Cross-domain Modeling and Optimization of High-speed Visual Servo Systems

Zhenyu Ye, Henk Corporaal, Pieter Jonker and Henk Nijmeijer

***Abstract*— High-speed visual servo systems are used in an increasing number of applications. Yet modeling and optimizing these systems remains a research challenge, largely because these systems consist of tightly-coupled design parameters across multiple domains, including image sensors, vision algorithms, processing systems, mechanical systems, control systems, among others. To overcome such a challenge, this work applies an axiomatic design method to the design of high-speed visual servo systems, such that cross-domain couplings are explicitly modeled and subsequently eliminated when possible. More importantly, methods are proposed to model the sample rate, measurement error, and delay of visual feedback based on design parameters across multiple domains. Lastly, methods to construct a holistic model and to perform cross-domain optimization are proposed. The proposed methods are applied to a representative case study that demonstrates the necessity of cross-domain modeling and optimization, as well as the effectiveness of the proposed methods.**

I. Introduction

High-speed visual feedback is increasingly used in mechatronics and robotics systems [1] [2]. Visual servo systems are an important category of these systems, and use visual feedback in closed-loop control [3]. The majority of high-speed visual servo systems are designed for applications in structured environments [4], where vision algorithms with limited adaptability are optimized for efficiency. In structured environments, predefined patterns of environments and objects of interest are often used by vision algorithms to derive measurements for control purposes. The focus of this paper is on such use cases.

By way of example, Fig. 1 illustrates a high-speed visual servo system that uses planar visual patterns to control planar motions. The design parameters of the system are across multiple domains including, but not limited to, illumination, optics, image sensors, vision algorithms, processing systems, mechanical systems and control systems.

A. Overview of cross-domain modeling framework

This paper applies the axiomatic design method [5] to explicitly describe the coupling of design parameters, and to link them to functional requirements. An example that involves design parameters from four domains, and links them to two functional requirements, is described in (1) and (2).

Zhenyu Ye is with the Department of Mechanical Engineering, Eindhoven University of Technology, The Netherlands and Connecterra B.V., The Netherlands z.ye@tue.nl

Henk Corporaal is with the Department of Electrical Engineering, Eindhoven University of Technology, The Netherlands h.corporaal@tue.nl

Pieter Jonker is with the Department of Biomechanical Engineering, Delft University of Technology, The Netherlands p.p.jonker@tudelf.nl

Henk Nijmeijer is with the Department of Mechanical Engineering, Eindhoven University of Technology, The Netherlands h.nijmeijer@tue.nl

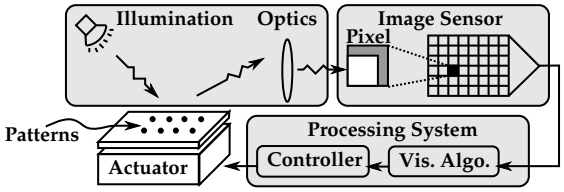


Fig. 1. An example of high-speed visual servo systems with planar visual patterns and a planar motion stage.

TABLE I

Dependent parameters and design parameters in (1).

Dependent parameter		Design parameter	
Symbol	Description	Symbol	Description
h	Sample period	I_s	Image sensor
ϵ	Measurement error	$alg.$	Vision algorithm
τ	Delay	$arch.$	Processing architecture
K	Controller gain	P	Plant

$$\begin{bmatrix} h \\ \epsilon \\ \tau \\ K \end{bmatrix} = \begin{bmatrix} \star & 0 & 0 & 0 \\ \star & \star & 0 & 0 \\ \star & \star & \star & 0 \\ \star & \star & \star & \star \end{bmatrix} \begin{bmatrix} I_s \\ alg. \\ arch. \\ P \end{bmatrix}. \tag{1}$$

$$\begin{cases} RMSE = f_1(\epsilon, h, \tau, K) \\ BW = f_2(\epsilon, h, \tau, K) \end{cases} \tag{2}$$

The matrix in (1) is called a *design matrix*, in which the symbol '$\star$' represents dependence and the symbol '0' represents independence. The two vectors on the right and the left side of the equation are design parameters and dependent parameters, as explained in Table I. In (2), two functional requirements are described as functions of dependent parameters. In this instance, the requirement of accuracy is noted as the root-mean-square error ($RMSE$) of tracking a reference signal, and the bandwidth is referred to as BW.

Each design parameter symbol in Table I represents a set of design parameters in each domain. By way of example, design parameters of image sensors (I_s) are typically image sizes and speeds of image read out. Design parameters of vision algorithms ($alg.$) are choices of algorithms and their variations. Design parameters of processing architecture ($arch.$) typically include types of processing platforms and available computational resources. Design parameters of plants (P) can be types of motor and motion stages.

The design equations expressed in (1) and (2) are examples that represent typical visual servo systems. For a specific use case, the design equations can have different design parameters, domains, coupling patterns, and requirements. Yet they can be similarly described using the same framework, as detailed in [5]. Therefore, this paper proceeds with the aforementioned example. Methods of modeling the coupling patterns in design equation (1) are explained and demonstrated in subsequent sections.

B. Related work

Methods proposed in this paper are based on adaptations of methods from previous work. This paper specializes generic methods, generalizes single-domain methods, and extends existing methods, such that a holistic and quantitative framework can be constructed for modeling and optimizing high-speed visual servo systems.

The modeling method of cross-domain coupling used in this work is based on the generic framework of axiomatic design [5] and the hierarchical representation of design parameters [6]. Using [5] [6], this paper identifies cross-domain coupling patterns of visual servo systems and proposes quantitative methods for modeling and optimizing these systems.

The design template of high-speed vision processing systems proposed in this paper is based on the processing architecture described in [7]. This paper generalizes and parameterizes the parallel architecture in [7] as well as applies high-level synthesis [8] to rapidly generate vision processing systems. The proposed method of high-level synthesis using algorithmic patterns is similar to that in [9]. This paper extends [9] by incorporating a parallel architecture template that supports high-speed vision processing.

The method of deriving controller gain is based on modeling methods and stability analysis methods used in networked control systems [10] [11]. This paper simplifies the methods in [10] [11], by removing time-varying delay and other network-induced effects from the model, and applying them to visual servo systems.

There is previous work that addresses specific coupling problems in the design matrix (1). In [12], couplings between illumination, optics, image sensors, and vision algorithms, as well as their impacts on measurement errors are explored. In [13], effects of processing resource on sample rate, delay, controller gain, and the resulted control performance are examined. In [14], a controller is optimized by taking into account a large delay induced by visual feedback.

C. Contributions

This paper contributes to the state of the art by providing a framework of methods for cross-domain modeling and optimization of high-speed visual servo systems. More specifically, this paper bridges a gap between generic methods of cross-domain modeling [5] [6] and methods that address a specific set of coupling problems in visual servo systems [12] [13] [14]. To the best knowledge of the authors, this paper is the first to bridge such a gap using a holistic and quantitative framework.

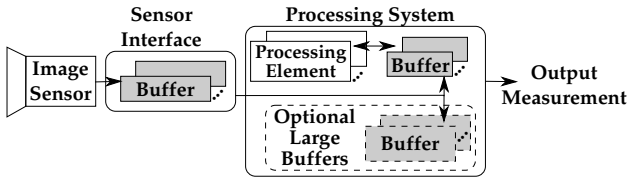


Fig. 2. A design template of high-speed vision systems with image buffers. The sensor interface and the processing system have buffers to hold multiple image lines or blocks. If the image algorithm requires to store large image frames, an additional large memory can be applied.

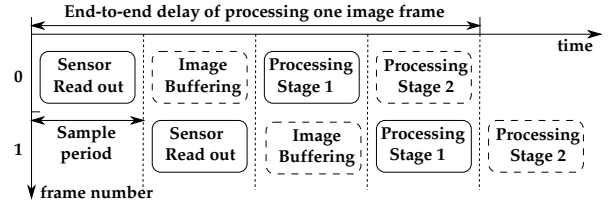


Fig. 3. A timing diagram achievable by the design template of Fig. 2. The stages marked with dashed boxes, that is, image buffering and additional stages of processing, can be omitted for simple vision algorithms. Based on the design template, the sample rate only depends on the read out time of the image sensor.

The organization of the paper is as follows. First, design and modeling methods are proposed and described in generic terms. Next, a representative case study and its design parameters are introduced. Subsequently, the proposed methods are applied to the case study and the results are discussed. In the end, conclusions are drawn.

II. Design and Modeling Methods

While this paper focuses on modeling methods, the applicability of modeling methods depends on the system's design. Therefore, design methods for high-speed visual servo systems are introduced first, followed by modeling methods.

A. Design of High-speed Vision Systems

Based on surveys of existing high-speed visual servo systems [1] [2] and current technologies, visual servo systems can be designed to have a coupling pattern described by (1), in which the sample period (h) depends only on the image sensor's design parameters. To achieve design equation (1), high-speed vision systems are often designed to process a frame in multiple pipeline stages, with buffers between each stage.

This paper proposes a design template of high-speed vision systems, illustrated in Fig. 2, that decouples the sample rate from vision algorithms and processing systems. The corresponding sample rate and end-to-end delay of the vision system are described in Fig. 3. In the proposed design template, the image buffering and processing can be pipelined into multiple stages, such that each of those stages takes less time than that of sensor read out. Therefore, the sample rate depends only on the time needed by sensor read out.

The proposed design method illustrated in Fig. 2 can be applied to a wide range of vision algorithms, and the resulted delay, described in Fig. 3, ranges from less than two sample periods to multiple sample periods. A case study that uses the proposed design method to achieve $h < \tau < 2h$ is detailed in the next section. The remaining coupling patterns of (1) are self-evident and can be achieved using common design methods.

B. Methods of Cross-domain Modeling

For the ease of modeling, the design matrix in (1) can be separated into two steps, described by design equations (3a) and (3b). In the first step, three dependent parameters $[h, \epsilon, \tau]$ are derived from design parameters $[I_s, alg., arch.]$. In the second step, controller gain K is derived from dependent parameters $[h, \epsilon, \tau]$ and from design parameters P.

$$\begin{bmatrix} h \\ \epsilon \\ \tau \end{bmatrix} = \begin{bmatrix} \star & 0 & 0 \\ \star & \star & 0 \\ \star & \star & \star \end{bmatrix} \begin{bmatrix} I_s \\ alg. \\ arch. \end{bmatrix}, \tag{3a}$$

$$K = \begin{bmatrix} \star & \star & \star & \star \end{bmatrix} \begin{bmatrix} h \\ \epsilon \\ \tau \\ P \end{bmatrix}. \tag{3b}$$

The benefit of performing the modeling in two separate steps is the reduced complexity of deriving controller gain K. More specifically, the controller gain K can be derived from $[h, \epsilon, \tau, P]$ of (3b) instead of from $[I_s, alg., arch, P]$ of (1). The latter contains significantly more parameters. The modeling methods for each of the dependent parameters in (3) are described subsequently.

1) Modeling sample period h: As discussed in Section II-A, a high-speed visual servo system can be designed to have its sample period (h) only depend on design parameters of image sensors (I_s). A typical high-speed image sensor has a configurable exposure time (t_{exp}). After exposure, a pre-configured number of pixels (N_p) is read out line-by-line at a certain speed (R_d) in pixels per second. Therefore, the sample period can be modeled as

$$h = t_{exp} + \frac{N_p}{R_d}. \tag{4}$$

2) Modeling measurement error ϵ: The measurement error (ϵ) is defined as the difference between a plant's actual position x and its corresponding quantized position $Q(x)$ measured by visual feedback. The quantization function Q is modeled using the method of [15], which has a sensitivity $\Delta \in \mathbb{R}_{>0}$. The measurement error is therefore modeled as

$$\epsilon = Q(x) - x = \Delta \left\lfloor \frac{x}{\Delta} + \frac{1}{2} \right\rfloor - x, \tag{5}$$

in which $\lfloor x \rfloor$ is a floor function defined as $\lfloor x \rfloor = max\{m \in \mathbb{Z} | m \leq x\}$.

The quantization sensitivity (Δ) of visual feedback can be obtained from simulations as follows. First, an image is placed at position x. After that, the image is moved from position x by a displacement d. The vision algorithm (lumped into a function g) should ideally detect the displacement d between the two images, but in practice it induces an error e defined as,

$$e(x, d) = g(x + d) - g(x) - d. \tag{6}$$

Subsequently, the sensitivity (Δ) can be modeled using the peak-to-peak error, denoted $P - P(e)$, of (6).

The procedure of deriving measurement error ϵ is as follows. First, an error profile $e(x, d)$ defined by (6) is obtained via simulation. Second, a peak-to-peak error $P - P(e)$ is derived from the error profile $e(x, d)$. Third, a quantization sensitivity Δ is set equal to $P - P(e)$. Subsequently, measurement error ϵ is modeled using Δ according to (5). An example of applying this procedure to a case study is provided in Section IV.

3) Modeling delay τ: To obtain the delay of different algorithmic choices and architectural choices, high-level synthesis [8] is used to rapidly generate customized processing systems, which are subsequently used to obtain the delay τ. To reduce the complexity of synthesizing every possible vision algorithm on every possible processing architecture, this paper proposes to perform high-level synthesis based on algorithmic patterns and architecture templates. The method is illustrated in Fig. 4, showing two examples of algorithmic patterns and an architecture template proposed in this work.

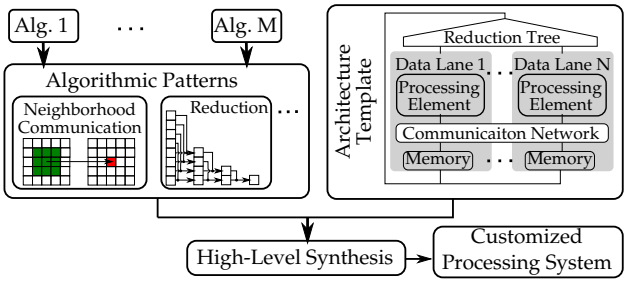


Fig. 4. High-level synthesis based on algorithmic patterns and architecture templates.

Algorithmic patterns, also called algorithmic skeletons, describe data access patterns of algorithms that are separated from the actual computations [16]. The definitions of common algorithmic patterns are detailed in [16] [17]. The two examples of algorithmic patterns displayed in Fig. 4 are neighborhood communication and reduction; the former computes one output element from a small window of input elements, while the latter computes one output element based on one vector or one block of pixels.

Architecture templates are parameterized building blocks of processing systems, including programmable processors [16] as well as dedicated circuits [18]. To perform rapid prototyping of algorithmic patterns on architecture templates, this work applies the high-level synthesis method and targets field-programmable gate arrays (FPGA) devices [8]. Using the aforementioned methods, this paper rapidly generates customized processing systems and obtains their delays by simulations.

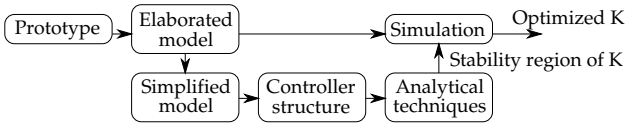


Fig. 5. Method of deriving controller gain using a combination of analytical method and simulation.

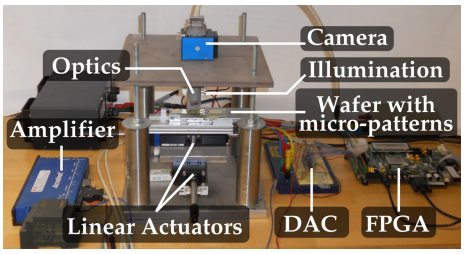


Fig. 6. A prototype representing a high-speed visual servo system described by Fig. 1. Abbreviation: Digital-to-Analog Converter (DAC); Filed-Programmable-Gate-Array (FPGA).

4) Deriving controller gain K: A combination of analytical methods and simulations are used in this paper to derive the controller gain. The process is summarized in Fig. 5. First, based on a prototype, an elaborated model of the plant is constructed and the model's parameters are identified. After that process, the elaborated model is simplified so that analytical methods can be applied. More specifically, the elaborated model is simplified into a quantized sampled-data system with time delay. Next, based on the model, an overall structure of the controller is chosen. Subsequently, the stability region of controller gains K is analytically derived based on the simplified model. The analytical method used in this work is simplified from those used in networked control systems [10] [11] by removing time-varying delay and other network-induced effects. In the end, within the stability region, the controller gains are optimized according to various criteria, based on simulations using the elaborated model. The corresponding performance is obtained simultaneously. An example of applying this method to a case study is provided in Section IV-C.

III. Case Study

To demonstrate the proposed design and modeling methods, a visual servo system prototype involving tightly-coupled design parameters across multiple domains is used as a case study. As discussed in Section I, this paper focuses on high-speed visual servo systems in a structured environment, and uses planar vision and planar motion as examples. A prototype that is representative for such use cases is implemented, and illustrated in Fig. 6.

As displayed in Fig. 6, the prototype has an X-Y motion stage driven by two linear actuators. The illumination system, optics, and camera are fixed to a static frame. A wafer with pre-defined micro-patterns on it is mounted on the motion stage. This prototype is applied to a use case of vision-based inkjet printing on non-uniform micro-structures, as detailed in [19]. In such a use case, visual measurements are used in a closed loop to compensate for deformations of flexible display substrates. A snapshot of micro-structures on the wafer and algorithms for detecting the centers of these structures are shown in Fig. 7, and to be described in Section III-B. Further details of the use case are available in [19]. The remaining parts of this paper focus on applying the proposed design and modeling methods to the case study.

TABLE II
Image sizes and their properties.

Image size $[px]$	Image Type	Resolution $[\mu m/px]$
120×75	synthetic by downsampling	6.67
160×100	in-situ measurement	5
200×125	synthetic by upsampling	4

Among hundreds of design parameters in the prototype system, this case study selects one or a few design parameters from each domain as examples. Design parameters from each domain are described in details in the remaining parts of this section. Despite the fact that only a limited number of design parameters are included, they are sufficient to demonstrate the importance of cross-domain modeling and the effectiveness of the proposed methods, which is supported by results in Section IV.

A. Image sensors

In this case study, the image size is chosen as a design parameter of image sensors. As shown in Table II, three image sizes are selected, and the zoom factor of the optical system is adjusted to let three different image sizes have the same field of view of $800 \times 500[\mu m]$. The size of field of view is chosen to cover multiple micro-patterns, which have pitches of $200 \times 80[\mu m]$, as illustrated in Fig. 7a.

The exposure time (t_{exp}) is experimentally determined to be approximately $50\mu s$. A shorter exposure time significantly increases image noise, while a longer exposure time only has a marginal improvement in image noise. Therefore, the exposure time is fixed at the aforementioned value. An image sensor capable of achieving 1600 frames-per-second is used in the case study, with an image read out rate (R_d) of $32px/\mu s$. With the exposure time and the image read out rate fixed, this case study only uses image sizes as a design parameter of image sensors.

B. Vision algorithms

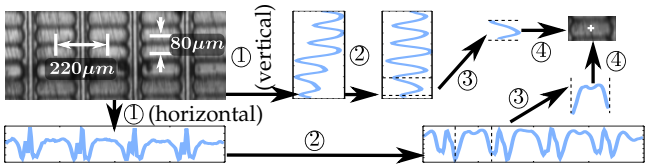
Three vision algorithms with different delay and accuracy trade-offs are considered. The algorithms are designed to detect the centers of pre-defined patterns, which are used as feedback for the controller. The details of the use case are provided in [19]. The processing stages of these algorithms are summarized in Table III, and visualized in Fig. 7.

Among these algorithms, Algorithm 3 is the most complicated and includes operations used by Algorithm 1 and 2. Therefore, this section only provides the details of Algorithm 3 and its implementation on the architecture template. The other two algorithms, with simpler processing stages, can be similarly implemented.

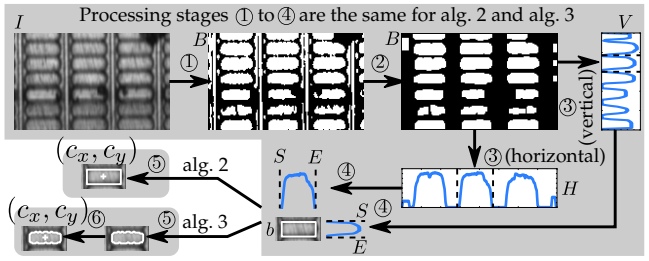
TABLE III

PROCESSING STAGES OF THREE VISION ALGORITHMS.

Stages	Algorithm 1	Algorithm 2	Algorithm 3
①	projection	binarization	binarization
②	1D filter	2D filter	2D filter
③	segmentation	projection	projection
④	1D moment	segmentation (bounding box)	segmentation (bounding box)
⑤	-	2D moment (bounding box)	segmentation (contour)
⑥	-	-	2D moment (contour)



(a) Algorithm 1



(b) Algorithm 2 and 3

Fig. 7. Illustration of each processing stage of three vision algorithms.

Algorithm 3 performs binarization at Stage ① , converting a gray scale image I to a binarized image B. At Stage ② , morphological filters are applied on the binary image B to remove noise. Afterwards, Stage ③ projects the 2D binary image B into a horizontal vector H and vertical vector V. At Stage ④ , the horizontal vector H and vertical vector V are segmented into start and end coordinates S and E, which can be used to form bounding box b. At Stage ⑤ , a mask is derived within the bounding box b to represent the contour of the predefined patterns. In the simplest implementation, the binary image B generated at Stage ② can be re-used as a mask. In such a case, Stage ⑥ performs a 2D image moment over the original image I within the bounding box b and uses the binary image B as a mask. Stage ⑥ can be described as follows:

$$c_x = \frac{\sum_{i,j\in b}(i \cdot I_{i,j} \cdot B_{i,j})}{\sum_{i,j\in b}(I_{i,j} \cdot B_{i,j})}, \quad c_y = \frac{\sum_{i,j\in b}(j \cdot I_{i,j} \cdot B_{i,j})}{\sum_{i,j\in b}(I_{i,j} \cdot B_{i,j})}. \tag{7}$$

in which $I_{i,j}$ is a pixel at column i and row j of the gray scale image; similarly, $B_{i,j}$ is a pixel of the binary image; c_x and c_y are the x and y coordinates of the center of the mask, weighted by each pixel's gray scale values.

TABLE IV

COMPLEXITY ANALYSIS AND MAPPING OF ALGORITHM 3.

Step	Complexity	Alg. Pattern	Mapping
①	$O(m \times n)$	Local	Parallel proc.
②	$O(m \times n)$	Neighbor	Parallel proc.
③	$O(m \times n)$	Reduction	Parallel proc.
④	$O(m + n)$	Scan	Sequential proc.
⑤	$O(m \times n)$	Local	Parallel proc.
⑥	$O(m \times n)$	Reduction	Parallel proc.

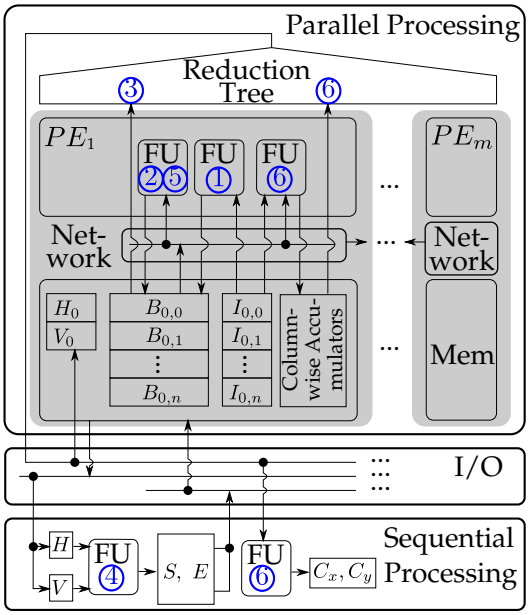


Fig. 8. Algorithm 3 mapped on the architecture template. The parallel processing circuit has m data lanes, and the first data lane is illustrated in details. A data lane communicates with its left and right neighbors via a neighborhood network. Abbreviations: Processing Elements (**PE**), Functional Unit (**FU**), Memory (**Mem**).

C. Processing architecture

Using Algorithm 3 as a case in point, this section analyzes each stage of the algorithm for its computational complexity, algorithmic pattern, and subsequently maps it to parallel or sequential processing circuits of the architecture template. The analysis is provided in Table IV, for an input image size of $m \times n$. The computational complexity is presented in big O notation which describes how the computation workload grows with regard to the input image size. The algorithmic patterns are explained in Section II-B.3, and a common list of them are detailed in [16] [17].

Most of the processing stages can be directly mapped on the architecture template of Fig. 4, except for stage ⑥ which is more complicated. Stage ⑥ is subsequently split into three sub-stages that can be efficiently implemented. First, column-wise multiply-accumulation operations are performed in parallel. Second, the reduction tree sums all the column-wise results. In the end, the sequential circuit performs divisions to derive c_x and c_y of (7).

With the aforementioned optimization, all six stages of Algorithm 3 can be mapped on the architecture template, as

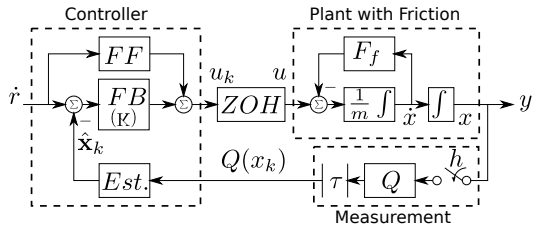
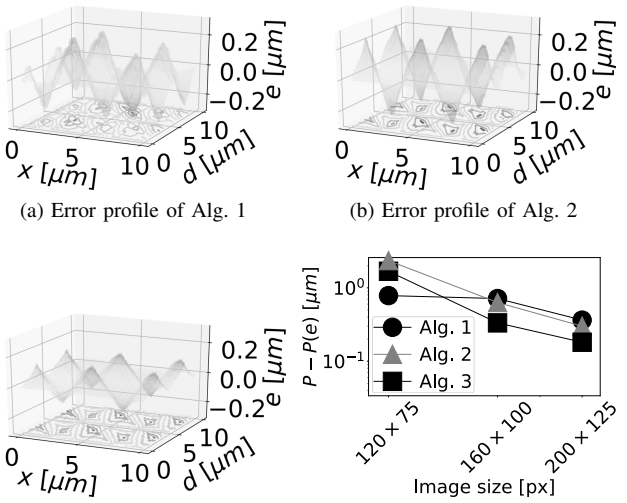


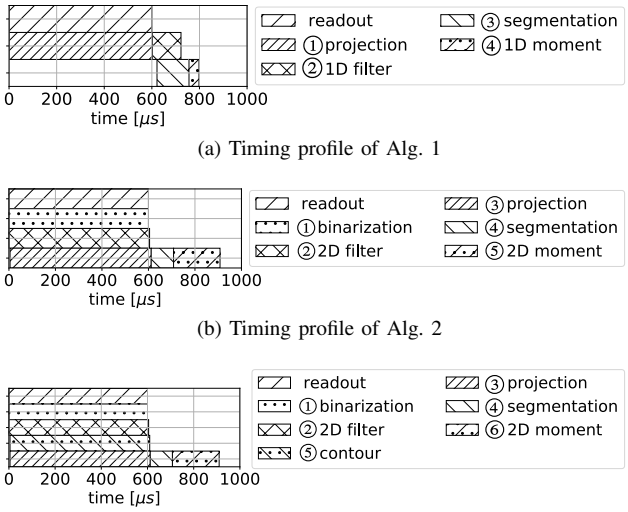
Fig. 9. System diagram of the plant, measurement system, and control system.



(a) Error profile of Alg. 1 (b) Error profile of Alg. 2

(c) Error profile of Alg. 3 (d) P-P(e) of different image sizes.

Fig. 10. Measurement error profile $e(x, d)$ of different algorithms on image of 160×100 px., shown in (a)-(c), and the corresponding P-P(e) when different image sizes are also considered, shown in (d).



(a) Timing profile of Alg. 1

(b) Timing profile of Alg. 2

(c) Timing profile of Alg. 3

Fig. 11. Timing profile of different algorithms. In this instance, the image size is 160×100 px., and there are 160 parallel data lanes in the processing architecture.

illustrated in Fig. 8. In this instance, the number of parallel data lanes is set equal to the number of columns in the image data, that is, equal to m. The synthesized system is capable of running at a clock speed of $100MHz$ and processing 100,000 frames-per-second given a fast image sensor. However, a clock speed of $17MHz$ is sufficient in this example because the read out time from the image sensor used in this case study is the bottleneck.

D. Plant

The plant consists of an X-Y motion stage driven by two linear actuators. Accordingly, the plant is modeled as a simple mass (m) with friction (F_f). The actuator is zero-order-hold (ZOH) and has an output force u. Together with the modeling of the measurement system and the control system, the system diagram is illustrated in Fig. 9. The measurement error is modeled as a quantization function Q as described in (5). The control system includes a state estimator ($Est.$), a feedforward controller (FF) and a feedback controller (FB). The feedforward controller is manually tuned to compensate for the friction. The feedback controller is a PID controller. Baseline values of the proportional, integral, and derivative gains are manually tuned. The proportional gain, noted K, is subsequently optimized using the procedure described in Fig. 5. The controller gain K is optimized for two criteria, accuracy and bandwidth, as detailed in Section IV-C. The plant's design parameters, such as the types of motor and motion stage, are reflected in this model's mass, friction, and output force.

IV. Results and Discussions

By applying the proposed methods to the case study, this section derives dependent parameters in (3a), the controller gain in (3b), and subsequently the overall performance in (2). The implications of the results are discussed in the end.

A. Measurement error ϵ

The measurement error ϵ can be derived from the error profile $e(x, d)$ as defined in (6) and by using the procedure described in Section II-B.2. The error profiles of different input image sizes and vision algorithms are revealed in Fig. 10. It can be observed that, Algorithm 1 is most precise among the three algorithms if a smaller input image size is used, while Algorithm 3 is most precise if a larger input image size is applied. The choice of image sizes and algorithms are more complicated when considering sample period and delay, as shown in the following section.

B. Sample period h and delay τ

As described in the design matrix (3a), the sample period h depends only on the image size. The delay τ, on the other hand, depends on the image size, vision algorithm, and processing architecture. By way of example, the number of parallel data lanes in the processing architecture is set to be the same as the number of columns in the image. The sample period and delay of the system, using an image size of 160×100 px., is illustrated in Fig. 11.

In Fig. 11, all three algorithms have a two-stage pipeline based on the design template of Fig. 2, and have a timing

TABLE V

CONFIGURATIONS OF ALGORITHMS, IMAGE SIZES, AND PARALLEL DATA LANES IN THE ARCHITECTURES; DERIVED DEPENDENT PARAMETERS; AND LABELS OF PARETO-OPTIMAL CONFIGURATIONS IN FIG. 12.

Alg.	Image size	Lanes	h $[\mu s]$	Δ $[\mu m]$	τ $[\mu s]$	Label
1	120×75	120	450	0.78	621.6	Ⓐ
	160×100	160	600	0.71	795.6	Ⓑ
	200×125	200	750	0.36	969.6	-
2	120×75	120	450	2.35	712.8	-
	160×100	160	600	0.63	902.4	-
	200×125	200	750	0.3	1092	-
3	120×75	120	450	1.68	712.8	-
	160×100	160	600	0.33	902.4	Ⓒ
	200×125	200	750	0.18	1092	Ⓓ

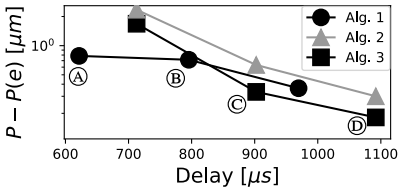


Fig. 12. The measurement error and delay of three vision algorithms at three different input image sizes. The configuration of design parameters with Pareto-optimal error-delay trade-offs are labeled Ⓐ Ⓑ Ⓒ Ⓓ .

diagram of Fig. 3. The first stage consists of image read out and parallel processing; the second stage consists of segmentation and post-segmentation processing. As explained in Fig. 3, the sample period is determined by the first stage, while the delay depends on both stages.

A summary of dependent parameters is provided in Table V. When plotting measurement errors against delays, as illustrated in Fig. 12, combinations of design parameters that have Pareto-optimal error-delay trade-offs are labeled Ⓐ Ⓑ Ⓒ Ⓓ in the figure. A configuration is called Pareto-optimal if there is no other configuration that strictly dominates it in all criteria, which are error and delay in this case. Using Pareto-optimal configurations simplifies the exploration of design space, as discussed in details in [20].

C. Controller gain and overall performance

As described in the design matrix (3b), the controller gain can be obtained from dependent parameters h, ϵ, τ, and design parameters of the plant P. Design parameters of the plant P, reflected in mass m and friction F_f, are included in the model but have fixed values in this case, because the number of design parameters in Table V are already sufficiently large. The controller's structure is explained in Section III-D. This section explores proportional gain K of the feedback controller, using the method described in Fig. 5.

Following Fig. 5, the elaborated model in Fig. 9 is reduced into a simplified model as shown in Fig. 13. The simplification assumes the friction is compensated by the feedforward controller; the state estimator is perfect; only the proportional gain K is present. The simplified model is a quantized sampled-data system with time delay, which is a special case of [11]. A stability region of K can be derived using analytical methods of [11]. Within the stability region, the

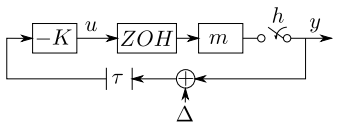


Fig. 13. A simplified model derived from Fig. 9. A stability region of K is derived analytically from this simplified model.

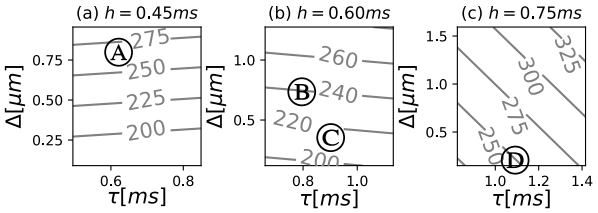


Fig. 14. RMSE of tracking a constant velocity reference (8), resulted from different sample periods (a)(b)(c), different delays (τ) and different measurement errors (ϵ) defined by quantization sensitivity (Δ). The tracking errors are illustrated as equal-$RMSE$ lines with actual $RMSE$ values in $[\mu m/s]$ annotated along the lines. The RMSE of Pareto-optimal configurations Ⓐ - Ⓓ are annotated along the equal-$RMSE$ lines.

proportional gain K is further optimized using the elaborated model in Fig. 9 and by simulation, for performance criteria of accuracy and bandwidth.

Accuracy and bandwidth are used in this case study as two examples of performance criteria, as mentioned in (2). Accuracy is measured as the root-mean-square error, noted $RMSE$, of tracking a constant velocity reference. Bandwidth is measured as the tracking errors of sinusoidal velocity references of different frequencies. The reference signals used for the evaluation of accuracy and bandwidth are, respectively, as follows:

$$\dot{r} = 0.03m/s, \tag{8}$$

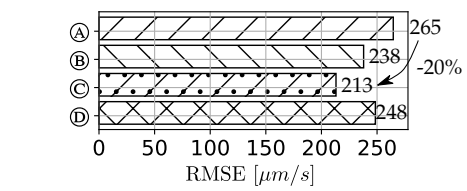
$$\dot{r} = A \cdot sin(2\pi ft) + B. \tag{9}$$

In this example, $A = 0.005m/s$, $B = 0.02m/s$, and the frequency f has values of 10, 20, and $50Hz$ to represent use cases that have different bandwidth requirements.

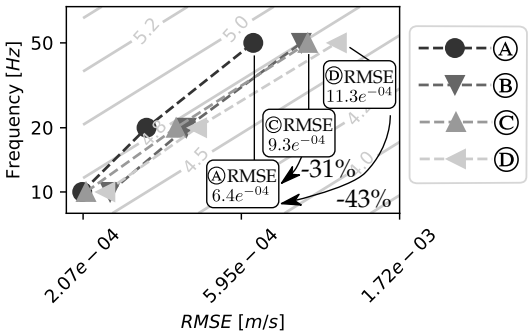
For the performance criterion of accuracy, the tracking errors of a constant reference that result from different ϵ, h, and τ, are displayed in Fig. 14. Four Pareto-optimal configurations are annotated in this figure. The same results are illustrated in Fig. 15a, for a direct comparison of different configurations. For the performance criterion of bandwidth, the tracking errors of sinusoidal references of different frequencies that result from different ϵ, h, and τ, are illustrated in Fig. 15b.

D. Discussions

The results of the case study, revealed in Fig. 15, have multiple implications. First, the results indicate that a significant improvement of system performance is obtained when using the proposed cross-domain modeling and optimization methods. Measured quantitatively, for a constant reference, a 20% reduction of tracking errors is achieved, as displayed in Fig. 15a; for high-bandwidth use cases, with a sinusoidal



(a) Constant reference



(b) Sinusoidal references of different frequencies

Fig. 15. RMSE of tracking: (a) a constant reference described in (8); (b) sinusoidal references of different frequencies f described in (9). In (b), besides four configurations Ⓐ - Ⓓ , equal-$(f/RMSE)$ lines are plotted, with the values of $log(f/RMSE)$ annotated along the lines.

reference of $50Hz$, a 43% reduction of tracking errors is achieved, as illustrated in Fig. 15b. Second, the results indicate that, with conflicting requirements such as accuracy and bandwidth, a system optimized for one set of requirements is possibly far from optimal for another set of requirements. As a case in point, among the four configurations, Ⓒ achieves the best accuracy when tracking a constant reference, as demonstrated in Fig. 15a; however, Ⓒ is far from optimal in high-bandwidth use cases where the tracking error of Ⓐ is 31% lower than that of Ⓒ , as shown in Fig. 15b. Third, more importantly, the results indicate that the demonstrated improvements are only achievable through cross-domain modeling and optimization. These results make a case for the necessity and effectiveness of the proposed methods.

V. Conclusions

This work applies the axiomatic design method to explicitly model the cross-domain couplings in visual servo systems. Design methods are proposed to decouple sample rate from vision algorithms and processing systems. Modeling methods are proposed to derive sample rate, measurement error, and delay of visual feedback based on design parameters across multiple domains. The proposed methods are applied to a representative case study, which demonstrates that, even when a small number of design parameters are considered, there are nontrivial cross-domain couplings and trade-offs. Quantitative results obtained from the case study demonstrate the necessity and effectiveness of the proposed methods.

References

[1] Q.-Y. Gu and I. Ishii, "Review of some advances and applications in real-time high-speed vision: Our views and experiences," *International Journal of Automation and Computing*, vol. 13, no. 4, pp. 305–318, Aug 2016.

[2] Y. Watanabe, H. Oku, and M. Ishikawa, "Architectures and applications of high-speed vision," *Optical Review*, vol. 21, no. 6, pp. 875–882, Nov 2014.

[3] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, Oct 1996.

[4] K. Goldberg, "What is automation?" *IEEE Transactions on Automation Science and Engineering*, vol. 9, no. 1, pp. 1–2, Jan 2012.

[5] N. P. Suh, "Axiomatic design theory for systems," *Research in Engineering Design*, vol. 10, no. 4, pp. 189–209, 1998.

[6] P. Hehenberger, F. Poltschak, K. Zeman, and W. Amrhein, "Hierarchical design models in the mechatronic product development process of synchronous machines," *Mechatronics*, vol. 20, no. 8, pp. 864 – 875, 2010, special Issue on Theories and Methodologies for Mechatronics Design.

[7] Y. He, Z. Ye, D. She, B. Mesman, and H. Corporaal, "Feasibility analysis of ultra high frame rate visual servoing on fpga and simd processor," in *Advanced Concepts for Intelligent Vision Systems*, 2011.

[8] R. Nane, V. M. Sima, C. Pilato, J. Choi, B. Fort, A. Canis, Y. T. Chen, H. Hsiao, S. Brown, F. Ferrandi, J. Anderson, and K. Bertels, "A survey and evaluation of fpga high-level synthesis tools," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 10, pp. 1591–1604, Oct 2016.

[9] S. Fernando, M. Wijtvliet, C. Nugteren, A. Kumar, and H. Corporaal, "(as)2: Accelerator synthesis using algorithmic skeletons for rapid design space exploration," in *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2015, pp. 305–308.

[10] M. B. G. Cloosterman, N. van de Wouw, W. P. M. H. Heemels, and H. Nijmeijer, "Stability of networked control systems with uncertain time-varying delays," *IEEE Transactions on Automatic Control*, vol. 54, no. 7, pp. 1575–1580, July 2009.

[11] S. van Loon, M. Donkers, N. van de Wouw, and W. Heemels, "Stability analysis of networked and quantized linear control systems," *Nonlinear Analysis: Hybrid Systems*, vol. 10, pp. 111 – 125, 2013, special Issue related to IFAC Conference on Analysis and Design of Hybrid Systems (ADHS 12).

[12] Y. Chen, T. Ogata, T. Ueyama, T. Takada, and J. Ota, "Automated design of the field-of-view, illumination, and image pre-processing parameters of an image recognition system," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, Aug 2017, pp. 1079–1084.

[13] R. Medina, S. Stuijk, D. Goswami, and T. Basten, "Exploring the trade-off between processing resources and settling time in image-based control through lqr tuning," in *Proceedings of the Symposium on Applied Computing*, ser. SAC '17. New York, NY, USA: ACM, 2017, pp. 1456–1459.

[14] M. B. Khamesee, M. Rapolti, and A. Amirfazli, "Controller design and optimization for large-delays image processing in visual closed-loop systems," *Mechatronics*, vol. 18, no. 5, pp. 251 – 261, 2008, special Section on Optimized System Performances Through Balanced Control Strategies.

[15] R. W. Brockett and D. Liberzon, "Quantized feedback stabilization of linear systems," *IEEE Transactions on Automatic Control*, vol. 45, no. 7, pp. 1279–1289, Jul 2000.

[16] W. Caarls, "Automated design of application-specific smart camera architectures," Ph.D. dissertation, Delft University of Technology, 2008.

[17] C. Nugteren, H. Corporaal, and B. Mesman, "Skeleton-based automatic parallelization of image processing algorithms for gpus," in *2011 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation*, July 2011, pp. 25–32.

[18] K. Benkrid, D. Crookes, J. Smith, and A. Benkrid, "High level programming for FPGA based image and video processing using hardware skeletons," in *The 9th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, FCCM 2001, Rohnert Park, California, USA, April 29 - May 2, 2001*, 2001, pp. 219–226.

[19] R. Pieters, Z. Ye, P. Jonker, and H. Nijmeijer, "Direct motion planning for vision-based control," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 4, pp. 1282–1288, Oct 2014.

[20] C. Haubelt and J. Teich, "Accelerating design space exploration using pareto-front arithmetics [soc design]," in *Proceedings of the ASP-DAC Asia and South Pacific Design Automation Conference, 2003.*, Jan 2003, pp. 525–531.