Running on Optical Rails: Theory, Implementation and Testing of Omnidirectional View-based Point-To-Point Navigation

David Dederscheck, Holger Friedrich, Christine Lenhart, Joachim Penc, Eduard Rosert, Maximilian Scherer, Rudolf Mester

Running on Optical Rails.

Theory, Implementation and Testing of Omnidirectional View-based Point-To-Point Navigation

David Dederscheck**, Holger Friedrich, Christine Lenhart, Joachim Penc, Eduard Rosert, Maximilian Scherer, and Rudolf Mester

Visual Sensorics and Information Processing Lab
Goethe University, Frankfurt, Germany
http://www.vsi.cs.uni-frankfurt.de

Abstract. Optical Rails is a purely view-based method for steering a robot through a network of positions in a known environment. Navigation is based on images aquired by an upward-looking omnidirectional camera; even a very modest quality of the optical system is sufficient, since all views are represented in terms of low-order basis functions (spherical harmonics). The theoretical concept of Optical Rails and a first preliminary validation using simulations have been presented very recently [5]; the present paper provides substantial advances in terms of the efficient computation of the spherical harmonics representation, considers the necessary processing particularly for inexpensive cameras, and describes how the link between view-based navigation information and the actual steering of a real robot is achieved. We present strategies for navigation along a prerecorded path that also allow for arbitrary movement of the robot between adjacent positions in the network.

1 Introduction

In this work, we present a method for guiding a mobile robot through a known environment, only relying on omnidirectional vision, and without use of any geometric information. Animals as well as humans use visual perception to a very large extent to orient themselves in their surroundings. When using visual information for guidance and navigation, most processing is related to recognition, association, and observing changes of similarity; it seems less plausible that geometry, e.g. triangulation, plays an explicit role in finding and following a known path. Our method is designed according to these principles. It has been very recently introduced in [5] as Optical Rails and tested using simulated data. In this article, we show the extension of our approach for successful implementation on a real robot vehicle. The method shares certain characteristics with landmark-based robot homing approaches [6, 11, 1, 3], view based homing [12], and with those methods that use epipolar geometry [2, 8]. It does, however, not require the use of artificial markers, laser scanners, SLAM, etc. In terms

** in alphabetical order

of utilizing spherical harmonics, it is founded upon previous work of Makadia and Daniilidis [8, 9]; the expansion into orthonormal basis functions for image signal representation leads to the essential innovation in Optical Rails: efficient differential pose change estimation and differential track following.

The representation of images in spherical harmonics (SH) is the key to the particular efficiency of Optical Rails: To control the motion of the robot, the gradient of a dissimilarity measure with respect to the motion parameters of the robot (translation, rotation) is obtained by comparing omnidirectional views. The computation of derivatives in image space is avoided by expanding the spherical image signals in basis functions; all further processing occurs only using the resulting low-dimensional coefficient vectors. Thus, with Optical Rails, pose derivatives can be obtained in a computationally inexpensive way using precomputed expressions. Another positive aspect is that by using a truncated SH expansion Optical Rails works well even without high resolution images or precision optics.

2 Optical Rails Concept

Imagine the desired course of the robot as a sequence of discrete waypoints, i.e. subsequent locations in the environment. Each of these waypoints is associated with an individual omnidirectional view. The position of the waypoints in space is neither determined nor necessary for steering the robot. Merely by comparing the view at its current location with the target view, the robot is able to drive towards the target waypoint. As soon as it gets close enough to its current target, the robot proceeds with the next waypoint in the sequence (*waypoint handover*). In this way, the robot visits one waypoint after the other and thus follows the prerecorded path. Multiple paths can be interconnected and extended to a network in which the robot can move freely.

In our approach, all views are represented as an expansion into spherical harmonics (SH), analogously to the Fourier representation in the plane [8, 5]. The sets of coefficients $a_{\ell m}$ of this expansion form *view descriptors* $\boldsymbol{a}$. The dissimilarity Q between the current view and the target view can be computed from the difference of the two corresponding view descriptors.

Svoboda and Pajdla [13] perform motion estimation on panoramic images using correspondences of randomly scattered points. In our case, the differential change of the dissimilarity measure Q – for infinitesimal pose changes of the robot – corresponds to the desired robot motion to decrease dissimilarity. For the task of robot navigation, we may then assume that minimizing dissimilarity also reduces the distance between the positions. This incremental differential process is analogous to classical differential matching processes (a.k.a. 'differential image registration') as used by Lucas and Kanade [7].

For computing the differential change of the dissimilarity with respect to pose changes, we require a model for the effect of a pose change on the spherical image signal. An exact prediction can be made only for the trivial case of a pure rotation; for exactly predicting the effects of a 2D translation, we would require

the depth structure of the scene. The changes w.r.t. translation and rotation are modeled by reprojecting the spherical signal on a virtual ceiling plane parallel to the ground plane (*gnomonic projection*) on which the robot moves.

3 A Model for Differential Pose Change Estimation

A natural choice for the dissimilarity measure Q between two views is the mean squared image signal difference obtained by integration across the hemisphere. Let $s(\theta, \phi)$ be the view at the current – yet unknown – robot position $\boldsymbol{p}_c$, and $\boldsymbol{a}$ the associated view descriptor; $\tilde{s}(\theta, \phi)$ is the view at the destination pose $\boldsymbol{p}_d$ with the corresponding view descriptor $\tilde{\boldsymbol{a}}$. Due to the fact that the basis functions used here are orthonormal, the dissimilarity Q between the two views $s(\theta, \phi)$ and $\tilde{s}(\theta, \phi)$ can be directly computed from their view descriptors $\boldsymbol{a}$ and $\tilde{\boldsymbol{a}}$:

$$Q := \int_{\theta=0}^{\pi/2} \int_{\phi=0}^{2\pi} (s(\theta, \phi) - \tilde{s}(\theta, \phi))^2 \cdot \sin\theta \, \mathrm{d}\phi \, \mathrm{d}\theta \; = \; ||\boldsymbol{a} - \tilde{\boldsymbol{a}}||_2^2 \tag{1}$$

where $\sin\theta$ is the Jacobian of integration in spherical coordinates θ, ϕ.

As we wish to minimize Q, we require the gradient $-\boldsymbol{g} = -\partial Q / \partial \boldsymbol{p}_c$ which represents the direction in which the robot should move (including rotation) to reduce the dissimilarity Q. Because $s(\theta, \phi)$ is dependent on the current robot pose $\boldsymbol{p}_c$, but the exact relation is inaccessible due to the unknown depth structure, we need a model that approximates the effects of a pose change.

To that purpose, we project the spherical image signal onto a ceiling plane. Then differential translations and rotations of the projected signal correspond to motions of the robot underneath the ceiling plane. This way we coarsely approximate the underlying geometry to benefit from the now analytically determinable effects of the motion.

Using this model, we compute the partial derivatives of Q with respect to the components of the current pose vector $\boldsymbol{p}_c$ and steer the robot in the direction of the destination.

Gnomonic Projection of a Hemispherical Signal. We project an image signal $s(\theta, \phi)$ defined on a hemisphere onto a ceiling plane by using the *gnomonic projection*. Thus we obtain spherical coordinates from Cartesian coordinates on the projection plane by

$$\boldsymbol{G}\,(x, y) = (\theta(x, y); \phi(x, y))^T \qquad \begin{array}{l} \theta(x, y) = \arctan(1, \sqrt{x^2 + y^2}) \\ \phi(x, y) = \arctan(x, y) \end{array} \tag{2}$$

The inverse mapping from spherical coordinates to Cartesian coordinates on the projection plane is obtained by

$$\boldsymbol{G}^{-1}\,(\theta, \phi) = (x(\theta, \phi); y(\theta, \phi))^T \qquad \begin{array}{l} x(\theta, \phi) = \tan(\theta) \cdot \cos(\phi) \\ y(\theta, \phi) = \tan(\theta) \cdot \sin(\phi) \end{array} \tag{3}$$

Integral of a Planar Projection of a Hemispherical Function. Since we aim to compute the derivative of Q in coordinates of the projection plane, we require the planar equivalent of our dissimilarity measure (1). Thus we substitute $\theta, \phi, \mathrm{d}\phi, \mathrm{d}\theta$ in (1) according to (2) and adjust the integration boundaries appropriately. The combined Jacobian $w(x, y)$ for this transform is

$$w(x,y) = \det|\mathbf{J}| \cdot \sin(\arctan(1, \sqrt{x^2+y^2})) = (1+x^2+y^2)^{-\frac{3}{2}} \tag{4}$$

where $\det|\mathbf{J}|$ is the Jacobian for the gnomonic transform $\boldsymbol{G}(x, y)$. We obtain the planar equivalent of Q for the projection:

$$Q = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} w(x,y) \cdot [s(\theta(x,y),\phi(x,y)) - \tilde{s}(\theta(x,y),\phi(x,y))]^2 \ \mathrm{d}x \ \mathrm{d}y \tag{5}$$

Differential Pose Change Estimation for 3 DoF Motion. Let $b(\boldsymbol{x})$ and $\tilde{b}(\boldsymbol{x})$ be the planar image signals acquired at the current pose $\boldsymbol{p}_c$ and the destination pose $\boldsymbol{p}_d$ of the robot, respectively. We define $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{d})$ as an isometric Euclidean transform which represents the pose translation and rotation with the *displacement parameter vector* $\boldsymbol{d}$:

$$\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}) = \mathbf{A}(\boldsymbol{d}) \cdot \boldsymbol{x}, \quad \mathbf{A}(\boldsymbol{d}) = \begin{pmatrix} \cos\varphi & -\sin\varphi & v_1 \\ \sin\varphi & \cos\varphi & v_2 \\ 0 & 0 & 1 \end{pmatrix}, \quad \boldsymbol{d} = \begin{pmatrix} v_1 \\ v_2 \\ \varphi \end{pmatrix}, \quad \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ 1 \end{pmatrix} \tag{6}$$

We now regard the dissimilarity Q between the view at the destination pose $\tilde{b}(\boldsymbol{x})$ and the transformed view at the current pose of the robot $b(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{d}))$, i.e. to the latter the Euclidean transform $\boldsymbol{f}$ has been applied:

$$Q = \int_A w(\boldsymbol{x}) \cdot \Big[b(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d})) - \tilde{b}(\boldsymbol{x}) \Big]^2 \mathrm{d}\boldsymbol{x} \tag{7}$$

The partial derivative of Q with respect to the parameter vector $\boldsymbol{d}$ at $\boldsymbol{d} = \mathbf{0}$ is:

$$\frac{\partial Q}{\partial d_i} = \int_A 2 \cdot w(\boldsymbol{x}) \cdot \Big[b(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d})) - \tilde{b}(\boldsymbol{x}) \Big] \cdot \left[\frac{\partial b(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}))}{\partial d_i} - \frac{\partial \tilde{b}(\boldsymbol{x})}{\partial d_i} \right] \mathrm{d}\boldsymbol{x} \Bigg|_{\boldsymbol{d}=\mathbf{0}} \tag{8}$$

As $\tilde{b}(\boldsymbol{x})$ does not depend on $\boldsymbol{d}$, $\frac{\partial \tilde{b}(\boldsymbol{x})}{\partial d_i}$ is 0. The derivative $\frac{\partial b(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}))}{\partial d_i}$ is obtained by applying the generalized chain rule:

$$\frac{\partial b(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}))}{\partial d_i} = \Big(\tfrac{\partial f_1(\boldsymbol{x},\boldsymbol{d})}{\partial d_i}, \tfrac{\partial f_2(\boldsymbol{x},\boldsymbol{d})}{\partial d_i}, \tfrac{\partial f_3(\boldsymbol{x},\boldsymbol{d})}{\partial d_i} \Big) \Big(\tfrac{\partial b(\boldsymbol{y})}{\partial y_1}, \tfrac{\partial b(\boldsymbol{y})}{\partial y_2}, \tfrac{\partial b(\boldsymbol{y})}{\partial y_3} \Big)^T \Big|_{\boldsymbol{y}=\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d})} \tag{9}$$

where $f_i(\boldsymbol{x}, \boldsymbol{d})$ are three individual components of the vector function $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{d})$. We perform the differentiation and substitute the results in (8).
With $\xi_1 := -x_1 \cdot \sin\varphi + x_2 \cdot \cos\varphi$ and $\xi_2 := -x_1 \cdot \cos\varphi - x_2 \cdot \sin\varphi$ we obtain:

$$\frac{\partial Q}{\partial \boldsymbol{d}} = 2\int_A w(\boldsymbol{x}) \Big[b(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d})) - \tilde{b}(\boldsymbol{x}) \Big] \Big(\tfrac{\partial b(\boldsymbol{y})}{\partial y_1}, \tfrac{\partial b(\boldsymbol{y})}{\partial y_2}, \xi_1 \tfrac{\partial b(\boldsymbol{y})}{\partial y_1} + \xi_2 \tfrac{\partial b(\boldsymbol{y})}{\partial y_2} \Big)^T \Big|_{\boldsymbol{y}=\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d})} \mathrm{d}\boldsymbol{x} \Bigg|_{\boldsymbol{d}=\mathbf{0}} \tag{10}$$

For these derivative terms depending on the parameter vector, we may now perform the transition to $\boldsymbol{d} = \mathbf{0}$. As $\boldsymbol{f}(\boldsymbol{x}, \mathbf{0})$ is the identity transform, we obtain:

$$\frac{\partial Q}{\partial \boldsymbol{d}} = \int_A 2 \cdot w(\boldsymbol{x}) \cdot \Big[b(\boldsymbol{x}) - \tilde{b}(\boldsymbol{x}) \Big] \cdot \Big(\tfrac{\partial b(\boldsymbol{x})}{\partial x_1}, \tfrac{\partial b(\boldsymbol{x})}{\partial x_2}, x_2 \tfrac{\partial b(\boldsymbol{x})}{\partial x_1} - x_1 \tfrac{\partial b(\boldsymbol{x})}{\partial x_2} \Big)^T \, \mathrm{d}\boldsymbol{x} \quad (11)$$

which represents the gradient $\boldsymbol{g}$ for continuous planar image signals.

Exploiting the Spherical Harmonic Representation. We represent image signals by linear combination of basis functions, which allows for drastic simplifications in the computation of gradient $\boldsymbol{g}$. Let $\breve{Y}_j$ be the gnomonically projected basis functions on the ceiling plane, i. e.

$$b(\boldsymbol{x}) = \sum_j a_j \cdot \breve{Y}_j(\boldsymbol{x}), \qquad \tilde{b}(\boldsymbol{x}) = \sum_j \tilde{a}_j \cdot \breve{Y}_j(\boldsymbol{x}) \quad (12)$$

where a_j are the coefficients of the linear combination, corresponding to the j-th entry of the view descriptor $\boldsymbol{a}$. Substituting an image signal $b(\boldsymbol{x})$ by $\sum_j a_j \cdot \breve{Y}_j(\boldsymbol{x})$ in (11) and correspondingly for $\tilde{b}(\boldsymbol{x})$, results in

$$\frac{\partial Q}{\partial \boldsymbol{d}} = 2 \int_A w(\boldsymbol{x}) \begin{pmatrix} \sum_j \sum_k (a_j - \tilde{a}_j) \breve{Y}_j(\boldsymbol{x}) a_k \frac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_1} \\ \sum_j \sum_k (a_j - \tilde{a}_j) \breve{Y}_j(\boldsymbol{x}) a_k \frac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_2} \\ \sum_j \sum_k (a_j - \tilde{a}_j) \breve{Y}_j(\boldsymbol{x}) a_k \left(x_2 \frac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_1} - x_1 \frac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_2} \right) \end{pmatrix} \mathrm{d}\boldsymbol{x} \quad (13)$$

Using the linearity of the integral operator, we obtain

$$\frac{\partial Q}{\partial \boldsymbol{d}} = 2 \begin{pmatrix} \sum_j \sum_k (a_j - \tilde{a}_j) a_k \int_A w(\boldsymbol{x}) \breve{Y}_j(\boldsymbol{x}) \frac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_1} \mathrm{d}\boldsymbol{x} \\ \sum_j \sum_k (a_j - \tilde{a}_j) a_k \int_A w(\boldsymbol{x}) \breve{Y}_j(\boldsymbol{x}) \frac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_2} \mathrm{d}\boldsymbol{x} \\ \sum_j \sum_k (a_j - \tilde{a}_j) a_k \int_A w(\boldsymbol{x}) \breve{Y}_j(\boldsymbol{x}) \left(x_2 \frac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_1} - x_1 \frac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_2} \right) \mathrm{d}\boldsymbol{x} \end{pmatrix} \quad (14)$$

Since the integrals in the sums do not depend on any of the coefficients a_i, we may precompute them. We obtain the precomputed coefficients $u_i(j,k)$:

$$u_1(j,k) = \int_{x_1=-\infty}^{\infty} \int_{x_2=-\infty}^{\infty} w(\boldsymbol{x}) \cdot \breve{Y}_j(\boldsymbol{x}) \tfrac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_1} \, \mathrm{d}x_2 \, \mathrm{d}x_1 \quad (15)$$

$$u_2(j,k) = \int_{x_1=-\infty}^{\infty} \int_{x_2=-\infty}^{\infty} w(\boldsymbol{x}) \cdot \breve{Y}_j(\boldsymbol{x}) \tfrac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_2} \, \mathrm{d}x_2 \, \mathrm{d}x_1 \quad (16)$$

$$u_3(j,k) = \int_{x_1=-\infty}^{\infty} \int_{x_2=-\infty}^{\infty} w(\boldsymbol{x}) \cdot \breve{Y}_j(\boldsymbol{x}) \left(x_2 \tfrac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_1} - x_1 \tfrac{\partial \breve{Y}_k(\boldsymbol{x})}{\partial x_2} \right) \mathrm{d}x_2 \, \mathrm{d}x_1 \quad (17)$$

With these integrals, the components of the gradient are computed using the following expression:

$$\boldsymbol{g}(\boldsymbol{a}, \tilde{\boldsymbol{a}}) \;:=\; \frac{\partial Q}{\partial \boldsymbol{d}} \;=\; \begin{pmatrix} \sum_j \sum_k (a_j - \tilde{a}_j) \cdot a_k \cdot u_1(j,k) \\ \sum_j \sum_k (a_j - \tilde{a}_j) \cdot a_k \cdot u_2(j,k) \\ \sum_j \sum_k (a_j - \tilde{a}_j) \cdot a_k \cdot u_3(j,k) \end{pmatrix} \quad (18)$$

Note that the negative gradient $-\boldsymbol{g}$ represents a pose change of the *camera*, and this vector has to be transformed into the *robot* coordinate frame in order to provide steering information.

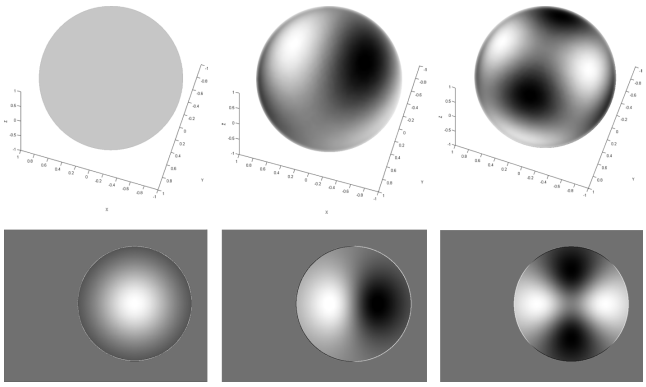

Fig. 1. From left to right: $Y_{0,0}$, $Y_{3,1}$, $Y_{4,2}$ (upper row) and their weighted planar projection for our calibrated fisheye camera. The circle outline consists of the weights obtained by the *integrative equivalent of interpolation*.

4 Spherical Expansion on a Plane

In the experiments performed here, image acquisition takes place with a wide angle lens; we obtain planar fisheye images with a field of view of approximately 160°. For calibration of the lens and modeling the projection from spherical coordinates to the planar image coordinates, we employ the *unified projection model* as described in [10], which can be applied to all single-viewpoint omnidirectional cameras. To obtain the calibration parameters, we use the MATLAB *calibration toolbox* supplied by [10].

Instead of projecting the image onto the unit sphere, we perform the SH-expansion directly in the image coordinates. To that end, we project the spherical basis functions onto a plane corresponding to the imaging process of the fisheye camera in order to obtain *projected basis functions*. The computation of these basis functions is time-consuming, but this needs to be done only once.

Let $b(\hat{x}, \hat{y})$ be the discrete image signal of the camera. Using the calibrated model of the camera and the lens, we obtain discrete basis functions $\hat{Y}_{\ell m}(\hat{x}, \hat{y})$ defined in discrete image coordinates $\hat{x}, \hat{y}$, which conform to the property

$$a_{\ell m} = \sum_{\hat{x},\hat{y}} \hat{Y}_{\ell m}(\hat{x}, \hat{y}) \cdot b(\hat{x}, \hat{y}) \tag{19}$$

and thus are the planar equivalent of the SH expansion on the sphere. Inverting the projection function of the camera model enables us to precompute $\hat{Y}_{\ell m}(\hat{x}, \hat{y})$ by sampling the spherical basis functions for each discrete image coordinate (pixel) and weighting them appropriately.

As the images acquired by the camera have a limited field of view, whereas our view descriptors represent a hemisphere of 180°, we wish to interpolate the missing signal using a radial nearest-neighbor interpolation. However, instead of performing this interpolation as a preprocessing step, we include it in the process

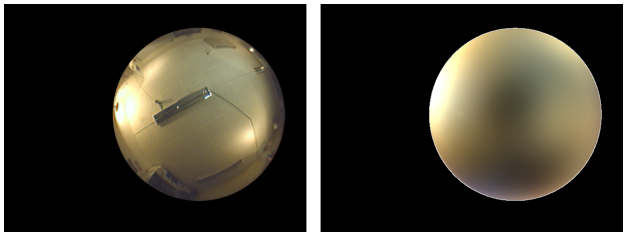

Fig. 2. Original (cropped) image signal (left) and its approximation using planar SH basis functions up to order 7 (right).

of image expansion by increasing the weight of the border pixels. We integrate over the area each interpolated border pixel would occupy on the sphere and obtain the *integrative equivalent of interpolation*. Hence, the relatively tedious interpolation of the input images is avoided.

5 Pose Tracking and Model-Based Motion Control

The navigation process in Optical Rails consists of a sequence of motions of the robot which are each governed by feedback control based on the gradient $-\boldsymbol{g}$. We obtain $-\boldsymbol{g}$ by acquiring an image from the camera, performing an expansion into SH to obtain the view descriptor $\boldsymbol{a}$ of this image, and then using (18) to compute the gradient. We refer to this process as *differential pose tracking*. The extension to following an Optical Rail composed of multiple waypoints is achieved by performing a *waypoint handover* upon reaching the current destination.

Recall that $-\boldsymbol{g}$ gives a camera pose change direction to approach the destination based on the view taken at the current pose. We obtain the steering direction of the robot vehicle towards the destination pose by transforming the gradient $-\boldsymbol{g}$ into the vehicle coordinate system. We define the coordinate system transform $\mathbf{M}$ and the *steering vector* $\boldsymbol{g}_s$, which denotes the proper motion direction of the robot:

$$\boldsymbol{g}_s := (\underbrace{\boldsymbol{g}_{s,x}}_{\text{lateral}}, \quad \underbrace{\boldsymbol{g}_{s,y}}_{\text{forward}}, \quad \underbrace{\boldsymbol{g}_{s,\varphi}}_{\text{rotation}})^T, \qquad \boldsymbol{g}_s = \mathbf{M} \cdot (-\boldsymbol{g}) \tag{20}$$

Tracking and Steering Control for a Non-Holonomic Robot. If the robot follows a track under ideal conditions, it should constantly approach the next target in the forward driving direction while exactly locking to the track in lateral direction. This is particularly important for a non-holonomic vehicle, which can only move in two degrees of freedom. The motion of these vehicles (differential drive, car, tricycle) can be described by a translation rate u_{forward} and a rotation rate u_{steering}. If a trajectory of such a vehicle is subsequently followed using Optical Rails, we may assume that at each time the bearing

(rotation) to the next waypoint is small, and that the lateral spacing to the next waypoint is close to zero.

Thus, in principle it would be sufficient to determine the proper ratio of rotation and forward translation rate to follow the track. However, due to the varying visual structure in different areas of a real environment, the response of the gradient is not uniform in its individual components. Hence, such a ratio will be largely distorted, leading to a cumulative error of the trajectory and oscillations.

We use PI control to determine the steering rate while driving at a fixed forward speed. This is less susceptible to the influences of the changing sensitivity of the individual components of the steering gradient $\boldsymbol{g}_s$; we obtain the *rotational control value* u_{rot} by

$$u_{\mathrm{rot}}(t) = K_{\mathrm{rot},p} \cdot \boldsymbol{g}_{s,\phi}(t) + K_{\mathrm{rot},i} \sum_{k=t-\Delta t}^{t} \boldsymbol{g}_{s,\phi}(k) \tag{21}$$

where $K_{\mathrm{rot},p}$ and $K_{\mathrm{rot},i}$ are weights for the proportional and integral control components, respectively.

A typical issue when approaching a view is a deviation perpendicular to the driving direction, often experienced as a lateral divergence after passing through a curve. A continuous response of the lateral gradient component $\boldsymbol{g}_{s,x}$ is indicative of this problem. Although the non-holonomic vehicle model does not allow direct lateral motions, a steering motion to the right while driving forward also moves the vehicle laterally to the right; we determine the *lateral correction control value* u_{lat} by

$$u_{\mathrm{lat}}(t) = K_{\mathrm{lat},p} \cdot \boldsymbol{g}_{s,x}(t) + K_{\mathrm{lat},i} \sum_{k=t-\Delta t}^{t} \boldsymbol{g}_{s,x}(k) \tag{22}$$

with the weighting factors $K_{\mathrm{lat},p}$ and $K_{\mathrm{lat},i}$ as above.

Finally, a weighted sum of the lateral correction and rotational control values determines the steering rate of the robot, whereas a constant forward velocity is used:

$$u_{\mathrm{forward}} \;=\; \mathrm{sign}(\boldsymbol{g}_{s,y}) \cdot K_{\mathrm{forward}}, \qquad u_{\mathrm{steering}} \;= \alpha_1 \cdot u_{\mathrm{lat}} + \alpha_2 \cdot u_{\mathrm{rot}} \tag{23}$$

To determine the motor control output for the robot drive system, the differential drive vehicle model is employed in connection with a calibration lookup table for motor speed linearization.

6 Recording a Track and Waypoint Selection

To record a track, greyscale images are acquired at a rate of approximately 5 images per second. For each image, the corresponding approximation in SH coefficients is stored in a list representing the raw data for Optical Rails.

Then, a *waypoint selection scheme* is applied to this sequence of view descriptors, which drastically reduces the number of reference views in the sequence. For the subsequent task of *track following*, the set of descriptors is reduced in such a way that subsequent waypoints have a minimum dissimilarity $Q_{\min}$. As views are typically recorded in rapid succession, we obtain a track T of approximately equi-dissimilar view descriptors $\tilde{\boldsymbol{a}}_i$.

$$T = [\tilde{\boldsymbol{a}}_0, \tilde{\boldsymbol{a}}_1, \ldots, \tilde{\boldsymbol{a}}_N] \tag{24}$$

Recall that in an area of convergence, closing in on a waypoint in terms of visual dissimilarity corresponds to approaching this waypoint geometrically; This is, however, on a different scale also depending on the structure of the environment.

The waypoint selection scheme ensures that waypoints which are recorded while the robot only moves on a very small scale are discarded. Therefore only distinct waypoints are kept, which is important for successfully following a track. Waypoints too similar lead to ambiguous handover; if the waypoints are too coarsely spaced, the radius of convergence of the gradient $\boldsymbol{g}$ can be exceeded.

7 Waypoint Handover

The task of waypoint handover in Optical Rails is to detect when the robot has reached its current destination and should steer towards the next waypoint. Without geometry, this decision has to be made based merely on visual information.

Let Q_c be the dissimilarity between the view at the current position and view at the current destination. An intuitive criterion for handover is the dissimilarity measure Q_c falling below a threshold T_c (denoted here as condition I).

$$Q_c < T_c \tag{25}$$

Even if we assume that all successive reference views on a prerecorded track have an equal dissimilarity Q, already minute (lateral or rotational) deviations from

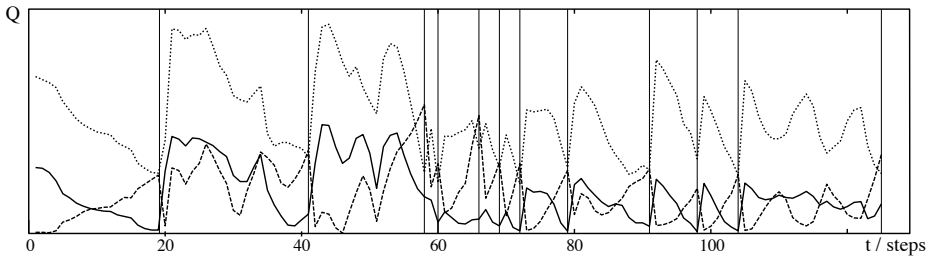

Fig. 3. Behavior of dissimilarity measures Q_p (*previous waypoint, dashed*), Q_c (*current waypoint, solid*), and Q_n (*next waypoint, dotted*) for a course along a track leading through a door (experimental results). Vertical bars indicate waypoint handovers.

the prerecorded track can lead to handover problems. In a real environment the effect of small pose deviations on the dissimilarity measure can be very different. There are regions where pose changes have little influence on Q_c, whereas in other regions the same pose changes cause substantial changes in dissimilarity. This is caused by the different visual structure of the corresponding views in those regions. Therefore, relying only on a constant threshold is an insufficient condition, as it leads to premature handover in regions where the distance measure is insensitive to motions and causes the robot to diverge from the track. Conversely, if the threshold is chosen too restrictive, in motion sensitive regions no handover will occur at all and the robot will halt.

We can, however, augment this criterion with additional conditions for handover in these motion sensitive areas. Let Q_p be the dissimilarity between the view at the current position and the previous waypoint and Q_n be the dissimilarity between the view at the current position and the next waypoint. We observe: While approaching the current destination, Q_p increases while Q_c and Q_n decrease. At the current destination Q_c is small; Q_p and Q_n are approximately equal (see Fig. 3). This is due to the utilized waypoint selection scheme (see Sec. 6), which provides approximately equal visual distance $Q_{\min}$ between successive waypoints. We use the equal visual distance assumption as an additional condition (condition II) for waypoint handover, since it is less dependent on the visual structure of the environment. Therefore we set a small threshold T_i on the difference of Q_n and Q_p, yet allow waypoint handover only if the dissimilarity Q_c is not greater than $T_{c,\max}$.

$$|Q_n - Q_p| < T_i \ \wedge \ Q_c < T_{c,\max} \tag{26}$$

However, if the visual appearance of the environment quickly changes from one waypoint to the next, equal visual distance between Q_p and Q_n cannot be assumed. In this case condition II would fail and handover would be missed. Beyond the destination, Q_n decreases and Q_c increases. Yet our waypoint selection scheme ensures densely spaced waypoints in these areas. Thus, if the robot is about as close to the current destination as it is to the next, handover can also be performed. We set an additional threshold T_d on the difference between Q_n and Q_c and obtain condition III.

$$|Q_n - Q_c| < T_d \tag{27}$$

These three conditions lead to combined handover criterion, which can handle all typical situations.

$$\mathcal{H} = \underbrace{Q_c < T_c}_{\text{condition I}} \ \vee \ \underbrace{(|Q_n - Q_p| < T_i \ \wedge \ Q_c < T_{c,\max})}_{\text{condition II}} \ \vee \ \underbrace{|Q_n - Q_c| < T_d}_{\text{condition III}} \tag{28}$$

8 Putting Optical Rails on a Real Robot

The theoretical concept of Optical Rails has already been extensively discussed [5]. This prior work uses a simulated robot moving in an artificial environment.

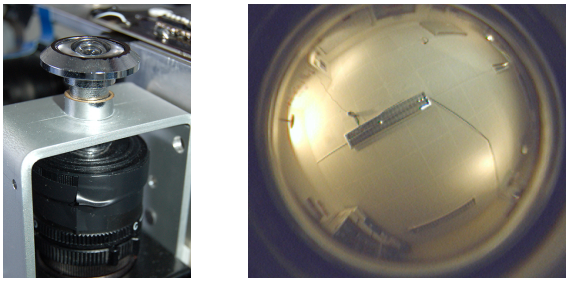

Fig. 4. Door peephole camera system (*left*) and raw omnidirectional image (*right*)

While the theory governing the pose-change gradient $\boldsymbol{g}$ is a common foundation for the Optical Rails concept, the extension of the method to an application for a real vehicle – performing maneuvers in a laboratory environment – comprises substantial differences. This applies in particular to the task of obtaining the view descriptors in an efficient way; we use the Frobenius inner product of planar precomputed basis images and the raw camera image (Sec. 4). This new core concept – in concert with the efficient computation of $\boldsymbol{g}$ – has brought Optical Rails to full real-time capabilities.

8.1 Mobile Robot Platform

For the implementation of Optical Rails on a real robot, we use a proprietary robot platform based upon standard PC components with a six wheel belt-driven differential drive. The robot is equipped with a variety of different sensors connected via CAN-Bus (IR, accelerometer, ultrasonic), and stereo vision to serve as a multi purpose development platform in our labs. For Optical Rails, however, all these standard sensors are neither required nor used.

The robot runs a distributed software framework on the GNU/Linux operating system. Using a network interface server, the robot is controlled and transmits omnidirectional image data to a PC running Matlab where the Optical Rails algorithm (Sec. 8.2) operates.

Omnidirectional Camera System. For omnidirectional image acquisition, we added a hemispherical camera system to the robot, which is based upon a standard Firewire IIDC camera in connection with a door peephole lens (Fig. 4). We use automatic exposure mode to provide a basic level of robustness to global illumination changes. This relatively cheap imaging setup provides more than sufficient image quality for the vision task using SH, since high-frequency signals are disregarded.

Vehicle Model / Drive System. During track following, we perform motions with a rotational and translational rate governed by the PI control approach

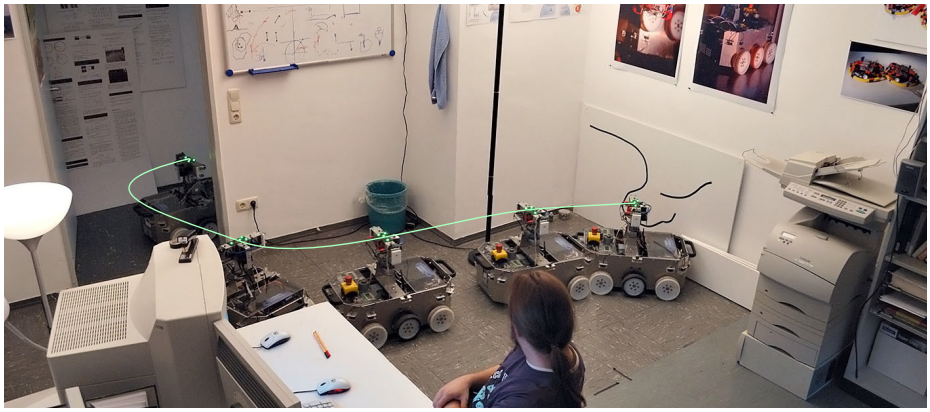

Fig. 5. Experimental result of Optical Rails: following a prerecorded track. The approximate trajectory of the robot has been superimposed

from Sec. 5 to attain minimal tracking and steering deviation. Since we use a differential drive vehicle, we have to translate the values from (23) to motor control values. This is performed using the inverse kinematics for the vehicle model in connection with a calibration of the drive characteristics.

8.2 The Optical Rails Algorithm.

Let T be a prerecorded sequence of reference views $\tilde{\boldsymbol{a}}_i$ as introduced in Sec. 6. The robot is initially located at the pose of the first view $\boldsymbol{a}_0$ in T. Track following takes place as follows:

1. $i = 1$
2. set the coefficient vector $\tilde{\boldsymbol{a}}_i$ as the destination view $\tilde{\boldsymbol{a}}$
3. grab the current image and calculate the coefficient vector $\boldsymbol{a}$
4. compute the gradient $\boldsymbol{g}$ from $\boldsymbol{a}$ to $\tilde{\boldsymbol{a}}$ and the dissimilarities Q_p, Q_c, Q_n
5. transform the gradient $\boldsymbol{g}$ to the steering vector $\boldsymbol{g}_s$
6. issue the PI controller subroutine with $\boldsymbol{g}_s$ as input value
7. translate the resulting rotation and translation rate into motor control values
8. if the combined handover criterion $\mathcal{H}$ is true, advance the destination to the next reference view in T (set $i = i + 1$)
9. If $i > N$, the final destination is reached and the robot stops.
10. Else ($i \leq N$) start over at 2.

Note that for the case of approaching the last waypoint $\tilde{\boldsymbol{a}}_N$, the dissimilarity measure Q_n cannot be computed, which requires special treatment.

9 Results

We have tested Optical Rails by a number of experiments in our laboratory with different tracks to be followed. Here we present one of the most difficult tracks,

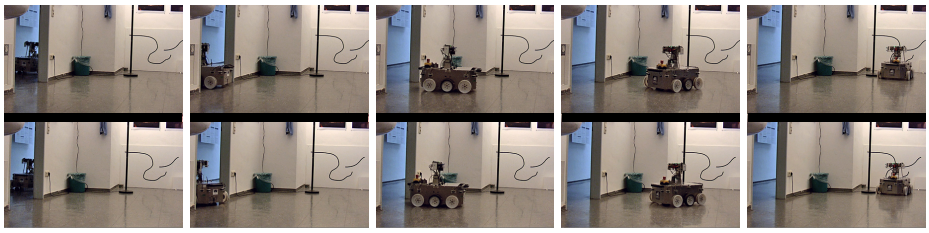

Fig. 6. Comparison of the originally recorded teach-in sequence (*upper images*) with autonomous track-following using Optical Rails (*lower images*).

in which the robot has to pass through a door (very rapidly changing views) and has to perform rather sharp turns. The result of this drive can be seen in Fig. 8.2 and is also supplied as a video, which shows the track as it is being recorded and as the robot drives this track autonomously. Even though the implementation includes transmission of the images via WLAN to a workstation performing the bulk of the computation in MATLAB, we can process up to 5 input images per second. For the experiments, the processing of only one image per second (and thus issuing only one command to the robot per second) has proven to be sufficient. The utilized constants and thresholds for the Optical Rails algorithm were learned and fine tuned on a track from inside our laboratory, through the door, into the corridor. Without changing the parameters, the robot was able to follow different subsequently recorded tracks. Although we currently use precomputed coefficients $u_i(j,k)$ only up to SH order 4 and planar basis functions up to order 7, the results acquired so far are quite satisfactory and prove the Optical Rails concept to be applicable to a real robot.

10 Outlook

Our experimental results show that Optical Rails can be applied to a real robot in an office environment. To robustify the gradient in situations where the planar projection model is largely violated, we plan to incorporate the depth structure of the environment computed of consecutive snapshots. With such an extension, we expect Optical Rails to perform very well even in very challenging environments.

The present implementation allows for one way track following from the beginning of a track. However, for applications in a network of interconnected tracks it is not only important to reverse the direction: To begin driving from an arbitrary point, an *ad hoc* localization step and path planning have to be added.

To obtain a smoother driving behavior, we plan to use model-based trajectory planning in combination with the existing calibration of our robot vehicle.

In addition, it is important to provide illumination invariance to the system, since illumination changes also affect the dissimilarity between views. Corresponding procedures have been proposed in a paper on spherical harmonics [4].

For universal applicability of the system an important task is to provide tolerance against occlusions. Suitable concepts based on statistical models and local

change detection already exist; the implementation still has to be performed. Yet, due to the holistic approach of Optical Rails, minor occlusions do not disturb the system.

References

1. A. Argyros, K. Bekris, S. Orphanoudakis, and Lydia E. Kavraki. Robot homing by exploiting panoramic vision. *Autonomous Robots*, 19(1):7–25, 2005.
2. R. Basri, E. Rivlin, and I. Shimshoni. Visual homing: Surfing on the epipoles. *International Journal of Computer Vision*, 33(2):117–137, 1999.
3. M. O. Franz, B. Schölkopf, H. A. Mallot, and H. H. Bülthoff. Where did I take that snapshot? *Biological Cybernetics*, 79:191–202, 1998.
4. H. Friedrich, D. Dederscheck, M. Mutz, and R. Mester. View-based robot localization using illumination-invarant spherical harmonics descriptors. In A. Ranchordas and H. Araujo, editors, *Proceedings of the International Joint Conference on Computer Vision and Computer Graphics Theory and Applications*, volume 2, pages 543–550. INSTICC and University of Madeira, 2008.
5. H. Friedrich, D. Dederscheck, E. Rosert, and R. Mester. Optical rails. view-based point-to-point navigation using spherical harmonics. In G. Rigoll, editor, *Pattern Recognition*, volume 5096 of *Lecture Notes in Computer Science (LNCS)*, pages 345–354, Heidelberg, 2008. Springer Verlag.
6. T. Goedeme, M. Nuttin, T. Tuytelaars, and L. Van Gool. Omnidirectional vision based topological navigation. *IJCV*, 74(3):219–236, 2007.
7. B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI81*, pages 674–679, 1981.
8. A. Makadia, C. Geyer, and K. Daniilidis. Radon-based structure from motion without correspondences. In *Proc. CVPR*, 2005.
9. A. Makadia, D. Gupta, and K. Daniilidis. Planar ego-motion without correspondences. IEEE Workshop on Motion and Video Computing, Breckenridge, 2005.
10. C. Mei and P. Rives. Single view point omnidirectional camera calibration from planar grids. In *IEEE International Conference on Robotics and Automation*, April 2007.
11. R. Möller, A. Vardy, S. Kreft, and S. Ruwisch. Visual homing in environments with anisotropic landmark distribution. *Autonomous Robots*, 23(3):231–245, 2007.
12. W. Stürzl and H. A. Mallot. Efficient visual homing based on Fourier transformed panoramic images. *Robotics and Autonomous Systems*, 54:300–313, 2006.
13. T. Svoboda, T. Pajdla, and V. Hlavac. Motion estimation using central panoramic cameras. In IEEE International Conference on Intelligent Vehicles, October 1998.