# Direct dynamic visual servoing at 1 kHz by using the product as 1.5D encoder

# Direct Dynamic Visual Servoing at 1 kHz by using the Product as 1.5D Encoder

J.J.T.H. de Best, M.J.G. van de Molengraft and M. Steinbuch

***Abstract*— This paper focusses on direct dynamic visual servoing at high sampling rates for machines used for the production of products that inherently consist of equal features placed in a repetitive pattern. A mechanical system is controlled on the basis of vision *only*. In contrast to kinematic visual servoing approaches, we do not use a hierarchical control structure. More specifically, the motor inputs are driven *directly* by the vision controller without the intervention of low level joint controllers. The product in view consists of a repetitive pattern, which is used as a 1.5D encoder purely on the basis of vision. Using fast image processing and a prediction based on a steady-state Kalman filter a 1 kHz direct visual servoing setup is created capable of using the repetitive pattern as a 1.5D encoder with an accuracy of 2 $\mu$m. The design is validated on an experimental setup.**

## I. INTRODUCTION

Many production processes take place on repetitive structures, for example in inkjet printing technology where droplets are placed in a repetitive pattern, or in pick and place machines used in the production of solar cell arrays or in LCD production. In each of these processes one or more consecutive steps are carried out on the repetitive structure to create the final product. Such production machines consist of a tool, for example a printhead, and a table or carrier on which the repetitive structure is to be produced. Key in obtaining a high product quality is to position the tool with respect to the object with a high accuracy. Within many production machines the position of the tool and the position of the object are measured separately as shown in Fig. 1(a). Often the absolute reference points of these measurements do not coincide, when for example several frame parts are in between the two. This is referred to as an *indirect* measurement. One can calibrate the offsets of these reference points with which the relative position between the tool and the object can be derived, assuming the frame parts are rigid. However, each frame part has a limited stiffness resulting in vibrations when forces act on it, which renders the relative position measurement to be incorrect. Secondly, due to thermal expansion the size of the frame part changes, which again affects the relative position of the tool with respect to the object. Thirdly, often the position of the table or carrier can be measured but the position of the object with respect to the table can not be measured.

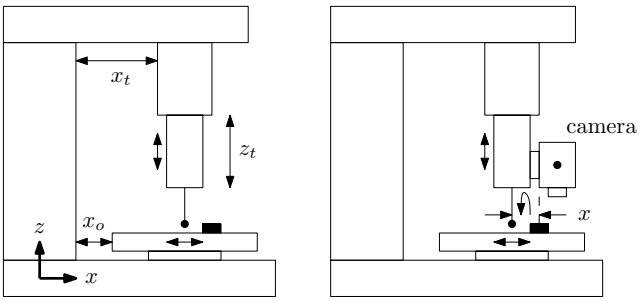
This work was supported by SenterNovem - Innovatiegerichte Onderzoeksprogramma's IOP.

J.J.T.H. de Best, M.J.G. van de Molengraft and M. Steinbuch are with the Eindhoven University of Technology, Department of Mechanical Engineering, Control Systems Technology Group, P.O. Box 513, 5600 MB, Eindhoven, The Netherlands, j.j.t.h.d.best@tue.nl, m.j.g.v.d.molengraft@tue.nl, m.steinbuch@tue.nl

To overcome these problems it is desirable to *directly* measure the relative position between the tool and the object. This can be realized by using a camera as sensor as shown in Fig. 1(b). Controlling a mechanical system by means of camera measurements is referred to as visual servoing [5], [7], [12].

Visual servoing has many classifications [15], [18]. The most familiar ones are 2D or image based visual servoing (IBVS) and 3D position based visual servoing (PBVS). In IBVS, the control actions are calculated on the basis of images directly, whereas in PBVS control actions are taken on the basis of cartesian measurements. The latter therefore includes a pose estimation. Furthermore, a less known classification is visual kinematic control versus visual dynamic control [4]. Most visual servoing literature focusses on kinematic visual control in which joint controllers are used to track the velocities that are calculated by a high level vision controller [1], [5], [7]. In the control design of the high level vision controller the dynamics of the underlying low level closed-loop joint control loops are often discarded. Furthermore, all joints are assumed to be rigid. Hence, a kinematic model of the system is adopted. Inappropriate tuning of the vision control loop might cause instabilities when ignoring the dynamics of the system. On the other hand, visual dynamic control takes into account the dynamics of the system [2], [3], [4], [16], [17] and does not rely solely on the kinematic model. Most of the existing schemes are still indirect, i.e. a hierarchical architecture containing low level velocity controllers and a high level vision controller. In direct visual servoing, the inner velocity control loops are absent, such that the total dynamics are visible to the vision-based controller.

In this paper a direct dynamic visual servoing design is




(a) *Indirect* measurement loop. (b) *Direct* measurement loop.

Fig. 1. Direct versus indirect measurement.

presented, capable of sampling at 1 kHz without the need for massive parallel processing as in [8], [9], [13] but instead using a commercially available, affordable camera. Two main differences between our approach and existing literature are:

1) we fully account for the machine frame dynamics in the control design and
2) we fully account for the machine driveline dynamics, so we drive the motors directly without the use of low level motor velocity controllers.

We compare our approach to a control scheme using classical measurement devices to show that the sensor placement results in different observed dynamics, which can jeopardize the attainable accuracy of the closed-loop system.

Furthermore, in the production of repetitive structures that contain a single type of feature, the tool is to be positioned with respect to these features. Therefore, the repetitive structure itself can be used as an encoder grid where the features represent the increments. However, when using vision the resolution is not restricted to that; when having two features within the field of view, a linear interpolation can be implemented to increase the resolution. As opposed to [6] we will not create an absolute encoder but an incremental encoder using a camera in combination with a one dimensional repetitive pattern, since we are only interested in relative positions. The position range perpendicular to this direction is limited by the field of view of the camera. Therefore, we call it a 1.5D encoder. This feature-based incremental encoder signal will be used as the feedback signal in the closed-loop control setup. So the main contribution of this paper is twofold:

1) a direct vision-based, dynamics aware position control will be designed together with a
2) vision-based repetitive structure incremental 1.5D encoder

Both aspects will be demonstrated with an experimental visual servoing setup.

In Section II the measurement principle to create a 1.5D encoder using the repetitive structure in combination with a camera will be given followed in Section III by the design of a model-based predictor. For combining the results of Section II and III a correction step is needed, as will be discussed in Section IV. The image processing algorithm will be discussed in Section V. The experimental setup used for validation of the proposed algorithm will be described in Section VI, the system identification in Section VII and in Section VIII, we will discuss the total integration followed by the experimental validation that is implemented in a closed-loop visual servoing control setting. Finally, conclusions and future work will be given.

## II. MEASUREMENT PRINCIPLE

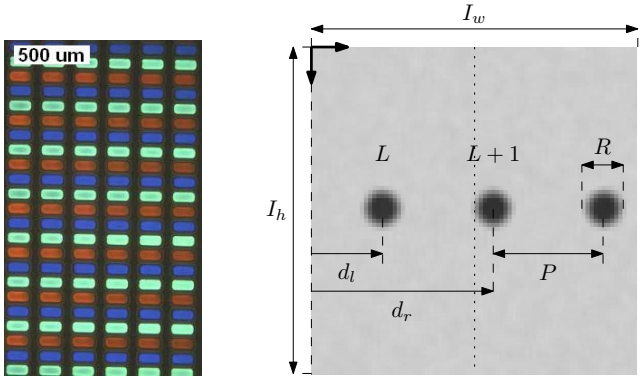
Within this research we focus on machines used for the production of structures that inherently consist of identical features placed in a repetitive pattern like OLED displays, see Fig.2(a). At this point we restrict the focus of the paper to a one dimensional repetitive structure for ease of explanation. In many manufacturing machines, production steps are carried out row by row or column by column, so in practice we need a two dimensional position measurement. In our case the second dimension is however restricted by the field of view. The focus in this paper will be on the position measurement along the repetitive structure. For now we will consider the features to be circular objects as shown in Fig. 2(b), with a diameter of $R$ pixels. The height and width of the image captured by the camera are $I_h$ and $I_w$ pixels, respectively, whereas the repetitiveness is characterized by the pitch between the features which is denoted by $P$. The number of features within the field of view for the presented method must be at least two. Within the image the horizontal pixel positions $d_l$ and $d_r$ of those two features that are located most near the image center are measured, see Fig. 2(b). These features are labeled $L$ and $L+1$, with $L \in \mathbb{Z}$, *irrespective* of the mutual distance. The measured position $y_v$ that will be used in the closed-loop visual control setting is now given by

$$y_v(k) = y_c(k) + y_f(k), \tag{1}$$

with $y_c$ being the coarse position, i.e. the integer feature label $L$ and the fine position $y_f$ which is the linear interpolation between the left and right feature label and is calculated as

$$y_f(k) = \frac{0.5I_w - d_l(k)}{d_r(k) - d_l(k)} \leq 1. \tag{2}$$

The output $y_v(k)$ indicates which feature label is in the center of the image and is measured in a sub-feature label sense. So, $y_v(k) = 1.0$ indicates that the feature labeled 1 is exactly in the center of the image, whereas $y_v(k) = 0.5$ indicates that the center of the image is exactly between the features with labels 0 and 1. Comparing this approach with a classical incremental encoder an important difference is that with this method we can interpolate the position between increments, whereas classical incremental encoders increment only when transitions are detected. Furthermore, in classical incremental encoders the pitch is assumed to be known and static because the position output is linear




(a) OLED display: a repetitive structure.

(b) One dimensional repetitive pattern.

Fig. 2. Repetitive structures.

dependent on that. This means for example that temperature effects and encoder imperfections are not taken into account. Since we are only interested in the position of the feature with respect to the center of the image we have created a *relative incremental* encoder. Note, that deviations in the pitch $P$ cause this measurement to be piecewise linear, i.e. the gain of the process varies along the structure. For now we will assume small deviations, i.e. $d_r(k) - d_l(k) \approx P$, such that linear control techniques can still be applied. The advantage of this method is that operators will be able to use feature-based positions instead of cartesian positions. Furthermore, the cumulative sum of the deviations of all pitches does not affect the new position measurement.

## III. MODEL-BASED PREDICTION

Key in obtaining the correct position is to determine the value of $L$ within the field of view. When for example the velocity is one pitch per sample the camera will record identical images every time step. Based on that information only, the measurement $y_v$ as described in the previous section gives the same value if $L$ is not incremented, i.e. we measure a velocity of zero while the structure is moving at the high velocity of one pitch per sample. If the velocity is even increased further aliasing effects cause the features to visually move slowly in the *wrong* direction. To tackle the problem of incrementing the value of $L$, a model-based solution will be applied. More specifically, we will design a steady-state Kalman filter [11], from which the one step ahead prediction will be used to estimate the value of $L$ in the next time step. Therefore, we will model the input-output behavior of the motion drive carrying the repetitive structure. The state space representation of the discrete time system is given by

$$\underline{x}(k+1) = A\underline{x}(k) + Bu + \underline{w}(k) \tag{3}$$

$$y(k) = C\underline{x}(k) \tag{4}$$

where $\underline{x}$ is the state vector, $u$ is the known force input and $\underline{w}$ is the process noise. The true position output of the system is denoted by $y$. The matrices $A$, $B$ and $C$ are the system, input and output matrices, respectively, whereas the time step is given by $k$. The exact matrices for our case will be given in Section VII. In this section a steady-state Kalman filter will be given that estimates the output $y$ given the input $u$ and the measurement $y_v$ given by

$$y_v(k) = C\underline{x}(k) + v(k), \tag{5}$$

where $v$ represents the measurement noise. For the process and measurement noise we assume

$$E(\underline{w}\underline{w}^T) = Q_w,\ E(vv^T) = Q_v,\ E(\underline{w}v^T) = 0. \tag{6}$$

The steady-state Kalman filter consists of a time update

$$\underline{\hat{x}}(k+1|k) = A\underline{\hat{x}}(k|k) + Bu(k), \tag{7}$$

and a measurement update

$$\underline{\hat{x}}(k|k) = \underline{\hat{x}}(k|k-1) + M(y_v(k) - C\underline{\hat{x}}(k|k-1)), \tag{8}$$

which combined lead to

$$\underline{\hat{x}}(k+1|k) = A(I-MC)\underline{\hat{x}}(k|k-1) + Bu(k) + AMy_v(k) \tag{9}$$

$$\hat{y}(k|k) = C(I-MC)\underline{\hat{x}}(k|k-1) + CMy_v(k). \tag{10}$$

The one step ahead prediction is given by

$$\hat{y}(k+1|k) = C\underline{\hat{x}}(k+1|k), \tag{11}$$

where, $\hat{y}(k+1|k)$ represent the estimate $y(k+1)$ on the basis of measurements up to time step $k$. This prediction is used to get an estimate $\hat{y}_c$ of the position of the repetitive structure in the next time step $k+1$:

$$\hat{y}_c(k+1|k) = \lfloor \hat{y}(k+1|k) \rfloor, \tag{12}$$

where $\lfloor . \rfloor$ is the floor function.

## IV. CORRECTION STEP

When the position of the feature is located around the center of the image it is hard to precisely detect whether it will be on the left side or on the right side of the image center. Therefore a correction step might be needed to correct the value of $y_c$. In cases the feature is located at the same side as it was estimated with respect to the center of the image, no correction is needed. However, if the feature was estimated on the left but is measured at the right a correction step $y_c(k) = \hat{y}_c(k) + 1$ is needed. Vice versa, a correction of $y_c(k) = \hat{y}_c(k) - 1$ is needed.

## V. FAST IMAGE PROCESSING IMPLEMENTATION

This section discusses the image processing algorithm used for detecting the pixel positions $d_l$ and $d_r$. At this point we will introduce search areas around each of the features within the field of view with a width $S_w$ and height $S_h$ such that only one feature is present within one search area as shown in Fig. 3. The goal is to search for only one feature within one search area such that labeling implementations to distinguish between multiple features in the image processing algorithms, which cause overhead, can be eliminated. Furthermore, we introduce $\hat{d}$, which is a pixel position estimate of the feature that is closest to the image center. By using a better prediction the search area can be reduced, which in turn again leads to a smaller computation time of the image processing. Obviously, the size of the search area depends on i) the feature size $R$ ii) the variation of the feature position and iii) the quality of the prediction $\hat{d}$. Naturally, this size should be larger if i) the feature size is large, ii) the variation of the feature position is large and iii) the prediction quality is low. The size of the feature is fixed and the variation of the position is a characteristic of the machine, so they cannot be altered. However, the estimate $\hat{d}$ can be influenced. Note that the pixel position estimation $\hat{d}$ can be obtained from the one step ahead prediction discussed in the Section III as follows

$$\hat{d}(k+1|k) = \begin{cases} 0.5I_w + (1 - (\hat{y}(k+1|k) - \hat{y}_c(k+1|k))P \\ \qquad \text{if } \hat{y}(k+1|k) - \hat{y}_c(k+1|k) \geq 0.5 \\ 0.5I_w - (\hat{y}(k+1|k) - \hat{y}_c(k+1|k)P \\ \qquad \text{if } \hat{y}(k+1|k) - \hat{y}_c(k+1|k) < 0.5 \end{cases} \tag{13}$$

Given this estimate together with the search area, the position of the feature within the search area is calculated. This is done as follows.

First, the image is thresholded within the search area with a static threshold level $TH$

$$T(i,j,k) = \begin{cases} TH - I(i,j,k) & \text{if } I(i,j,k) \leq TH \\ 0 & \text{if } I(i,j,k) > TH \end{cases} \tag{14}$$

Here, the image data is denoted by a function of the form $I(i,j,k)$, with indices $i \in \{s_t, \ldots, s_t + S_h\}$, $j \in \{s_l(k), \ldots, s_l(k) + S_w\}$ indicating the pixel elements and $k$ indicating the time step. The position $(s_t, s_l(k))$ indicates the top left corner of the search area, see Fig. 3. This position is given by $s_l(k) = \hat{d}(k) - 0.5S_w$ and $s_t = 0.5(I_h - S_h)$. Therefore, we assume that the vertical positions of the features only vary within $S_h - R$ with respect to the vertical center of the image. As a result, we can also measure the vertical position within a limited range. This position can be used in the feedback loop to keep the features within the field of view. However in the remainder we will focuss on the horizontal position measurement. The resulting thresholded image is given by $T(i,j,k)$.

Secondly, the horizontal center of gravity within the search area of the thresholded image $T(i,j,k)$ is calculated as

$$d(k) = \frac{\sum_{i=S_t}^{S_t+S_h} \sum_{j=S_l(k)}^{S_l(k)+S_w} iT(i,j,k)}{\sum_{i=S_t}^{S_t+S_h} \sum_{j=S_l(k)}^{S_l(k)+S_h} T(i,j,k)}. \tag{15}$$

If $d(k) \geq 0.5I_w$ we have found the feature at the right of the center of the image and we call this distance $d_r(k) = d(k)$. From Fig. 3 it can be seen that $d_r(k)$ is slightly different from $\hat{d}(k)$ indicating the estimation error. Next, if the feature is found at the right of the center of the image, another feature is searched at the left of the image center with an estimate given by $\hat{d}_l(k) = d_r(k) - P$. Vice versa, if $d(k)$ was found to satisfy $d(k) < 0.5I_w$ we have found the left feature with position $d_l(k) = d(k)$ and we would search for the right feature with an estimate given by $\hat{d}_r(k) = d_l(k) + P$. We end up having two sub pixel positions $d_r(k)$ and $d_l(k)$.

## VI. EXPERIMENTAL SETUP

The setup that is used for experimental validation is depicted in Fig. 4. It consists of two stacked linear motors forming an xy-table. The data-acquisition is realized using an EtherCAT [10] data-acquisition system, where DAC, I/O, and ADC modules are installed to drive the current amplifiers of the motors, to enable amplifiers and to measure the position of the xy-table. Hence, this position is only used for comparison and is *not* used in the final control algorithm as such. A Prosilica GC640M high-performance machine vision camera [14] with Gigabit Ethernet interface (GigE Vision) capable of reaching a frame rate of 197 Hz full frame (near VGA, 659×493) is mounted above the table. The GigE interface allows for very fast frame rates and long cable lengths. The captured images are monochrome images with 8 bit intensity values. To obtain a frame rate of 1 kHz we make use of a region of interest (ROI); we readout only a part of the image sensor as large as 80×80 pixels. The used objective is a Fuijinon DF6HA-1 lens, with a focal length $f$ of 6 mm. With a height $h$ of 11 cm between the camera and the table and a pixel size $p$ of 9.9 $\mu$m we can calculate the resulting field of view according to the pinhole camera model as

$$\frac{I_w ph}{f} \times \frac{I_h ph}{f}, \tag{16}$$

which in this case is $14.5 \times 14.5$ mm. The exposure time is set to its minimum, which is 10 $\mu s$. The illumination is realized using power LEDs. The data-acquisition is integrated in a Linux environment running a 2.6.28.3 preemptible low-latency kernel and the real-time executable is build using the real-time workshop (RTW) of Matlab/Simulink. The repetitive structure consists of circular black dots with a radius of 1 mm and a pitch of 4 mm.

## VII. SYSTEM IDENTIFICATION

For the horizontal direction frequency response functions (FRFs) are measured. Two kinds of FRFs are measured: one

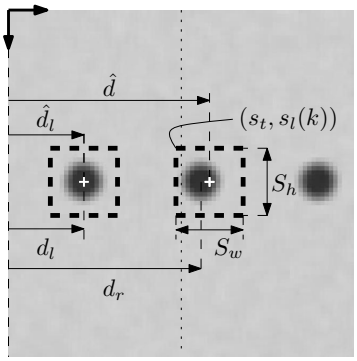



Fig. 3. Measurements of $d_r$ and $d_l$ using the search areas.

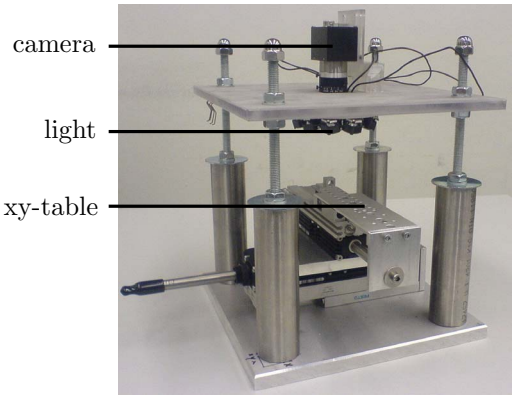



Fig. 4. Experimental visual servoing setup.

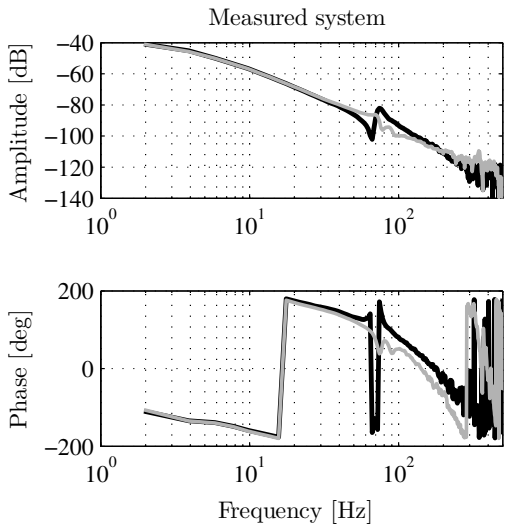



Fig. 5. Measured FRFs from motor input $u$ to position output $y$, **black**: using motor encoder, **gray**: using camera.

from motor input $u$ to position output $y_{mot}$ using the motor encoder and one from motor input $u$ to position output $y_{cam}$ using the camera with the position measurement as described in the previous sections. In the ideal case, bot $y_{mot}$ and $y_{cam}$ would represent a measurement of the product position $y$. The result is given in Fig. 5. Different dynamics are present if the position measurement from the camera is used instead of the motor encoder. In case of using the camera as sensor all relevant dynamics are measured including vibrations caused by the limited stiffness of the frame. Furthermore, it can be seen that different time delays are present when using the camera in the feedback instead of the motor encoder. The time delay when using the camera is larger, due to the necessary image acquisition and image processing time. When the camera is used the time delay is 3.5 ms.

For frequencies below 50 Hz the two FRFs are quite similar and can be approximated by a second order integrator with delay given in a discrete time model as (3) and (4) with

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, B = \begin{pmatrix} 0 \\ 0 \\ 0 \\ T^2/2m \\ T/m \end{pmatrix}, \quad (17)$$
$$C = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \end{pmatrix},$$

where $m$ is the mass (including all motor and amplifier gains) of the system and $T = 0.001$ is the sampling time. The mass $m$ is estimated as 0.184 using the measured FRFs.

## VIII. INTEGRATION

The integration of all blocks within the control scheme is depicted in Fig. 6. The steady-state Kalman filter is designed using the matrices $A$, $B$ and $C$ given in the previous section

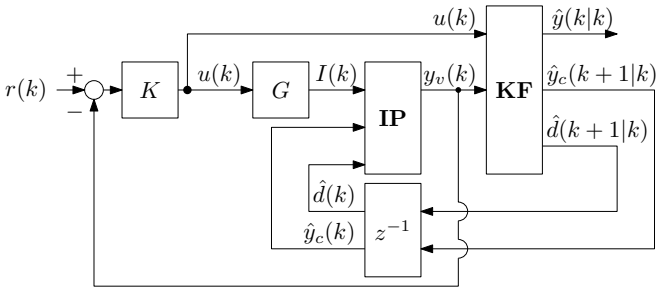



Fig. 6. Control scheme.

together with

$$Q_w = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & q_w \end{pmatrix}, Q_v = q_r, \quad (18)$$

with $q_r = 9.6 \cdot 10^{-11}$, which is the variation of the measurement $y_v(k)$. The value $q_w$ is used as a tuning variable, under the assumption that there is no uncertainty in the delay of the system. The value is determined to be $1 \cdot 10^{-5}$. In this scheme the controller $K$ is connected to the system $G$ with motor input $u$ and image output $I$. This image is processed in the image processing block **IP** using the estimates $\hat{y}_c(k)$ and $\hat{d}(k)$. These estimates are the previous outputs of the Kalman filter **KF**. Note, that the Kalman filter is used only for i) incrementing the value of $L$ and ii) estimating the position of the feature closest to the image center in the next time step. The filtered position output of the Kalman filter $\hat{y}(k|k)$ is *not* used for feedback, but only to track the measured features over time. It is chosen not to use $y(k|k)$ since in that case the dynamics of the Kalman filter attribute to the system dynamics.

## IX. EXPERIMENTS AND RESULTS

Two experiments have been carried out. In the first experiment a reference with a constant velocity of 0.1 m/s is applied with a final position of 0.04 m, which is well outside the field of view. In the second experiment the reference to be tracked is a sine wave with an amplitude of 0.015 m and a frequency of 2 Hz. The outputs $y_v(k)$ are given in the top figures of Fig. 7 and Fig. 8. During these experiments a one step ahead prediction of the output $\hat{y}_v(k+1)$ is calculated as $\hat{y}_v(k+1) = \hat{y}(k+1|k)$, i.e. using the prediction given in Section III. In the lower figures of Fig. 7 and Fig. 8 this estimate is compared to the real value of $y_v(k+1)$ and this prediction error is depicted in gray. The black curve shows the prediction error when the prediction is taken as the current position, i.e. $\hat{y}_v(k+1) = y_v(k)$. In both figures it can be seen that the prediction error using $\hat{y}_v(k+1) = y_v(k)$ is much larger than using the prediction $\hat{y}_v(k+1) = \hat{y}(k+1|k)$. Using the pitch of 4 mm, the prediction error can be calculated and is less than $\pm 10$ $\mu$m. The quality of the position measurement is characterized by

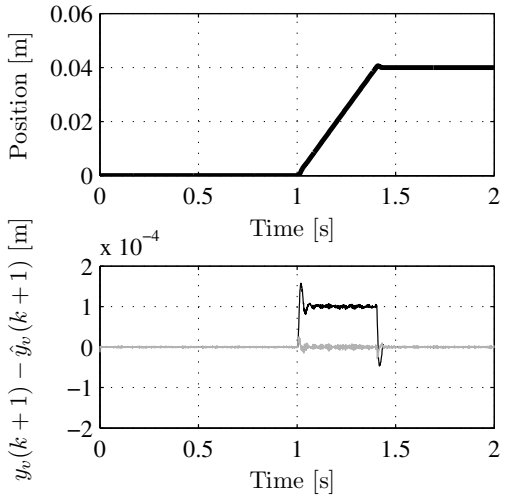



Fig. 7. Prediction results for ramp reference, **black**: $y_v(k+1) - y_v(k)$, **gray**: $y_v(k+1) - \hat{y}(k+1|k)$.

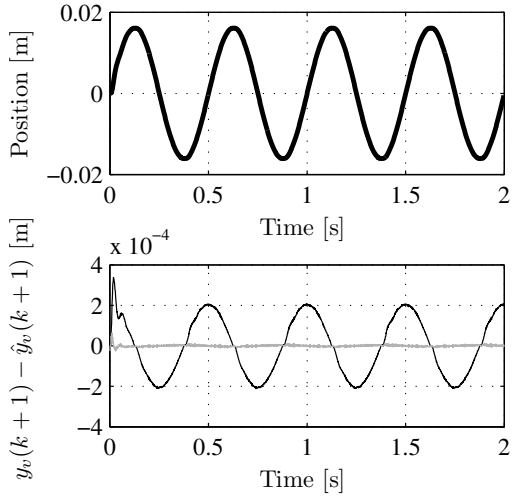



Fig. 8. Prediction results for sine reference, **black**: $y_v(k+1) - y_v(k)$, **gray**: $y_v(k+1) - \hat{y}(k+1|k)$.

a $3\sigma$ bound of 2 $\mu$m, with $\sigma$ the standard deviation of the measurement.

## X. CONCLUSIONS AND FUTURE WORK

In this paper a direct dynamic visual servoing setup is created that controls a motion system with 1 kHz visual feedback, without the intervention of low level joint controllers. Different dynamics have been identified when using the motor encoder or measurements of the camera. By using the camera all the relevant dynamics are measured directly. In the control design it is possible to account for these dynamics. Secondly, an algorithm is developed that uses the repetitive pattern to create a 1.5D incremental optical encoder with an accuracy of 2 $\mu$m and capable of sampling at 1 kHz in combination with velocities up to 0.2 m/s. The sample rate of 1 kHz is realized by reading out a part of the vision sensor to reduce the data flow. In the image processing steps prediction techniques are used to further reduce the amount of data to be analyzed. The advantage of the proposed method is that feature-based positions can be used instead of cartesian positions, that drift due to the cumulative sum of the deviations of the pitches.

Future work will concentrate on expanding the given methods to more dimensions by including more translations and rotations. Furthermore, an investigation is planned to get insight on the robustness of the proposed method if the pitch is uncertain, i.e. $P = \overline{P} + \delta P$, with $\overline{P}$ the mean pitch and $\delta P$ the variation of the pitch.

## REFERENCES


[1] F. Chaumette and S. Hutchinson. Visual servo control part 1: Basic approaches. *Robotics and Automation Magazine, IEEE*, 13(4): pp 82–90, 2006.

[2] P.I. Corke. Dynamic issues in robot visual-servo systems. In *Robotics Research, Int. Symp.*, pp 488–498, 1995.

[3] P.I. Corke and M.C. Good. Dynamic effects in high-performance visual servoing. In *Robotics and Automation, Proc. IEEE Int. Conf.*, volume 2, pp 1838–1843, 1992.

[4] P.I. Corke and M.C. Good. Dynamic effects in visual closed-loop systems. *Robotics and Automation, IEEE Trans.*, 12(5): pp 671–683, 1996.

[5] K. Hashimoto. A review on vision-based control of robot manipulators. *Advanced Robotics*, 17(10): pp 969–991, December 2003.

[6] B. Hassler and M Nolan. Using a C.C.D. to make a high accuracy absolute linear position encoder. In *Proc. SPIE Optoelectric Devices and Applications*, volume 1338, pp 231–240, 1990.

[7] S. Hutchinson, G.D. Hager, and P.I. Corke. A tutorial on visual servo control. *Robotics and Automation, IEEE Trans.*, 12(5): pp 651–670, 1996.

[8] I Ishii, Y Nakabo, and M Ishikawa. Target tracking algorithm for 1ms visual feedback system using massively parallel processing. In *Robotics and Automation, Proc. IEEE Int. Conf.*, pp 2309–2314, April 1996.

[9] M Ishikawa, A. Morita, and N. Takayanagi. High speed vision system using massively parallel processing. In *Intelligent Robotics and Systems, Proc. IEEE Int. Conf.*, pp 373–377, July 1992.

[10] D Jansen and H. Buttner. Real-time ethernet the ethercat solution. *Computing and Control Engineering, IEEE Jour.*, 15(1): pp 16–21, 2004.

[11] R.E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D): pp 35–45, 1960.

[12] E. Malis. Survey of vision-based robot control. In *European Naval Ship Design, Captain Computer IV Forum*, 2002.

[13] Y. Nakabo, M. Ishikawa, H. Toyoda, and S. Mizuno. 1 ms column parallel vision system and its application of high speed target tracking. In *Robotics and Automation, Proc. IEEE Int. Conf.*, volume 1, pp 650–655, 2000.

[14] Prosilica. Ultra-compact gige vision cameras - gc640 fast cmos vga camera - 200 fps, 2009.

[15] A.C. Sanderson and L.E. Weiss. Image-based visual servo control using relational graph error signals. In *Robotics and Automation, IEEE Int. Conf.*, pp 1074–1077, 1980.

[16] P.J. Sequira Goncalves. Kinematic and dynamic 2d visual servoing. In *European Control Conference, IEEE Int. Conf.*, 2001.

[17] P.J. Sequira Goncalves and J.R. Caldas Pinto. Dynamic visual servoing of robotic manipulators. In *Emerging Technologies and Factory Automation, IEEE Int. Conf.*, volume 2, pp 560– 565, 2003.

[18] L. Weiss, A. Sanderson, and C. Neuman. Dynamic sensor-based control of robots with visual feedback. *Robotics and Automation, IEEE Trans.*, 3(5): pp 404–417, 1987.