Improving force controlled planar contour following using on-line eye-in-hand vision based feedforward

J. Baeten, J. De Schutter
Affiliation : Katholieke Universiteit Leuven, Dept. Mechanical Engineering
Address : Celestijnenlaan 300B, B-3001 Heverlee

Abstract

The limited bandwidth of sensor-based feedback control, restricts the execution speed of a force controlled planar contour following task if the shape of the workpiece is unknown. This paper shows how appropriate feedforward control of the task frame orientation, calculated on-line from an eye-in-hand camera image, results in a faster or more accurate executed task.

However, keeping the contour in the camera field of view in addition to maintaining a force controlled contact, imposes additional requirements on the controller. This double control problem is specified in the task frame formalism and executed in a hybrid position/force control environment. It is solved using the redundancy for the rotation in the plane, which exists for rotational symmetric tools. The orientation of the task frame can then be controlled independent of the end effector. Experimental results are presented to validate the approach.

1 Introduction

In planar force controlled contour following, the robot is holding a tool and is required to follow the contour of a workpiece. When pose and shape of the workpiece are unknown, a sensor, in casu a force sensor, is used to modify or even generate on-line the tool trajectory. Due to the limited bandwidth of the sensor-based feedback control loop, the execution speed of the task has to be limited in order to prevent loss of contact or occurrence of excessive contact forces.

This work shows how on-line vision based feedforward control improves the performance w.r.t. a pure force feedback controlled task. While maintaining the force controlled contact, the controller has to keep the camera, also mounted on the robot end effector, over the contour at all times.

The approach presented in this paper can be applied to all actions that scan surfaces along planar paths with a rotational symmetric tool: cleaning, polishing . . . It is especially useful for one-off tasks in which accurate positioning or calibration of the workpiece is costly or impossible.

1.1 Framework

Hybrid position/force control: In case of a point contact between tool and workpiece, the force controlled contour following task can easily be defined in the task frame formalism [1]. In this formalism, desired actions are specified separately for each direction of an orthogonal frame related to the tool: the task frame. Each direction $(x,y,z)$ is considered once as an axial direction and once as a polar direction. Basically, there are three types of directions: velocity, force and tracking directions.

Velocity and force directions are specified by setting a desired velocity respectively contact force. They correspond with control loops (d1) and (d2) of figure 1. For a tracking direction (d3), however, no desired velocity is specified. The controller will use this direction to automatically adjust the pose of the task frame w.r.t. the workpiece.

Due to the rotational symmetry of the tool, the relation between the task frame and the robot end effector is not necessarily fixed. Hence, there is a redundancy for the task frame orientation.

Adding the vision system: The camera is mounted on the end effector ahead of the force sensor with the optical axis normal to the plane. Two new control specifications arise. First, the contour has to be kept in the camera field of view by visual servoing. The task frame direction used to this end is a vision (controlled) direction (d4). Second, feedforward signals (d5) calculated from the local vision information of the contour, are added in order to reduce errors in both tracking and vision controlled directions. The better the feedforward velocity anticipates the motion of the task frame, the smaller the tracking error [2].

Since the camera is looking ahead, the delay time
and low bandwidth of the vision system can be compensated. Hence, they will not affect the stability of tracking and vision control loops (d3,d4).

However, generating the correct feedforward signal at the correct time instant, requires careful matching between the contour information, extracted in advance, and the actual tool position.

1.2 Related work

Unlike previously reported hybrid position/force control with force sensor and camera [6, 4] this work uses an eye-in-hand camera, rather than a fixed one. Also the use of the task frame formalism to merge both vision and force sensors distinguishes our work from [6, 4] or from visual servoing such as [5]. Furthermore, no identification of vision jacobian or optical flow are necessary. Many researchers reported vision based feedforward control. In contrast to this work, however, they mostly use off-line generated models e.g. [3] or partially known paths or workpieces.

1.3 Used variables and notation

Figure 2a) gives the global experimental set-up defining the task, the camera and the robot end effector frames. They are denoted by subscripts t, cam and ee respectively.

Following notations are used: \( P \) indicates an absolute pose [mm] or [rad]; \( \Delta P_e \) is the relative pose (or error) of the contour w.r.t. the frame \( q \); \( F \) is a force [N] or torque [Nm], \( u \) is a generalized velocity [mm/sec] or [rad/sec] with superscripts \( c \), \( m \) and \( d \) referring to commanded, measured and desired signals resp.; superscript \( ff \) indicates the feedforward signal. The directions are indicated between brackets with \( x \) and \( y \) for translations and \( \theta \) for a rotation around the z-axis.

Figure 3a) shows how the tracking (orientation) error \( \Delta P_e(\theta) \) between the task frame and the workpiece tangent and normal, is identified based on the velocities \( u_c^x(x) \) and \( u_c^y(y) \). Figure 3b) defines the local contour variables as seen by the camera.

2.1 Double contact control

The desire to keep the contour in the camera field of view, yields a ‘virtual double contact’. The first contact (point A in figure 2b) is between workpiece and force probe, which must move at the specified velocity. The second (virtual) contact (point B in figure 2b) coincides with the intersection of the optical axis and the contour plane.

To maintain this double contact following control actions are taken:

**Force control:** Maintain normal contact force by: \( u_c^y(y) = K_p(y)K_{inv}(y)[F_m^y(y) - F_c^y(y)] \), with \( K_p(y) \) and \( K_{inv}(y) \) the y-direction control gain and tool compliance.

![Figure 1: Used hybrid force/position control scheme. (*)Difference of corresponding pairs only.](image1)

![Figure 2: a) Global Situation, b) Top view of contour for two time instants.](image2)

![Figure 3: Top view for a) Contact variables in task frame, b) Local variables relative to camera.](image3)
**Velocity control:** Move task frame (point $A$) with given tangential velocity $u_1^v(x)$. 

**Force tracking:** Compensate tracking angle error $\Delta P_t(\theta) \equiv u_1^v(y)/u_1^v(x)$ by rotating the task frame. This action does not change the actual position of the end effector.

**Visual servoing:** Rotate end effector around $z_{ec}$-direction by $\omega_1 = \omega_1 + \omega_2$. Component $\omega_1 = -K_p(\theta), \Delta P_{cam}(x)/r_{AB}$ controls point $B$ towards the contour. $r_{AB}$ [mm] is the fixed distance from $A$ to $B$. Component $\omega_2$ moves $B$ tangent to the contour in $B$ with velocity $\dot{v}_B$ (direction known, magnitude unknown) while compensating for the velocity of $A$. (This is in fact a feedforward signal on the position control of $B$). Its value follows from:

\begin{equation}
\Delta r_{AB} \times \omega_2 + \Delta v_A = \dot{v}_B
\end{equation}

According to the notations of figure 4 and neglecting the velocity $u_1(y)$, $\omega_2$ is solved as:

\begin{equation}
\omega_2 \equiv \frac{u_1(x) \sin(\Delta P_{cam}(\theta) - \gamma)}{r_{AB} \cos(\Delta P_{cam}(\theta))}
\end{equation}

with $\gamma$ the angle between $\Delta v_A$ and $v_{cam}$.

**2.2 Matching the vision data**

At this point, we need to determine the link between the actual position of the task frame and the data of the contour, as collected by the vision system. This is done as shown in figure 5: First, the measured tool pose $P_t^m$ and camera pose $P_{cam}^m$ are corrected to compensate for the tool deformation under the current contact forces $F_t^m$. (Mark that the camera frame is calibrated w.r.t. the end effector.) Next, the absolute contour pose is calculated using the image measurement $\Delta P_{cam}$. Offsetting these data points by the tool radius gives the desired (absolute) pose of the task frame $P_t^a$, which is logged in a look-up table. Finally, the calculated predicted pose $\hat{P}_t$ of the task frame for the next time instant is used as a (interpolating) pointer into the look-up table.

**2.3 Calculating the feedforward**

The last step implements the feedforward ($d5$ in figure 1) to reduce/eliminate the tracking angle error $\Delta P_t(\theta)$ using the following control signal:

\begin{equation}
u_1^{ff}(\theta) = u_1^v(x) \cdot \kappa
\end{equation}

This equation uses the curvature $\kappa$ of the contour in the current contact point. An obvious way to calculate $\kappa$ is the use of a fitted contour model. This poses however some problems: for a start, the calculation of the curvature from the fitted contour is very noise sensitive due to the second derivative. Furthermore, the fitted contour may differ from the real contour, especially for simple models or may be computationally intensive for more complicated models. Finally, not the curvature in one point is needed but the mean curvature for the arc length travelled during one time interval. In order to avoid all of the previous mentioned problems, the curvature is calculated as the mean change in orientation of the contour over the travelled arc length $ds$:

\begin{equation}
\kappa = \frac{dP_t(\theta)}{ds}.
\end{equation}

$\kappa$ results from the least square solution of a set of first order equations $P_t(\theta) = \kappa \cdot s + cte$ in about 9 $(P_t(\theta), s)$ pairs, lying symmetrically around the matched position (i.e. position $h$ in figure 5).

The feedforward velocity calculated according to equations 3 and 4, is then added to the feedback control actions described in section 2.1.
Ideal

\[ F_y(t) \text{ versus time} \]

Curves 1, 2, and 3

Figure 7: Top: Measured normal contact forces \( F_y^n(t) \) without and with using vision based feedforward on the tracking direction \( \theta \). Bottom: Actual and ideal feedforward signals.

3 Experiments

Experimental set-up: The experimental set-up consists of a KUKA 361 robot, with an eye-in-hand SONY CCD XC77 camera with 6 mm lens together with a SCHUNK force sensor. Instead of the commercial controller, our own software environment is used, running on a T801 transputer board. The image processing is implemented on a TI-C40 DSP unit, with frame grabber and transputer link. The robot controller and force acquisition is running at 100 Hz. The image processing is limited by the non-interlaced frame rate of 25 Hz. The force probe is about 600 mm. The camera is placed about 300 mm above the workpiece resulting in a camera resolution of about 3 pix/mm.

Implementation of image processing: The image processing algorithm calculates the position \( \Delta P_{\text{cam}} \) of the contour relative to the camera frame in three steps starting from a 256 grey level 128 by 128 image.

First, some local contour points are extracted. Second, a total least squares fit through the contour points determines position \( \Delta P_{\text{cam}}(x) \) and orientation \( \Delta P_{\text{cam}}(\theta) \) of the contour w.r.t. the center of the image. The third step logs the offset contour data in absolute coordinates and calculates the feedforward according to equations 3 and 4 as explained in sections 2.2 and 2.3.

Results: Figure 6 shows the (uncorrected) paths travelled by camera and task frame. The plotted task frame and end effector frame directions illustrate the variable relation between them during task execution. The maximum error between (by vision) logged and (by tool) travelled paths (after corrections) is about 1 mm. This validates the used matching method.

4 Conclusion

This paper shows how the quality of the force controlled planar contour following task improves significantly by adding vision based feedforward on the tracking direction. This reduces tracking errors resulting in faster and/or more accurate task execution. The feedforward signal is calculated from an on-line generated local data set of the contour.

Keeping the contour in the camera field of view, while maintaining a force controlled contact, however, imposes additional requirements on the controller. The task frame orientation is controlled independently from the end effector frame. The shown experimental results validate the used approach.

References


