Trajectory Generation for 1000 fps Direct Visual Servoing*

Roel Pieters, Pieter Jonker and Henk Nijmeijer
Dynamics and Control Group, Department of Mechanical Engineering
Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven
The Netherlands
email:{r.s.pieters, p.p.jonker, h.nijmeijer}@tue.nl

Abstract

This paper presents a trajectory generation method by using a repetitive product pattern as visual encoder for control feedback. A direct, high-speed coupling between camera measurements and local motor control motivates the omission of local motor encoders. By exploiting the repetitive structure of the product and sampling at a high update rate (i.e., 1000 fps), a velocity-constraint polynomial trajectory is designed for image-based velocity feedback. We present algorithms for image processing and trajectory generation with a frame rate of 1000 fps and an image size of 619×75 *pixels.*

1 Introduction

In present day, most visual servoing approaches use visual input (at e.g. video rate) only as reference trajectory for motion control. Stability of the complete system is only guaranteed when using an additional local motor controller or by updating the visual feedback loop fast enough [8]. When regarding this additional local motor controller, high accuracy position (or velocity) measurements are obtained via motor encoders situated within a motor housing. We disregard these encoders and use a camera positioned at the end-effector to obtain high accuracy and high speed (i.e., 1000 fps) position measurements.

In order for this visual servoing method to be successful, a few requirements have to be taken into account. Firstly, a balance has to be found between a high sampling rate and the size of the image to extract useful information. Secondly, since the image effectively becomes the encoder, a repetitive pattern is necessary that acts as reference for motion control.

Positioning can then be effected by controlling a product (i.e., a repetitive pattern feature) towards the centre of the camera's field of view. In this case lens distortions and inaccuracies in internal and external camera parameters have little effect on performance, due to the minimization of the error function (i.e., uncalibrated visual servoing). Since this causes a zero velocity when the feature is in the centre of the field of view, the result will be a stop-and-go positioning.

However, when such trajectory is unwanted, a more complex trajectory generation method has to be employed. For example, when regarding industrial inkjet printing, a constant velocity over each product feature is required for accurate printing. This then involves the design of a polynomial trajectory with setpoints obtained from the repetitive product pattern. Since in this method the complete image is used for control, the calibration of the optical system is necessary.

*This research is supported by Agentschap NL - IOP Precision Technology - Fast Focus On Structures (FFOS).

2 Similar work

Similar work is done by Namiki et al. [7] which describes an eye-to-hand visual servoing application where the joint loop is also directly controlled by the vision sensor. In this, the position between a hand and an object as well as a collision avoidance between grasped object and other objects is controlled directly. This research uses the well-known 1 ms sensory-motor fusion system, which has been developed several years ago by the Ishikawa-Komuro Laboratory [6] and is still a state-of-the-art. In several other projects, this vision system, consisting of two 2 DOF pan-tilt units with a pixel parallel processing array, forwards visual information to a larger robotic manipulator for different visual control tasks (e.g., batting [13]). Other high speed vision systems are e.g., [3] and [4].

These approaches are all focused on visual servoing for industrial robotic manipulators. Research on microscopic imaging and positioning systems is for example by Ogawa et al. which proposes a visual control system for tracking and directing motile cells using a high-speed tracking system [8]. Other, more traditional examples can be found in [1] and [15], a short survey in [5].

Other research involving repetitive patterns is limited to camera calibration techniques or structured, coded light applications. Camera calibration uses a predefined (usually printed) pattern to determine the camera's intrinsic and extrinsic parameters [14]. Structured coded light applications involve a camera-projector system that projects an array of structured light onto objects to identify their shape or depth [10].

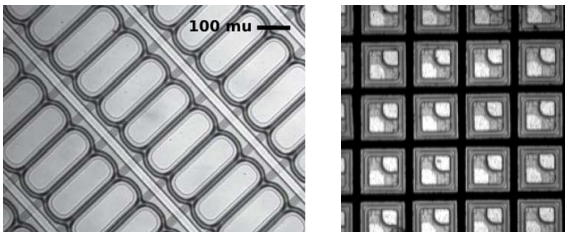
Our method differs from existing visual servoing techniques by exploiting the repetitiveness of the product pattern as visual encoder for 2D planar positioning. In this, only off-the-shelf components are used and a single, direct feedback loop connects camera measurements to a local motor controller. Our work extends from [2] by using complex feature structures and defining a polynomial trajectory for positioning.

3 Direct High-Speed Visual Servoing

In order to compute a trajectory for visual motion control, the initial step is extracting visual encoder points from our repetitive product pattern (for clarity, a single structure in the repetitive product pattern is from now on called a 'feature'). This involves an image processing algorithm for subpixel feature detection and the calibration of the visual system.

3.1 Repetitive Product Pattern

Examples of repetitive patterns are for instance organic LED displays (OLEDs, Fig. 1.a) or semiconduc-



(a) OLED substrate (b) Wafer substrate

Figure 1. Repetitive product patterns. High resolution image of OLED substrate (a). High Resolution image of transistors on a wafer (b).

tors on a wafer substrate (Fig. 1.b). In both cases a positioning task has to align the product head with respect to a repetitive pattern feature and perform a task with micrometer accuracy (typical size of one OLED: $80 \times 220\ \mu m$). In the case of OLED manufacturing, this additional task consists of inkjet printing, in the case of semiconductor manufacturing, a pick-and-place task has to be carried out. Despite the difference in manufacturing, a similar approach towards using the product as encoder can be taken. One approach uses a feature as an image-based visual servoing goal; minimizing the error in image space. A second approach uses multiple features in the field of view to form a trajectory which has to be followed. For both approaches it holds that while moving, new features are added as setpoint or used for trajectory generation.

3.2 Image processing

In order to use the product as encoder, fast image processing extracts individual encoder points from an image. From a greyscale image, the first step is segmentation into a binary format with Otsu's method [9]. Mathematical morphology (contour following) then iteratively locates each feature and defines its position by a bounding box (i.e., x, y and width and height as variables). The final step is to compute the sub-pixel accurate centre location inside each feature as follows:

The two vertical edges of each structure are known from contour following and used to determine a pixel accurate x-position for each of the four points:

$$\begin{aligned} hx[1] &= box.x + 0.2 * box.width \\ hx[2] &= box.x + 0.8 * box.width \end{aligned} \quad (1)$$

This is then the start point to localize the maximum edge gradient position in sub-pixel accuracy in y-direction. With interpolation a local maximum is obtained using five neighbouring points and calculating their gradient norm. Simply said, this is fitting three points (i.e., representing the edge gradient) to a quadratic equation and finding its maximum.

With $L = \{l(j)|j \in \aleph\}$ an infinite line of pixels with a peak at coordinate 0 corresponding to the middle pixel $l(0)$ and ∇ a general derivation operator (symmetric) the gradient norm becomes:

$$\begin{aligned} a &= |\nabla l(-1)| \\ b &= |\nabla l(0)| \\ c &= |\nabla l(1)| \end{aligned} \quad (2)$$

The maximum of the parabola (i.e., the highest slope in intensity) passing through $(-1, a)$, $(0, b)$ and $(1, c)$ is now found by:

$$y_m = \frac{a - c}{2(a - 2b + c)} \quad (3)$$

This maximum y_m is calculated for $hx[x]$ and its direct neighbors, i.e., $hx[x] - 1$ and $hx[x] + 1$, and the average of these three values is then passed as local y-maximum. From the four found sub-pixel accurate points, a line is fit from the two pairs of opposite points on both vertical and horizontal edges. The crossing of these lines determine the final centre coordinates of the feature [11].

The presented centre detection algorithm is designed for rectangular features. When a feature is square shaped (as for transistors on a wafer, to be seen in Fig. 1.b), a similar method can be used to determine each centre location.

3.3 Parametric microscopic camera calibration

When using the complete image as reference and measurement tool, camera calibration is a necessity. Since microscopic imaging systems have a small depth of focus, traditional camera calibration techniques are no longer valid. One technique for calibrating microscopic optical systems with small depth of focus, is presented in [12] and [16]. In this, the camera model is modified for the parallel case and calibration is based on Tsai's two-step method [14].

4 Trajectory generation

When feature points have effectively been extracted from an image, these can then be interpreted to form a velocity-based trajectory. Two methods can be employed for positioning: an image-based visual servoing method and a constant-velocity method. The former means that positioning actuates to individual features (stop-and-go) and can be executed uncalibrated, the latter positions with a fixed, predetermined velocity over each feature-point.

For image-based visual servoing, an error minimization positions the feature in the centre of the image and subsequent feature points are iteratively set as target. The constant velocity method uses multiple feature points from one row (see Fig. 2) to compute a trajectory and is designed as follows:

Determining a trajectory through $n + 1$ points can be solved by means of a polynomial of degree n:

$$q(t) = a_0 + a_1(t) + \ldots + a_n(t^n) \quad (4)$$

in which t represents time.

The polynomial coefficients can be computed by solving a linear system of equations $(n + 1)$: Given the feature points (t_k, q_k) with t_k the k-th time instant, q_k the k-th via-point $(k = 0, \ldots, n)$ and considering additional constraints on the polynomial coefficients regarding initial and final velocities and accelerations, we can build the vectors $\boldsymbol{q}$, $\boldsymbol{a}$, and the so-called Vandermonde matrix $\boldsymbol{T}$ of order $n + m$ (i.e., $n + 1$ points, m additional constraints). Since $t_{k+1} > t_k$,

with $k = 0, \ldots, n-1$, the matrix $\boldsymbol{T}$ is square and invertible, the coefficients a_k can be computed as:

$$\boldsymbol{a} = \boldsymbol{T}^{-1}\boldsymbol{q}. \tag{5}$$

Using a polynomial interpolation method to determine a trajectory has the advantage that all points n are crossed and that the trajectory is smooth. A drawback is the computational effort needed and the fact that for large values of n numerical errors may occur. Since the goal is to position along individual features on a single row, the control (and thus the trajectory generation) for x- and y-direction is partitioned. In one direction, a trajectory is created for multiple feature points (e.g., $n > 5$, pitch > 100 μm). For the other direction, a similar number of points computes a trajectory that deviates only a few percent (i.e., pitch < 5 μm).

5 Experimental results

In this section experimental results are given to validate the use of a repetitive product pattern and trajectory generation for control purposes.

5.1 Experimental setup

The camera and lens used for experiments are standard of-the-shelf industrial components: a Prosilica GE680 camera with a frame rate of 200 fps full frame (640×480 pixels) combined with a standard 1.5x magnifying lens results in a pixel size of 3.08 μm. The camera is connected via a Gigabit Ethernet interface (GigE Vision) to a standard notebook with 2 GB of RAM and 2.4 GHz Intel Core 2 Duo CPU.

Coaxial lighting is applied which has the advantage that the light that enters the camera sensor is reflected mainly from axial illumination. This is due to the use of a beamsplitter which directs light from a power LED source downwards onto the OLED substrate which subsequently is reflected up into the camera.

The platform to be controlled is a planar xy-table (2 DOF) actuated by 2 linear motors. The camera and the lighting itself are fixed. A controlled height and orientation motion is not (yet) present.

The maximum image size for data transfer at 1 kHz using GigE is 75 lines, disregard of image height. When using the full effective range (due to lighting) of the sensor, the image resolution is 619×75 pixels in which 2×8 full features (OLEDs, see Fig. 2) are located.

5.2 Feature detection

Figure 3 shows the output of the image processing algorithm, figure 2 shows the actual image from which a trajectory is generated and table 1 shows the accuracy of the feature detection method.

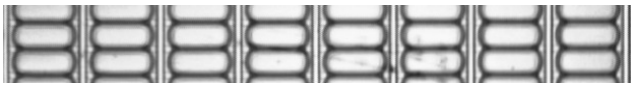


Figure 2. Cropped OLED image. The image size (619×75) allows for a frame rate of 1 kHz using Gigabit Ethernet (GigE) communication.

(a) Thresholded image (b) Detected pattern

Figure 3. Output of centre detection algorithm. Fig. 'a' shows the output after thresholding with Otsu's method. Fig. 'b' shows the found OLEDs outlined with a rectangle. On horizontal lines the points are shown where the optimal vertical edges are detected.

Table 1. Detection algorithm results

	Centre detection
Accuracy	0.2 [px]
	0.62 [μm]
Reproducibility (3σ)	0.36 ± 0.06 [px]
	1.11 ± 0.18 [μm]
Timing (per feature)	0.045 [ms]

On average, nine features take 0.40 ms of processing time. Equivalently, 16 features need 0.71 ms of processing time, which is within timing limits. The network load for transferring these images at 1 kHz is roughly 50 MB/s.

The algorithm to detect feature centres and generate the trajectory in real-time is controlled by the timer of the camera, which is more stable (i.e., less jitter) than a standard linux (or Xenomai) timer.

5.3 Visual trajectory generation

Using a polynomial to form a trajectory has the advantage that a velocity reference trajectory can be constructed by simple differentiation. The design of such trajectory involves choices in number of points and the constraints on velocity and acceleration. To clarify this, a comparison is made between several simulated trajectories.

In figure 4 it is shown that in order to anticipate to future feature positions it is more beneficial to use more points to construct a trajectory. The figure shows a trajectory generation for one 7th order polynomial with 4 points and three 3rd order polynomials with 2 points. The polynomials visit the same feature point locations (at $t = 0, 1, \ldots, 3$ seconds, relative standard deviation $\sim$11%) and have equal constraints on start- and end-point (i.e., fixed velocity and no acceleration constraint). The difference is due to the in-between feature velocity and acceleration constraints and the fact that the higher order polynomial incorporates more points to generate a trajectory. This results in a more smooth velocity profile (and thus lower acceleration) for higher order, multipoint polynomial trajectories. The feature velocity on feature points for both trajectories, however, is the same.

Figure 5 shows that constraints on velocity and acceleration on feature points have to be chosen carefully when designing a polynomial trajectory. When a constraint on acceleration is incorporated, this generally results in a very oscillating trajectory (for position, velocity and acceleration). A constraint on only velocity

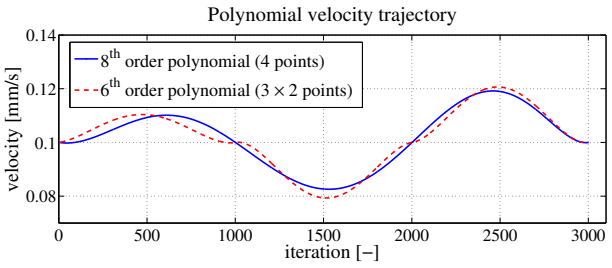


Figure 4. Trajectory point comparison. The solid line depicts three velocity trajectories sequentially, each computed from 2 points. The dashed line depicts a velocity trajectory computed from the same 4 points. Due to the extra constraints on velocity and acceleration and the amount of trajectory points, the subsequent 2-point trajectories are less smooth.

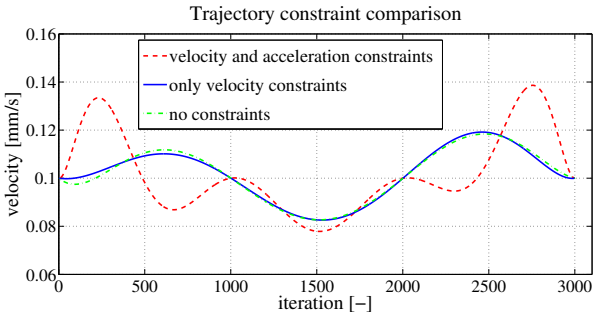


Figure 5. Polynomial constraint comparison. The dashed (red) line shows that with acceleration and velocity constraints on feature points, an overall oscillating velocity signal is computed. The solid (blue) line shows that only constraints on velocity give the most moderate velocity trajectory.

gives the most smooth or moderate trajectory and is therefore preferable. Also here it holds that the feature velocity on feature points for all trajectories is equal.

For the high-speed visual servoing task, the polynomial trajectory is computed by using eight points ($n + 1 = 8$) with an additional velocity constraint on each point ($m = 8$), generating a 15-th degree polynomial. For validation on an experimental setup, (reference) velocity measurements will be taken from visual data.

6 Conclusions

We have presented a framework for using a repetitive product pattern as intelligent reference tool. In this, the low-level motor encoders are disregarded and direct, fast (i.e., 1 kHz) feedback is obtained between image (619×75 pixels) and motor. A method for detecting simple, repetitive rectangular structures is presented, which reaches sub-micron accuracy. Positioning can then be executed in an image-based way (stop-and-go) or by generating a polynomial trajectory. A polynomial trajectory is computed by using a sequence of feature points and adding additional constraints on velocity and acceleration on feature point centres. We show that using more feature points results in a more smooth trajectory and that constraints on velocity and acceleration have to be chosen carefully when designing a polynomial trajectory.

Future work focusses on miniaturization of the visual control system by implementing image processing, trajectory generation and control on FPGA, which will reduce delay (direct connection between sensor and processing unit) and computation time.

References

[1] H. Bilen et al., "A comparative study of conventional visual servoing schemes in microsystem applications", in IEEE IROS, pp. 1308-1313, 2007.

[2] J.J.T.H. de Best et al., "Direct Dynamic Visual Servoing at 1 kHz by Using the the Product as One Dimensional Encoder," 7th IEEE International Conference on control and Automation, pp 361-366, 2009.

[3] R. Ginhoux et al., "Beating heart tracking in robotic surgery using 500 Hz visual servoing, model predictive control and an adaptive observer", in IEEE Int. Conf. on Robotics and Automation, pp. 274-279, 2004.

[4] C. F. Graetzel et al., "A 6000 Hz computer vision system for real-time wing beat analysis of Drosophila", in IEEE / RASEMBS Int. Conf. on Biomedical Robotics and Biomechatronics (BioRob), pp. 1-6, 2006.

[5] P. Kallio et al., "Control Issues in Micromanipulation", in Int. Symp. on Micromechatronics and Human Science, pp. 135-141, 1998.

[6] Y. Nakabo et al., "1 ms Column Parallel Vision System and It's Application of High Speed Target Tracking", in IEEE ICRA, vol. 1, pp 650-655, 2000.

[7] A. Namaki et al., "A Hierarchical Control Architecture for High-Speed Visual Servoing", in Int. Journal of Robotics Research, vol. 20(10-11), pp. 873-888, 2003.

[8] N. Ogawa et al., "Microrobotic Visual Control of Motile Cells Using High-Speed Tracking System", in Robotics and Automation, Trans., 21(4): pp. 704-712, 2005.

[9] N. Otsu, "A Threshold Selection Method from Gray-level Histograms," in IEEE Transactions on Systems, Man and Cybernetics, vol. 9 (1), pp. 62-66, 1979.

[10] J. Pagès et al., "A camera-projector system for robot positioning by visual servoing", in conf. on Computer Vision and Pattern Recognition Workshop, 2006.

[11] R.S. Pieters et al., "High Performance Visual servoing for controlled micrometer positioning", in IEEE WCICA, pp. 379-384, 2010.

[12] R.S. Pieters et al., "Product Pattern based Camera Calibration for microrobotics", in IEEE IVCNZ, 2010 (accepted, in process).

[13] T. Senoo et al., "High-Speed Batting Using a Multi-Jointed Manipulator", in Robotics and Automation, Proc. IEEE Int. Conf., pp. 1191-1196, 2004.

[14] R.Y. Tsai, "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses", in IEEE J. of Rob. and Autom., 3, pp. 323-344, 1987.

[15] B. Vikramaditya, B. J. Nelson, "Visually Guided Microassembly Using Optical Microscopes and Active Vision Techniques", in IEEE ICRA, pp. 3172-3177, 1997.

[16] Y. Zhou, B.J. Nelson, "Calibration of a Parametric Model of a Microscope", in Optical Engineering., 38, pp. 1989-1995, 1999.