Joint Conference on Robotics ISR2006 / ROBOTIK 2006, May, 2006, Munich, Germany

IMPEDANCE-BASED SMOOTHING FOR VISUAL SERVOING ALONG EDGES

Friedrich Lange **Mirko Frommberger** **Gerd Hirzinger**
Institute of Robotics and Mechatronics
Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)
Germany

Speaker: Friedrich Lange, DLR e-mail: Friedrich.Lange@dlr.de

Topic: Visual servoing
Keywords: visual servoing, tracking, impedance control, hierarchical control

Abstract
In contrast to explicit sensor-based control where the goal is to minimize the difference between sensed and desired sensor values, impedance-based control also considers the required accelerations and thus smoothes robot motion. Two approaches to impedance control are developed and applied to the sensor-based method in [1] which uses predictive control to track unknown contours at high speed. With the impedance approach the contour is not more required to be smooth.

1. Introduction

Visual servoing is well known as the tracking of a randomly moving target by a camera or a robot [2,3,4]. For this task the performance is limited, among others, by the possible accelerations of the robot or the camera. But there are also servoing tasks which can be executed almost without tracking error. This concerns tracking of contours. In industry, camera-based tracking of contours is used e.g. to spray glue or sealing strips if the robot motion cannot be specified in advance because of workpiece tolerances or inaccurate fixation of the parts (see e.g. [5]). This kind of visual servoing is simpler than tracking a randomly moving target since the desired path is visible in advance. Therefore the specifications will be more demanding. The challenge is to control the robot very accurately at high speed but without using an expensive high speed camera.

For this purpose the authors presented a control method [1] which evaluates the images of a camera mounted on a robot in order to model the contour. We predict the course of the contour beyond the current tool center point - see the red markers in Figure 2. Smooth contours as curved lines are very accurately tracked by this method. The drawback is that discontinuities, corners or even noisy edge data yield high accelerations since the robot tries to execute directly the sensed path. Therefore, this method is not well suited for the edges and vertices of Figure 1. The paper presents an impedance-based approach for camera-based servoing, following the procedures of force control.

In contrast to current research [6], we assume that problems with image processing, feature detection and projections are solved, which holds for our simple black and white scenario. In addition, the initial configuration is supposed to be close to

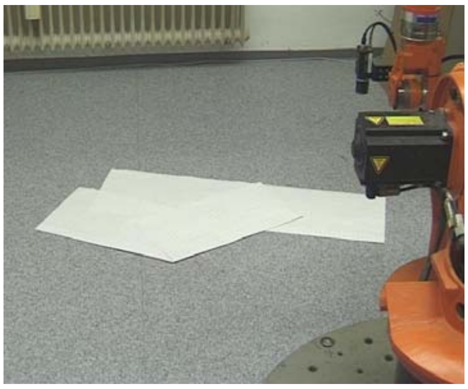


Figure 1. Originally the robot is programmed to execute a linear motion, back and forth. But image data of the robot-mounted camera are used to modify the desired path in order to track the edges of randomly spread sheets.

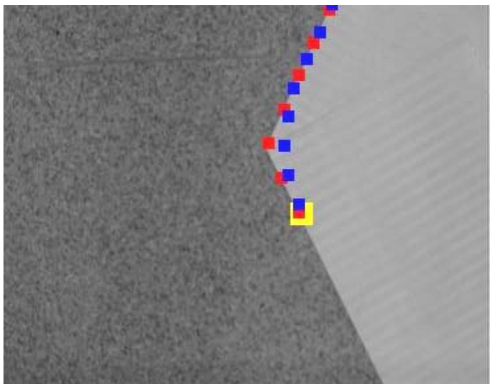


Figure 2. View from the robot-mounted camera. Sensed edge points (red) and the smoothed path (blue) computed from the present camera pose (yellow) are depicted.

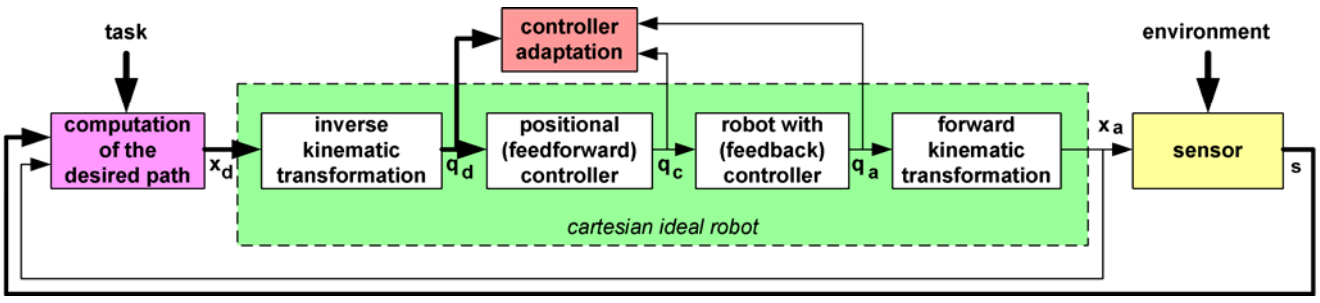

task
computation of the desired path
x_d
inverse kinematic transformation
q_d
controller adaptation
positional (feedforward) controller
q_c
robot with (feedback) controller
q_a
forward kinematic transformation
x_a
cartesian ideal robot
environment
sensor
s

Figure 3. Control architecture

the target configuration, so that large rotations as in [7] do not appear.

The paper is organized as follows: in the next section we summarize previous work on visual servoing along contours. Section 3 shows the limits of the reported method and presents two impedance-based alternatives. Section 4 presents illustrative experimental results and gives some hints for the parameter selection.

2. Sensor based control with prediction

Our approach of sensor based control uses accurate position control for the inner loop. The desired positions are computed from sensor data in an outer loop (see Figure 3). Therefore we first report on the structure of the position control from [1], and then concentrate on the integration of sensor data.

2.1. How to reach path accuracy

Position control is divided into two parts as well. The cascaded control system provided by the robot manufacturer - KUKA in this case - is used as the feedback controller. This controller is assessed by a so called sensor interface, similar to [8]. This interface accepts joint position commands and provides the measured joint angles, both of them in sampling steps of 12 ms.

An adaptive feedforward controller is added since position commands and actual positions usually differ because of the robot dynamics. The inputs to this controller are the desired joint angles at the current and future time steps. Thus an adapted feedforward controller is able to reach that the commanded and the actually reached joint angles shall coincide, provided that the adaptation process has generated a controller that compensates the robot dynamics.

Because of the delays in the robot dynamics, the feedforward controller has to process future desired values. A preview of about twice the time constant of the robot is required. This is no problem for given robot paths but in the case of sensor based motion, it requires desired path predictions of n_d time steps. The resulting feedforward controller is

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

where $\mathbf{K}_{qi}$ stands for the adapted parameters and $\mathbf{q}_d$ and $\mathbf{q}_c$ are the desired and commanded joint angles respectively. This controller can be seen as a filter of the desired positions.

Position control is done on joint level since in Cartesian space there are many dynamical couplings between individual components. In contrast, in joint space the robot model is almost linear, due to the position independent motor dynamics. Therefore we provide the kinematic transformations at the input and output of the positional control block in Figure 3.

2.2. Handling sensor data

The task of the leftmost block in Figure 3 is to provide the desired positions that solve the given problem, thereby processing sensor data if there is any uncertainty in the task description.

In this concern we assume that the sensor data represent the difference between the actual sensor pose and a sensed object. The following steps are executed if the task is to keep a predefined distance with respect to this object:

1. Computation of the absolute object pose from
   - the sensed pose with respect to the sensor and
   - the sensor pose at the measurement instant
2. Addition of the desired distance
3. Transfer of the desired pose to the position control of the robot.

In this notation the object pose represents position and orientation of the part of the object that has to be tracked. So a typical object pose is a point of a sensed edge, where the point is moving in time along the edge.

In the case of a camera as the sensor, the absolute object pose $\mathbf{T}_o$ is computed (e.g. in homogeneous coordinates) by the transformation of the sensed object pose ${}^c\mathbf{T}_o$ from the camera system to the robot base system or world system. This gives

$$\mathbf{T}_o(k) = \mathbf{T}_c(k) \cdot {}^c\mathbf{T}_o(k) \tag{2}$$

where $\mathbf{T}_c$ represents the pose of the camera in the world system. For an eye-in-hand configuration, $\mathbf{T}_c$ is computed from the joint angles at the instant of the exposure and the given kinematics that reflect the actual end-effector pose $\mathbf{T}_a$. Then the mounting ${}^a\mathbf{T}_c$ of the camera with respect to the robot end-effector is considered by

$$\mathbf{T}_c(k) = \mathbf{T}_a(k) \cdot {}^a\mathbf{T}_c . \tag{3}$$

In the case that the sensor data do not provide all degrees of freedom (dof) of the transformation, we assume the unknown components to be nominal. Only the sensed components are tracked.

The “addition” of the desired distance in the above list represents a transformation as well. The desired frame of the robot end-effector is

$$\mathbf{T}_d(k) = \mathbf{T}_o(k) \cdot {}^o\mathbf{T}_d \tag{4}$$

where ${}^o\mathbf{T}_d$ represents the desired “distance” between the desired pose and the object. For edge tracking this pose difference is usually expressed in the object reference system. In addition to a translational difference it may include the orientation between end-effector and object reference.

This approach differs from the usual control approach in that we do compute the absolute desired pose of the end-effector. Usual control algorithms take the sensed control error ${}^a\mathbf{T}_d$ between actual and desired poses to compute the commanded pose difference for the end-effector - neglecting the absolute pose.

The motivation for using absolute poses is that in this way the predictions of future time steps are represented correctly, even if we have changes in the orientation. Furthermore, as in the usual approach, a common translational error in $\mathbf{T}_a$ and $\mathbf{T}_c$ has no effect.

Another difference between usual feedback control of sensor data and this approach is the absence of controller parameters. A pose is computed and every control issues are left to the position control. Only smoothing of noise from sensor data is allowed. So this approach is only feasible for tracking tasks with small uncertainty. Planning of the initial trajectory of approximation to the object has to be solved in a different way.

2.3. Predictive use of sensor data

Since the used position control requires future desired poses, the evaluation of the sensor data has to be done in a predictive way. Besides the current desired frame of the end-effector, the desired poses at n_d future time steps are computed. With a camera as sensor and the particular task of tracking an edge, this prediction reduces to measuring the course of the contour and to keep the values that correspond to future time steps.

This means that the above list is executed for each time step of interest. In practice the edge is represented by a polynomial, which facilitates the point 1 of the above list.

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 edge without extra filtering of the sensor data.

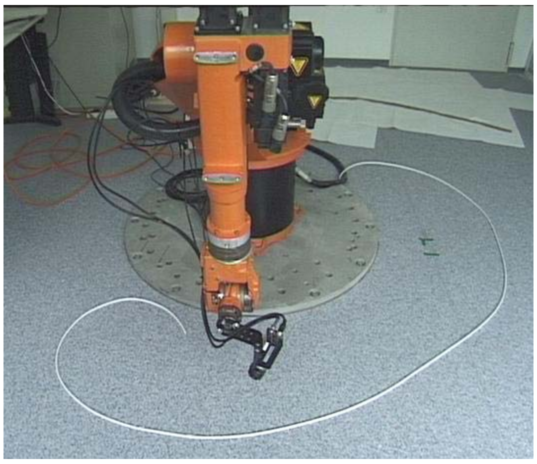


Figure 4. Originally the robot is programmed to execute a quarter of a circle around its base, back and forth. Image data of the robot-mounted camera are used to modify the radial distance in order to track the wire.

Previous work [1,9,10,11] demonstrates that tracking of a curved line (see Figure 4) at high speed of 0.7 m/s is possible with a mean path error of 0.3 mm (see Figure 5). If the orientation of the camera is servoed as well, we can reach an angular accuracy of 0.02 rad (about 1 degree) with slightly inferior translational accuracy of 0.4 mm. 3D trajectories, i.e. motion which is not restricted to the horizontal plane, has been not as accurate in our experiments, but 1 mm mean path error can be reached. [12] summarizes the video-clips of the experiments.

3. Specification of a smooth desired path

A problem may occur with the presented method if a line has to be followed that is not continuous. Figure 1 gave an example of such a line. At the vertices between straight edges the robot tries to execute a velocity step without considering acceleration limitations.

To prevent this case, we apply impedance-based control. A filter that smoothes the sensed edge so that the resulting contour can be tracked by the robot is located between point 2 and point 3 of the above list. 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.

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.

3.1. Impedance-based control

For reasons of clarity we now restrict on Cartesian positions $\mathbf{x}$ instead of positions and orientations as in (2) to (4). So vectors are used instead of matrices of homogeneous coordinates, and therefore the orientation is not affected by the impedance-based control.

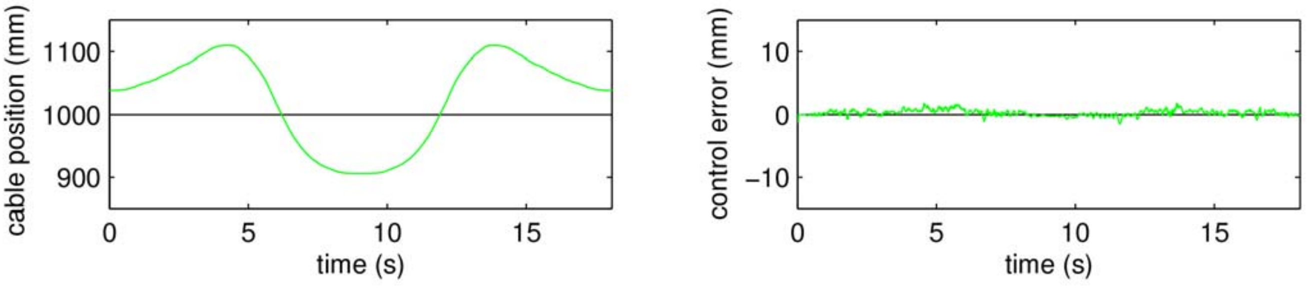


Figure 5. Radial component of the cable position and of the control error

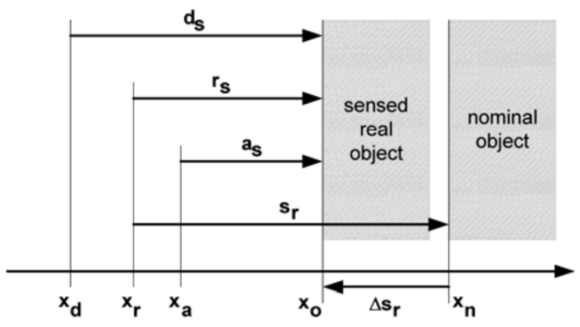


Figure 6. Robot tool positions and sensed distances to a fixed object

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

$$\mathbf{E}\cdot {}^r\mathbf{x}_d + \mathbf{D}\cdot {}^r\dot{\mathbf{x}}_d + \mathbf{M}\cdot {}^r\ddot{\mathbf{x}}_d = {}^r\mathbf{x}_a + {}^a\mathbf{s} - \mathbf{s}_r = \Delta\mathbf{s}_r \tag{5}$$

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

Note that within this section the index $*_d$ represents filtered desired values which are usually different from the result of equations like (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 $\mathbf{T}_o$ and $-\mathbf{s}_r$ to ${}^o\mathbf{T}_d$.

With $\mathbf{E} = \mathbf{I}$ and $\mathbf{D} = \mathbf{M} = 0$, (5) represents explicit sensor-based control as in Section 2. If, in contrast, we specify $\mathbf{M} > 0$, 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 $\mathbf{M} > 0$ we further specify $\mathbf{D} > 0$ 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

$$\begin{aligned}\mathbf{E}\cdot {}^r\mathbf{x}_d(k+i) + \frac{\mathbf{D}}{2T_0}\cdot\left({}^r\mathbf{x}_d(k+i+1) - {}^r\mathbf{x}_d(k+i-1)\right) + \frac{\mathbf{M}}{T_0^2}\cdot\left({}^r\mathbf{x}_d(k+i+1) - 2\,{}^r\mathbf{x}_d(k+i) + {}^r\mathbf{x}_d(k+i-1)\right)\\ = {}^r\mathbf{x}_a(k) + {}^a\mathbf{s}(k+i/k) - \mathbf{s}_r.\end{aligned} \tag{6}$$

${}^a\mathbf{s}(k+i/k)$ reflects the sensed edge position for time step $(k+i)$, sensed in the image at time step k. Equation (6) gives a system of equations for ${}^r\mathbf{x}_d(k+i)$ 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 $\mathbf{D} = \mathbf{M} = 0$ a vertex or a discontinuity of ${}^r\mathbf{x}_d(k+i)$ would be reproduced. In contrast, $\mathbf{M} > 0$ 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 (5) 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

$$\mathbf{E}\cdot {}^d\mathbf{x}_o + \mathbf{D}\cdot {}^d\dot{\mathbf{x}}_o + \mathbf{M}\cdot {}^d\ddot{\mathbf{x}}_o = \mathbf{s}_r + (\mathbf{E} - \mathbf{I})\cdot {}^r\mathbf{x}_o + \mathbf{D}\cdot {}^r\dot{\mathbf{x}}_o + \mathbf{M}\cdot {}^r\ddot{\mathbf{x}}_o \tag{7}$$

can be derived in the same way as (5). The right hand side expression is independent of the resulting desired robot motion $\mathbf{x}_d$.

3.2. Modified approach

Experiments with impedance-based control according to Section 3.1 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 positional 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.

$$\mathbf{E} \cdot {}^{r}\mathbf{x}_d(k+i) = {}^{r}\mathbf{x}_a(k) + {}^{a}\mathbf{s}(k+i/k) - \mathbf{s}_r \tag{8}$$

$$\frac{\mathbf{D}}{2T_0} \cdot \left({}^{r}\mathbf{x}_d(k+i+1) - {}^{r}\mathbf{x}_d(k+i-1) \right) = 0 \tag{9}$$

$$\frac{\mathbf{M}}{T_0^2} \cdot \left({}^{r}\mathbf{x}_d(k+i+1) - 2\,{}^{r}\mathbf{x}_d(k+i) + {}^{r}\mathbf{x}_d(k+i-1) \right) = 0 \tag{10}$$

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

3.3. Implementation

Instead of solving (6) or minimizing the mean errors of (8) to (10) 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 (6) or (8) to (10) 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 ${}^{r}\mathbf{x}_d(l) = \Delta\mathbf{s}_r(l)$.

4. Experiments

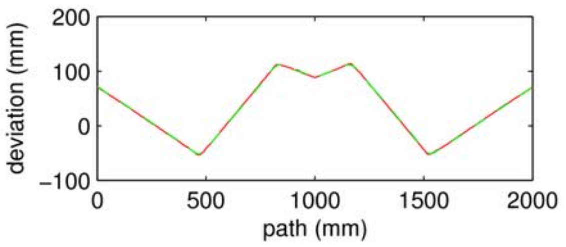
Both methods are demonstrated using the layout in Figure 1. The programmed path is a straight line in y-direction, executed back and forth at 0.7 m/s. The robot is a KUKA KR6 controlled with a KRC1 industrial controller with sampling steps of 12 ms. The camera is a standard monochrome device using a frame grabber that allows to get a new image field every 20 ms. Images are used to control the x-component. In contrast to previous experiments reported in Section 2.3, 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 25 sampling steps of the controller which is sufficient with $n_d = 15$.

Instead of the continuous lines in previous papers we now track straight edges with vertices, again with $\mathbf{s}_r = 0$. The results are displayed in Figure 7 for the two methods. With explicit image-based control (Figure 7a), which is identical for both variants, 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[1]. Nevertheless a mean tracking error of less than 1 mm is reached. Experiments with small D as in the second row of Figure 7 give similar results.

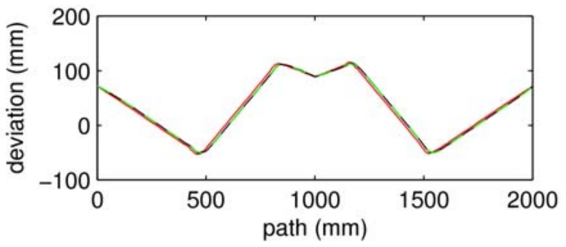
In contrast, by specifying $M = 0.001\,\mathrm{s}^2$ and $D = 0.06\,\mathrm{s}$ we define a desired trajectory that leaves the edges but that shows smooth transitions (see Figure 2). The third row of Figure 7 shows that the method of Section 3.1 computes a delayed trajectory whereas the modified method tracks the edges more closely. The same displacement can be seen in the last row of Figure 7. Here, with both methods, the original edge can hardly be found. Resuming, the modified method outperforms the other one, except for special cases where both methods are applicable (see [13] for other examples). Note that the vertex which is halfway in Figure 7 is not smoothed because this is the point of reversing.

Figure 8 shows the executed motion with respect to the edges, not to the reference path as in Figure 7. We see that with the first method the edges are not accurately tracked (left hand side of Figure 8), while with the modified variant the desired path is close to the sensed edges, except at the vertices. Figure 8b shows the reachable accuracy. The actually reached trajectory is not as close to the edge as the desired motion which has been computed with $M = 0.0001\,\mathrm{s}^2$ and $D = 0.02\,\mathrm{s}$.

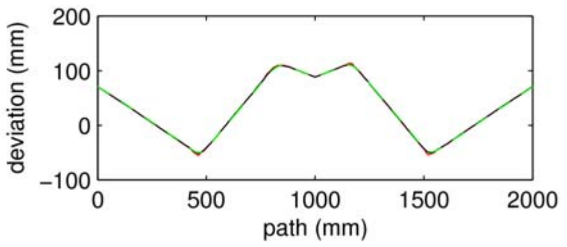
[1] For convenience, accelerations exceeding the limitations are scaled to the feasible values.



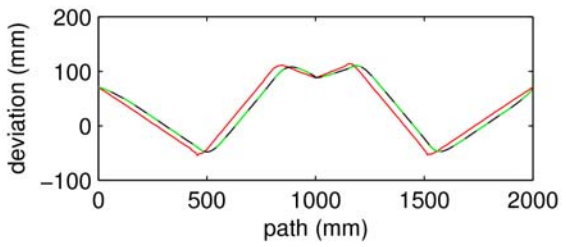
a *Explicit sensor-based control*



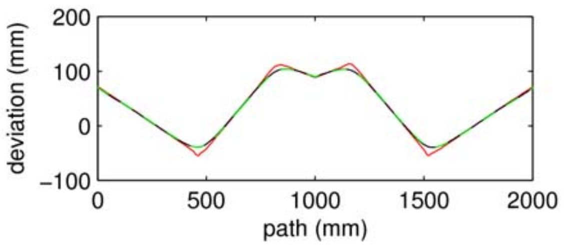
b *Impedance-based sensor-based control according to Section 3.1 with* $D = 0.02$ *s and* $M = 0.0001$ s^2



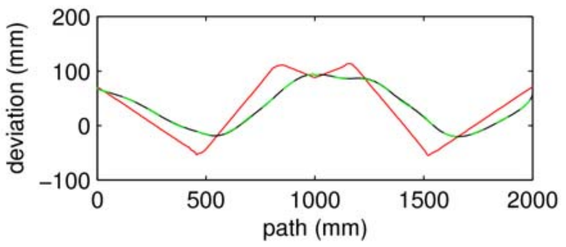
c *Impedance-based sensor-based control according to Section 3.2 with* $D = 0.02$ *s and* $M = 0.0001$ s^2



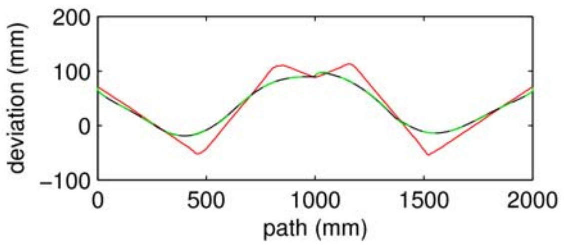
d *Impedance-based sensor-based control according to Section 3.1 with* $D = 0.06$ *s and* $M = 0.001$ s^2



e *Impedance-based sensor-based control according to Section 3.2 with* $D = 0.06$ *s and* $M = 0.001$ s^2



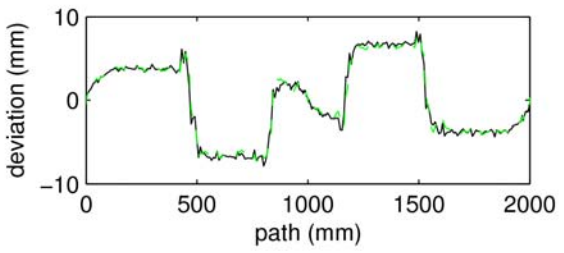
f *Impedance-based sensor-based control according to Section 3.1 with* $D = 0.2$ *s and* $M = 0.01$ s^2



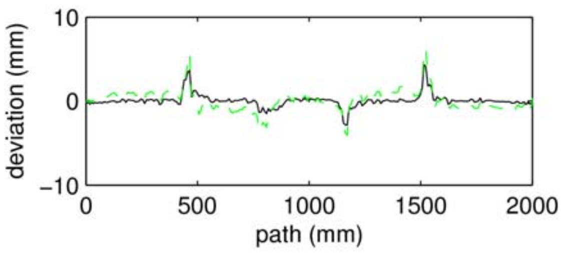
g *Impedance-based sensor-based control according to Section 3.2 with* $D = 0.2$ *s and* $M = 0.01$ s^2

Figure 7. Plots of ${}^{r}x_{d}$ *when tracking the sensed edge (red), back and forth, using camera-based impedance control. Desired (black) and actual (dashed green) path are almost identical.*

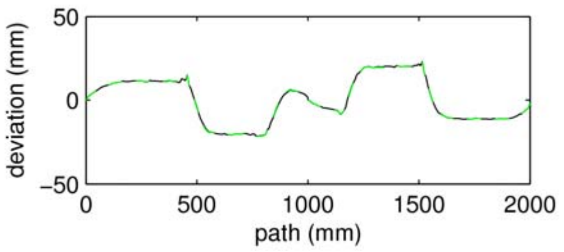
As regards the parameters, with the method of Section 3.1 we primarily tune M in order to smooth the trajectory and then set $D = 2\sqrt{E \cdot M}$ in order to suppress oscillations. For the modified version of impedance-based control, the tuning of D controls the amount of smoothing at the vertices, since this reduces the speed error as well. M is tuned to reduce accelerations caused by noise or by the limited field of view. Figure 9 shows the influence of the two parameters with a different layout of the sheets.



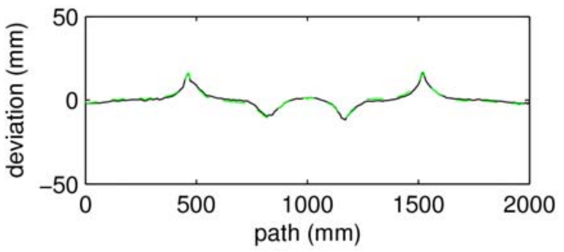
a *Impedance-based sensor-based control of Figure 7b according to Section 3.1, yielding a mean distance to the edge of 5 mm.*



b *Impedance-based sensor-based control of Figure 7c according to Section 3.2, yielding a mean distance to the edge of 1.3 mm.*



c *Impedance-based sensor-based control of Figure 7d according to Section 3.1, yielding a mean distance to the edge of 12 mm.*



d *Impedance-based sensor-based control of Figure 7e according to Section 3.2, yielding a mean distance to the edge of 4 mm.*

Figure 8. Plots of the distance ${}^{o}x_d$ *to the edge.*

5. Conclusion

The paper presents two methods that enable the execution of a tracking task for which explicit sensor-based control would exceed the acceleration limits of the robot. This is reached by tolerating a reduction of the standards of position accuracy 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.

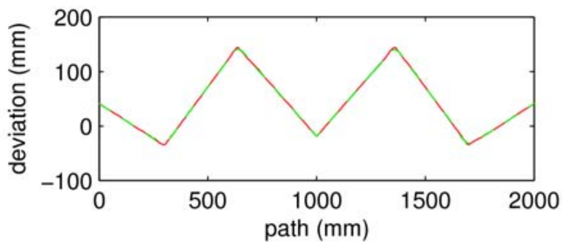
Both methods are qualified to smooth given robot paths or sensor-based trajectories, where for the latter it is favourable if a predictive sensor is used. We adopt the separation of the position control in an inner loop and the determination of the desired trajectory in the outer loop. The latter is modified by this kind of impedance-based approach. It is therefore independent of the implementation of the robot control and of existing control errors.

Acknowledgement

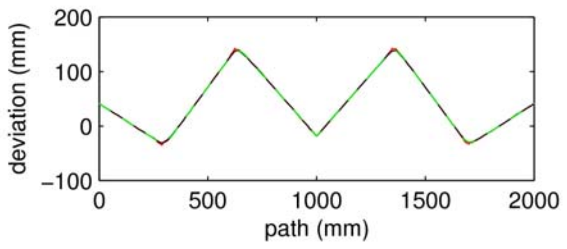
This work has been partially supported by the Bayerische Forschungsstiftung.

References

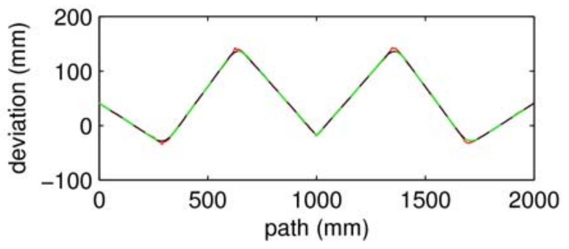
[1] F. Lange and G. Hirzinger. Spatial control of high speed robot arms using a tilted camera. In *Proc. Int. Symposium on Robotics (ISR 2004)*, Paris, France, March 2004.
[2] Danica Kragic and Ville Kyrki. Tutorial on visual tracking: 2d/3d methods and cue integration. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, Barcelona, Spain, April 2005.
[3] Y. Nakabo, 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.
[4] R. Ginhoux 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.
[5] QUISS GmbH. QUISS Products RTVision.t, 2005. www.quiss.com/en/produkte/rtvision_t.html/.
[6] A. I. Comport, 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.
[7] O. Tahri 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.
[8] F. Pertin and J.-M. Bonnet des Tuves. Real time robot controller abstraction layer. In *Proc. Int. Symposium on Robots (ISR)*, Paris, France, March 2004.



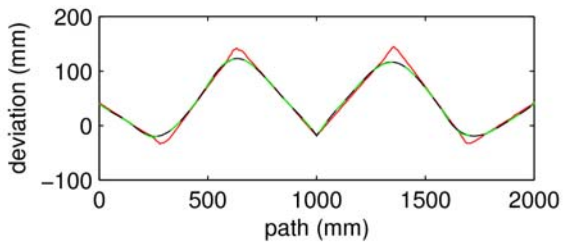
a Explicit sensor-based control



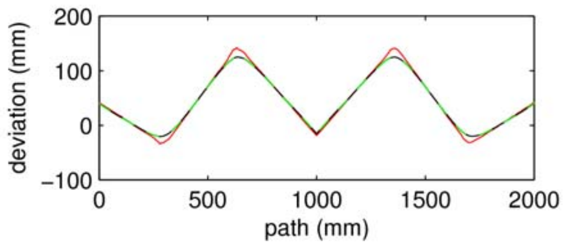
b $D = 0.02\ s$ *and* $M = 0.0001\ s^2$



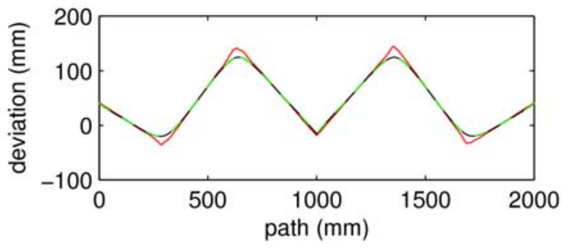
c $D = 0.02\ s$ *and* $M = 0.001\ s^2$



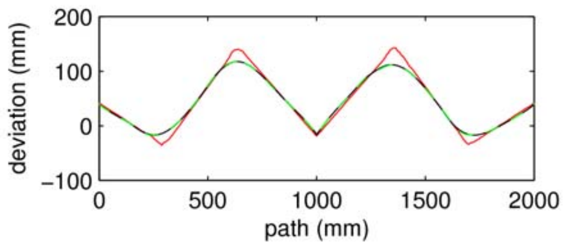
d $D = 0.02\ s$ *and* $M = 0.01\ s^2$



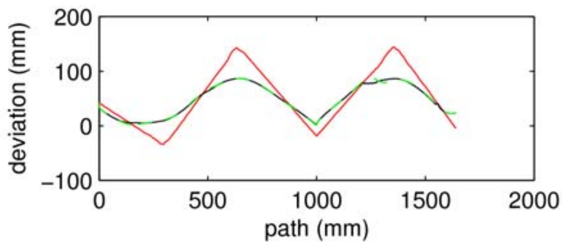
e $D = 0.06\ s$ *and* $M = 0.0001\ s^2$



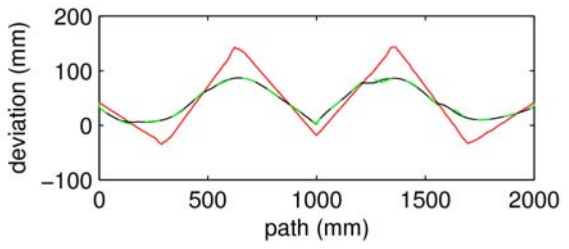
f $D = 0.06\ s$ *and* $M = 0.001\ s^2$



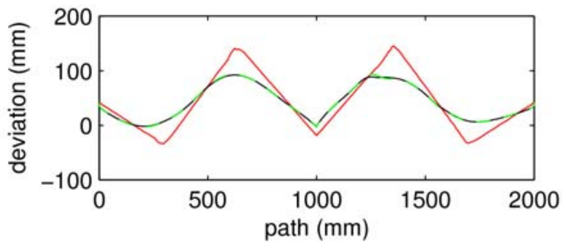
g $D = 0.06\ s$ *and* $M = 0.01\ s^2$



h $D = 0.2\ s$ *and* $M = 0.0001\ s^2$



i $D = 0.2\ s$ *and* $M = 0.001\ s^2$



j $D = 0.2\ s$ *and* $M = 0.01\ s^2$

Figure 9. Plots of ${}^{r}x_{d}$ *when tracking the sensed edge (red), back and forth, using the modified camera-based impedance control. Desired (black) and actual (dashed green) path are almost identical.*

[9] F. Lange 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.
[10] F. Lange 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.
[11] F. Lange. Predictive sensor based control of robots with positional interface, 2005. http://www.robotic.de/?id=42.
[12] F. Lange. Video clips, 2005. http://www.robotic.de/?id=43.
[13] F. Lange, 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, submitted.

Brief curriculum vitae of authors
Dr.-Ing. Friedrich Lange
received his Dipl.-Ing. degree from the Technical University of Darmstadt in 1982 and his doctor's degree from the University of Karlsruhe in 2003. He joined DLR, the German Aerospace Center, in 1983 and now works within the Institute of Robotics and Mechatronics. His first works covered learning and adaptation methods to improve the accuracy of position controlled robots, e.g. by using neural nets. In the last years his work focuses on the use of sensor data from force-torque and vision sensors for robot control.

Mr. Mirko Frommberger
studied product development in the field of mechatronics at the University of Applied Sciences of Bielefeld where he received his Dipl.-Ing. (FH) degree in 2002. Since 2001 he is member of the Institute of Robotics and Mechatronics at the German Aerospace Center DLR. There he is responsible for the operation of industrial robots. Other work focuses on sensors, programming of microprocessors, and robustness of electronics with respect to radiation during space missions.

Prof. Dr.-Ing. Gerd Hirzinger
received his Dipl.-Ing. degree and the doctor's degree from the Technical University of Munich, in 1969 and 1974 respectively. In 1991 he received a joint professorship from the Technical University of Munich, and in 2003 a honorary professorship at the Harbin Institute of Technology in China. Since 1992 he has been director at DLR's Institute of Robotics and Mechatronics. He received numerous national and international awards, e.g. in 1994 the Joseph-Engelberger-Award for achievements in robotic science and in 1995 the Leibniz-Award, the highest scientific award in Germany and the JARA (Japan robotics association) Award. In 1996 he received the Karl-Heinz-Beckurts-Award, Germany's most important award for outstanding promotion of the partnership between science and industry, and in 1997 the IEEE-Fellow Award. In 2004 he got the order of merit of the Federal Republic of Germany and became member of the "wall of fame" of the Heinz Nixdorf Computer Museum. In 2005 he received the IEEE Pioneer Award of the Robotics and Automation Society and the "honorary citizenship" of Budapest Tech.