We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

186,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

25

Spatial Vision-Based Control of High-Speed Robot Arms

Friedrich Lange and Gerd Hirzinger

1. Introduction

Industrial robots are known to execute given programs at high speed and at the same time with high repeatability. From non industrial scenarios as in (Nakabo et al., 2005) or (Ginhoux et al., 2004) we know that cameras can be used to execute high-speed motion even in those cases in which the desired path is not a priori given but online sensed. In general, such visual tracking tasks of following a randomly moving target by a camera are not accurate enough for industrial applications. But there the work-piece will scarcely move randomly. So in this chapter we concentrate on another type of visual servoing, in which the path that has to be followed is fixed in advance, but not given by a robot program.

A robot-mounted camera is a universal sensor which is able to sense poses with high accuracy of typically less than a millimeter in the close-up range. At the same time with high shutter speed the robot may move fast, e.g. at 1 m/s. In contrast to expensive high-speed cameras, yielding a high frame rate of e.g. 1 kHz as in (Nakabo et al., 2005), we restrict on a standard CCIR camera, to meet the requirements of a cost-effective hardware. Off-the-shelf cameras are fundamental for the acceptance in industry. So an important feature of our method is an appropriate control architecture that tolerates low sampling rates of sensors. The camera is mounted at the robot arm and measures the work-piece pose (given by boundary lines) with respect to the tool center point (tcp) of the robot. More precisely, the camera is mounted laterally to provide enough space for a tool. So with constant moving sense we have a predictive sensor as in (Meta-Scout, 2006). With alternating moving sense the camera has to be tilted so that the corresponding nominal line point $\mathbf{p}_n$ comes approximately to the center of the image (see Fig. 1d). In this case segments of the lines might be occluded by the tool. A tilted mounting yields a priori unknown distances of the different line points and thus complicates the equations. As well, the task frame defined by the tcp, and the camera frame are different.

In contrast to current research (Comport et al., 2005), we assume that problems with image processing, feature detection and projections are solved, which

Open Access Database www.i-techonline.com

Source: Industrial-Robotics-Theory-Modelling-Control, ISBN 3-86611-285-8, pp. 964, ARS/plV, Germany, December 2006, Edited by: Sam Cubero

holds for our simple black and white scenario. In addition, the initial configuration is supposed to be close to the target configuration, so that large rotations as in (Tahri and Chaumette, 2005) do not appear.

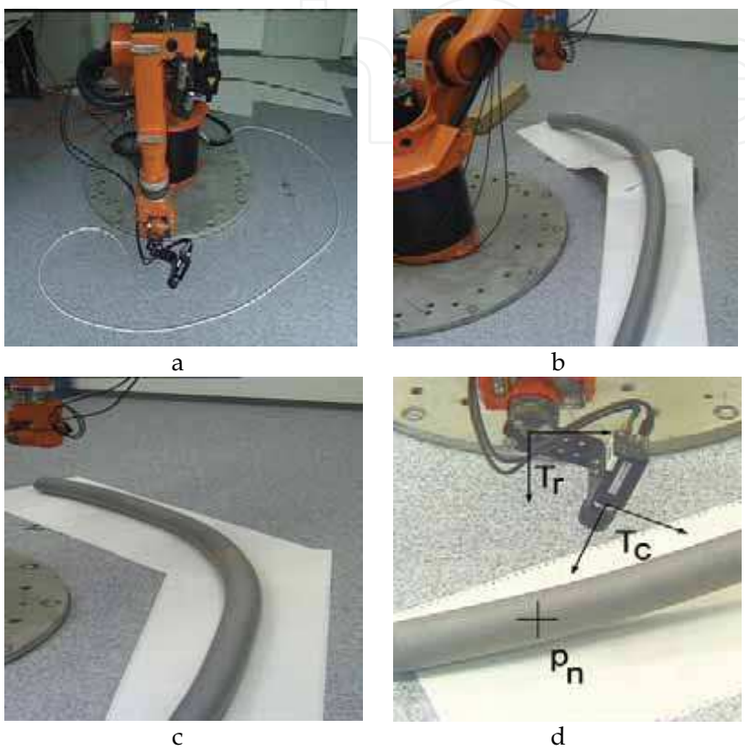


Figure 1. Sample tasks with natural and artificial background and tilted mounted camera

We show several planar and spatial sample tasks (Fig. 1). They are used to sense (curved) lines or edges which are coarsely parallel to the desired motion. This allows to *refine* a coarsely programmed path at full speed.
According to the notation of this chapter the programmed path is called *reference path* and the corresponding lines of the nominal scenario are *nominal lines*. In reality the *sensed lines* will differ from the nominal lines, thus defining the *desired path*, where a path is given by trajectories of positions and orientations.
In our experiments we program a horizontal circular path which is online modified in radial and in vertical direction using image data. A possible application is the spraying of glue to flexible or imprecisely positioned work-pieces. Other applications are laser cutting, or soldering. In these cases the desired path is determined during motion, e.g. by surveying of the boundary lines of the work-piece. For all these tasks high accuracy at high speed is required.

Similar approaches have been proposed predominantly with simple scenarios, see e.g. (Meta-Scout, 2006) or (Lange & Hirzinger, 2002). Other methods handle with multiple lines with point-by-point given nominal location, thus allowing both translational and rotational sensor induced path corrections. However they require complex computations as (Lange & Hirzinger, 2003), or work at lower speed as (Gangloff & de Mathelin, 2002) or (Rives & Borrelly, 2000). The complexity comes from rotations between nominal and real world which in practice are not significant. Therefore we now present a method which denies time consuming iterations to solve systems of trigonometric equations.
The chapter is organized as follows: In section 2 we present a control concept that allows different sampling rates for the robot controller and the sensor. This leads to a universal position controller (section 3) and a task-dependent computation of the desired path (section 4). If the sensible lines yield a non continuous or even a non contiguous path, a smoothing method is required to compute an executable desired path. This is explicated in section 5. Experimental results of continuous paths and from the tracking of lines with vertices are then demonstrated in section 6.

2. Control concept

Instead of directly using sensor data to control the robot, we use a predictive architecture which separates position control from the sensing of the desired path (see Fig. 2). Position control provides what we call a Cartesian ideal robot. This means that for all sampling steps its arm position $\mathbf{x}_a$ is identical to the desired pose $\mathbf{x}_d$. The realization of such an ideal robot will be explained in the sequel. Sensor data affect the system by a module that computes the desired poses at the sampling steps. This is presented in section 4.

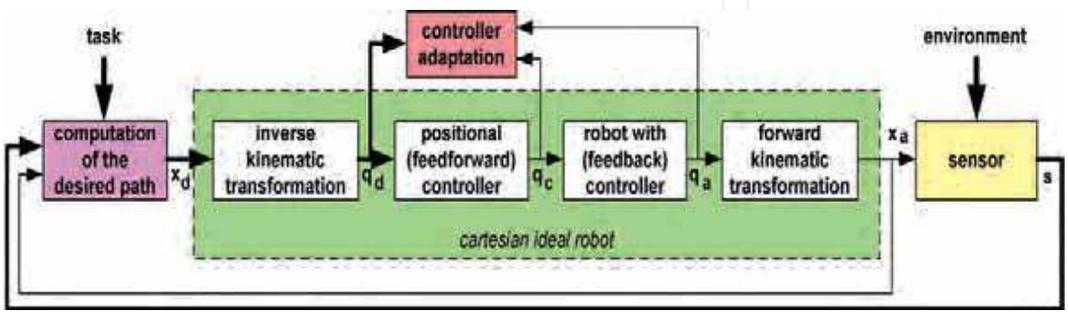


Figure 2. Architecture of control

The fusion of the two parts, position control and the computation of the desired path, yields control from image data.
This concept allows different sampling rates of the camera and the robot controller. Our approach is to integrate image data in each case in the next positional sampling step. As in (Zhang et al., 2003), time delays owing to processor load are tolerable as long as the instant of exposure is known.

3. Position control

The core of this chapter is not restricted to a special position control method. A standard industrial position controller will do if the industrial interface allows online modified references. Such an interface, usually called sensor interface, is required since the desired positions $\mathbf{x}_d$ are not assumed to be available long before the motion.
A standard sensor interface as (Pertin & Bonnet des Tuves, 2004) allows to send online desired values and to receive present positional values. For the purpose of this chapter we refer to the desired values of the sensor interface as a *command* vector $\mathbf{q}_c$, to distinguish between *desired positions* $\mathbf{q}_d$ (of the interface of the ideal robot) and the input to the industrial controller. The actual joint values are called *arm position* $\mathbf{q}_a$.
For curved paths to be executed at high speed we recommend to increase accuracy using advanced control methods which process not only the current desired pose but also the desired speed and acceleration or - what we use - the desired poses of multiple future sampling steps as (Clarke et al., 1987) or (Grotjahn & Heimann, 2002).

Fig. 2 shows an adaptive feed-forward controller as in (Lange & Hirzinger, 1996) as an add-on to the standard industrial feedback controller. A possible controller equation could be

$$\mathbf{q}_c(k) = \mathbf{q}_d(k) + \sum_{i=0}^{n_d} \mathbf{K}_{qi} \cdot (\mathbf{q}_d(k+i) - \mathbf{q}_d(k)) \tag{1}$$

where $\mathbf{K}_{qi}$ stands for the adapted parameters. This controller can be seen as a filter of the desired positions. This proposed controller works in joint space since there are substantial couplings when considering robot dynamics in Cartesian space. In joint space the estimation of couplings can be restricted to some additional adaptive parameters.
This feed-forward controller uses the special predictive interface from (Lange & Hirzinger, 1996), defined either in joint space or in Cartesian coordinates: In

every sampling step the desired poses of the next n_d sampling steps are required, with n_d corresponding to twice the time constant of the controlled robot. This is a key aspect for the realization of an ideal robot.
Bold face lines in Fig. 2 represent data for current and future time steps, i.e. predictions of sensor data are required.
In the case of time-invariant environments, using predictions of the progression of the desired path enables the position controller to compensate all dynamical delays that otherwise would directly affect performance. The uncertainty becomes small enough so that the robot can track a continuous line with very little path error.

4. Computation of the desired path

The task of this computation is to determine the desired poses of next n_d sampling steps. It is advantageous to use a reference path with respect to which nominal lines are defined. Then, the task is to find the path which lies with respect to the real (sensed) lines, in the same way as the reference path to the nominal lines. This path is called *desired path* and is to be transferred to the ideal robot.
Thus with a camera as sensor and the particular task of tracking lines, prediction of the next n_d sampling steps of the desired path reduces to measuring the course of the contour and to keep the values that correspond to future time steps.
For each time-step we compute the transformation matrix ${}^{r}\mathbf{T}_{d}$ which describes the frame of a point of the *desired path* $\mathbf{T}_{d}$ with respect to the frame of the corresponding point of the *reference path* $\mathbf{T}_{r}$. The so called *sensor correction* ${}^{r}\mathbf{T}_{d}$ is computed from the postulation that the *desired path* is defined by the actually *sensed line* $\mathbf{p}_{s}$ in the same way as the *reference path* is given by the *nominal line* points $\mathbf{p}_{n}$. $\mathbf{p}$ is a point vector and, as $\mathbf{T}$, expressed in homogeneous coordinates. All these variables depend on time. The time index k is omitted, however, in order to simplify the notation.

This gives the fundamental equation

$$ {}^{d}\mathbf{p}_{s} = {}^{r}\mathbf{p}_{n} \tag{2} $$

and then

$$ {}^{r}\mathbf{p}_{s} = {}^{r}\mathbf{T}_{d} \cdot {}^{d}\mathbf{p}_{s} = {}^{r}\mathbf{T}_{d} \cdot {}^{r}\mathbf{p}_{n}. \tag{3} $$

Neglecting rotational sensor corrections yields

$$ {}^{r}\mathbf{p}_s = {}^{r}\mathbf{p}_d + {}^{r}\mathbf{p}_n . \tag{4} $$

If we have at least two lines and their nominal line positions ${}^{r}\mathbf{p}_{n_i}$ we can compute the components of ${}^{r}\mathbf{p}_d$, namely ${}^{r}x_d$ and ${}^{r}z_d$ if y is understood as the direction of motion. ${}^{r}y_d$ is defined to be zero since we evaluate lines points which are in the x-z-plane of the reference system. Because of the neglected rotational sensor corrections this means also

$$ {}^{r}y_s = 0 = {}^{d}y_s . \tag{5} $$

In the image the lines are detected by single points. Here we assume that the camera is oriented such that lines are approximately vertical in the image (see Fig. 3). We use a simple edge detection algorithm which evaluates a single image row. Horizontal line searching gives the best accuracy since it allows processing of single image fields. The detected line points are corrected by a rectification algorithm using previously identified camera and lens distortion parameters.

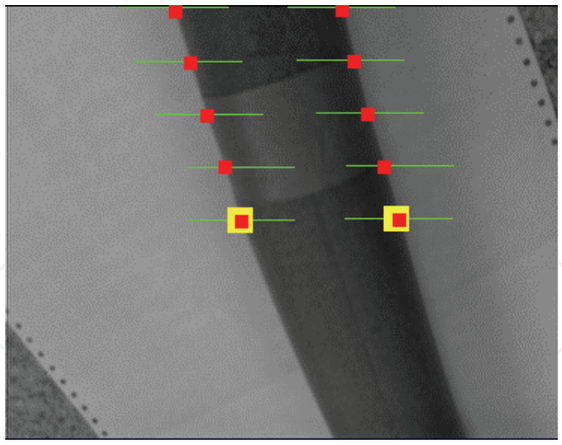


Figure 3. View from the robot mounted camera with 2 lines and 5 sensed points each. The desired positions of the lines[1] in the image are marked with big yellow blocks. The windows for line search are shown by horizontal lines.

[1] The desired position of a line in the image is the image point of the current nominal line point when the tcp pose of the exposure is the reference pose. If these desired line positions coincide in the image with the sensed lines, the actual pose complies with the desired path.

At least for curved paths the sensed line points will not share image row with the corresponding nominal image points. So horizontal search starting at the expected image position of a time step will give image points corresponding to the reference of different time instants. Therefore we represent the lines in the image by polynomials. All future computations are then based on these *line polynomials*

$$\xi = f(\eta) \tag{6}$$

with ξ und η as horizontal and vertical normalized image coordinates of a line point.

Using the projection equation

$${}^{c}\mathbf{p}_s = \begin{pmatrix} \xi \cdot {}^{c}z_s & \eta \cdot {}^{c}z_s & {}^{c}z_s & 1 \end{pmatrix}^T, \tag{7}$$

where ${}^{c}z_s$ is the distance between the camera and the line point, we get the pose vector ${}^{r}\mathbf{p}_s$ from the reference pose of the tcp to a sensed line point

$${}^{r}\mathbf{p}_s = \begin{pmatrix} {}^{r}x_s \\ {}^{r}y_s \\ {}^{r}z_s \\ 1 \end{pmatrix} = {}^{r}\mathbf{T}_c \begin{pmatrix} \xi \cdot {}^{c}z_s \\ \eta \cdot {}^{c}z_s \\ {}^{c}z_s \\ 1 \end{pmatrix}. \tag{8}$$

With equation (5) we get

$$0 = {}^{r}y_s = \begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix} \cdot {}^{r}\mathbf{T}_c \cdot {}^{c}\mathbf{p}_s = ({}^{r}\mathbf{T}_{c10} \cdot \xi + {}^{r}\mathbf{T}_{c11} \cdot \eta + {}^{r}\mathbf{T}_{c12}) \cdot {}^{c}z_s + {}^{r}\mathbf{T}_{c13}, \tag{9}$$

where for example ${}^{r}\mathbf{T}_{c10}$ is the component (1,0) of the transformation matrix ${}^{r}\mathbf{T}_c$. This transformation matrix expresses the actual pose of the camera with respect to the reference pose of the tcp. At least for curved reference paths or a camera which is tilted with respect to the tool or the lines (see Fig. 1d) this transformation differs from identity.

The equations for ${}^{r}x_s$ and ${}^{r}z_s$ are a little bit more complicated to evaluate. With $\mathbf{T}_r$ and $\mathbf{T}_d$ having the same orientation, using equation (4) and (2) we can write

$${}^{r}x_s = {}^{r}x_d + {}^{d}x_s = {}^{r}x_d + {}^{r}x_n, \tag{10}$$

where ${}^{r}x_n$ is the nominal distance of the line with respect to the reference trajectory of the tcp. This distance is given, e.g. as the space between the desired cutting edge and a visible edge on the work-piece. ${}^{r}x_d$ is the wanted path correction which is the same for different lines:

$${}^{r}x_d = ({}^{r}\mathbf{T}_{c00} \cdot \xi + {}^{r}\mathbf{T}_{c01} \cdot \eta + {}^{r}\mathbf{T}_{c02}) \cdot {}^{c}z_s + {}^{r}\mathbf{T}_{c03} - {}^{r}x_n. \tag{11}$$

By comparing ${}^{r}x_d$ of two lines we get

$$\begin{aligned} &({}^{r}\mathbf{T}_{c00} \cdot \xi_0 + {}^{r}\mathbf{T}_{c01} \cdot \eta_0 + {}^{r}\mathbf{T}_{c02}) \cdot {}^{c}z_{s_0} + {}^{r}\mathbf{T}_{c03} - {}^{r}x_{n_0} \\ &\qquad = \\ &({}^{r}\mathbf{T}_{c00} \cdot \xi_1 + {}^{r}\mathbf{T}_{c01} \cdot \eta_1 + {}^{r}\mathbf{T}_{c02}) \cdot {}^{c}z_{s_1} + {}^{r}\mathbf{T}_{c03} - {}^{r}x_{n_1}. \end{aligned} \tag{12}$$

Likewise we compute

$${}^{r}z_d = ({}^{r}\mathbf{T}_{c20} \cdot \xi + {}^{r}\mathbf{T}_{c21} \cdot \eta + {}^{r}\mathbf{T}_{c22}) \cdot {}^{c}z_s + {}^{r}\mathbf{T}_{c23} - {}^{r}z_n \tag{13}$$

and thus

$$\begin{aligned} &({}^{r}\mathbf{T}_{c20} \cdot \xi_0 + {}^{r}\mathbf{T}_{c21} \cdot \eta_0 + {}^{r}\mathbf{T}_{c22}) \cdot {}^{c}z_{s_0} + {}^{r}\mathbf{T}_{c23} - {}^{r}z_{n_0} \\ &\qquad = \\ &({}^{r}\mathbf{T}_{c20} \cdot \xi_1 + {}^{r}\mathbf{T}_{c21} \cdot \eta_1 + {}^{r}\mathbf{T}_{c22}) \cdot {}^{c}z_{s_1} + {}^{r}\mathbf{T}_{c23} - {}^{r}z_{n_1}. \end{aligned} \tag{14}$$

where ${}^{r}z_{n_0}$ and ${}^{r}z_{n_1}$ are the nominal distances to the lines, as e.g. the distance between the laser and the work-piece. Usually the distances to the two lines are the same.

In the case of two lines, with two equations (9), equation (12), equation (14), and two equations (6) we have a total of six equations to determine $\xi_0, \xi_1, \eta_0, \eta_1, {}^{c}z_{s_0}$, and ${}^{c}z_{s_1}$. Because of the nonlinearity, equation (6), a numerical solution is required which usually converges within 2 iterations. The wanted components ${}^{r}x_d$ and ${}^{r}z_d$ of the path correction are calculated by inserting the computed variables into the equations (11) and (13), using any of the lines.

If only one line is visible, we need a priori information, e.g. the distance of the line or, more precisely, the plane in which the line lies. If this plane is parallel to the x-y-plane of the tool frame we can use ${}^{r}z_d = 0$. In this case the system of equations is limited to equations (6), (9), and (13) to determine ξ, η, and ${}^{c}z_s$ which are inserted into equation (11).

Strictly speaking we use a priori information as well with two lines. It is the assumption that the plane of the lines is parallel to the x-y-plane of the tool. A varying distance can be interpreted as a non zero roll angle. So the specification is restricted to the pitch angle of the plane, i.e. the rotation around the y-axis of the reference frame of the tcp. The computation of this value needs at least three lines, see (Gangloff and de Mathelin, 2002).

For every capture we compute path corrections for multiple reference poses $\mathbf{T}_r$ but fixed camera pose $\mathbf{T}_c$ which is the pose at the time instant of the exposure. To avoid the computation for all n_d sampling steps (see section 3), we represent the path corrections also as polynomials, using again parameter estimation methods. The resulting polynomials allow to readout the wanted path modifications with minimal effort. Therefore the method is still suitable for $n_d \approx 20$. With an appropriate feed-forward control it yields minimal path errors.

The indirect computation of the desired poses by using path modifications with respect to a reference path is advantageous since curved paths with varying orientation are allowed. Solely the path correction itself is assumed to be done without rotations.

5. Computation of a smooth desired path

A problem may occur with the presented method if a line has to be followed that is not as continuous as the lines of Fig. 1. With the path of Fig. 4 the robot tries to execute a velocity step at the vertices between straight edges without considering acceleration limitations.

To prevent this case, we apply impedance-based control. A filter smoothes the sensed edge so that the resulting contour can be tracked.

Figure 4. Sample task with vertices within the line

Regarding Fig. 5 this means that instead of the sensed edge (red points) a smoothed edge (blue points) is used to compute the desired path for the inner position control loop. In force control scenarios such a filter is well known and is called impedance filter. We adopt this expression although in this case sensor data do not represent forces.

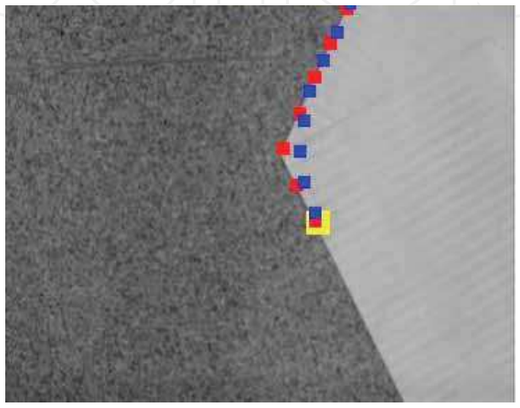


Figure 5. View from the robot-mounted camera. Sensed edge points (red) and the computed smoothed path (blue) are depicted. The yellow block visualizes the current desired pose.

It is worth noting that this kind of impedance-based control does not affect the control law but only the desired trajectory. The stiff position control is maintained. This is useful if different compliances are desired in the individual components of the Cartesian pose vector. A typical example is a horizontal motion where the height of the end-effector with respect to the floor has to be accurately kept while the horizontal components are sensor controlled and therefore have to be compliant with respect to vertices of the target line.

5.1 Impedance-based control

For reasons of clarity we now restrict on Cartesian positions $\mathbf{x}$ instead of positions and orientations as in section 4. So vectors are used instead of matrices of homogeneous coordinates, and therefore the orientation is not affected by the impedance-based control.
It is assumed further that there is a reference trajectory, the originally programmed motion $\mathbf{x}_r(k)$. With respect to this motion the impedance law

$$\mathbf{E} \cdot {}^{r}\mathbf{x}_d + \mathbf{D} \cdot {}^{r}\dot{\mathbf{x}}_d + \mathbf{M} \cdot {}^{r}\ddot{\mathbf{x}}_d = {}^{r}\mathbf{x}_a + {}^{a}\mathbf{s} - \mathbf{s}_r = \Delta\mathbf{s}_r \tag{15}$$

defines the desired trajectory ${}^{r}\mathbf{x}_d$ where $\mathbf{x}_a$ is the actual pose of the tcp. ${}^{a}\mathbf{s}$ are the sensed edge data, expressed with respect to the tcp. We here assume that the transformation from the camera system to the tcp system is known. $\mathbf{s}_r$ represents a reference sensor value, i.e. a specified distance between tcp and edge. Fig. 6 demonstrates this setup.

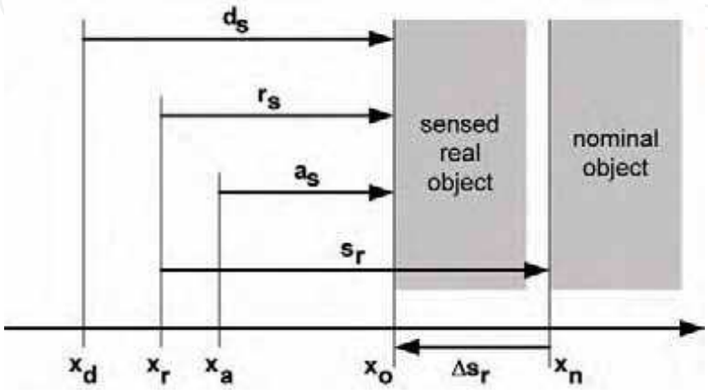


Figure 6. Robot tool positions and sensed distances to a fixed object

Note that within this section the index $*_d$ represents filtered desired values which are usually different from ${}^{r}\mathbf{T}_d$ or ${}^{r}\mathbf{p}_d$ in equations like (3) or (4). The latter will now be denoted by $\mathbf{x}_a + {}^{a}\mathbf{s} - \mathbf{s}_r$ where $\mathbf{x}_a + {}^{a}\mathbf{s}$ corresponds to an object pose $\mathbf{T}_o$ and $-\mathbf{s}_r$ to ${}^{o}\mathbf{T}_d$.

With $\mathbf{E} = \mathbf{I}$ and $\mathbf{D} = \mathbf{M} = \mathbf{0}$, (15) represents explicit sensor-based control as in section 4. If, in contrast, we specify $\mathbf{M} > \mathbf{0}$, the resulting trajectory will smooth the corners since then the accelerations are weighed in addition to the deviations from the trajectory of explicit sensor-based control. With $\mathbf{M} > \mathbf{0}$ we further specify $\mathbf{D} > \mathbf{0}$ in order to avoid oscillations.

With ${}^{r}\dot{\mathbf{x}}_d(k) = ({}^{r}\mathbf{x}_d(k+1) - {}^{r}\mathbf{x}_d(k-1))/2T_0$

and ${}^{r}\ddot{\mathbf{x}}_d(k) = ({}^{r}\mathbf{x}_d(k+1) - 2\,{}^{r}\mathbf{x}_d(k) + {}^{r}\mathbf{x}_d(k-1))/T_0^2$

we compute ${}^{r}\mathbf{x}_d(k+i)$ from images taken at the time step k by

$$\mathbf{E}\cdot{}^{r}\mathbf{x}_d(k+i) + \frac{\mathbf{D}}{2T_0}\cdot\left({}^{r}\mathbf{x}_d(k+i+1) - {}^{r}\mathbf{x}_d(k+i-1)\right) \tag{16}$$

$$+\frac{\mathbf{M}}{T_0^2}\cdot\left({}^{r}\mathbf{x}_d(k+i+1) - 2\,{}^{r}\mathbf{x}_d(k+i) + {}^{r}\mathbf{x}_d(k+i-1)\right) = {}^{r}\mathbf{x}_a(k) + {}^{a}\mathbf{s}(k+i/k) - \mathbf{s}_r.$$

${}^{a}\mathbf{s}(k+i/k)$ reflects the sensed edge position for time step $(k+i)$, sensed in the image at time step k. Equation (16) gives a system of equations for ${}^{y}x_d(k+i)$ with given k. So the solution computes the desired trajectory from the actual position and the sensor values at time step k.

The result can be seen considering a vertex or a discontinuity between ${}^{a}\mathbf{s}(k+i/k)$ and ${}^{a}\mathbf{s}(k+i+1/k)$. With $\mathbf{D}=\mathbf{M}=\mathbf{0}$ a vertex or a discontinuity of ${}^{r}\mathbf{x}_d(k+i)$ would be reproduced. In contrast, $\mathbf{M}>\mathbf{0}$ will smooth the trajectory. Smoothing is not restricted by causality. Instead, the desired trajectory is affected far before the discontinuity of ${}^{a}\mathbf{s}(k+i/k)$. It is still affected after the discontinuity, as well.

Equation (15) defines the compliance between the reference trajectory and the new desired position. The same compliance is valid for the more intuitive relation between the sensed edge and the desired system since can be derived in the same way as (15). The right hand side expression is independent of the resulting desired robot motion $\mathbf{x}_d$

$$\mathbf{E}\cdot{}^{d}\mathbf{x}_o+\mathbf{D}\cdot{}^{d}\dot{\mathbf{x}}_o+\mathbf{M}\cdot{}^{d}\ddot{\mathbf{x}}_o=\mathbf{s}_r+(\mathbf{E}-\mathbf{I})\cdot{}^{r}\mathbf{x}_o+\mathbf{D}\cdot{}^{r}\dot{\mathbf{x}}_o+\mathbf{M}\cdot{}^{r}\ddot{\mathbf{x}}_o \tag{17}$$

.

5.2 Modified approach

Experiments with impedance-based control according to Section 5.1 show a time delay between the sensed and the optimized trajectory. This comes from the fact that a discontinuity of the sensed edge does not cause a position error in relation to an acceleration dependent expression and a velocity dependent expression but the sum of all three terms is minimized. So for example a negative acceleration may compensate a positive position error.

Therefore we define three independent equations that have to be minimized altogether for all time steps.

$$\mathbf{E}\cdot{}^{r}\mathbf{x}_d(k+i)={}^{r}\mathbf{x}_a(k)+{}^{a}\mathbf{s}(k+i/k)-\mathbf{s}_r \tag{18}$$

$$\frac{\mathbf{D}}{2T_0}\cdot\left({}^{r}\mathbf{x}_d(k+i+1)-{}^{r}\mathbf{x}_d(k+i-1)\right)=\mathbf{0} \tag{19}$$

$$\frac{\mathbf{M}}{T_0^2}\cdot\left({}^{r}\mathbf{x}_d(k+i+1)-2\,{}^{r}\mathbf{x}_d(k+i)+{}^{r}\mathbf{x}_d(k+i-1)\right)=\mathbf{0} \tag{20}$$

Minimization is done in a least squares sense where $\mathbf{E}$, $\mathbf{D}$, and $\mathbf{M}$ are the weighting factors.

5.3 Implementation

Instead of solving (16) or minimizing the mean errors of (18) to (20) in every time step, a filter can be computed since we have a linear system of equations. This filter gives ${}^{r}\mathbf{x}_d(l)$ with $k \le l \le k + n_i$ from the values of ${}^{r}\mathbf{x}_a(k) + {}^{a}\mathbf{s}(k+i/k)$ with $0 \le i \le n_i$. This filter is called impedance filter. Its coefficients are found by a single optimization process in the systems (16) or (18) to (20) respectively.
We now have to specify the number n_i of elements of this filter. To be able to compute n_d time steps of ${}^{r}\mathbf{x}_d(l)$ we need $n_i \gg n_d$. In practice however, n_i is usually limited by the visible range of the edge. n_i has been chosen sufficiently large if ${}^{r}\mathbf{x}_d(l)$ proves to be time invariant. This requires that the initial conditions ${}^{r}\mathbf{x}_d(l)$ with $l < k$ have been computed in the same way. The end conditions ${}^{r}\mathbf{x}_d(l)$ with $l > k + n_i$ are set to ${}^{r}\mathbf{x}_d(l) = \Delta\mathbf{s}_r(l)$.

6. Experiments

As first experiments we consider a bent tube that has to be followed at a speed of 0.7 m/s by a KUKA KR6/1 robot (see Fig. 1b or 1c) using the industrial controller KRC1. Image processing, i.e. detection of the boundary lines of the tube, runs in field mode of the camera at a rate of 50 Hz. This is asynchronous to the control rate of 83 Hz.
Previously identified camera reconstruction errors are compensated. E.g. lens distortion is identified according to (CalLab, 1999) and the sensed line points are rearranged accordingly. In addition, some parameters as the constant time shift between the time instant of the exposure and the reception of the image in the computing module are estimated and incorporated thereafter, see (Lange and Hirzinger, 2005) for details.
All tasks for robot control and image processing run in parallel on a 400 MHz processor of the industrial controller KRC1. Control uses the operating system VxWorks and vision runs under Windows95 which is one of the VxWorks tasks. Thus the additionally required hardware is limited to a camera and a standard frame grabber.
Table 1 displays the reached accuracy.

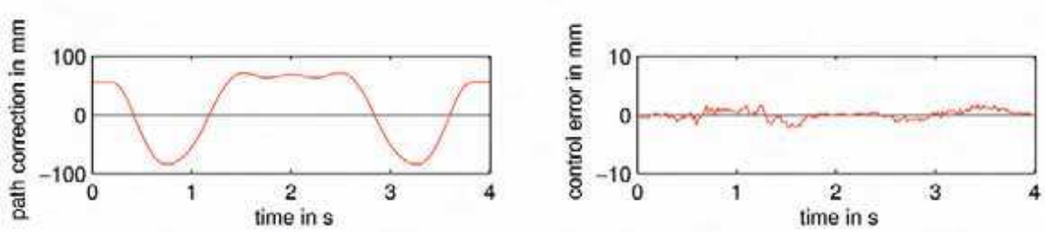
Experiment with	**1 line**	**2 lines**
Mean pixel error	1.2 pixel	1.4 pixel
Maximum pixel error	2.9 pixel	4.4 pixel
Mean path error (horizontal, vertical)	0.3 mm	0.3 mm 0.9 mm
Maximum path error (horizontal, vertical)	1.0 mm	1.0 mm 2.4 mm
Mean deviation from reference path	51 mm	37 mm 65 mm
Maximum deviation from reference path	77 mm	78 mm 98 mm

Table 1. Path error when following a bent tube with 0.7 m/s evaluating in the horizontal plane (1 boundary line) or in space (2 boundary lines) respectively

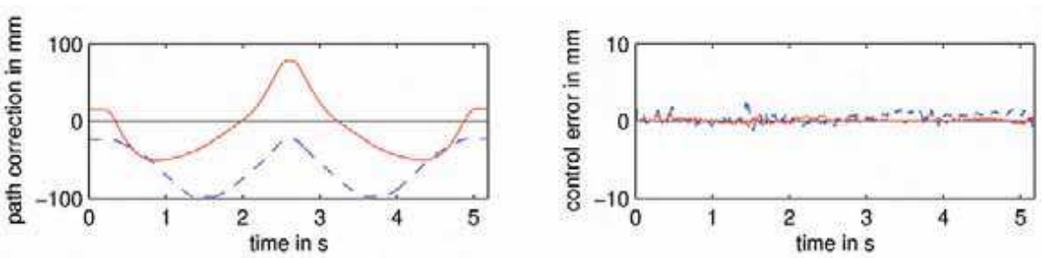
When tracking only one boundary line (Fig. 1c), the tube lies horizontally on the floor. In this case, in spite of big differences between nominal and actual tube, a mean pixel error of less than 1 pixel is reached even without filtering of the edge data.

With a non planar layout (Fig. 1b) we evaluate both boundary lines to determine a spatial path. The vertical accuracy is inferior because on the one hand the measuring of the distance is ill-conditioned due to the geometrical setup, and on the other hand because elastic oscillations of the robot are excited. Compliance in the robot joints prevents accurate measurements of the camera pose, which influences the detected line poses. Nevertheless the mean control error in both degrees of freedom (dof) falls below one millimeter (Fig. 7b and Fig. 7c). The reached path accuracy equals thus the complex method of (Lange & Hirzinger, 2003).

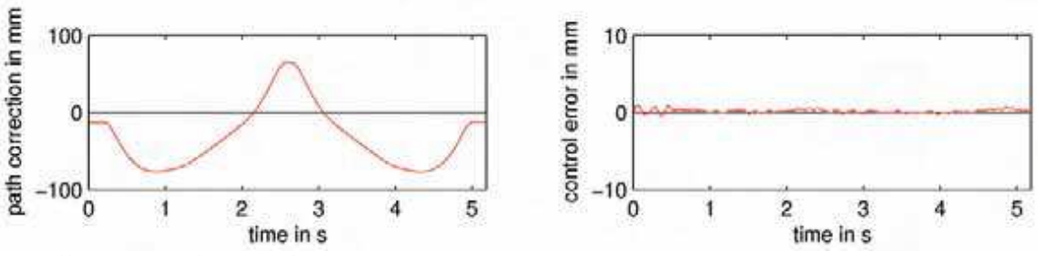
As a second experiment the robot has to follow, as well at 0.7 m/s, a light cable which lies on the ground (see Fig. 1a and Fig. 7a). This demonstrates that, in spite of the structured floor, there is no artificial background required



a Following a cable as in Figure 1a



b Following a tube in space as in Figure 1b (solid = horizontal, dashed = vertical)



c Following a planar tube as in Figure 1c

Figure 7. Sensor induced path correction and control error (note the different scales)

Path corrections are horizontal only, and computed by different methods. Because of bigger deviations from the reference path, path errors (see Table 2) are somewhat bigger than in the previous experiment. This is caused by approximation errors between the polynomials and the real shape of the line. Besides, exposure instant uncertainties will cause additional errors.

The errors of the method of this chapter are listed in the right-hand column. They are compared, first, with a method which, using the same control architecture, works without the predictive interface of the position controller, and therefore without feed-forward control. So with this method in the inner control loop, only the standard feedback controller is used. In contrast, the computation of the desired path is unchanged. Depending on the shape of the line, only a small increase of the path errors appears with this setup. This is because although the position control is not predictive, the vision system still uses the upper and lower regions of the image, thus allowing predictions of the desired path.

Control	without use of sensor data	by visual servoing with PD controller without prediction	with position controller without feed-forward with prediction	with position controller with feed-forward with prediction
Mean pixel error	-	35 pixel	1.9 pixel	1.3 pixel
Max. pixel error	-	63 pixel	4.7 pixel	3.6 pixel
Mean path error	95 mm	9.7 mm	0.8 mm	0.6 mm
Max. path error	157 mm	19.9 mm	2.4 mm	1.5 mm

Table 2. Path error when following a cable using different methods

A further alternative is a classical visual servoing algorithm which only evaluates the location of the line in the center of the image, i.e. without any prediction. Control is implemented using a coarsely optimized PD algorithm for the Cartesian signal in x direction of the reference system. This controller ignores the camera positions.

$$\begin{aligned} {}^{r}x_{cx}(k) &= {}^{r}x_{cx}(k-1) \\ &+K_P \cdot {}^{r}e_x(k) + K_D \cdot ({}^{r}e_x(k) - {}^{r}e_x(k-1)) \end{aligned} \tag{21}$$

Because of the tilted mounted camera (see Fig. 1d), a control error of

$${}^{r}e_x = ({}^{r}z_n - {}^{a}\mathbf{T}_{c23}) \cdot \frac{{}^{a}\mathbf{T}_{c00} \cdot \xi + {}^{a}\mathbf{T}_{c01} \cdot \eta + {}^{a}\mathbf{T}_{c02}}{{}^{a}\mathbf{T}_{c20} \cdot \xi + {}^{a}\mathbf{T}_{c21} \cdot \eta + {}^{a}\mathbf{T}_{c22}} + ({}^{a}\mathbf{T}_{c03} - {}^{r}x_n) \tag{22}$$

is evaluated. ${}^{a}\mathbf{T}_{cij}$ are the elements of the (constant) transformation matrix between tcp and camera. In this experiment the performance is poor, and without limitation[2] the robot would leave the allowed range of accelerations.

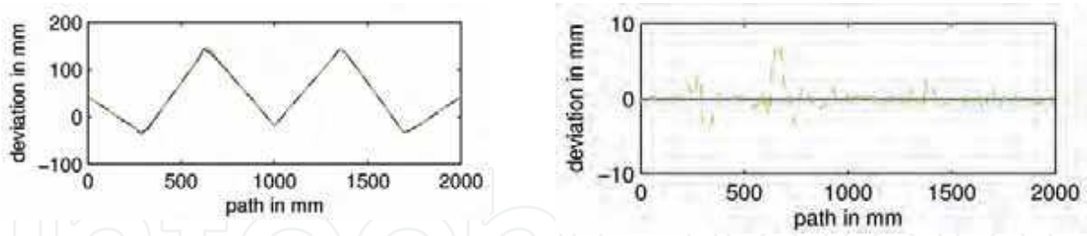
For the non continuous layout of the sheets in Fig. 4 we restrict on the modified method of section 5.2 because this method outperforms the other one, except for special cases, as in (Lange et al., 2006), where both methods are applicable. This time the programmed path is a straight line in y-direction, executed back and forth at 0.7 m/s and with $\mathbf{s}_r = 0$. In contrast to the previous experiments, with 600 mm the distance between the camera and the edge is about twice as much as before. This reduces the positional resolution but it enlarges the visible region to about n_i = 25 sampling steps of the controller which is sufficient with $n_d = 15$. Such an extension is useful to fulfil $n_i \gg n_d$.

The results are displayed in Fig. 8, on the left hand side with respect to the reference path and on the right hand side with respect to the sensed edge. With explicit image-based control (Fig. 8a) or small D as in Fig. 8b we result in high accelerations at the vertices in the sheets. On the other side, the edges are tracked as closely as possible. This desired path is accurately executed by the ideal position controlled robot besides a couple of time-steps in which the acceleration limits of the robot are reached. Nevertheless a mean tracking error of about 1 mm is reached.

By specifying $M = 0.001\,\mathrm{s}^2$ and $D = 0.06\,\mathrm{s}$ we define a desired trajectory that leaves the edges but that shows smooth transitions (see Fig. 5). Bigger impedance parameters, as in Fig. 8d are not recommended since then n_i limits the smoothing. Note that the vertex which is halfway on the left hand side diagrams of Fig. 8 is not smoothed because this is the point of reversing.

Videos of the reported and additional experiments can be found in (Lange, 2006).

[2] For convenience, accelerations exceeding the limitations are scaled to the feasible values.



a Explicit sensor-based control yielding a mean distance to the edge of 1.2 mm

b Impedance-based sensor-based control with D = 0.02 s and M = 0.001 s^2 yielding a mean distance to the edge of 1.7 mm

c Impedance-based sensor-based control with D = 0.06 s and M = 0.001 s^2 yielding a mean distance to the edge of 5 mm

d Impedance-based sensor-based control with D = 0.2 s and M = 0.001 s^2 yielding a mean distance to the edge of 23 mm

Figure 8. Plots of ${}^{r}x_d$ (left) and ${}^{o}x_d$ (right) when tracking the sensed edge (red), back and forth, using camera-based impedance control. Desired (black) and actual (dashed green) path are almost identical, besides the experiment without smoothing.

7. Conclusion

The article shows that vision systems are also applicable for tracking tasks with high robot speed. Even accurate control of the robot is possible. The crucial fact is the predictive image evaluation, maybe in combination with an adaptive positional feed-forward control.
The results are reached by means of a cost effective method for which additional hardware is limited to standard components. In order to guarantee low costs, image processing, computation of the desired path and dynamical control are executed using only the standard processor of an industrial robot controller, in our case the KUKA KRC1.
For non continuous or non contiguous paths we propose smoothing similar to impedance control. Then a reduction of the standards of position accuracy is tolerated for specified components. The user has to specify parameters in order to find a compromise between a smooth trajectory and minimum path deviations with respect to the sensed edges. The modified impedance-based method effects that the robot follows the sensed edge as accurate as possible, except for vertices or discontinuities where the path is smoothed. Among the two shown methods the latter it is favourable if a predictive sensor is used.
The methods are demonstrated in tasks where the location and the shape of planar or spatial lines are used to modify the robot path during motion. For continuous lines the mean resulting path error is below 1 mm, whereas for lines with vertices, exact tracking is not executable with a robot. Then the amount of smoothing determines the deviation from the lines.
Future work will improve the measurements of the camera pose e.g. using an observer, thus avoiding oscillations caused by joint elasticity.

Acknowledgements
This work has been partially supported by the Bayerische Forschungsstiftung.

8. References

CalLab.http://www.robotic.dlr.de/VISION/Projects/Calibration/CalLab.html, 1999.

Clarke, D. W. , C. Mohtadi, and P. S. Tuff. Generalized predictive control - part I. the basic algorithm. *Automatica*, 23(2):137–148, 1987.

Gangloff, J. A. and M. F. de Mathelin. Visual servoing of a 6-dof manipulator for unknown 3-d profile following. *IEEE Trans. on Robotics and Automation*, 18(4):511–520, August 2002.

Ginhoux, R. et al. Beating heart tracking in robotic surgery using 500 Hz visual servoing, model predictive control and an adaptive observer. In *Proc. 2004 IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 274–279, New Orleans, LA, April 2004.

Comport, A. I., D. Kragic, E. Marchand, and F. Chaumette. Robust real-time visual tracking: Comparison, theoretical analysis and performance evaluation. In *Proc. 2005 IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2852–2857, Barcelona, Spain, April 2005.

Grotjahn, M. and B. Heimann. Model-based feedforward control in industrial robotics. *The International Journal on Robotics Research*, 21(1):45–60, January 2002.

Lange, F. and G. Hirzinger. Learning of a controller for non-recurring fast movements. *Advanced Robotics*, 10(2):229–244, April 1996.

Lange, F. and G. Hirzinger. Is vision the appropriate sensor for cost oriented automation? In R. Bernhardt and H.-H. Erbe, editors, *Cost Oriented Automation (Low Cost Automation 2001)*, Berlin, Germany, October 2001. Published in IFAC Proceedings, Elsevier Science, 2002.

Lange, F. and G. Hirzinger. Predictive visual tracking of lines by industrial robots. *The International Journal on Robotics Research*, 22(10-11):889–903, Oct-Nov 2003.

Lange, F. and G. Hirzinger. Calibration and synchronization of a robot-mounted camera for fast sensor-based robot motion. In *Proc. 2005 IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3911–3916, Barcelona, Spain, April 2005.

Lange, F.. Video clips. http://www.robotic.de/?id=43., 2006.

Lange, F., M. Frommberger, and G. Hirzinger. Is impedance-based control suitable for trajectory smoothing? In *Preprints 8th IFAC Symposium on Robot Control (SYROCO 2006)*, Bologna, Italy, Sept. 2006.

Meta-Scout GmbH. SCOUT joint tracking system. http://www.scout-sensor.com/index-engl.html.

Nakabo, Y., T. Mukai, N. Fujikawa, Y. Takeuchi, and N. Ohnishi. Cooperative object tracking by high-speed binocular head. In *Proc. 2005 IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1585–1590, Barcelona, Spain, April 2005.

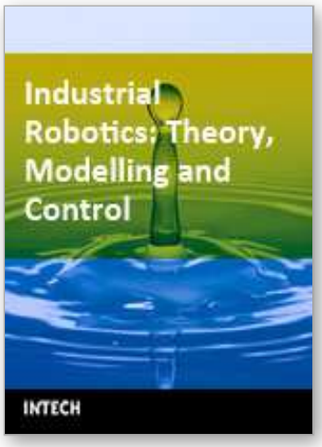
Pertin, F. and J.-M. Bonnet des Tuves. Real time robot controller abstraction layer. In *Proc. Int. Symposium on Robots (ISR)*, Paris, France, March 2004.

Rives, P. and J.-J. Borrelly. Real-time image processing for image-based visual servoing. In M. Vincze and G. D. Hager, editors, *Robust vision for vision-based control of motion*, pages 99–107. IEEE Press, 2000.

Tahri, O. and F. Chaumette. Complex objects pose estimation based on image moments invariants. In *Proc. 2005 IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 438–443, Barcelona, Spain, April 2005.

Zhang, J., R. Lumia, J. Wood, and G. Starr. Delay dependent stability-limits in high performance real-time visual servoing systems. In *Proc. IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pages 485–491, Las Vegas, Nevada, Oct. 2003.

IntechOpen



Industrial Robotics: Theory, Modelling and Control

Edited by Sam Cubero

ISBN 3-86611-285-8

Hard cover, 964 pages

Publisher Pro Literatur Verlag, Germany / ARS, Austria

Published online 01, December, 2006

Published in print edition December, 2006

This book covers a wide range of topics relating to advanced industrial robotics, sensors and automation technologies. Although being highly technical and complex in nature, the papers presented in this book represent some of the latest cutting edge technologies and advancements in industrial robotics technology. This book covers topics such as networking, properties of manipulators, forward and inverse robot arm kinematics, motion path-planning, machine vision and many other practical topics too numerous to list here. The authors and editor of this book wish to inspire people, especially young ones, to get involved with robotic and mechatronic engineering technology and to develop new and exciting practical applications, perhaps using the ideas and concepts presented herein.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Friedrich Lange and Gerd Hirzinger (2006). Spatial Vision-Based Control of High-Speed Robot Arms, Industrial Robotics: Theory, Modelling and Control, Sam Cubero (Ed.), ISBN: 3-86611-285-8, InTech, Available from: http://www.intechopen.com/books/industrial_robotics_theory_modelling_and_control/spatial_vision-based_control_of_high-speed_robot_arms

INTECH
open science | open minds

InTech Europe

University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China

Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2006 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen