sqrt{b}$ is the distance between the image plane and the tip of the mirror. This equation can easily be solved for β_i resulting in

$$\beta_i = \arcsin \frac{(1+b)\cdot\cos\gamma_i - 2\sqrt{b}}{2\sqrt{b}\cos\gamma_i - (1+b)} \tag{8}$$

The angle α_i can be estimated for the whole 360° field of view directly from the image to

$$\alpha_i = \text{atan2}\left(\frac{\nu_i}{u_i}\right) \tag{9}$$

We use here as well as later in the text the atan2() function instead of the arctan() function, because it calculates the angle directly for the 360° range of values.

3) ***Laser Range Finder:*** A laser range finder is capable of direct distance measurements to the observed landmark D_i (Fig. 1) in addition to the two angles (α_i, β_i). This allows the direct usage of the form presented in Equation (3). Most laser range finders are capable merely of a horizontal scan, which does not give any readings at angles $\beta_i \neq 0$. A full 3D laser system acquiring (α_i, β_i, D_i) is necessary for our system or the motion needs to be restricted to the x-z plane in the *replay step* as well.

The direct measurement of D_i gives an important advantage for the replay step described in Section II-C.1.

B. Teaching Step

As mentioned earlier in the text, our system calculates the pose from the error between an expected position of a landmark in the image and its actual current image position. Although the resulting system allows a motion in all 6 degrees of freedom, it is simpler to build a model from a movement restricted to an x-z plane with a fixed value Y_p for y-coordinate of this plane that corresponds to the average height of the user (Fig. 1).

1) Recording of the Landmark Positions: In our current implementation the system starts with a selection of the landmarks that are supposed to be used in the further processing. This selection is done manually by the user, who selects rectangular regions in the image that define the landmarks. Once the landmarks are selected, SSD tracking routines from our image processing library XVision [9] are used to track their positions in consecutive image frames.

The system waits until it detects an initial motion and then it starts recording the path. Images $\mathcal{I}^t$ are acquired from a monocular camera in equidistant time intervals ΔT and for each frame the position of the trackers in the image p_i^t is estimated. These positions are stored in a matrix $\mathcal{M}_p$, which is updated with each new frame $t \in 1, ..., K$.

$$\mathcal{M}_p = \begin{pmatrix} p_1^1 & p_1^2 & & p_1^K \\ p_2^1 & p_2^2 & \cdots & p_2^K \\ & & \cdots & \\ p_N^1 & p_N^2 & \cdots & p_N^K \end{pmatrix}, \; p_i^t = \begin{pmatrix} \alpha_i \\ \beta_i \end{pmatrix} \tag{10}$$

At the end of the teaching phase, which in our system is recognized as a stop, the estimation step is performed to calculate the values y_i for all landmarks.

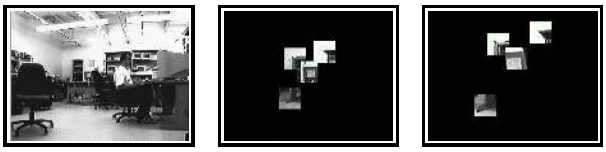

Fig. 3. Tracking of selected patterns in an example scene.

2) Estimation of the Landmark Height (y_i): There are several options for estimating the height y_i of a landmark. The first one is to choose landmarks in specific height relative to the sensor height Y_p that is constant during the *teaching step*. It is useful in situations where an omnidirectional camera system is used that monitors the upper hemisphere as a field of view and the landmarks are markers on the ceiling or when the values y_i are measured and assigned manually by the person generating the model.

The second alternative allows to estimate the values y_i for each tracked landmark P_i from the information collected during the teaching step of the system, if the odometry information is available during the generation of the model. The odometry information gives us an information about the relative motion T in a path segment, where a specific point P_i was observed. This relative motion of the robot is directly related to the pseudo-motion of the static landmark described by the vector $\Delta = -T$ that can be observed in the sensor image for each tracked point (Fig. 4).

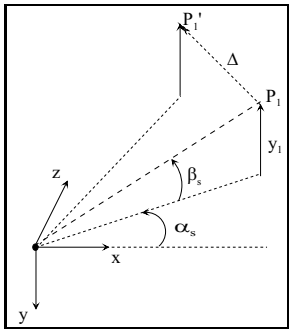

Fig. 4. Relative motion of the robot appears in the image as a motion of the tracked point P_1 along a vector Δ that is identical for all points.

The vector Δ in Fig. 4 can be expressed in image coordinates as follows

$$\Delta = y_i \cdot \begin{pmatrix} \frac{\cos\alpha_e}{\tan\beta_e} - \frac{\cos\alpha_s}{\tan\beta_s} \\ 0 \\ \frac{\sin\alpha_e}{\tan\beta_e} - \frac{\sin\alpha_s}{\tan\beta_s} \end{pmatrix} \tag{11}$$

The angles (α_s, β_s) describe the position of the tracked point at the beginning of the path segment and (α_e, β_e) are the observed angles at the end of the path. From Equation (11) we can directly estimate the value y_i using the odometry information.

Although formulated for a single motion, better results are obtained by formulating this system for several point along the path segment and solving a least-squares problem.

C. Replay Step

A conventional perspective camera with an f=4mm lens has a horizontal opening angle of approximately $82°$. This limits the allowed rotations of the camera if the tracked landmarks need to stay visible all the time. A landmark can disappear due to several factors. It can disappear, because it got occluded by the room structure, or because the cone of the projected space cannot span over the current set of landmarks any more, or the user may have turned the camera too far. In any case it is obvious that the desired field of view for the used camera should be as wide as possible and omnidirectional cameras are preferred for this kind of systems.

Independent of the used system, natural occlusions in the world require a switch between different sets of landmark to allow navigation over a long path. We subdivide the entire replay action into navigation in local segments, where a specific set of landmarks is visible and a hand-off action, where new landmarks are selected.

1) Navigation in a Local Segment: In a general case, the camera can be rotated around the three Euler angles $(\varphi_x, \varphi_y, \varphi_z)$ defining the rotation matrix $\tilde{R}$ and translated along the vector $\vec{T}$ from the expected position that defines the origin of the current calculation. This transformation defines the matrix ${}^c x_m$ from Equation (2) to

$${}^c x_m = \begin{bmatrix} \tilde{R} & \vec{T} \\ 0^T & 1 \end{bmatrix} \tag{12}$$

We use the matrix ${}^c x_m$ to describe the point transformation between the stored point location $P_i(\alpha_i, \beta_i, y_i)$ and the observed point P_i^*. We can write the Cartesian coordinates for P_i^* using (3). The Cartesian coordinates of the new point are known up to a scale λ_i due to the camera projection. What we know about each landmark are the two angles from the current observation (α_i^*, β_i^*). These angles define the point to be along the following vector

$$P_i^* = \lambda_i \cdot \begin{pmatrix} \cos\beta_i^* \cos\alpha_i^* \\ \sin\beta_i^* \\ \cos\beta_i^* \sin\alpha_i^* \end{pmatrix} = \lambda_i \cdot \vec{n_i^*} \tag{13}$$

The vector $\vec{n_i^*}$ in Equation (13) is a unit vector pointing in the direction of the tracked landmark P_i^*. Using (12) and (13) we can write the following equation:

$$P_i^* = \lambda_i \cdot n_i^* = \tilde{R} \cdot P_i + \vec{T} \tag{14}$$

A major problem in Equation (14) is the fact that the exact solution depends on the knowledge of λ_i for a specific landmark that corresponds to the knowledge of D_i^* (known only in case of a laser range finder) for a tracked landmark in Fig. 1. As mentioned already before the system is used to correct the position of a handheld camera. The major challenge in this case is the strongly varying orientation angle of the camera, while the camera may be almost at the correct Cartesian position. Using this assumption we set λ_i for a frame t in our system to

$$\hat{\lambda_i^t} = \begin{cases} D_i, & t = 0 \quad (\text{initial step}) \\ \lambda_i^{t-1}, & t > 0 \end{cases} \tag{15}$$

the distance to the expected landmark position D_i from the reference frame as an initial guess, or to the result estimated for the previous frame. The closer this estimate $\hat{\lambda_i^t}$ is to the true value λ_i^t the faster the convergence of the following algorithm. For $\lambda_i^t = \hat{\lambda_i^t}$ we are be able to calculate the pose directly. Since the following processing is independent on the history of the previous frame we will omit the t-index in the following equations.

Computing the absolute orientation is the process of determining $\tilde{R}$ and $\vec{T}$ from corresponding pairs P_i^* and P_i. With three or more non-collinear points, $\tilde{R}$ and $\vec{T}$ can be obtained as a solution to the following least-squares problem as described in [8].

$$\min_{\tilde{R},\vec{T}} \sum_{i=1}^{n} \|\tilde{R}P_i + \vec{T} - P_i^*\|^2, \quad \text{subject to } R^T R = I. \tag{16}$$

Such a constrained least squares problem [7] can be solved in closed form using quaternions [11], [15], or singular value decomposition (SVD) [10], [16], [11], [15].

The SVD solution proceeds as follows. Let P_i and P_i^* denote lists of corresponding vectors and define

$$\bar{P} = \frac{1}{n}\sum_{i=1}^{n} P_i, \quad \bar{P^*} = \frac{1}{n}\sum_{i=1}^{n} P_i^*, \tag{17}$$

that is, $\bar{P}$ and $\bar{P^*}$ are the centroids of $\{P_i\}$ and $\{P_i^*\}$, respectively. Define

$$P_i' = P_i - \bar{P}, \quad P_i'^* = P_i^* - \bar{P^*}, \tag{18}$$

and

$$M = \sum_{i=1}^{n} P_i'^* P_i'^T. \tag{19}$$

In other words, $\frac{1}{n}M$ is the sample cross-covariance matrix between $\{P_i\}$ and $\{P_i^*\}$. It can be shown that, if $\tilde{R}^*$, $\vec{T}^*$ minimize (16), then they satisfy

$$\tilde{R}^* = \text{argmax}_{\tilde{R}} \text{tr}(\tilde{R}^T M) \tag{20}$$

$$\vec{T}^* = \bar{P^*} - R^* \bar{P}. \tag{21}$$

Let (U, Σ, V) be a SVD of M, that is $U^t M V = \Sigma$. Then the solution to (16) is

$$\tilde{R}^* = VU^T \tag{22}$$

Note that the optimal translation is entirely determined by the optimal rotation, and all information for finding the best rotation is contained in M as defined in (19). Hence, only the position of the 3D points relative to their centroids is relevant in the determination of the optimal rotation matrix.

These two values from (21) and (22) are used to calculate a new guess for P_i^* (14).
These new better approximations are used to repeat the whole calculation. The iteration is terminated once the change in $|\Delta\vec{T}| < \epsilon_d$ is smalled than the required accuracy ϵ_d.

We assumed the following order of rotations for the matrix $\tilde{R}$. A point is rotated first around the y-axis (φ_y) corresponding to an azimuthal rotation followed by a rotation around the x-axis (φ_x) representing the tilt angle of the system and a rotation around the optical axis (φ_z). This results in the structure of the rotation matrix $\tilde{R}$ presented in (23) on the next page.

From (23) we can see that a multiplication with the following vectors allows an easy recovery of the rotational angles by back-substitution of vector rotation results, like shown in for the most important heading angle φ_y. This angle can be estimated to

$$\varphi_y = \arctan \frac{\tilde{R}_{31}}{\tilde{R}_{33}}. \tag{24}$$

2) Landmark Hand-off: We established in Section II-C.1 that a minimum number of N=3 landmarks is necessary to calculate the pose deviation to the expected position from where the reference measurement $\mathcal{M}_p$ was acquired (Section II-B.1). This position defines the origin for the current frame. If the system is able to track the minimum required number of landmarks then we can calculate the position error Δw^t using Equations (21) and (24). This position error is identical for all tracked landmarks, because it represents the error in the robot position. Therefore, we can use (14) to calculate the correction values P_i^* for an expected new landmark P_i.

An example for a correct prediction of the tracker position is shown in Fig. 5 for a position error of $\Delta w^t = (-1.3, 0, -1.3, \frac{\pi}{10})^T$. We used the position the valid landmarks to estimate the position error Δw^t to the expected position. The estimated value was used to correct the saved position of the new landmarks shown as a horizontal cross from their actual position shown as a diagonal cross. In this example a simulation was used to remove any detection and tracking errors. The small deviation is due to the approximation used in our

$$\tilde{R} = \begin{bmatrix} \cos\varphi_z \cos\varphi_y + \sin\varphi_z \sin\varphi_x \sin\varphi_y & \sin\varphi_z \cos\varphi_x & -\cos\varphi_z \sin\varphi_y + \sin\varphi_z \sin\varphi_x \cos\varphi_y \\ -\sin\varphi_z \cos\varphi_y + \cos\varphi_z \sin\varphi_x \sin\varphi_y & \cos\varphi_z \cos\varphi_x & \sin\varphi_z \sin\varphi_y + \cos\varphi_z \sin\varphi_x \cos\varphi_y \\ \cos\varphi_x \sin\varphi_y & -\sin\varphi_x & \cos\varphi_x \cos\varphi_y \end{bmatrix} \quad (23)$$

approach. This correction allows a successful re-start of a tracker at a new expected location to continue tracking.

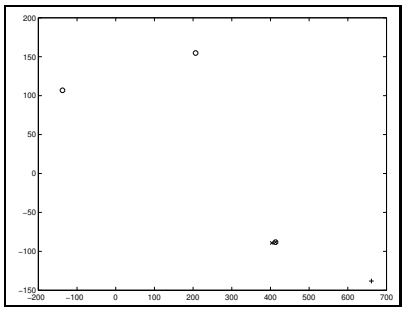

Fig. 5. Prediction of the actual feature position (diagonal cross) from a saved expectation (horizontal cross) under consideration of a pose error $\Delta w^t = (-1.3, -1.3, \frac{pi}{10})^T$ that is estimated from the error in the position of the remaining features during the segment switch.

III. Results

The algorithm has been implemented on a laptop computer with a Pentium-III 700MHz processor under Linux OS. The system uses an IEEE1394 video camera for real-time image acquisition. Both omnidirectional vision using catadioptric systems and standard cameras are supported.

A. Restrictions on Camera Motion

In case of a configuration with a standard camera, the frame rate of the camera together with a maximal allowed shift of the tracked landmark between two camera frames set a limit on the maximal rotation speed $\dot{\varphi}_{p\in x,y,z}$ of the camera system. An analog NTSC camera allows frame rates up to 60 Hz allowing faster rotations of the camera, but since the current system uses a notebook with an IEEE1394 camera the frame rate is limited to 30 Hz.

The tracking is using an SSD-(sum of square differences)-tracker from the XVision library. This tracker type allows in our configuration a maximal landmark displacement between two images of $d_t = 3$ pixels. The camera has a resolution of 640x480 pixels is equipped with an f=4mm lens to allow a wide field of view. The pixel-size of the camera chip is $p_{sx} = 0.011mm$.

The largest changes in the image are caused by the rotation of the camera. The maximal shift of the landmarks between frames results in following angular rotations for a pixel around the center of the camera image.

$$\Delta\varphi = \arctan\frac{d_t \cdot p_{sx}}{f} = 0.47^\circ \quad (25)$$

The maximal rotational velocity ω_{cam} of the camera depends on the frame rate r_{cam} and is limited in our case to

$$\omega_{cam} = \Delta\varphi \cdot r_{cam} = \arctan\frac{d_t \cdot p_{sx}}{f} \cdot r_{cam} = 14\text{degrees}/s \quad (26)$$

This values scales linear with the frame rate of the sensor.

B. Error in the Pose Estimation

The following tests show the convergence of the proposed vision-based pose estimation algorithm. In all tests a set of 4 landmarks at the positions (0,1,3), (2,2,4), (2,-1,2) and (1,2,3). The system was tested with an angular error of $\varphi_x = -20^\circ$, $\varphi_y = 50^\circ$, $\varphi_z = 10^\circ$. The position was varied at $\pm 1m$ in x and z-direction around the pre-taught position.

The Fig. 6 shows the remaining position error in the transformation matrix in dependence on the deviation from the expected position.

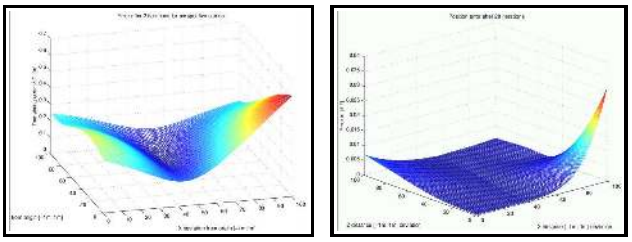

Fig. 6. Remaining position errors for a set of four landmarks at the positions (0,1,3),(2,2,4),(2,-1,2),(1,2,3): (left) after 2 iterations in the *replay step*; (right) after 20 iterations in the replay step.

The system has a wide convergence radius tested up to several meters deviation and as shown in Fig. 7 the accuracy deterioration can be compensated by additional iterations through the algorithm described in Section II-C.

As already expected in Section II-C the system has a very good convergence of the angular error. The increase of the error in one of the corner is due to ill conditioned configuration of landmarks for this specific pose.

The position errors on a real camera system are usually an order of a magnitude smaller than the errors plotted in the figures above. A real camera system was able to guide robustly through our lab environment. The increasing position errors in larger distances from the expected positions can be neglected, because the system creates in all cases a vector in the correct direction and just with slightly wrong

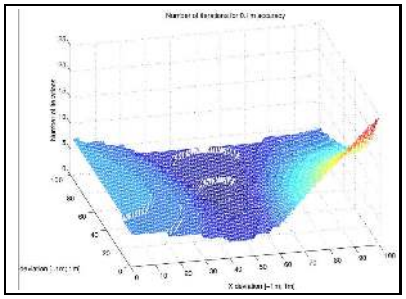

Fig. 7. Number of iterations for the same set of landmarks as in Fig. 6 to keep the position accuracy $\Delta T < 0.1m$.

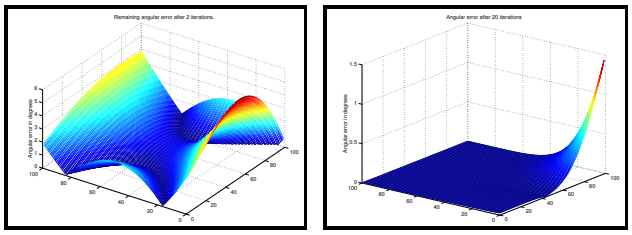

Fig. 8. Angular error for the landmark confi guration from Fig. 6 after: (left) 2 iterations; (right) 20 iterations.

magnitude that improves with the decreasing distance to the desired position. That means that once the system comes closer to the reference point the error decreases.

IV. Conclusions and Future Work

We have presented a navigation system capable of operation under significant pose errors between the expected and the actual position. Especially, its capability to operate robustly under large rotational errors distinguish it from pure Image Jacobian approaches. Due to the linearizations in the Jacobian matrix the area of convergence is limited. Storing the representation of the world as a combination of invariant parameters for the teaching step together with image parameters of the actual observations make this approach very simple to implement and it does not require any odometry information at all for model generation in case the values Y_p for the heights of the landmarks are known.

The correspondence problem between the saved references representing the desired path and the current image is solved by tracking the landmarks in consecutive images.

Acknowledgments

This work was supported by the DARPA MARS project. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agency.

V. References

[1] S. Baker and S.K. Nayar. A Theory of Catadioptric Image Formation. *Proc. of IEEE International Conference on Computer Vision*, Bombay, 1998.

[2] A.M. Bruckstein and T.J. Richardson. On Hyperbolic Mirrors for Omnidirectionality. *ATT Bell Laboratories Lab Notes*, February 1996.

[3] D. Burschka and S. Blum. Identification of 3d Reference Structures for Video-Based Localization. In *Proc. 3rd Asian Conf. on Computer Vision (ACCV'98)*, volume 1, pages 128–135, 1998.

[4] D. Burschka and G. Hager. Vision-based control of mobile robots. In *Proc. International Conference on Robotics and Automation*, pages 1707–1713, 2001.

[5] Darius Burschka, Jeremy Geiman, and Gregory D. Hager. Optimal Landmark Configuration for Vision-Based Control of Mobile Robots. In *Proc. of International Conference on Robotics and Automation (ICRA)*, 2003. To appear.

[6] Daniel F. DeMenthon and Larry S. Davis. Model-Based Object Pose in 25 Lines of Code. *International Journal of Computer Vision*, 15:123–141, June 1995.

[7] O. Faugeras, *Three-Dimensional Computer Vision*, The MIT Press, 1993.

[8] G. Hager, C-P. Lu, and E. Mjolsness. Object pose from video images. *PAMI*, 22(6):610–622, 2000.

[9] G. Hager and K. Toyama. The XVision System: A General-Purpose Substrate for Portable Real-Time Vision Applications. *Computer Vision and Image Understanding*, 69(1):23–37, 1995.

[10] B. K. P. Horn, H. M. Hilden, and S. Negahdaripour, "Closed-form solution of absolute orientation using orthonomal matrices," *J. Opt. Soc. Amer.*, vol. A-5, pp. 1127–1135, 198.

[11] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternion," *J. Opt. Soc. Amer.*, vol. A-4, pp. 629–642, 1987.

[12] B. K. P. Horn. *Robot Vision*. MIT Press, 1986.

[13] Ezio Malis, Franois Chaumette, and Sylvie Boudet. 2D 1/2 visual servoing. *IEEE Transactions on Robotics and Automation*, 15(2):238–250, April 1999.

[14] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, 1998.

[15] M. W. Walker, L. Shao, and R. A. Volz, "Estimating 3-D location parameters using dual number quaternions," *CVGIP: Image Understanding*, vol. 54, no. 3, pp. 358–367, 1991.

[16] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pat. Anal. Machine Intell.*, vol. 9, pp. 698–700, 1987.