Robotics and Autonomous Systems 11 (1993) 113–127
Elsevier

Trajectory generation from noisy positions of object features for teaching robot paths

Aleš Ude *

Institute for Real-Time Computer Systems and Robotics, University of Karlsruhe, Kaiserstr. 12, 76128 Karlsruhe, Germany

Communicated by F.C.A. Groen
Received November 6, 1992
Revised April 5, 1993

Abstract

Ude, A., Trajectory generation from noisy positions of object features for teaching robot paths, Robotics and Autonomous Systems, 11 (1993) 113–127.

In this paper we discuss a method for generating a trajectory describing robot path using a sequence of noisy positions of features belonging to a moving object obtained from a robot's sensor system. In order to accurately estimate this trajectory we show how uncertainties in the positions of object feature points can be converted into uncertainties in parameters describing the object pose (3-D position and orientation). Noisy estimations of the object poses, together with their uncertainties, are then used as an input to an algorithm that approximates the trajectory describing the robot path. The algorithm is based on natural vector splines and belongs to a family of non-parametric regression techniques which enable the estimation of the trajectory without requiring its functional form to be known. Since dilemma between specifying the trajectory either in Cartesian or in joint coordinates always exists, we present both alternatives. Some simulation results are given which illustrate the accuracy of the approach.

Keywords: Vision-based robot programming; Robot path planning; Smoothing splines; Motion estimation; Uncertainty

1. Introduction

Robot programming is a critical factor in achieving flexibility in robot applications. Intelligent robot systems must possess the ability to interact with human operators efficiently in order to accomplish a variety of tasks within a short period of time. Present teaching methods as teach-in, play-back and programming in robot language systems with or without an interface to a CAD system cannot fulfill this requirement. To overcome this difficulty, most of the present research in the field of robot programming was concentrated on programming with different relief systems such as kinematical models of the robot

Aleš Ude was born in Ljubljana, Slovenia, in 1965. He received the Dipl.Ing. degree in applied mathematics from the University of Ljubljana in 1990. In the same year he joined the Department of Automation, Biocybernetics and Robot Systems, Jožef Stefan Institute, Ljubljana, Slovenia, where he is a research assistant. Since March 1991 he is a visiting scientist at the Institute for Real-Time Computer Systems and Robotics, University of Karlsruhe, Germany, where he is working toward the Ph.D. degree. His research interests include robotics, computer vision, numerical analysis and statistics.

* Fax: +49-721-606-740; e-mail: ude@ira.uka.de

to be taught and 3-D sensor systems. Our research falls into the latter category. It deals with a development of a teaching method which will enable the operator with expertise only in a target task and without any specialized knowledge in robotics to program robot paths quickly.

Our method represents a variant of 'Teaching by Showing' [11,12] and is based on the use of a robot's visual system. We have selected binocular stereo as a cheap and common method for sensing 3-D structure of a workcell. The basic idea is as follows: the operator first moves an object to be manipulated (or a specially designed teaching tool) along the prespecified trajectory. The action is recorded and reconstructed by an image analysis system which produces a list of discrete and noisy object poses as its output. Using these noisy measurements, a smooth and continuous approximation of the true trajectory is generated. Finally, a program in a robot language system is generated and the robot is ready to execute the task. Our system, which is still under development, comprises the following functions:

(i) The extraction of the image features from a stereo image sequence.
(ii) The simultaneous matching of the image features extracted from a stereo image pair with the modelled object features and the object tracking.
(iii) The computation of 3-D object poses together with their uncertainties and the estimation of the object's trajectory.
(iv) The transformation of the trajectory into the robot language system.

The assumption that the underlying camera system is binocular stereo has its role in the first two stages of the process. Our imaging computer can produce separate lists of extracted image features such as distinct points or straight line segments from a stereo image pair in real-time. The proposed matching procedure and object tracking algorithms are completely based on a CAD model of the moving object and will be described elsewhere. Only the third stage of our system is considered in this article, i.e. we concentrate on the task of estimating the trajectory of the moving object for teaching robot paths when range data and the 3-D model of the object are available. Most of the results are valid for any range sensor which enables measurement of 3-D positions of object feature points and estimation of uncertainties in these measurements. The binocular stereo assumption is only used to illustrate the approach. For a review of range sensors used in robotics see Nitzan [16].

We assume in this article that image feature points have been extracted from each pair of stereo images, that they have been related to the object feature points and that they have been tracked through the whole sequence. A sequence of poses of the moving object can then be computed. In previous systems similar to ours [11], only a couple of important passing points such as approach-point, grip-point, and withdraw-point on the object's trajectory could have been programmed. But in many applications, for instance in the presence of obstacles, the whole trajectory of the robot's end-effector must be controlled.

A lot of research has been done in 3-D motion estimation from long image sequences in recent years. Most of it assumes a model for 3-D rigid body motion. For instance, a motion estimation scheme based on superposition of constant precession rotational motion and polynomial translational motion has been proposed. The parameters of the model are estimated by applying a least-squares criterion [21] or an extended Kalman filter [22]. Although this and similar approaches may allow the parameters of the model to evolve in time, they are not applicable to our problem since we are looking for a smooth estimate of the object's trajectory over a longer period of time without assuming a specific parametrical model for the rigid body motion. Furthermore, these approaches primarily concentrate on the estimation of the object motion parameters rather than the object trajectory which is a more meaningful quantity in terms of robot tasks such as for example a pick and place operation. In this article we propose a new method which concentrates on the estimation of the moving object's trajectory.

It is a common belief among vision researchers that uncertainties in image features must be considered explicitly in order to obtain accurate results when estimating motion parameters of a moving object. It has been reported [15] that the uncertainties in the reconstructed 3-D coordinates of the object feature points obtained from a binocular stereo system can be modelled as Gaussian which leads to a good performance in motion analysis. An interesting method to relate the uncertainties in the reconstructed feature point positions to the uncertainties in the parameters describing the object pose has

been proposed in [13]. Unfortunately, the non-minimal representation of the orientation by quaternions used there is not appropriate for our application. For this reason we modified the approach to work with the minimal representation of the orientation by rotation vectors. We also developed an alternative approach in which the pose and the uncertainties are obtained directly from the projected object features.

We applied a non-parametric regression technique based on smoothing vector splines in order to obtain a smooth estimate of the object's trajectory. Non-parametric regression using smoothing splines is a flexible and widely-applicable approach to curve estimation [20,6,19]. The possibility of the automatic choice of the amount of smoothing makes it particularly attractive. We present how the trajectory in the Cartesian space can be generated from a sequence of object poses together with their uncertainties previously extracted from a sequence of stereo images using smoothing vector splines. As it is well known, Cartesian paths are prone to various difficulties relating to workspace and singularities [4]. Furthermore, they must be converted into equivalent joint quantities at run-time which can represent a significant computational burden. For these reasons we outline an algorithm for the reconstruction of the trajectory in the robot's joint space as well.

This article is divided into six sections. In Section 2 calculation and representation of the pose in the form suitable for the object's trajectory reconstruction is discussed. Section 3 relates the uncertainties in the estimated feature point positions to the uncertainties in the parameters determining the pose. Section 4 discusses estimation of the moving object's trajectory while some experimental results are given in Section 5. The concluding remarks are gathered in Section 6.

2. Calculating and representing the object pose

To study the pose of the rigid object, we rigidly attach a coordinate frame, called object-centered coordinate frame, to it. For simplicity we assume that all object feature points are visible in each stereo image pair. The centroid of the feature points is chosen to be the origin of the object-centered coordinate frame. We consider motions of the object as motions of the frame attached to the moving object relative to the frame attached to the stationary object. The frame attached to the stationary object is assumed to coincide with the world coordinate system. Let the feature points of the stationary object be $\{\boldsymbol{m}_i\}_{i=1}^n$. A displacement of every feature point $\boldsymbol{p}_i'$ belonging to the moving object from its initial to its current position is given by

$$\boldsymbol{p}_i' = R'\boldsymbol{m}_i + \boldsymbol{t}', \tag{1}$$

where R' is a 3×3 *rotation matrix* and $\boldsymbol{t}'$ is a 3-D *translation vector*. At least three noncolinear points are needed in order to obtain both $\boldsymbol{t}'$ and R' without ambiguity. However, in the presence of noise, which is unavoidable in image processing, the positions of the estimated object feature points $\boldsymbol{p}_i$ returned by the system differ from the positions of the actual object feature points $\boldsymbol{p}_i'$:

$$\boldsymbol{p}_i = \boldsymbol{p}_i' + \boldsymbol{e}_i, \quad i = 1, \ldots, n, \tag{2}$$

where $\boldsymbol{e}_i$ are zero mean, mutually independent random vectors. Therefore R' and $\boldsymbol{t}'$ can be obtained only approximately by minimizing

$$\sum_{i=1}^{n} \left\| \boldsymbol{p}_i - \left(\tilde{R}\boldsymbol{m}_i + \tilde{\boldsymbol{t}} \right) \right\|^2 \tag{3}$$

over all rotation matrices $\tilde{R}$ and over all translation vectors $\tilde{\boldsymbol{t}}$. Defining $\boldsymbol{p} = 1/n \sum_{i=1}^n \boldsymbol{p}_i$, $\boldsymbol{m} = 1/n \sum_{i=1}^n \boldsymbol{m}_i$ and $\boldsymbol{v}_i = \boldsymbol{p}_i - \boldsymbol{p}$, $i = 1, \ldots, n$, it has been shown [10] that the solution to (3) is given by $\boldsymbol{t} = \boldsymbol{p} - R\boldsymbol{m} = \boldsymbol{p}$ (since we assume that the centroid $\boldsymbol{m}$ coincides with the origin of the world coordinate system) and by R minimizing

$$g(\tilde{R}) = \sum_{i=1}^{n} \left\| \boldsymbol{v}_i - \tilde{R}(\boldsymbol{m}_i - \boldsymbol{m}) \right\|^2 = \sum_{i=1}^{n} \left\| \boldsymbol{v}_i - \tilde{R}\boldsymbol{m}_i \right\|^2 \tag{4}$$

over all rotation matrices. It follows that the translational and the orientational component of the displacement can be estimated separately. We have used a closed-form solution given in [1] to determine the optimal rotation matrix.

The pose of the object can be parametrized by the rotation matrix R and the translation vector $t = p$. While a parametrization of the translation by p fulfills the requirement that a parametrization must be minimal, the same is not true for a parametrization of the orientation by the rotation matrix R. If non-minimal representations were used, the singularities in covariance matrices describing the uncertainties in the estimated poses could not be avoided. Moreover, the dimensionality of the problem of approximating the object trajectory would be increased and since non-minimal representations are subject to constraints, the problem would become constrained. Among several minimal representations of orientation we have chosen a parametrization by a 3-D rotation vector r because it does not contain, unlike in robotics frequently used Euler angles, any singularities. The rotation vector is defined by $r = \theta n$, where n is a unit vector in the direction of the axis of rotation and θ is the angle of rotation about this axis. Defining antisymmetric matrix $X(r)$ to be

$$X(r) = \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix},$$

every rotation matrix can be written as

$$R(r) = e^{X(r)} = I + \sin(\theta)X(n) + (1 - \cos(\theta))X(n)^2,$$

which is just the Rodrigues' formula for the rotation $R(n, \theta)$. An inverse mapping of the rotation matrix into the rotation vector is given in [2] and is used to convert the rotation matrix obtained by minimizing (4) into the rotation vector.

Let the underlying camera system be binocular stereo and let both cameras be modelled by a simple pinhole camera model. Let $B_l = [b_{lij}]$, $B_r = [b_{rij}]$ be the 3×4 calibration matrices mapping the object points given in homogeneous coordinates [18] into the left and right image, let $u_{li} = [x_{li}, y_{li}]^T$, $u_{ri} = [x_{ri}, y_{ri}]^T$ be the projections of the point p_i into the left and right image plane and let A_l and A_r be the mappings into the left and right image plane, respectively:

$$u_{l(r)i} = A_{l(r)}(p_i) = \begin{bmatrix} \dfrac{b_{l(r)11}p_{i1} + b_{l(r)12}p_{i2} + b_{l(r)13}p_{i3} + b_{l(r)14}}{b_{l(r)31}p_{i1} + b_{l(r)32}p_{i2} + b_{l(r)33}p_{i3} + b_{l(r)34}} \\ \dfrac{b_{l(r)21}p_{i1} + b_{l(r)22}p_{i2} + b_{l(r)23}p_{i3} + b_{l(r)24}}{b_{l(r)31}p_{i1} + b_{l(r)32}p_{i2} + b_{l(r)33}p_{i3} + b_{l(r)34}} \end{bmatrix}. \tag{5}$$

Given the measurements u_{li}, u_{ri}, the 3-D position of the point p_i used in (2) can be obtained by use of triangulation equations [18]:

$$p_i = h(u_{li}, u_{ri}) = (P^T P)^{-1} P^T z, \tag{6}$$

$$P = \begin{bmatrix} b_{l11} - b_{l31}x_{li} & b_{l12} - b_{l32}x_{li} & b_{l13} - b_{l33}x_{li} \\ b_{l21} - b_{l31}y_{li} & b_{l22} - b_{l32}y_{li} & b_{l23} - b_{l33}y_{li} \\ b_{r11} - b_{r31}x_{ri} & b_{r12} - b_{r32}x_{ri} & b_{r13} - b_{r33}x_{ri} \\ b_{r21} - b_{r31}y_{ri} & b_{r22} - b_{r32}y_{ri} & b_{r23} - b_{r33}y_{ri} \end{bmatrix}, \quad z = \begin{bmatrix} b_{l34}x_{li} - b_{l14} \\ b_{l34}y_{li} - b_{l24} \\ b_{r34}x_{ri} - b_{r14} \\ b_{r34}y_{ri} - b_{r24} \end{bmatrix}.$$

However, in the presence of noise, the computed location is only an approximation of the true location. As illustrated in *Fig. 1* for errors caused by quantization, the estimated location of p_i can lie anywhere in the shaded region surrounding the true location [15]. It is common to assume that errors in image feature point coordinates u_{li}, u_{ri} are mutually independent and Gaussian [15,13] with covariance matrices $\sigma^2 V_{li}$ and $\sigma^2 V_{ri}$. As we shall see later the actual scale parameter σ is not needed during the

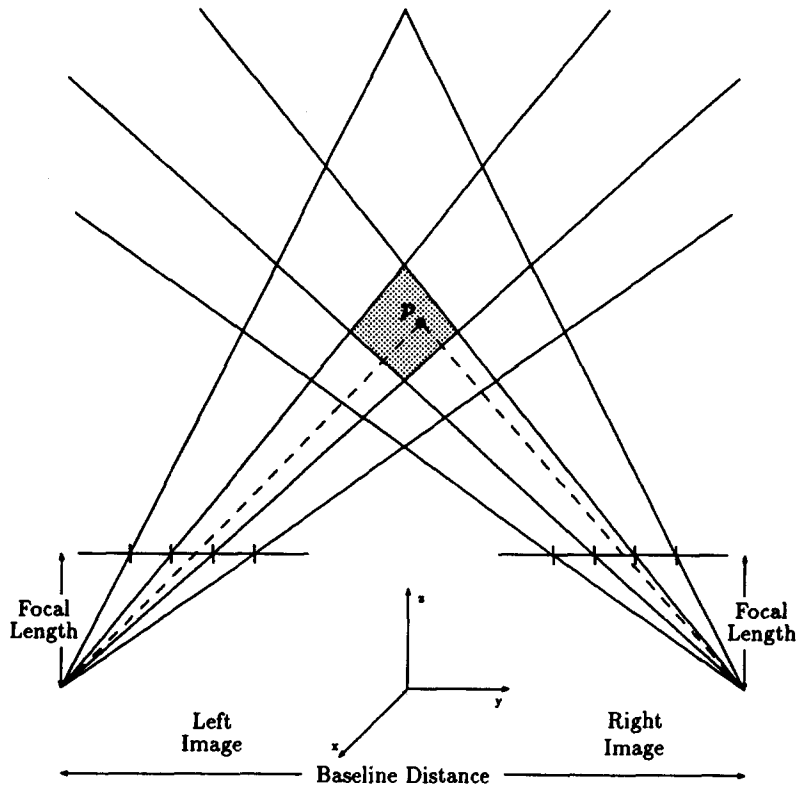

Fig. 1. Stereo geometry showing triangulation uncertainty (aligned cameras).

approximation process. If we further assume that the pixels are quadratic, then one may often set $V_{li} = V_{ri} = I_2$, where I_2 is a 2×2 identity matrix. With these assumptions, one can estimate the object pose directly from the image points matched with the object points without using triangulation formulae first:

$$\text{minimize} \sum_{i=1}^{n} \left\| \boldsymbol{u}_{li} - A_l\left(R(\tilde{\boldsymbol{r}})\boldsymbol{m}_i + \tilde{\boldsymbol{t}}\right)\right\|^2_{V_{li}^{-1}} + \left\| \boldsymbol{u}_{ri} - A_r\left(R(\tilde{\boldsymbol{r}})\boldsymbol{m}_i + \tilde{\boldsymbol{t}}\right)\right\|^2_{V_{ri}^{-1}}, \tag{7}$$

over all translation vectors $\tilde{\boldsymbol{t}}$ and over all rotation vectors $\tilde{\boldsymbol{r}}$. $R(\tilde{\boldsymbol{r}})$ denotes the rotation matrix corresponding to the rotation vector $\tilde{\boldsymbol{r}}$. We have used the notation

$$\|\boldsymbol{x}\|^2_V = \boldsymbol{x}^{\mathrm{T}} V \boldsymbol{x}.$$

This minimization problem is a highly nonlinear one and can be solved only iteratively with the help of the Levenberg–Marquardt method for solving nonlinear least-squares problems. Note that Eq. (7) can be extended to include the feature points observed by only one of both cameras. Furthermore, by leaving out the terms belonging to one of the cameras, the object pose can be calculated even when only one camera is available (since the structure of the object is known). The object pose obtained by minimizing (7) is more accurate than the one calculated by using the triangulation formulae (6) first and by minimizing (3) afterwards. On the other hand, minimizing (3) can be performed with noniterative methods. It is possible to consider the uncertainties also when determining the pose by use of 3-D object point positions as in (3). However, the resulting minimization problem can again be solved only

iteratively. Since minimization of (7) gives better results, we shall not consider this possibility in the rest of the paper. The object pose and its uncertainties can also be calculated when other image features are used. For the case of straight line segment correspondences see [23].

3. Modelling of uncertainties in the object pose

Let us assume that 3-D feature point positions $\boldsymbol{p}_i$ were obtained by use of triangulation Eqs. (6) and that the distribution of noise in image feature points is Gaussian. Since triangulation is a nonlinear mapping, the true distribution of noise in 3-D feature point positions is in general non-Gaussian (see [3] for the case of quantization errors). Nevertheless, it may be approximated as Gaussian [15] with mean at $\boldsymbol{h}(\boldsymbol{u}_{li}, \boldsymbol{u}_{ri})$ and covariance matrix $\sigma^2 V_i$, where

$$V_i = J_i \begin{bmatrix} V_{li} & 0 \\ 0 & V_{ri} \end{bmatrix} J_i^{\mathrm{T}} \in R^{3\times 3}, \tag{8}$$

J_i is the 3×4 Jacobian of the mapping $\boldsymbol{h}$ at $(\boldsymbol{u}_{li}, \boldsymbol{u}_{ri})$. This approximation holds true when the distance between object feature points and both cameras is not too large or, equivalently, when the disparity between image feature points is not too small [15,13]. The Jacobian of the mapping $\boldsymbol{h}$ can be easily computed using the chain rule for the derivative of a product of two matrix functions and the following formula for the derivative of an inverse of a matrix function:

$$\left(P(x)^{-1}\right)' = -P(x)^{-1} P(x)' P(x)^{-1}.$$

We are interested in the distribution of error in the parameters describing the pose. We first consider the case when the pose is obtained by minimizing (3). The estimated translation vector $\boldsymbol{t}$ fulfills

$$\boldsymbol{t} = \boldsymbol{p} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{p}_i = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{p}_i' + \boldsymbol{e}_i = \boldsymbol{t}' + \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{e}_i. \tag{9}$$

Let $\boldsymbol{e}_i$ be, as in Eq. (2), zero mean, mutually independent and normally distributed random vectors with covariance matrices $\sigma^2 V_i$. In the case of binocular stereo, the covariances can be estimated by (8). It follows from (9) that the translation vector $\boldsymbol{t}$ is normally distributed, too, with covariance matrix $\sigma^2 \Sigma_p$, where

$$\Sigma_p = \frac{1}{n^2}\sum_{i=1}^{n} V_i. \tag{10}$$

The estimation of the covariance of the rotation vector requires more effort. We proceed somehow similar as in [13]. First we relate the true rotation R' to the estimated rotation R by use of a *differential rotation matrix* δR:

$$R' = \delta R R. \tag{11}$$

With reference to (1), (2), (9), and using the fact that multiplication with the orthogonal matrix δR preserves the norm, we may write $R'\boldsymbol{m}_i = \boldsymbol{p}_i' - \boldsymbol{t}' = \boldsymbol{p}_i - \boldsymbol{p} + (1/n\sum_{j=1}^{n} \boldsymbol{e}_j - \boldsymbol{e}_i) = \boldsymbol{v}_i + \boldsymbol{f}_i$. Therefore minimum of (4) is achieved at

$$g(R) = \sum_{i=1}^{n} \|\boldsymbol{v}_i - R\boldsymbol{m}_i\|^2 = \sum_{i=1}^{n} \|\boldsymbol{v}_i - \delta R^{\mathrm{T}} R' \boldsymbol{m}_i\|^2 = \sum_{i=1}^{n} \|(\delta R - I)\boldsymbol{v}_i - \boldsymbol{f}_i\|^2.$$

Assuming a small rotation angle and neglecting higher order terms, the differential rotation matrix can be estimated by $\delta R \approx I + X(\boldsymbol{\varepsilon})$ [9], where ε_x, ε_y and ε_z are small rotations around the coordinate axes. With this approximation we can write:

$$g(R) \approx \sum_{i=1}^{n} \|X(\boldsymbol{\varepsilon})\boldsymbol{v}_i - \boldsymbol{f}_i\|^2 = \sum_{i=1}^{n} \|X(-\boldsymbol{v}_i)\boldsymbol{\varepsilon} - \boldsymbol{f}_i\|^2. \tag{12}$$

Hence minimization of (4) for the rotation matrix R is equivalent to the minimization of (12) for the differential rotation vector $\boldsymbol{\varepsilon}$. But this is equivalent to solving the overdetermined system $A\boldsymbol{\varepsilon} = f$ in a least-squares sense, where

$$A = \begin{bmatrix} X(-v_1) \\ \cdots \\ X(-v_n) \end{bmatrix}, \quad A \in R^{3n\times 3} \text{ and } f = \begin{bmatrix} f_1 \\ \cdots \\ f_n \end{bmatrix}, \quad f \in R^{3n}.$$

The solution to this problem is $\boldsymbol{\varepsilon} = A^+ f$, where A^+ denotes the pseudoinverse of the matrix A. Since f can be written as

$$f = B\begin{bmatrix} \boldsymbol{e}_1 \\ \cdots \\ \boldsymbol{e}_n \end{bmatrix}, \quad B = \begin{bmatrix} \left(\frac{1}{n}-1\right)I_3 & \frac{1}{n}I_3 & \cdots & \frac{1}{n}I_3 \\ \frac{1}{n}I_3 & \left(\frac{1}{n}-1\right)I_3 & \cdots & \frac{1}{n}I_3 \\ & \cdots & \cdots & \\ \frac{1}{n}I_3 & \frac{1}{n}I_3 & \cdots & \left(\frac{1}{n}-1\right)I_3 \end{bmatrix}, \quad B \in R^{3n\times 3n},$$

where I_3 is a 3×3 identity matrix, we have $\boldsymbol{\varepsilon} = A^+ B\boldsymbol{e}$. We are interested in the distribution of noise contained in the rotation vector $\boldsymbol{r}$. To obtain this, we must relate $\boldsymbol{\varepsilon}$ to $\mathbf{d}\boldsymbol{r} = \boldsymbol{r}' - \boldsymbol{r}$, where $R' = \mathrm{e}^{X(r')}$ and $R = \mathrm{e}^{X(r)}$. We can write two first order approximations for R':

$$R' \approx R + X(\boldsymbol{\varepsilon})R, \tag{13}$$

$$R' \approx R + \frac{\partial R}{\partial r_x}\mathrm{d}r_x + \frac{\partial R}{\partial r_y}\mathrm{d}r_y + \frac{\partial R}{\partial r_z}\mathrm{d}r_z = R + X(H(\boldsymbol{r})\,\mathbf{d}\boldsymbol{r})R, \tag{14}$$

where

$$H(\boldsymbol{r}) = \begin{bmatrix} h(\theta)r_x^2 + f(\theta) & h(\theta)r_x r_y - g(\theta)r_z & h(\theta)r_x r_z + g(\theta)r_y \\ h(\theta)r_x r_y + g(\theta)r_z & h(\theta)r_y^2 + f(\theta) & h(\theta)r_y r_z - g(\theta)r_x \\ h(\theta)r_x r_z - g(\theta)r_y & h(\theta)r_y r_z + g(\theta)r_x & h(\theta)r_z^2 + f(\theta) \end{bmatrix} \tag{15}$$

and

$$\theta = \|\boldsymbol{r}\|, \quad f(\theta) = \frac{\sin(\theta)}{\theta}, \quad g(\theta) = \frac{1-\cos(\theta)}{\theta^2}, \quad h(\theta) = \frac{1-f(\theta)}{\theta^2}.$$

For the proof of (14) see [2]. It can be shown that the matrix $H(\boldsymbol{r})$ is invertible everywhere. By equating both first order terms we obtain $X(\boldsymbol{\varepsilon}) = X(H(\boldsymbol{r})\,\mathbf{d}\boldsymbol{r})$ or, equivalently, $\mathbf{d}\boldsymbol{r} = H(\boldsymbol{r})^{-1}\boldsymbol{\varepsilon} = H(\boldsymbol{r})^{-1}A^+B\boldsymbol{e}$. The covariance of the rotation vector can thus be approximated by $\sigma^2\Sigma_r$, where

$$\Sigma_r = H(\boldsymbol{r})^{-1}A^+B\ \mathrm{diag}(V_i)\left(H(\boldsymbol{r})^{-1}A^+B\right)^{\mathrm{T}}. \tag{16}$$

Let $\boldsymbol{p}_j$ be the translation vectors and let $\boldsymbol{r}_j$ be the rotation vectors which minimize (3) at the time instants $\{t_j\}_{j=1}^N$ at which the stereo images are recorded. Let $\sigma^2\Sigma_{pj}$ and $\sigma^2\Sigma_{rj}$ be the covariance matrices of the translation and the rotation vector at the time instants t_j calculated as given in (10) and (16), respectively. Let vector functions $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ respectively represent the true trajectory of the translation vector and the rotation vector. We have arrived to the following regression *Model 1*:

$$\left.\begin{aligned} \boldsymbol{p}_j &= \boldsymbol{\alpha}(t_j) + \boldsymbol{\epsilon}_j, \\ \boldsymbol{r}_j &= \boldsymbol{\beta}(t_j) + \boldsymbol{\tau}_j, \end{aligned}\right\}, \quad j = 1,\ldots,N, \tag{17}$$

where ϵ_j and τ_j form a set of zero mean, mutually independent random vectors with covariance matrices $\sigma^2\Sigma_{pj}$ and $\sigma^2\Sigma_{rj}$, respectively. The vector functions α and β are usually referred to as the regression curves. When this model is used, the translational and the orientational part of the pose can be estimated separately.

If the translation vector p_j and the rotation vector r_j are calculated by minimizing (7), the uncertainties can be estimated as well. Denoting

$$g(\tilde{s}) = \begin{bmatrix} A_l(R(\tilde{r})m_1 + \tilde{t}) \\ A_r(R(\tilde{r})m_1 + \tilde{t}) \\ \cdots \\ A_l(R(\tilde{r})m_N + \tilde{t}) \\ A_r(R(\tilde{r})m_N + \tilde{t}) \end{bmatrix}, \quad u = \begin{bmatrix} u_{l1} \\ u_{r1} \\ \cdots \\ u_{lN} \\ u_{rN} \end{bmatrix}, \quad \tilde{s} = \begin{bmatrix} \tilde{t} \\ \tilde{r} \end{bmatrix}, \quad V = \operatorname{diag}(V_{l1}, V_{r1}, \ldots, V_{lN}, V_{rN}), \tag{18}$$

(7) can be rewritten as

$$(u - g(\tilde{s}))^T V^{-1}(u - g(\tilde{s})). \tag{19}$$

The covariance matrix of the minimizer of (19), or equivalently of (7), is given by $\sigma^2\Sigma_b$, where

$$\Sigma_b = (J^T V^{-1} J)^{-1}, \quad J = \frac{\partial g}{\partial s}(s). \tag{20}$$

$\partial g/\partial s(s)$ denotes the Jacobian of g computed at the solution $s = [p^T, r^T]^T$ of (7). Hence the minimization of (7) results in the following regression *Model 2*:

$$\gamma(t_j) = s_j + \nu_j, \tag{21}$$

where $s_j = [p_j^T, r_j^T]^T$ are the minimizers of (7) at the time instants $\{t_j\}_{j=1}^N$, ν_j are zero mean, mutually independent random vectors with the covariance matrices $\sigma^2\Sigma_{bj}$ obtained from (20), and γ is the true trajectory. Note that in the regression Model 1 the uncertainties in the translation and the rotation vector are decoupled, which is not the case in Model 2. It follows that the translational and the orientational part of the pose must be estimated simultaneously in this case. However, Model 2 is the more accurate one.

Because of the advantages of trajectories specified in a robot joint space, one is often tempted to estimate the trajectory at the joint level. This can be done provided the kinematics of the manipulator is known. Let $k: R^6 \to R^6$ be the mapping representing the direct kinematics of a manipulator with six degrees of freedom. Given a set of object poses s_j together with their covariance matrices $\sigma^2\Psi_j$ which can be obtained either by use of Model 1 or Model 2, one can convert them into the joint coordinates using the formulae:

$$q_j = k^{-1}(s_j), \quad \Phi_j = J_j \Psi_j J_j^T, \quad J_j = \left(\frac{\partial k}{\partial q}(q_j)\right)^{-1}. \tag{22}$$

In this case, we obtain the following regression *Model 3*:

$$q_j = \chi(t_j) + \varphi_j, \quad j = 1, \ldots, N, \tag{23}$$

where χ is the true joint trajectory and φ_j are zero mean, mutually independt random vectors with covariance matrices $\sigma^2\Phi_j$.

4. Estimation of the object's trajectory

After the whole motion of the object has been recorded and tracked, we are given the regression Model 1, 2, or 3. All of them have the following general form:

$$z_j = \vartheta(t_j) + \kappa_j, \quad j = 1, \ldots, N, \tag{24}$$

where $z_j \in R^M$ are the mesurements, ϑ is the true trajectory and κ_j are zero mean, independent random vectors with covariance matrices $\sigma^2\Sigma_j$. The form of the regression curve ϑ is unknown because, in general, the object can follow any continuous path. Since only discrete approximations of the points on the trajectory are available, we cannot expect to reconstruct it exactly. The aim of the reconstruction is to find the trajectory along which the robot end-effector will move, therefore we require the trajectory to be smooth. If the measurements were simply interpolated, this requirement would not be fulfilled. On the other hand, the trajectory must not be too distant from the measurements. Hence we must seek for the compromise between smoothness and goodness-of-fit. This motivates us to search for the curve that best fits our data in the class

$$W_2^m[a, b] = \left\{ \mu : [a, b] \to R^M; \quad \begin{array}{l} \mu^{(j)} \text{ is absolutely continuous}, \quad j = 0, \ldots, m-1, \\ \mu_k^{(m)} \text{ are square integrable}, \quad k = 1, \ldots, M, \end{array} \right.$$

where $a = t_1$ and $b = t_N$. The natural measure of smoothness associated with every component of the function $\mu \in W_2^m[a, b]$ is $\int_a^b \mu_k^{(m)}(t)^2 \, dt$ while the standard measure of goodness-of-fit for the regression model (24) is $1/N \sum_{j=1}^N \| z_j - \mu(t_j) \|^2_{(\sigma^2\Sigma_j)^{-1}}$. One possibility for combining these two criterion functions to obtain an overall performance measure is to estimate the function μ by the function μ_λ which minimizes [6]:

$$\frac{1}{N} \sum_{j=1}^N \left\| z_j - \mu(t_j) \right\|^2_{(\sigma^2\Sigma_j)^{-1}} + \sum_{k=1}^M \tilde{\lambda}_k \int_a^b \mu_k^{(m)}(t)^2 \, dt \tag{25}$$

over $\mu \in W_2^m[a, b]$. The parameter $\tilde{\lambda}$ governs the tradeoff between smoothness and goodness-of-fit. It can be easily seen that the scale parameter σ^2 can be left out since minimizing (25) is equivalent to minimizing

$$\frac{1}{N} \sum_{j=1}^N \left\| z_j - \mu(t_j) \right\|^2_{\Sigma_j^{-1}} + \sum_{k=1}^M \lambda_k \int_a^b \mu_k^{(m)}(t)^2 \, dt, \tag{26}$$

where the relation between the new and the old smoothing parameter is given by $\lambda = \sigma^2\tilde{\lambda}$. In the special case $m = 2$, minimization of (26) gives rise to the trajectory fitting data well and having the smallest possible overall second derivative. This property of the fitted trajectory is a very favourable one since the acceleration capabilities of the mechanism are limited.

We denote by $\mathcal{NS}^{2m}(t_1, \ldots, t_N)$ the vector space of *natural splines* [6] with knots at the $\{t_i\}$, i.e. the set of functions $s(t) = \sum_{i=0}^{m-1} \theta_i t^i + \sum_{i=1}^N \delta_i (t - t_i)_+^{2m-1}$, where

$$(x)_+ = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases},$$

satisfying

(i) s is a piecewise polynomial of degree $2m - 1$ on any subinterval $[t_i, t_{i+1})$,
(ii) s has $2m - 2$ continuous derivatives,
(iii) s has a $(2m - 1)$st derivative that is a step function with jumps at $t_1, \ldots, t_N$, and
(iv) s is a polynomial of degree $m - 1$ outside of $[a, b]$.

The following theorem can be established to solve the problem (26):

Theorem 1. *Let $x_1, \ldots, x_N$ be a basis for $\mathcal{NS}^{2m}(t_1, \ldots, t_N)$,*

$$x_j(t) = \sum_{i=0}^{m-1} \theta_{ij} t^i + \sum_{i=1}^N \delta_{ij} (t - t_i)_+^{2m-1}$$

and suppose that $N \geq 2m$. For fixed $0 < \lambda_k < \infty$, $k = 1, \ldots, M$, there is a unique minimizer, μ_λ, of (26) in $W_2^m[a, b]$. Moreover, $\mu_\lambda \in \mathcal{NS}^{2m}(t_1, \ldots, t_N)$ and therefore has the form $\sum_{j=1}^N \beta_{\lambda j} x_j(t)$. The coefficients $\beta_\lambda = [\beta_{\lambda 1}^T, \ldots, \beta_{\lambda N}^T]^T \in R^{MN}$ are the solution to

$$(X + \Sigma G)\beta_\lambda = z, \tag{27}$$

where

$$X = \begin{bmatrix} x_1(t_1)I_M & x_2(t_1)I_m & \dots & x_N(t_1)I_M \\ & \dots & \dots & \\ x_1(t_N)I_M & x_2(t_N)I_m & \dots & x_N(t_N)I_M \end{bmatrix},$$

$$G = (-1)^m(2m-1)!N\begin{bmatrix} \delta_{11}\Lambda & \delta_{12}\Lambda & \dots & \delta_{1N}\Lambda \\ & \dots & \dots & \\ \delta_{N1}\Lambda & \delta_{N2}\Lambda & \dots & \delta_{NN}\Lambda \end{bmatrix},$$

$\Sigma = \mathrm{diag}(\Sigma_i)$ *and* $\Lambda = \mathrm{diag}(\lambda_k)$. $X, G, \Sigma \in R^{MN\times MN}$, $\Lambda, I_M \in R^{M\times M}$ *and* $z = [z_1^{\mathrm{T}}, \dots, z_n^{\mathrm{T}}]^{\mathrm{T}} \in R^{MN}$.

The proof is based on the straightforward generalization of the proof given in [6] for the case when $\boldsymbol{\mu}$ is scalar instead of vector function and is omitted. A similar theorem showing that natural vector splines are the solution to the minimization problem (26) has been given in [8]. Note that the matrix of the system (27) depends on the selected basis functions of the space of natural splines. By choosing the basis based on B-splines [14,5], the matrix of the system (27) becomes banded what significantly reduces the computational cost of the algorithm and also results in a numerically stable algorithm. There exists an alternative way to write Eqs. (27). Similarly as 1-D case [6], premultiplying (27) by $X^{\mathrm{T}}\Sigma^{-1}$ yields

$$(X^{\mathrm{T}}\Sigma^{-1}X + \Omega)\boldsymbol{\beta}_\lambda = X^{\mathrm{T}}\Sigma^{-1}z, \tag{28}$$

where

$$\Omega = \left\{\left(N\int_a^b x_i^{(m)}(t)x_j^{(m)}(t)\,\mathrm{d}t\right)\Lambda\right\}_{i,j=1}^{N} \in R^{MN\times MN}.$$

The matrix of system (28) is positive definite.

The case when some of the measurements z_i are given exactly can be included by setting the corresponding covariances Σ_i to ξI_M with $\xi = 0$ when Eq. (27) is used, or by taking ξ 'small enough' when Eq. (28) is used. Such ξ can always be found because

$$\lim_{\xi\to 0}\boldsymbol{\mu}_{\lambda,\xi} = \boldsymbol{\mu}_{\lambda,0},$$

where $\boldsymbol{\mu}_{\lambda,\xi}$ denotes the solution to the smoothing problem (26) given ξ and $\boldsymbol{\lambda}$.

It remains to show how to choose the smoothing parameter $\boldsymbol{\lambda}$. If it is to small or too large, the resulting trajectory is undersmoothed or oversmoothed, respectively. Ideally, one would like to choose the smoothing parameter to minimize the *average squared error*:

$$\mathrm{ASE}(\boldsymbol{\lambda}) = \frac{1}{N}\sum_{j=1}^{N}\left\|\boldsymbol{\mu}_\lambda(t_j) - \boldsymbol{\vartheta}(t_j)\right\|^2. \tag{29}$$

However, this minimization is in practice impossible because the true function $\boldsymbol{\vartheta}$ is unknown. In the scalar case, several methods for choosing $\boldsymbol{\lambda}$ have been suggested. The *cross-validation* and the *generalized cross-validation* scores [20,6] are the most widely used. The vector counterpart of the cross-validation score is given by

$$\mathrm{CV}(\boldsymbol{\lambda}) = \frac{1}{N}\sum_{j=1}^{N}\left\|z_j - \boldsymbol{\mu}_{\lambda(j)}(t_j)\right\|_{\Sigma_j^{-1}}^2, \tag{30}$$

where $\boldsymbol{\mu}_{\lambda(j)}$ denotes the solution to the smoothing problem (26) with the j-th data point, (t_j, z_j), omitted. By minimizing (30) we obtain a curve which is least sensitive to the omission of one of the measurements. It can be shown that (30) can be converted into the following, computationally more appealing form [8]:

$$\mathrm{CV}(\boldsymbol{\lambda}) = \frac{1}{N}\sum_{j=1}^{N}\left\|\left(I_M - H_{(jj)}(\boldsymbol{\lambda})\right)^{-1}\left(z_j - \boldsymbol{\mu}_\lambda(t_j)\right)\right\|_{\Sigma_j^{-1}}^2, \tag{31}$$

where $H_{(jj)}(\boldsymbol{\lambda})$ is the j-th $M \times M$ block diagonal submatrix of the hat matrix

$$H(\boldsymbol{\lambda}) = X(X^{\mathrm{T}}\Sigma^{-1}X + \Omega)^{-1}X^{\mathrm{T}}\Sigma^{-1}. \tag{32}$$

For various technical reasons, it is often desirable to replace the CV-score by the generalized cross-validation score. In [8], the following score has been suggested as the vector version of the generalized cross-validation:

$$\mathrm{GCV}(\boldsymbol{\lambda}) = \frac{1}{N}\sum_{j=1}^{N} \frac{\left\| z_j - \mu_{\lambda}(t_j) \right\|^2_{\Sigma_j^{-1}}}{\left(\frac{1}{N}\mathrm{trace}(I - H(\boldsymbol{\lambda})) \right)^2}. \tag{33}$$

Both scores depend on the central bands of the hat matrix $H(\boldsymbol{\lambda})$. A method for their calculation, which is directly applicable to the vector case, is sketched in the discussion of Silverman's paper [19]. Some results on the performance of both scores in the vector case are given, though in a slightly different setting, in [8].

There exists a theoretically better model for the calculation of the trajectory than Model 1, 2 and 3. Let A_{l}, A_{r}, $V_{\mathrm{l}j}$, $V_{\mathrm{r}j}$, $\boldsymbol{u}_{\mathrm{l}j}$, $\boldsymbol{u}_{\mathrm{r}j}$, $\tilde{\boldsymbol{t}}$ and $\tilde{\boldsymbol{r}}$ have the same meaning as in (7), let $\boldsymbol{k}$ and $\boldsymbol{q}$ have the same meaning as in (22), let

$$\begin{bmatrix} \tilde{\boldsymbol{t}} \\ \tilde{\boldsymbol{r}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{k}_p(\boldsymbol{q}) \\ \boldsymbol{k}_r(\boldsymbol{q}) \end{bmatrix} = \boldsymbol{k}(\boldsymbol{q})$$

and let

$$\boldsymbol{g}_{\mathrm{l}j}(\boldsymbol{q}) = A_{\mathrm{l}}\big(R(\boldsymbol{k}_r(\boldsymbol{q}))\boldsymbol{m}_j + \boldsymbol{k}_p(\boldsymbol{q})\big), \quad \boldsymbol{g}_{\mathrm{r}j}(\boldsymbol{q}) = A_{\mathrm{r}}\big(R(\boldsymbol{k}_r(\boldsymbol{q}))\boldsymbol{m}_j + \boldsymbol{k}_p(\boldsymbol{q})\big)$$

for some appropriate mappings $\boldsymbol{g}_{\mathrm{l}j}$, $\boldsymbol{g}_{\mathrm{r}j}$. We can now set the following nonlinear regression *Model 4*:

$$\boldsymbol{u}_{\mathrm{l}j} = \boldsymbol{g}_{\mathrm{l}j}\big(\boldsymbol{\theta}(t_j)\big) + \boldsymbol{\zeta}_{\mathrm{l}j}, \quad \boldsymbol{u}_{\mathrm{r}j} = \boldsymbol{g}_{\mathrm{r}j}\big(\boldsymbol{\theta}(t_j)\big) + \boldsymbol{\zeta}_{\mathrm{r}j}, \quad j = 1, \ldots, N,$$

where $\boldsymbol{\zeta}_{\mathrm{l}j}$, $\boldsymbol{\zeta}_{\mathrm{r}j}$ are zero mean, mutually independent random vectors with the covariance matrices $\sigma^2 V_{\mathrm{l}j}$ and $\sigma^2 V_{\mathrm{r}j}$, respectively, and $\boldsymbol{\theta}$ is the true trajectory in the joint space. To estimate this trajectory, the following nonlinear criterion must be minimized

$$\frac{1}{N}\sum_{j=1}^{N} \left\| \boldsymbol{u}_{\mathrm{l}j} - \boldsymbol{g}_{\mathrm{l}j}\big(\boldsymbol{\mu}(t_j)\big) \right\|^2_{V_{\mathrm{l}j}^{-1}} + \frac{1}{N}\sum_{j=1}^{N} \left\| \boldsymbol{u}_{\mathrm{r}j} - \boldsymbol{g}_{\mathrm{r}j}\big(\boldsymbol{\mu}(t_j)\big) \right\|^2_{V_{\mathrm{r}j}^{-1}} + \sum_{i=1}^{M} \lambda_i \int_a^b \mu_i^{(m)}(t)^2 \, \mathrm{d}t \tag{34}$$

over all trajectories $\boldsymbol{\mu} \in W_2^m[a, b]$. Any function $\boldsymbol{\mu}_{\lambda}$ achieving minimum of (34) is a natural vector spline. However, unlike the linear case, there may be multiple minima. An iterative algorithm based on the Levenberg–Marquardt method was given [7]. Although the joint trajectory obtained with the help of this model is a theoretically better approximation of the true joint trajectoy than the one obtained with the help of Model 3, we do not recommend its use because it is computationally much more expensive.

5. Simulation results

A simulation experiment was carried out to test the performance of the natural vector splines for the estimation of the trajectory of a moving object. The simulated cameras were modelled as 512×512 pixel resolution imaging surfaces with the focal length $f = 28$ units. The stereo setup had a baseline length $b = 500$ units and both cameras were aligned. The object was modelled as a wire-framed cube with side length 100 units and with its eight vertices being used as the object feature points. The trajectory of the cube was generated by a computer with the course 2000–3000 units away from the cameras. The whole stereo image sequence had 101 frames. The cube's vertices were projected into both image planes at every time instant. Then independent, normally distributed noise was added to the true image coordi-

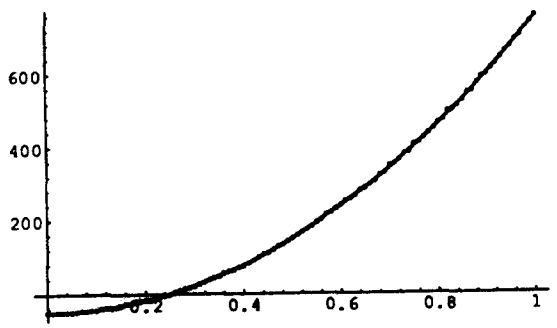

Fig. 2. Translation vector: x-component.

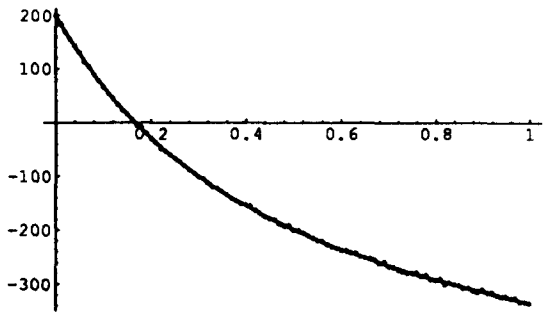

Fig. 3. Translation vector: y-component.

nates to simulate the noisy measurements and the images were digitized. The cube's poses were computed by minimizing either (3) or (7) and their uncertainties were computed as described in Section 3. The sequence of minimizations of (3) resulted in the same situation as in Model 1 whereas the outcome of the sequence of minimizations of (7) was the situation described by Model 2. Given a particular choice of the smoothing parameter λ, an approximation to the true trajectory was calculated by either minimizing (26) twice for the translation and the rotation vector with $M = 3$ when Model 1 was used or by minimizing (26) once for the pose with $M = 6$ when Model 2 was used. The smoothing parameter was determined by minimizing either the ASE, CV or GCV score.

The accuracy of the spline smoothing performed according to Model 1 is illustrated in *Figs. 2* to *7*, where the solid curves represent the output produced by the smoothing procedure while the dots represent the estimated coefficients of the cube's poses. The cross-validation score was used in this case. The noise was effectively smoothed out. The figures show us that the relative error in the estimated translation vector is smaller than its counterpart in the estimated rotation vector, i.e. the estimation of the orientation is more sensitive than the estimation of the position. One could have expected this; the estimate of the translation vector was obtained by direct use of the reconstructed object feature points $\boldsymbol{p}_i$ while the rotation vector was obtained by use of $\boldsymbol{v}_i$ given in (4). The length of $\boldsymbol{p}_i$ is of the same order as the distance of the object to the camera system while the length of $\boldsymbol{v}_i$ is of the same order as the size of the object. Since the distance of the object to the camera system is normally larger than its size, the relative errors in the coordinates of $\boldsymbol{v}_i$ are larger than their counterparts in the coordinates of $\boldsymbol{p}_i$. The error model for the uncertainties in the orientation parameters was developed under the assumption that

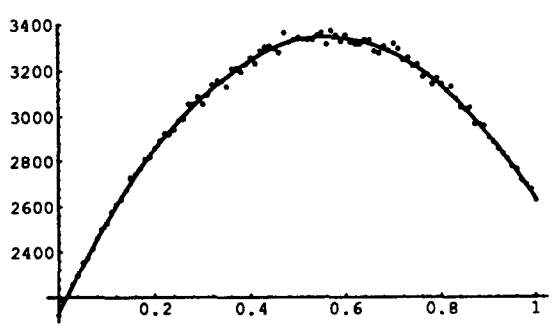

Fig. 4. Translation vector: z-component.

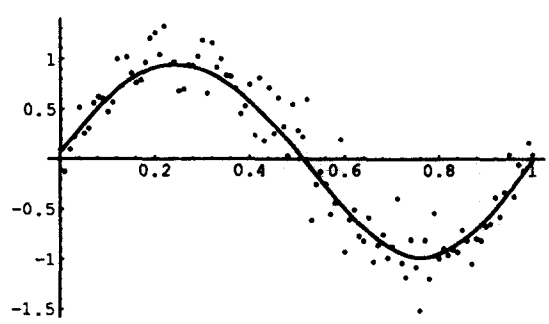

Fig. 5. Rotation vector: x-component.

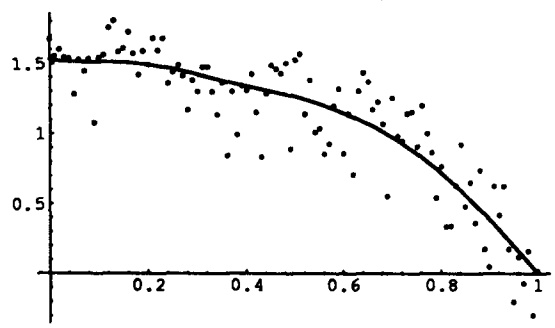

Fig. 6. Rotation vector: y-component.

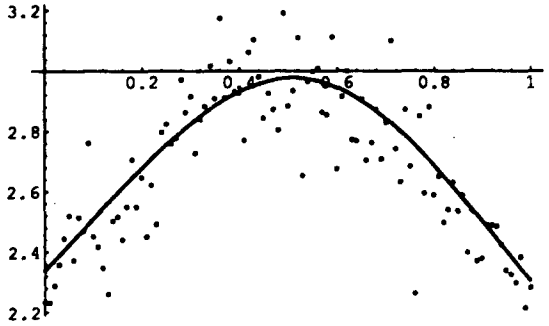

Fig. 7. Rotation vector: z-component.

the true and the estimated rotation matrix can be related by the differential rotation matrix. Therefore it must be used with care; Formula (16) becomes unvalid once the error in the estimated rotation vector crosses a certain limit.

To demonstrate the accuracy of the cross-validation and the generalized cross-validation, we did a Monte Carlo simulation with 200 runs, each with a different measurement noise realization. For each run we approximated the trajectory by minimizing ASE, CV and GCV scores with either the calculated covariance matrices or with the covariance matrices set to the identity matrix. The summary statistics of the experiment is gathered in *Tables 1* and *2*. λ_{ASE}, λ_{CV} and λ_{GCV} denote the estimated optimal smoothing parameter obtained by minimizing ASE, CV and GCV scores, respectively. By comparing the results given in both tables, one can clearly see that the computation of the covariance matrices improves the accuracy of the approximating trajectory. As expected, the trajectory obtained by use of Model 2 is slightly more accurate than the one obtained by use of Model 1, but the difference is not too dramatical. We expect that the difference between both estimates would increase if the distance between the object and the camera increased. On the other hand, the computation of the trajectory by use of Model 1 is more efficient because the dimensionality of the problem is smaller than the dimensionality of the problem when Model 2 is used. In *Table 1* we can see that the GCV estimate slightly outperformed the CV estimate when the translation or the pose trajectory were approximated while they gave essentially

Table 1
Summary statistics for 200 replications of the experiment (with computed covariance matrices).

	$ASE(\lambda_{ASE})$	$ASE(\lambda_{CV})$	$\frac{ASE(\lambda_{CV})}{ASE(\lambda_{ASE})}$	$ASE(\lambda_{GCV})$	$\frac{ASE(\lambda_{GCV})}{ASE(\lambda_{ASE})}$
Model 1 (Trans.) Expectation	1.299035	2.390818	2.081714	2.033360	1.731614
Model 1 (Trans.) *SD*	0.714876	1.115787	1.019384	0.931543	0.757778
Model 1 (Orient.) Expectation	0.000596	0.000785	1.343166	0.000790	1.360374
Model 1 (Orient.) *SD*	0.000229	0.000311	0.343194	0.000307	0.385751
Model 2 (Pose) Expectation	1.328998	2.357847	1.961145	1.927753	1.558913
Model 2 (Pose) *SD*	0.743312	1.121874	0.904297	0.920151	0.597591

Table 2
Summary statistics for 200 replications of the experiment (with covariance matrices set to identity).

	$ASE(\lambda_{ASE})$	$ASE(\lambda_{CV})$	$\frac{ASE(\lambda_{CV})}{ASE(\lambda_{ASE})}$	$ASE(\lambda_{GCV})$	$\frac{ASE(\lambda_{GCV})}{ASE(\lambda_{ASE})}$
Model 1 (Trans.) Expectation	2.426939	2.942644	1.228655	24.096578	11.322139
Model 1 (Trans.) *SD*	0.917074	1.269231	0.321666	4.258267	4.544574
Model 1 (Orient.) Expectation	0.000867	0.001180	1.406090	0.001185	1.436916
Model 1 (Orient.) *SD*	0.000308	0.000463	0.527131	0.000580	0.860098
Model 2 (Pose) Expectation	2.284835	2.857492	1.274271	24.411552	12.344560
Model 2 (Pose) *SD*	0.907756	1.752328	0.556727	3.411459	4.703860

the same results when the orientation trajectory was approximated. Surprisingly, the proposed GCV estimate completely failed when the translation or the pose trajectory were estimated with covariance matrices set to identity (see *Table 2*). This result raises some doubt in the correctness of the vector generalization of the GCV score given in [8]. In the scalar case, the GCV score is obtained from the CV score (31) by replacing the weights $H_{(kk)}(\boldsymbol{\lambda})$ by their average $1/N\sum_{k=1}^{N}H_{(kk)}(\boldsymbol{\lambda})$. The resulting score is:

$$\sum_{j=1}^{N}\left\|\left(I-\frac{1}{N}\sum_{k=1}^{N}H_{(kk)}(\boldsymbol{\lambda})\right)^{-1}\left(\mathbf{z}_j-\boldsymbol{\mu}_{\lambda}(t_j)\right)\right\|_{\Sigma_j^{-1}}^{2}. \tag{35}$$

Clearly, the scores (33) and (35) coincide in the scalar case; but in the vector case, they are different. At the moment we are using the implementation of vector spline smoothing obtained from the GCV directory of NETLIB [8] where the score (33) is used as the GCV score. We are working on our own implementation of the algorithm which will include the minimization of the suggested score (35) as well.

6. Concluding remarks

In this paper we described how a robot's visual system can be used for the reconstruction of trajectories for teaching robot paths. Using the robot's visual system, the human operator can transfer his knowledge about the course of the desired motion to the robot. We demonstrated how uncertainties in the feature point positions can be converted to the uncertainties in the estimated object poses. We extended the theory of smoothing splines to include the case when vector measurements and their uncertainties are available. We showed how a continuous approximation of the true trajectory can be calculated. We gave some simulation results to demonstrate the accuracy of the approach.

Further investigations should be done in order to improve the usefulness of our method. We concentrated in this article on the additive noise model. However, especially when a binocular vision system is used, another type of error frequently occurs: outliers. The second term in Eq. (26) makes the method fairly resistant to this type of noise although robustness cannot be guaranteed. However, for large values of sample size, it is unlikely that the outliers would cause any problems [17]. Another problem is that because of the physical constraints in the robot joints, the robot cannot track every path. To assure the feasibility of the reconstructed trajectory, the limits in joint velocities and accelerations must be considered. This means in the language of smoothing splines that the suggested method needs to be extended to include the case when inequality constraints are imposed on the derivatives of the trajectory. At the moment we are working on the implementation of the algorithm which will contain this

extension. We shall present our results in the future. We are also developing other modules of the system described in the introduction which will enable us to carry out some experiments with real data.

Acknowledgement

This research has been performed at the Institute for Real-Time Computer Systems and Robotics, University of Karlsruhe, Germany. It was supported in part by the International Bureau Jülich in the frame of the cooperation between the Institute for Real-Time Computer Systems and Robotics and Jožef Stefan Institute, University of Ljubljana, Slovenia. The author is grateful to Professor Rüdiger Dillmann for his support during the research. He also thanks the anonymous reviewer for his comments.

References

[1] K.S. Arun, T.S. Huang and S.D. Blostein, Least-squares fitting of two 3-D point sets, *IEEE Trans. Pattern Anal. Machine Intell.*, PAMI 9 (5) (1987) 698–700.

[2] N. Ayache, *Artificial Vision for Mobile Robots: Stereo Vision and Multisensory Perception*, Artificial Intelligence (MIT Press, Cambridge, MA, 1991).

[3] S.D. Blostein and T.S. Huang, Error analysis in stereo determination of 3-D point positions, *IEEE Trans. Pattern Anal. Machine Intell.*, PAMI 9 (6) (1987) 752–765.

[4] J.J. Craig, *Introduction to Robotics; Mechanics & Control* (Addison-Wesley, Reading, MA, 1989).

[5] C. de Boor, *A Practical Guide to Splines* (Springer-Verlag, New York, 1978).

[6] R.L. Eubank, *Spline Smoothing and Nonparametric Regression* (Marcel Dekker, New York, 1988).

[7] J.A. Fessler, Nonparametric fixed-interval smoothing of nonlinear vector-valued measurements, *IEEE Trans. Signal Processing* 39 (4) (1991) 907–913.

[8] J.A. Fessler, Nonparametric fixed-interval smoothing with vector splines, *IEEE Trans. Signal Processing* 39 (4) (1991) 852–859.

[9] H. Goldstein, *Classical Mechanics* (Addison-Wesley, Reading, MA, 1980).

[10] T.S. Huang, S.D. Blostein and E.A. Margerum, Least-squares estimation of motion parameters from 3-D point correspondences, *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Miami Beach, FL (June 1986) 198–201.

[11] M. Ishii, S. Sakane, M. Kakikura and Y. Mikami, A 3-D sensor system for teaching robot paths and environments, *Int. J. Robotics Res.* 6 (2) (1987) 45–59.

[12] Y. Kuniyoshi, M. Inaba and H. Inoue, Seeing, understanding and doing human task, *Proc. IEEE Int. Conf. Robotics and Automation*, Nice (May 1992) 2–9.

[13] S. Lee and Y. Kay, A Kalman filter approach for accurate 3-D motion estimation from a sequence of stereo images, *CVGIP: Image Understanding* 54 (2) (1991) 244–258.

[14] T. Lyche and L.L. Schumaker, Computation of smoothing and interpolating natural splines via local bases, *SIAM J. Numer. Anal.* 10 (6) (1973) 1027–1036.

[15] L. Matthies and S.A. Shafer, Error modeling in stereo navigation, *IEEE J. Robotics Automat.* RA 3 (3) (1987) 239–248.

[16] D. Nitzan, Three-dimensional vision structure for robot applications, *IEEE Trans. Pattern Anal. Machine Intell.* 10 (3) (1988) 291–309.

[17] T. Robinson and R. Moyeed, Making robust the cross-validatory choice of smoothing parameter in spline smoothing regression, *Commun. Statist.-Theory Meth.* 18 (2) (1989) 523–539.

[18] R.J. Schalkoff, *Digital Image Processing and Computer Vision* (Wiley, New York, 1989).

[19] B.W. Silverman, Some aspects of the spline smoothing approach to non-parametric regression curve fitting, *J. R. Statist. Soc., ser. B* 47 (1) (1985) 1–52.

[20] G. Wahba, *Spline Models for Observational Data* (SIAM, Philadelphia, 1990).

[21] J. Weng, T.S. Huang and N. Ahuja, 3-D motion estimation, understanding, and prediction from noisy image sequences, *IEEE Trans. Pattern Anal. Machine Intell.*, PAMI 9 (3) (1987) 370–389.

[22] G.-S.J. Young and R. Chellappa, 3-D motion estimation using a sequence of noisy stereo images: Models, estimation, and uniqueness results, *IEEE Trans. Pattern Anal. Machine Intell.* 12 (8) (1990) 735–759.

[23] Z. Zhang and O.D. Faugeras, Determining motion from 3D line segment matches: A comparative study, *Image and Vision Computing* 9 (1) (1991) 10–19.