Event- and Deadline-Driven Control of a Self-Localizing Robot With Vision-Induced Delays

Eelco P. van Horssen, Jeroen A. A. van Hooijdonk, Duarte Antunes, *Member, IEEE*, and W. P. M. H. Heemels, *Fellow, IEEE*

***Abstract*—Control based on vision data is a growing field of research and it is widespread in industry. The amount of data in each image and the processing needed to obtain control-relevant information from this data lead to significant delays with a large variability in the control loops. This often causes performance deterioration since in many cases the delay variability is not explicitly addressed in the control design. In this paper, we approach this problem by applying the ideas of recently developed model-based control design methods, which are tailored to address stochastic delays directly, to the motion control of an omnidirectional robot with a vision-based self-localization algorithm. The completion time or delay of the Random Sample Consensus (RANSAC) based localization algorithm is identified as a stochastic random variable with significant variability, illustrating the practical difficulties with data processing. Our main aim is to show that the novel deadline-driven and event-driven control designs significantly outperform a traditional periodic control implementation for a stochastic optimal control performance index.**

***Index Terms*—Deadline-driven control, event-driven control, feedback control, image-processing, robotics, stochastic optimal control, stochastic time-delay, vision.**

I. Introduction

VISION-BASED control has been a research topic for many years, both in the domains of computer-vision and control, with industrial robotics as an important application domain. Due to the complex nature of visual data, the time taken by image-processing algorithms can not only be quite large but also it can vary significantly with the properties of the image, such as color, detectability and spread of features, and possibility of segmentation. Moreover, the vision algorithms themselves make complex decisions, which introduces significant runtime delay as well. Mainly worst-case approaches to delay are taken in vision-based systems to prevent the loss of data. Recently, Prakash *et al.* [1] formally showed that this can significantly limit the achievable performance compared to a model-based approach for the delay.

One prominent example of processing algorithms causing delay is the Random Sample Consensus (RANSAC) [2] method, which is widely used in image processing to eliminate data outliers. It achieves this by randomly and iteratively selecting a minimal subset of data to compute the parameters of a model; the algorithm then checks which elements of the entire data set are consistent with this model, and stops if enough inliers are found (according to a threshold), and continues otherwise. When RANSAC is used, the method often runs without memory from previous runs, and therefore the delays can be assumed to be independent and identically distributed. In this paper, the delay caused by a simple RANSAC-based algorithm is considered in order to illustrate the practical application of the novel modeling and control design methods developed in [1], which uses models for variations in delay. Specifically, our main contribution is the application of the novel control design methods in a practical setting, rather than the practical setup itself. We have adapted the control design methods developed in [1] to the practical setting considered in this paper. The practical setting is a constrained omnidirectional robot with a vision-based localization algorithm that has been devised to provide a good basis for the control design methods, which are designed for linear systems. While the practical setting is limited from a general robotics perspective, it still resembles many interesting practical and industrial cases of motion-control with processing delays. It provides a basis for further development of the design of control algorithms using models of processing delay.

There are several works on vision-based control in the literature and in particular works using RANSAC. For example, a slightly different RANSAC-based localization procedure was used in [3], motivating the use of RANSAC for localization purposes in vision-based control. A background in robot navigation can be found in [3]–[8]. However, such works do not consider the timing/latency/delay aspect of the control problem, illustrating the separation-of-concerns approach that is usually taken. Robust approaches to time-delay, such as model-matching

Manuscript received April 14, 2018; revised October 13, 2018 and December 22, 2018; accepted January 30, 2019. Date of publication February 21, 2019; date of current version September 30, 2019. This work was supported in part by the Netherlands Organisation for Scientific Research (NWO-TTW) under Grant 12697 "Control Based on Data-Intensive Sensing," and in part by the Innovational Research Incentives Scheme under the VICI Grant 11382 "Wireless Control Systems: A New Frontier in Automation" awarded by NWO-TTW. *(Corresponding author: Eelco Van Horssen.)*

The authors are with the Control Systems Technology Group, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands (e-mail: eelcovanhorssen+tue@gmail.com; jeroenvhooijdonk@gmail.com; d.antunes@tue.nl; m.heemels@tue.nl).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2019.2899553

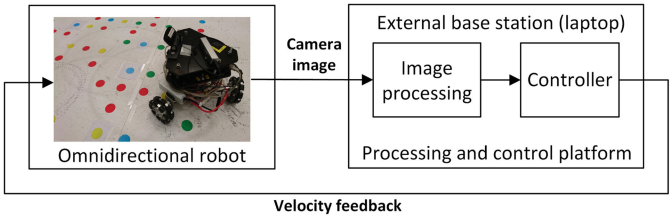

Fig. 1. Image-based feedback loop for an omnidirectional robot.

approaches [9]–[12], which account for the worst-case, are conservative since the (stochastic) nature of the delay is not taken into account in the design, which is often sufficient when the goal is only stability. The early works [13], [14] illustrate the impact that latency can have on performance in the context of different processing structures. Other earlier ideas to model the stochastic delay can be found in [15]–[18] as discussed in [1]. In a different context, Demirel [19] also suggested a procedure to find an optimal deadline.

Typically, in vision-based control applications, the vision and data-acquisition system is developed separately from the control, and the control design uses either a conservative model of the data-acquisition system, for example, considering the worst-case delay [9]–[12], or the data-acquisition characteristics are neglected [3]–[8] in the control design. Often, these conservative and possibly inaccurate models are subsequently used for periodic control design. Instead, in this paper, we show that it is possible to develop an appropriate model of the image-processing algorithm and use it in aperiodic control designs resulting in superior performance over traditional periodic control solutions.

Recently, Prakash *et al.* [1] presented novel model-based control design methods for systems with stochastic actuation delays. In particular, a deadline-driven control algorithm was developed, which updates the closed-loop system with a fixed period but only if the delay is smaller than a deadline, as well as an event-driven control algorithm, which immediately uses new data when it becomes available. The new controller designs can improve performance in a stochastic optimal control performance index sense. In this paper, we adapt the ideas of [1] to design output-feedback controllers for systems with stochastic measurement delays. In particular, we show feasibility of the model-based control design methods for practical vision-based applications by demonstrating the design for the vision-based control loop that is depicted in Fig. 1. An omnidirectional robot moves in an environment with colored markers. By processing camera images, the robot can determine its position and use this information for feedback control in a tracking problem. We compare both novel design methods to a traditional periodic control implementation which accommodates for (almost) all delays. Using postprocessed images obtained at a higher sampling rate than the processing rate, the tracking error and the process disturbance are identified and the system model is validated.

The rest of the paper is organized as follows. Section II provides an overview of the experimental system and the main problem formulation. The tracking control designs considering stochastic delays are detailed in Section III.

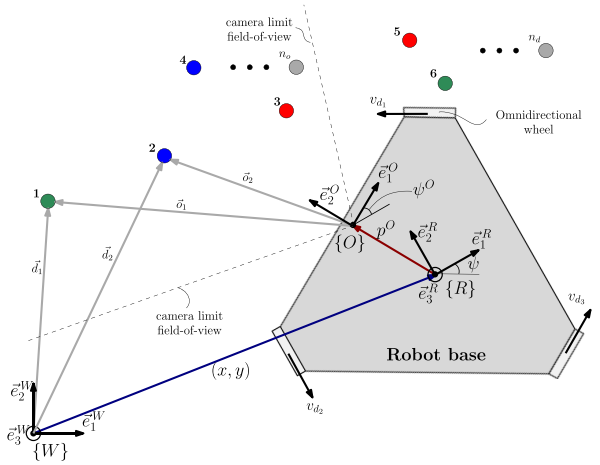

Fig. 2. Top view of the robot base with three wheels in an environment with colored markers on the ground. The coordinate frames of the world W, the robot base R, and the observing camera O are indicated. Relevant vectors for the localization algorithms and the main rotational direction of the wheels are depicted.

Section IV discusses the proposed vision-based localization method. Experimental results are presented in Section V. Section VI concludes this paper.

II. System Description and Problem Formulation

An *omnidirectional robot* moves on a planar field with coplanar colored markers, as can be seen in Fig. 1. The robot is equipped with an onboard camera which observes the environment in front of the robot. Through a high-speed Wi-Fi connection, an external processing and control platform, which we call the *base station*, acquires camera images and processes them to obtain a position estimate of the robot in the environment. Using the position estimate, the controller determines a velocity feedback control signal, which is transmitted back to the robot using the Wi-Fi connection. The delay in the Wi-Fi connection is assumed negligible compared to the image-processing delay. The robot computes and applies the appropriate wheel motion corresponding to the feedback signal, which closes the loop. More details on the hardware implementations are discussed in Section V.

We model and control the robot as a rigid body in the two-dimensional ground plane (see, e.g., [8]). We introduce a (robot) body-fixed frame $\{R\}$ relative to a world-fixed frame $\{W\}$, depicted in Fig. 2. The origin of the robot frame is fixed to the geometrical center of the triangular base of the robot. For both frames, the first two coordinate axes lie on the surface plane, the last coordinate axis is pointing out of the surface plane, therefore $\vec{e}_3^R$ and $\vec{e}_3^W$ are parallel. The robot has three degrees of freedom: translations x and y in directions $\vec{e}_1^R$ and $\vec{e}_2^R$, and a rotation ψ around $\vec{e}_3^R$, respectively.

We denote the pose of the robot relative to the world frame by $p = [x, y, \psi]^\intercal$. The velocity of the robot expressed in the robot frame is denoted by $v^R = [v_x^R, v_y^R, v_\psi^R]^\intercal$. The basic rotation matrix [20], [21]

$$R_z(\psi) = \begin{bmatrix} R(\psi) & 0 \\ 0 & 1 \end{bmatrix}, R(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix}, \psi \in \mathbb{R}$$

relates a pose in the robot frame to a pose in the world frame, such that the velocity in the world frame can be written, for $t \in \mathbb{R}_{\geq 0}$, as

$$\dot{p}(t) = R_z\big(\psi(t)\big)v^R(t). \tag{1}$$

The image-based feedback control input u_C of the robot is taken as the velocity setpoint for the robot in the sense that $v^R(t) = u_C(t)$, $t \in \mathbb{R}_{\geq 0}$. In this paper, we are mainly interested in motions in x and y. To apply the designs for linear systems in [1], we assume the robot orientation to be fixed, i.e., we assume $\psi(t) = \psi_0$ for all $t \in \mathbb{R}_{\geq 0}$, with $\psi_0 \in [0, 2\pi)$. From (1) it follows that we can consider the system dynamics to be given by the linear model

$$\dot{p}(t) = R_z\big(\psi_0\big)u_C(t), \quad t \in \mathbb{R}_{\geq 0}\,. \tag{2}$$

The camera is positioned at a fixed height $h \in \mathbb{R}_{>0}$ above the ground at pose $p^O = [x^O, y^O, \psi^O]^\intercal$ in the robot frame on the ground plane, as depicted in Fig. 2, and has observation frame $\{O\}$ on the surface. A localization algorithm is used to calculate an estimate of the position of the robot, denoted $\hat{p}$, based on the camera images. The 3-D pose of the camera and its relation to the localization problem are discussed in Section IV.

A. Control Problem

The robot aims to follow a predefined (differentiable) reference trajectory $p^{\text{ref}} : \mathbb{R}_{\geq 0} \to \mathbb{R}^2 \times [0, 2\pi)$, where the third element corresponding to the rotation of the robot is a constant equal to ψ_0 for all $t \in \mathbb{R}_{\geq 0}$. A feed-forward control action u^{FF} corresponding to this trajectory for model (2), detailed in Section III, is implemented in the robot. Due to model mismatch and disturbances, the true system does not match model (2) exactly and the robot deviates from the desired trajectory p^{ref}.

We model the effects of model mismatch and the disturbances as independent continuous-time additive Wiener processes on the elements of $\dot{p}$, which in discrete time can be considered as additive Gaussian white noise disturbances. Therefore, we can consider the standard form continuous-time stochastic differential equation plant model [22]

$$dp(t) = (A_c p(t) + B_c u(t))dt + B_w\, dw(t), \quad t \in \mathbb{R}_{\geq 0} \tag{3}$$

where $A_c = 0$, $B_c = R_z\big(\psi_0\big)$, and $B_w = \text{diag}([B_{w,x}, B_{w,y}, B_{w,\psi}])$ where $B_{w,x}, B_{w,y}, B_{w,\psi}$ are the standard deviations of the disturbances, and where $p(t) \in \mathbb{R}^{n_x}$ is the state and $u(t) \in R^{n_u}$ is the applied control input at time $t \in \mathbb{R}_{\geq 0}$, and w is an n_w-dimensional Wiener process with incremental covariance $I_{n_w}\, dt$ [22], where I_{n_w} is an identity matrix of size $n_w \times n_w$. Note that since I_{n_w} and B_w are not time-varying, the variance of the noise is not time-varying in continuous time. Note that while dw is the standard white noise, the units of $B_w dw$ physically correspond to displacement disturbances in terms of linear and angular positions. Moreover, $u = u^{FF} + u^{FB}$ where u^{FB} is the control feedback term. Note that (A_c, B_c) is controllable and B_c has full rank and is invertible.

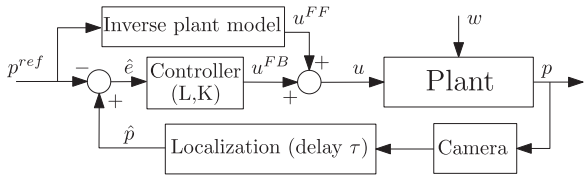

Fig. 3. Reference tracking control scheme with feed-forward and vision-based feedback loop. The error estimate $\hat{e}$ uses the delayed reference. In the current paper, the robot itself is the "Plant."

The control objective is the minimization of a linear quadratic Gaussian type cost function [22]

$$J := \limsup_{T\to\infty} \frac{1}{T} \int_0^T \mathbb{E}[g_c(p(t) - p^{ref}(t), u^{FB}(t))]dt \tag{4}$$

where $g_c(e, u) := e^\intercal Q_c e + u^\intercal R_c u$ with positive definite matrices $Q_c, R_c \succ 0$, which incorporates a cost on the deviation from the reference trajectory $p^{\text{ref}}(t)$ and a penalty for the applied feedback control action $u^{FB}(t)$.

The main control problem considered in this paper is the design of a sampled-data, piecewise constant, control action

$$u^{FB}(t) = u_k \text{ for all } t \in [t_k, t_{k+1}) \tag{5}$$

with $u_k \in \mathbb{R}^3$ and t_k the control update time, $k \in \mathbb{N}$, that minimizes (4) subject to a stochastic localization delay and the process disturbances. In this paper, we show that the delay τ between taking an image and the time of presenting the pose estimate $\hat{p}$ to the controller can be modeled by a stochastic process described by a probability distribution function (pdf) $f : \mathbb{R}_{>0} \to \mathbb{R}$ or the corresponding cumulative distribution function (cdf) $F : \mathbb{R}_{>0} \to [0, 1]$. To address the control problem with stochastic measurement delays, we shall apply the control design ideas for system with stochastic actuation delays presented in [1]. The control strategies, and how they can be applied in the case at hand, are detailed in Section III.

III. Control Design

In this section, we explain the control algorithms implemented in the robot. The velocity control input $u(t)$, $t \in \mathbb{R}$ is implemented digitally and consists of the sum

$$u(t) = u^{FF}(t) + u^{FB}(t), \qquad t \in \mathbb{R}_{\geq 0}\,.$$

First, we discuss the feed-forward control u^{FF}, which is used to steer the robot along a reference trajectory. Second, we discuss the image-based feedback control u^{FB}, which corrects for deviations from the reference and minimizes (4). The complete image-based reference tracking control scheme is depicted in Fig. 3.

A. Feed-Forward Control

Because we are interested in the feedback performance, the feed-forward update rate is significantly higher than the feedback update rate, such that the feed-forward can be considered as ideal with respect to the feedback problem. For a given differentiable reference trajectory $p^{\text{ref}} : \mathbb{R}_{\geq 0} \to \mathbb{R}^2 \times [0, 2\pi)$ and considering (3), the input u^{FF} that satisfies $p(t) = p^{\text{ref}}(t)$ for

all $t \in \mathbb{R}$ if $B_w = 0$ is given by

$$u^{FF\star}(t) = B_c^{-1}\dot{p}^{\text{ref}}(t), \qquad t \in \mathbb{R}_{\geq 0}\,. \tag{6}$$

Recall that $B_c = R_z(\psi_0)$ is a rotation matrix, which is always invertible. As an approximation to (6), in the digital domain, at intervals of $T_{FF} = \frac{1}{20}$ s, corresponding to a 20 Hz update rate, we implemented, for $t \in [nT_{FF}, (n+1)T_{FF}), n \in \mathbb{N}$, the piecewise constant input

$$u^{FF}(t) = B_c^{-1}\frac{1}{T_{FF}}(p^{\text{ref}}((n+1)T_{FF}) - p^{\text{ref}}(nT_{FF}))$$

such that $p(nT_{FF}) = p^{ref}(nT_{FF}), n \in \mathbb{N}$ if $B_w = 0$.

B. Feedback Control With Stochastic Delays

To compensate for the effect of the disturbance w, which is not possible by feed-forward control, the camera images are used in a tracking control configuration. As explained in Section II-A, the localization step can be modeled by a stochastic delay with pdf f. We denote the delay of processing the image obtained at time t_k by τ_k for all $k \in \mathbb{N}$, which for our setup are independent identically distributed (i.i.d.) realizations of distribution f. Next, we discuss how, for case (5) and measurement delay with pdf f and cdf F, we can use the ideas of [1] for event-driven and deadline-driven control design.

First, we define the tracking error $e(t) = p(t) - p^{\text{ref}}(t), t \in \mathbb{R}_{\geq 0}$ resulting in a model for the error dynamics given by

$$de(t) = (A_c e(t) + B_c u^{FB}(t))dt + B_w\, dw(t),\ t \in \mathbb{R}_{\geq 0} \tag{7}$$

which follows from (3) and (6).

By discretization of the error system (7) at sampling times t_k, $k \in \mathbb{N}$, we obtain

$$e_{k+1} = A(h_k)e_k + B(h_k)u_k + w_k$$

where $h_k := t_{k+1} - t_k$ and $A(h) := e^{A_c h}$ and $B(h) := \int_0^h e^{A_c s}B_c ds$.

The disturbance is a sequence of zero-mean independent random vectors $w_k \in \mathbb{R}^{n_w}$, $k \in \mathbb{N}$, with covariance $\mathbb{E}[w_k(w_k)^\intercal] = W(h_k)$, where $W(h) := \int_0^h e^{A_c s}B_w B_w^\intercal e^{A_c^\intercal s} ds$ for $h \in \mathbb{R}_{\geq 0}$. Note that the autocovariance $W(h_k)$ depends only on the sampling interval h_k and not on the sampling instance t_k.

The average cost (4) can be written as

$$J = \limsup_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\sum_{k=0}^{N(T)-1} g(e_k, u_k, h_k)\right] \tag{8}$$

where $N(T) := \min\{L \in \mathbb{N}_{[1,\infty]} \mid \sum_{k=0}^{L} h_k > T\}$ and $g(e,u,h) := e^\intercal Q(h)e + 2e^\intercal S(h)u + u^\intercal R(h)u + \alpha(h)$, with

$$\begin{bmatrix} Q(h) & S(h) \\ S(h)^\intercal & R(h) \end{bmatrix} := \int_0^h e^{\left[\begin{smallmatrix} A_c & B_c \\ 0 & 0 \end{smallmatrix}\right]^\intercal s} \begin{bmatrix} Q_c & 0 \\ 0 & R_c \end{bmatrix} e^{\left[\begin{smallmatrix} A_c & B_c \\ 0 & 0 \end{smallmatrix}\right] s} ds$$

and

$$\alpha(h) := \text{trace}\left(Q_c \int_0^h \int_0^t e^{A_c s}B_w B_w^\intercal e^{A_c^\intercal s} ds dt\right)$$

which is the cost associated with the intersampling behavior of the Wiener process.

Since our system has simple dynamics with $A_c = 0$ and $B_c = R_z(\psi_0)$, we have that $A(h) = I$, $B(h) = hR_z(\psi_0)$, $W(h) = hB_w B_w^\intercal$, and $\alpha(h) = \frac{h^2}{2}\text{trace}(Q_c B_w B_w^\intercal)$. Next, we explain the deadline-driven and event-driven control schemes for this system.

1) Deadline-Driven Control: In deadline-driven control, the control is updated after a fixed deadline, i.e., $h_k = D \in \mathbb{R}_{>0}$ for all $k \in \mathbb{N}$, according to

$$u_k = -K_d \bar{e}_k$$

where $\bar{e}_k$ is an estimate of the current state e_k, obtained using the Kalman filter

$$\bar{e}_k = \begin{cases} \bar{e}_{k-1} + B(D)u_{k-1} + L_d(\hat{e}_{k-1} - \bar{e}_{k-1}) & \text{if } \tau_{k-1} \leq D \\ \bar{e}_{k-1} + B(D)u_{k-1} & \text{if } \tau_{k-1} > D \end{cases}$$

where $\hat{e}_{k-1} = \hat{p}(t_{k-1}) - p^{\text{ref}}(t_{k-1})$ and $\hat{p}(t_{k-1})$ is the pose estimate from the image taken at time t_{k-1}, which is only obtained if the delay is less than the deadline, i.e., if $\tau_{k-1} \leq D$. The control gain K_d is obtained by solving, for positive definite P_d, the Riccati equation

$$\begin{aligned} P_d &= A(D)^\intercal P_d A(D) + Q(D) + K_d^\intercal G_d K_d \\ K_d &= G_d^{-1}(B(D)^\intercal P_d A(D) + S(D)^\intercal) \\ G_d &= B(D)^\intercal P_d B(D) + R(D) \end{aligned}$$

where the constant $D \in \mathbb{R}_{>0}$ can be chosen arbitrarily. Note that the probability that $\tau_{k-1} \leq D$ is given by $F(D)$. Our setup assumes to obtain perfect measurements of the complete state, which is a result of the chosen image acquisition and image-processing methods. The estimator gain is given by $L_d = A(D)$ and the infinite horizon covariance Θ_d of the estimate $\bar{e}_k$ is given by the solution to the Lyapunov equation

$$\Theta_d = \tilde{A}(D)^\intercal \Theta_d \tilde{A}(D) + W(D)$$

which only has a solution for values of $D \in \mathbb{R}$ for which $\tilde{A}(D) := \sqrt{1 - F(D)}A(D)$ is Schur. Since in our case $A(D) = I$ for all $D \in \mathbb{R}_{>0}$, $\tilde{A}(D)$ is Schur if $F(D) > 0$. This control design minimizes (8) for a particular value of $D \in \mathbb{R}_{\geq 0}$, and the minimal value is given by

$$J_d := \frac{1}{D}\left[\text{trace}(P_d W(D) + \Theta_d K_d^\intercal G_d K_d) + \alpha(D)\right]. \tag{9}$$

By computing (9) for a large set of values of $D \in \mathbb{R}$, we determine the optimal value of D that minimizes J_d.

Remark 1: Since $h_k = D \in \mathbb{R}_{>0}$ is a constant for all $k \in \mathbb{N}$ for the deadline-driven case, the control updates of this controller type are periodic.

2) Event-Driven Control: In event-driven control, the control is updated immediately after processing, i.e., $h_k = \tau_k$ for all $k \in \mathbb{N}$, according to

$$u_k = -K_e \bar{e}_k,$$

where $\bar{e}_k$ is an estimate of the current state e_k, obtained using the Kalman filter

$$\bar{e}_k = \bar{e}_{k-1} + B(\tau_{k-1})u_{k-1} + L_e(\hat{e}_{k-1} - \bar{e}_{k-1})$$

where $\hat{e}_{k-1} = \hat{p}(t_{k-1}) - p^{\text{ref}}(t_{k-1})$ and $\hat{p}(t_{k-1})$ is the pose estimate from the image taken at time t_{k-1}. The control gains K_e are obtained by solving, for positive definite P_e, the stochastic Riccati equation [23]

$$\begin{aligned} P_e &= \overline{A(\tau)^\intercal P_e A(\tau)} + \overline{Q(\tau)} + K_e^\intercal G_e K_e \\ K_e &= G_e^{-1}(\overline{B(\tau)^\intercal P_e A(\tau)} + \overline{S(\tau)}^\intercal) \\ G_d &= \overline{B(\tau)^\intercal P_e B(\tau)} + \overline{R(\tau)} \end{aligned}$$

where τ has pdf f and $\overline{X(\tau)^\intercal Y Z(\tau)}$ denotes $\mathbb{E}[X(\tau)^\intercal Y Z(\tau)]$ which can be computed by the Lebesgue–Stieltjes integral $\int_0^\infty X(s)^\intercal Y Z(s) dF(s)$. Note that, in this case, we always obtain a pose estimate $\hat{p}(t_{k-1})$, but the delays τ_k and thus the feedback control update intervals h_k are probabilistic. Note that while the covariance $W(h_k)$ for the deadline-driven approaches was constant, since $h_k = D$, it is time-varying for the event-driven approach due to the dependence of the sampling interval on the varying delay τ_k. The estimator gain is given by $L_e = \overline{A(\tau)}$ and the infinite horizon covariance Θ_e of the estimate $\bar{e}_k$ is given by $\Theta_e = \overline{W(\tau)}$.

This control design minimizes (8) for the event-driven case and the minimal value is given by

$$J_e := \frac{1}{\overline{\tau}}\left[\text{trace}(P_e \overline{W(\tau)} + \Theta_e K_e^\intercal G_e K_e) + \overline{\alpha(\tau)}\right] \tag{10}$$

where $\overline{\tau}$ is the expected value of the delay.

IV. Vision-Based Localization

Vision-based localization is a vast and challenging topic in robotics. One often considered problem is simultaneous localization and mapping (SLAM), which is a well-studied method [24]. To approach this problem, feature detectors as SIFT (Scale-Invariant Feature Transform [25]) and SURF (Speeded Up Robust Features [26]) are often used. However, in the present paper, mapping of the environment, as done in SLAM, is not required and the features (the round markers) have simple geometries. Thus, only the localization problem is considered and advanced feature detectors such as SIFT and SURF are not needed. In computer-vision literature, the RANSAC [2] method is often used to match features in series of images. Since we are mainly interested in the performance of the control algorithms for stochastic delay, we consider a simple vision-based localization procedure.

For reasons of space, we only sketch the steps taken in the localization procedure.

The exact 3-D pose of the camera with respect to $\{R\}$ and its intrinsic parameters, including a geometric distortion model [24], are determined *a priori* by means of camera calibration using the Camera Calibrator from the Computer Vision Toolbox [25] of MATLAB. By an exact perspective transformation from the camera pose, using the fact that all markers lie in the same plane, computations can be done with respect to the observation frame $\{O\}$ which is fixed w.r.t. $\{R\}$. The image-processing software in the base station has a database of all markers in the environment with their color and their positions relative to the world frame $\{W\}$. After the base station receives a new camera image, it accurately determines the color and location of the centroids of the colored markers with respect to $\{O\}$ by using distortion correction, Hue-Saturation-Value (HSV) color segmentation, morphological cleaning and closing, and blob analysis from [25] and the Image Processing Toolbox (IPT) [26]. Then it implements a RANSAC-based [2] localization algorithm, detailed in Section IV-A, to match the configuration of the observed markers to the configuration of a random set of markers in the database. Finding a positive match results in an estimate of the robot pose in the environment. Due to the random and memory-less nature of the algorithm, the completion time of the algorithm is random, independent and identically distributed.

A. Localization Algorithm

The database set $\mathcal{D} = \{d_i\}_{i\in\{1,2,\ldots\}}$ contains the position vectors

$$\vec{\boldsymbol{d}}_i = \begin{bmatrix} \vec{\boldsymbol{e}}_1^W & \vec{\boldsymbol{e}}_2^W \end{bmatrix} \begin{bmatrix} d_{i,x} \\ d_{i,y} \end{bmatrix} \text{ for all } i \in \{1,2,\ldots\} \tag{11}$$

of the centers of all markers in the world frame, as depicted in Fig. 2, the color of the marker, and its Euclidean distance to each other marker, given, for all $i, j \in \{1,2,\ldots\}$, by

$$\boldsymbol{l}(d_i, d_j) = \sqrt{(d_{i,x} - d_{j,x})^2 + (d_{i,y} - d_{j,y})^2}.$$

Markers observed in a new image are represented in the observation set $\mathcal{O} = \{o_i\}_{i\in\{1,2,\ldots\}}$ and we determine vectors

$$\vec{\boldsymbol{o}}_i = \begin{bmatrix} \vec{\boldsymbol{e}}_1^O & \vec{\boldsymbol{e}}_2^O \end{bmatrix} \begin{bmatrix} o_{i,x} \\ o_{i,y} \end{bmatrix} \text{ for all } i \in \{1,2,\ldots\} \tag{12}$$

pointing to the centroids of the landmarks in the observation frame $\{O\}$, as depicted in Fig. 2, as well as the color and Euclidean interdistances $\boldsymbol{l}(o_i, o_j)$, $i, j \in \{1,2,\ldots\}$. Note that $\vec{\boldsymbol{e}}_1^O, \vec{\boldsymbol{e}}_2^O$ are the coordinate axes of frame $\{O\}$ depicted in Fig. 2.

The pose of the robot can be determined by matching elements in $\mathcal{O}$ and $\mathcal{D}$. A correct match between an element $o \in \mathcal{O}$, with o_x, o_y as defined in (12), and an element $d \in \mathcal{D}$, with d_x, d_y as defined in (11), should satisfy the relation

$$\begin{bmatrix} d_x \\ d_y \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + R(\psi)\begin{bmatrix} x^O \\ y^O \end{bmatrix} + R(\psi)R(\psi^O)\begin{bmatrix} o_x \\ o_y \end{bmatrix} \tag{13}$$

where $p^O = [x^O, y^O, \psi^O]^\intercal$ is the pose of frame $\{O\}$ w.r.t. frame $\{R\}$, which follows from coordinate transformations. Note that $p^O = [x^O, y^O, \psi^O]^\intercal$ is fixed and known and that (13) is nonlinear in the rotation ψ.

Note that (13) provides only two nonlinear equations for three unknowns (x, y, ψ), which is not sufficient to obtain a unique pose. Therefore, we need two or more markers from each set $\mathcal{O}$ and $\mathcal{D}$ at the same time, which provides us with four or more equations for three unknowns. We solve this problem in two steps. First, we solve the nonlinear difference equations (case with two markers $o_1, o_2 \in \mathcal{O}, d_1, d_2 \in \mathcal{D}$)

$$R(\hat{\psi})R(\psi^O)\begin{bmatrix} o_{1,x} - o_{2,x} \\ o_{1,y} - o_{2,y} \end{bmatrix} - \begin{bmatrix} d_{1,x} - d_{2,x} \\ d_{1,y} - d_{2,y} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{14}$$

Algorithm 1: RANSAC Pose Estimation Algorithm.

Step 1: Random sampling

Randomly select $o_i, o_j \in \mathcal{O}, i \neq j$ and $d_l, d_m \in \mathcal{D}, l \neq m$ with matching colors for the pairs (i, l) and (j, m). For some small tolerance ϵ_1, check the condition

$$\frac{|\boldsymbol{l}(d_l, d_m) - \boldsymbol{l}(o_i, o_j)|}{\boldsymbol{l}(o_i, o_j)} \leq \epsilon_1. \tag{15}$$

Condition (15) checks whether the Euclidean distances of the pairs (i, j) and (l, m) are the same, otherwise the pairs cannot correspond to a match. If (15) is met, continue to Step 2, otherwise return to Step 1 and take new pairs of markers.

Step 2: Model estimation

Estimate $\hat{p} = [\hat{x}, \hat{y}, \hat{\psi}]^\intercal$ by the procedure explained above, based on (14).

Step 3: Consensus

Select randomly an element $o_k \in \mathcal{O} \setminus \{o_i, o_j\}$. Map o_k to a point $\tilde{d}$ in the world frame by

$$\begin{bmatrix} \tilde{d}_x \\ \tilde{d}_y \end{bmatrix} = \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} + R(\hat{\psi}) \begin{bmatrix} x^O \\ y^O \end{bmatrix} + R(\hat{\psi}) R(\psi^O) \begin{bmatrix} o_{k,x} \\ o_{k,y} \end{bmatrix}. \tag{16}$$

Compute the Euclidean distance $\boldsymbol{l}(\tilde{d}, d_n)$ to all markers $d_n \in \mathcal{D} \setminus \{l, m\}$ (with the same color). If for some n, for some small tolerance ϵ_2, $\boldsymbol{l}(\tilde{d}, d_n) \leq \epsilon_2$, then the marker o_k is explained. Otherwise, the pairs (o_i, o_j) and (d_l, d_m) are rejected.

This procedure can be repeated for all other elements in $\mathcal{O} \setminus \{o_i, o_j\}$. If a chosen sufficient number of explained markers (at least one), is reached, the algorithm accepts the pose p to be equal to the estimate $\hat{p}$. Otherwise the algorithm returns to Step 1.

for $\hat{\psi}$ by Newton's method [27] with initial value ψ_0.Second, we substitute $\hat{\psi}$ for ψ in (13) for each marker and solve for x and y in a least-squares sense to obtain estimates $\hat{x}$ and $\hat{y}$. By considering more markers, the pose estimation can be made more robust against measurement errors.

For the open problem of selecting markers in $\mathcal{O}$ and $\mathcal{D}$ to use to solve the pose estimation problem, we take the RANSAC [2] algorithm, which is often applied in practical applications of computer vision to solve problems of this nature. The application of RANSAC to the pose estimation problem is explained in the RANSAC pose estimation algorithm (see Algorithm 1).

V. Experimental Results

In this section, we explain the steps taken to identify the relevant system parameters and we provide the experimental results.

A. Robot Setup

The omnidirectional robot platform provided by SuperDroid Robots [28] has three 4-in 90° Swedish (omni) wheels powered by IG32 265RPM gear motors with quadrature encoders (538 counts per revolution) and gearbox (gear ratio $g_r = 0.0521$) separately controlled by an Arduino UNO at 1 kHz. Image capturing by the webcam (Logitech C525, configured as an IP camera), communication with the base station, and sending instructions to the Arduino UNO (via python script over USB serial bus) are implemented on a RaspberryPi 3 (RPi3). Wireless data is transferred via user datagram protocol connection using a dedicated wireless router (D-link DIR-850L). The processing and control platform (base station) is an Intel i7-3630QM CPU @2.40 GHz with 8 GB RAM running 64-bit Windows 7.

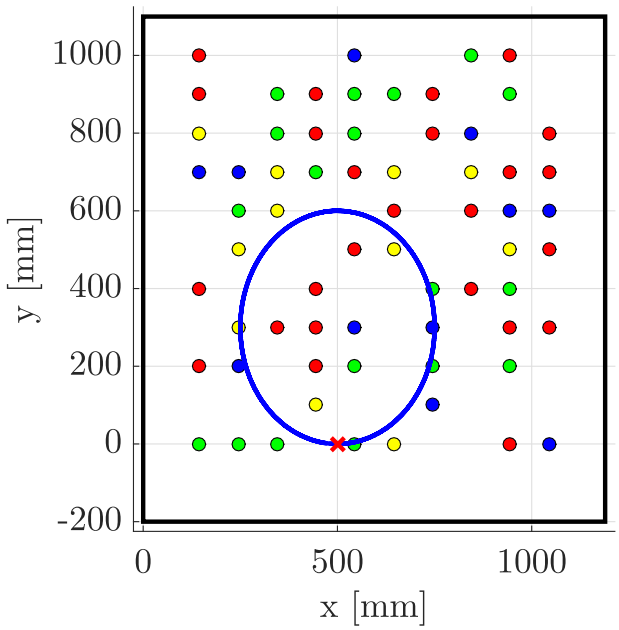

Fig. 4. Environment (in the world frame) with markers and reference (blue) with initial position (red X).

During experiments, an external computer samples the camera at a fast rate of $f_{\text{validate}} = 25$ Hz for validation data.

B. Imaging and Processing Parameters

Images of 320×240 RGB pixels are obtained from the camera. As pose estimation parameters, we took: $\epsilon_1 = 0.1$, a tolerance of 10^{-6} for $\hat{\psi}$, at least three markers required, and at least 85% of the observed markers should be explained by the database with $\epsilon_2 = 0.015$ m. Since several patterns of markers are repeated in the environment (see Fig. 4), a pose could be falsely accepted due to these artifacts. Since we are not interested in dealing with this case in this paper, we add an additional check whether an accepted pose is within 0.08 m of the current estimate of the pose. If this is not the case, we continue the localization algorithm. As we will see next, the process noise is small enough such that it is highly unlikely to trigger this threshold.

While it is possible to use more elaborate schemes with variations on the image size, for which compression methods could limit the impact on changes in delays, the imaging and processing methods are assumed to have fixed settings and variations are not considered in this paper.

TABLE I
ELLIPSOIDAL REFERENCE PARAMETERS

$R_{x,0}$[m]	$R_{y,0}$[m]	ψ_0 [rad]	R_x [m]	R_y [m]	f_{ref} [Hz]	ϕ_0 [rad]
0.50	0.00	$\frac{11}{6}\pi$	0.25	0.30	$\frac{1}{20} = 0.05$	-0.5π

C. Environment and Reference Trajectory

The workspace of the robot contains 65 colored markers (25 red, 18 green, 11 yellow, and 11 blue) that are randomly placed on a square grid with 10 cm distances.

The robot origin $\{R\}$ is chosen at the center of the robot as in Fig. 2. During the experiments we use the following ellipsoidal reference trajectory

$$p^{\text{ref}}(t) = \begin{bmatrix} x^{\text{ref}}(t) \\ y^{\text{ref}}(t) \\ \psi^{\text{ref}}(t) \end{bmatrix} = \begin{bmatrix} R_x \cos(2\pi f_{\text{ref}} t + \phi_0) + R_{x,0} \\ R_y \sin(2\pi f_{\text{ref}} t + \phi_0) + R_{y,0} \\ \psi_0 \end{bmatrix}$$

with the parameters listed in Table I, which is depicted in the robot workspace in Fig. 4. Before each experiment, the robot is driven to $p^{\text{ref}}(0) = [0.5, 0, \frac{11}{6}\pi]^\intercal$ with high accuracy, such that we can assume $p(0) = p^{\text{ref}}(0)$.

Remark 2: Since we are interested in the performance of the high-level vision-based control system, we have mitigated nonlinear effects in the low-level control of the robot by using a smooth reference trajectory with nonzero speed to eliminate start–stop friction effects which cannot be directly modeled/controlled. An ellipsoid was chosen as a variation on a circular path to create a more varied scenario. Using trajectories that incur nonlinear effects in the setup requires the study of controllers for more complex models and is subject of future work.

D. Model of Image Processing

The probability distribution function of the time needed for the steps in Algorithm 1 to process the image data is determined experimentally. A representative series of images (with variations in numbers and colors of markers) is obtained from the robot by open-loop experiments using only the feed-forward control action. By processing the image series with Algorithm 1 on the control and processing platform, a large set of delay values can be obtained (note that images may be reused several times). Computing a normalized histogram of the delay values provides us with a discrete probability distribution for the delay. For our system, the delay is determined with a resolution of 0.001 second intervals. From the histogram, we can obtain a piecewise constant cdf F that can be used to compute the controllers in Section III.

E. Identification of Noise Parameters

The validation data, obtained on the external computer at $f_{\text{validate}} = 25$ Hz, is used to compute the trajectory of the robot at $T_{\text{validate}} = \frac{1}{f_{\text{validate}}} = 0.04$ s intervals during a feed-forward experiment of 200 s. We use linear interpolation with interval $\Delta T \in \mathbb{R}_{>0}$ to obtain an approximation of the actual continuous trajectory of the robot.

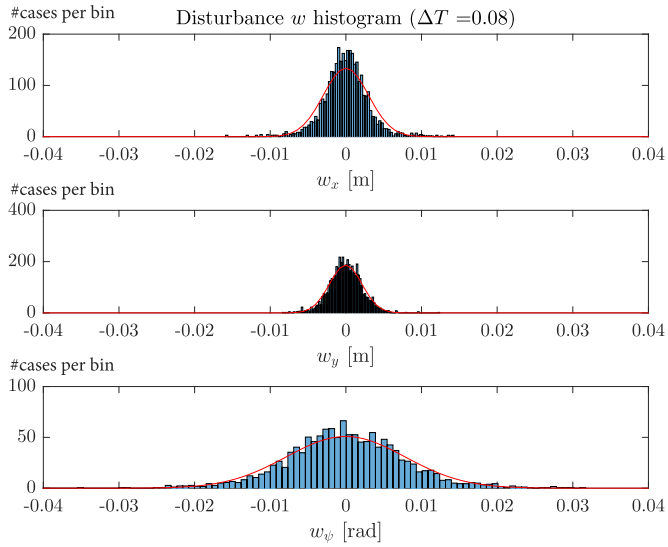

Fig. 5. Experimental process noise probability distribution with normal distribution fit (red).

The knowledge of the feed-forward input and the actual position can be used to determine the process noise parameter B_w. Since u^{FF} is known, we can use the formula

$$w(t_i, \Delta T) = p(t_i + \Delta T) - (p(t_i) + B(\Delta T) u^{FF}(t_v)) \quad (17)$$

to determine the total impact $w(t_i, \Delta T)$ of the process noise in the interval $[t_i, t_i + \Delta T]$ where $t_i \in \mathbb{R}_{[0,200]}$. Here, we assume that u^{FF} is approximately constant in this interval $[t_i, t_i + \Delta T]$, which is supported by the smoothness of the reference trajectory. Due to the stochasticity, the value of $w(t_i, \Delta T)$ varies with the interpolation instant t_i. The process noise is computed using (17) with $\Delta T = 0.08$ s (i.e., at twice the Nyquist sampling interval T_{validate}). The autocorrelation of the noise was computed, showing little to no correlation between noise instances.

The histogram in Fig. 5 depicts the probability distribution of the values of the process noise on the state. We observe that the distribution is approximately Gaussian with zero mean and standard deviations $[\sigma_x(\Delta T), \sigma_y(\Delta T), \sigma_\psi(\Delta T)] = [0.0037, 0.0037, 0.0071]$. Clearly, the artifact threshold discussed in Section V-B is much higher than the standard deviation. Normality tests indicate that for w_x a slight deviation from Gaussianity (outside the standard 95% confidence bounds) may be present. A detailed analysis of the source of this deviation is subject of future work. Because of the limited data available for the normality tests and because this is required in our modeling and design approach, we assume that all disturbances are Gaussian.

Following the assumption in (3) that the process noise is a Wiener process, we assume that the variance $\mathbb{E}[w(t, \Delta T)(w(t, \Delta T))^\intercal] = W(\Delta T) = \Delta T B_w B_w^\intercal$ for any $t \in \mathbb{R}_{\geq 0}$. Hence, we obtain

$$B_w = \frac{1}{\sqrt{\Delta T}} \text{diag}([\sigma_x(\Delta T), \sigma_y(\Delta T), \sigma_\psi(\Delta T)]) = \text{diag}([0.013, 0.013, 0.025]).$$

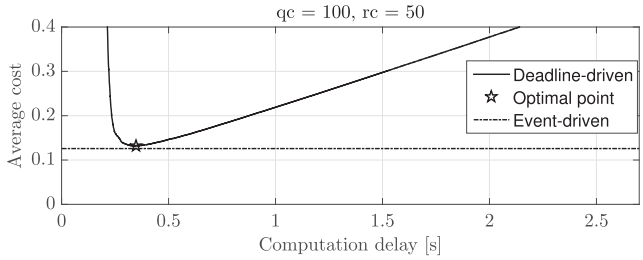

Fig. 6. Theoretical average cost for different values of the deadline D.

F. Control Objective and Theoretical Cost

We consider the cost (4) to be defined by $Q_c = q_c I$ and $R_c = r_c I$ with $q_c = 100$ and $r_c = 50$. Using these values and the previously obtained parameters for the delay and process noise, we compute the theoretical performance (9) and (10). Analytically, we obtain a trade-off for the deadline as depicted in Fig. 6, similar to the one in [1]. The optimal deadline is found to be $D_d = 0.347$ s and $J_d = 0.132$. The event-driven controller is expected to perform better by 4.78% with $J_e = 0.126$ and $\bar{\tau} = 0.334$ s.

Using the methodology in Section III, we find $L_d = L_e = I$ (which follows from the fact that we have $A_c = 0$, full state measurements and no measurement noise) and

$$K_d = \begin{bmatrix} 0.9757 & -0.5633 & 0 \\ 0.5633 & 0.9757 & 0 \\ 0 & 0 & 1.1266 \end{bmatrix},$$

$$K_e = \begin{bmatrix} 0.9004 & -0.5198 & 0 \\ 0.5198 & 0.9004 & 0 \\ 0 & 0 & 1.0397 \end{bmatrix}.$$

We compare the results to a conservative (worst-case) deadline-driven design (which we will refer to with abbreviation *d-con*, using a deadline $D_{d\text{-}con} = 1$ s, corresponding to a traditional periodic design in which 99% of the images produce a pose estimate at the deadline. The gains are given by $L_{d\text{-}con} = I$ and

$$K_{d\text{-}con} = \begin{bmatrix} 0.6853 & -0.3956 & 0 \\ 0.3956 & 0.6853 & 0 \\ 0 & 0 & 0.7913 \end{bmatrix}$$

for an expected performance of $J_{d\text{-}con} = 0.219$, which is 65–75% worse than the proposed designs. Conversely, the proposed designs are expected to perform approximately 40% better than the traditional design.

G. Experimental Control Performance

In this subsection, we validate the theoretical results by performing experiments with each type of controller for $t \in [0, 100]$ s in which the robot follows the ellipsoidal reference five times. Recall that, for brevity, we sometimes refer to the event-driven, optimal deadline-driven, and conservative deadline control policies by *e*, d, and *d-con*, respectively. The pose and pose error for each policy are depicted in Figs. 7 and 8, respectively. The feedback input is depicted in Fig. 9. The error for the conservative controller is generally larger, as expected, causing larger control inputs. In Section V-E, we have identified that $B_w = \text{diag}([0.013, 0.013, 0.025])$. Hence, for the deadline-driven approaches with $D_d = 0.347$ s and $D_{d\text{-}con} = 1$ s, the autocovariance is given by $W(D_d) = \text{diag}[(5.86 \times 10^{-5}, 5.86 \times 10^{-5}, 21.7 \times 10^{-5})]$ and $W(D_{d\text{-}con}) = \text{diag}[(16.9 \times 10^{-5}, 16.9 \times 10^{-5}, 62.5 \times 10^{-5})]$, respectively. For the event-driven approach, the autocovariance depends on the varying delay. The autocovariances of the three methods are depicted in Fig. 10.

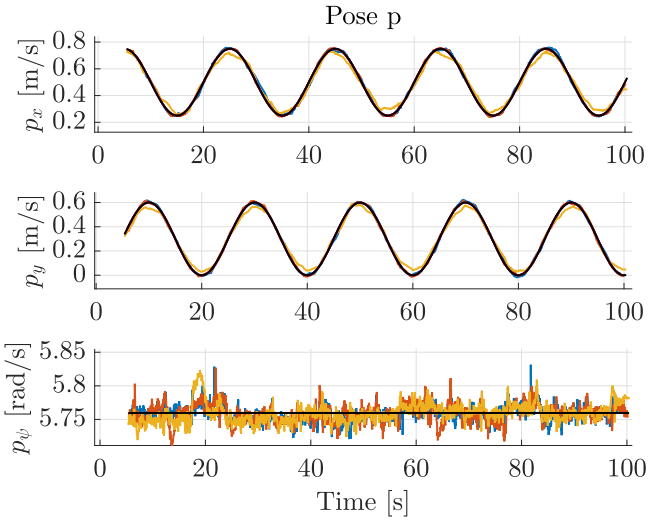

Fig. 7. Robot pose for each control [*e* in red, *d(-optimal)* in blue, *d-conservative* in yellow, and reference in black].

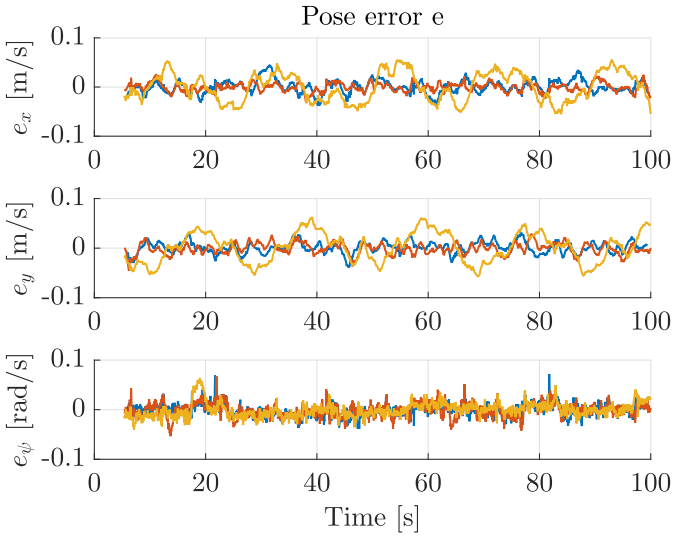

Fig. 8. Pose error for each control [*e* in red, *d(-optimal)* in blue, and *d-conservative* in yellow].

The transient starting behavior (up to 7 s) is omitted to analyze only the average cost (4). Using the error and input data, we compute the experimental performance of the methods to be given by $J_d^{\text{exp}} = 0.0571$, $J_e^{\text{exp}} = 0.0564$, and $J_{d\text{-}con}^{\text{exp}} = 0.2228$. This shows that the event-driven controller indeed performs slightly better than the optimal deadline-driven controller

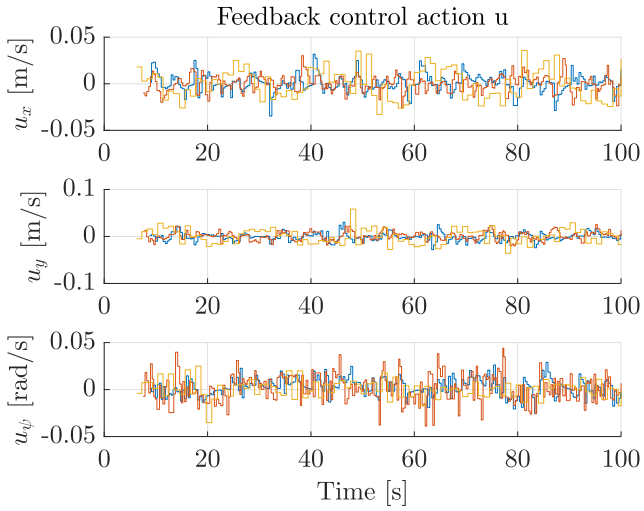

Fig. 9. Feedback input signal for each control [*e* in red, *d(-optimal)* in blue, and *d-conservative* in yellow].

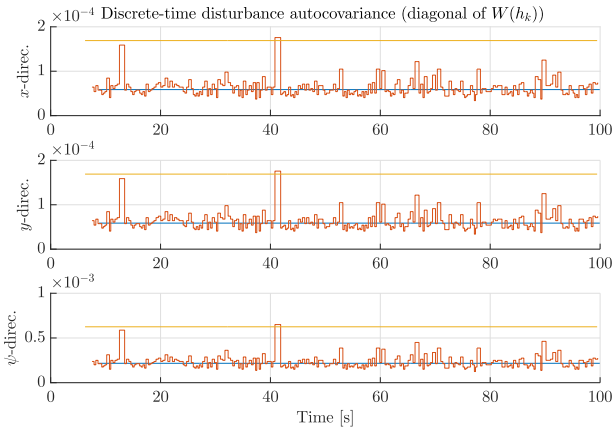

Fig. 10. Discrete-time disturbance autocovariance [*e* in red, *d(-optimal)* in blue, and *d-conservative* in yellow].

(1.3%), as expected from the theoretical results, and that both proposed methods perform significantly better than the conservative design (by almost 75%). The experimental results support the theoretical results and, in particular, the claim that the proposed methods to deal with stochastic delay outperform a traditional worst-case periodic design. While $J^{\text{exp}}_{d\text{-}con}$ approximates the theoretical cost closely, the experimental performance for J^{exp}_{d} and J^{exp}_{e} is significantly lower than the theoretical cost J_d and J_e. We believe that this is due to unmodeled (nonlinear) effects (in the low-level control) which causes very small disturbances to be poorly suppressed.

In Fig. 11, the processing delay for each of the methods is depicted, showing the accumulation of delay at the deadlines (0.33 and 1 s) for the deadline-driven implementations. The autocorrelation of the samples was computed, showing no significant correlation between samples, thereby validating the i.i.d. assumption.

The choice of feedback sampling period was dictated by the image-processing steps, the choice of the feed-forward rate at 20 Hz was sufficient to approximately follow the chosen

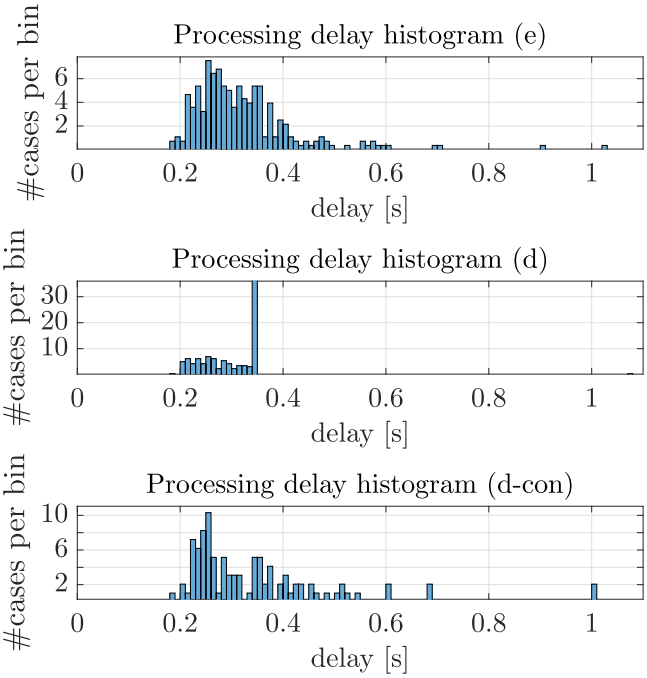

Fig. 11. Processing delay histogram for each control method.

reference because of the limited aggressiveness of the stochastic disturbances.

VI. Conclusion

In this paper, we experimentally demonstrated the strength of model-based event-driven and deadline-driven control algorithms based on [1] for real-time vision-based feedback control. The results illustrated that model-based methods for stochastic measurement delays could be applied in practice and yield significant performance gains compared to traditional (periodic worst-case) designs. The fact that we considered omnidirectional robots with a fixed orientation allowed us to apply the framework for control of linear systems to control the two-dimensional position; considering robots with nonholonomic constraints would entail extensions to the nonlinear realm and would require further research of the control framework. In future work, more complex robot models, models of "smarter" localization methods and robustness studies will be of interest. Moreover, the extent to which our results can be applied to control loops with delays induced by vision algorithms relying on (deep) neural networks is a topic of interest. Future studies should also investigate the more complex dynamics and uncertainties of more advanced physical robots.

References

[1] S. Prakash, E. P. van Horssen, D. Antunes, and W. P. M. H. Heemels, "Self-triggered and event-driven control for linear systems with stochastic delays," in *Proc. Amer. Control Conf. (ACC)*, 2017, pp. 3023–3028.

[2] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[3] S. Se, D. G. Lowe, and J. J. Little, "Vision-based global localization and mapping for mobile robots," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 364–375, Jun. 2005.

[4] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual navigation for mobile robots: A survey," *J. Intell. Robot. Syst.*, vol. 53, p. 263, Nov. 2008.

[5] L. Jayatilleke and N. Zhang, "Landmark-based localization for unmanned aerial vehicles," in *Proc. IEEE Int. Syst. Conf. (SysCon)*, Apr. 2013, pp. 448–451.

[6] R. Lin, Z. Wang, R. Sun, and L. Sun, "Image features-based mobile robot localization," in *Proc. IEEE Int. Conf. Inf. Autom.*, 2012, pp. 304–310.

[7] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: A survey," *Artif. Intell. Rev.*, vol. 43, no. 1, pp. 55–81, Jan. 2015.

[8] P. Corke, *Robotics, Vision and Control – Fundamental Algorithms in MATLAB* (Springer Tracts in Advanced Robotics, vol. 73). Germany: Springer, 2011.

[9] S.-S. Wang and B.-S. Chen, "Optimal model-matching control for time-delay systems," *Int. J. Control*, vol. 47, no. 3, pp. 883–894, 1988.

[10] S. Yi and A. G. Ulsoy, "Time-delayed vision-based dc motor control via rightmost eigenvalue assignment," in *Proc. Amer. Control Conf.*, 2014, pp. 5564–5569.

[11] H. Wu, L. Lou, C. C. Chen, S. Hirche, and K. Kuhnlenz, "Cloud-based networked visual servo control," *IEEE Trans. Ind. Electron.*, vol. 60, no. 2, pp. 554–566, Feb. 2013.

[12] L. R. G. Carrillo, E. S. Espinoza, and S. Mondié, "Controller's parameters tuning in presence of time-delay measurements: An application to vision-based quad-rotor navigation," in *Proc. 51st IEEE Conf. Decision Control (CDC)*, 2012, pp. 5667–5672.

[13] P. Krautgartner and M. Vincze, "Performance evaluation of vision-based control tasks," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1998, pp. 2315–2320.

[14] P. M. Sharkey and D. W. Murray, "Delays versus performance of visually guided systems," *IEEE Proc. - Control Theory Appl.*, vol. 143, no. 5, pp. 436–447, Sep. 1996.

[15] J. Nilsson, B. Bernhardsson, and B. Wittenmark, "Stochastic analysis and control of real-time systems with random time delays," *Automatica*, vol. 34, no. 1, pp. 57–64, Jan. 1998.

[16] B. Lincoln and B. Bernhardsson, *Optimal Control Over Networks With Long Random Delays*. Perpignan Cedex, France: Laboratoire de Théorie des Systèmes (LTS), University of Perpignan, 2000.

[17] J. Nilsson, "Real-time control systems with delays," Ph.D. dissertation, Dept. Autom. Control, Lund Inst. Technol., Lund, Sweden, 1998.

[18] L. A. Montestruque and P. Antsaklis, "Stability of model-based networked control systems with time-varying transmission times," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1562–1572, Sep. 2004.

[19] B. Demirel, "Architectures and performance analysis of wireless control systems," Ph.D. dissertation, Dept. Autom. Control, Sch. Elect. Eng. KTH Royal Inst. Technol., 2015.

[20] H. P. Oliveira, A. J. Sousa, A. P. Moreira, and P. J. Costa, "Modeling and assessing of omnidirectional robots with three and four wheels," in *Contemporary Robotics – Challenges and Solutions*, A. D. Rodic, Ed., 2009. [Online]. Available: https://www.intechopen.com/books/contemporary-robotics-challenges-and-solutions/modeling-and-assessing-of-omni-directional-robots-with-three-and-four-wheels

[21] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control* (Advanced Textbooks in Control and Signal Processing). London: Springer, 2010.

[22] K. J. Åström, *Introduction to Stochastic Control Theory*. Amsterdam: Elsevier, 1970.

[23] W. L. de Koning, "Infinite horizon optimal control of linear discrete time systems with stochastic parameters," *Automatica*, vol. 18, no. 4, pp. 443–453, Jul. 1982.

[24] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000.

[25] MATLAB, *Computer Vision Toolbox v6.2*. Natick, Massachusetts: The MathWorks Inc., 2016.

[26] MATLAB, *Image Processing Toolbox v9.2*. Natick, Massachusetts: The MathWorks Inc., 2016.

[27] M. T. Heath, *Scientific Computing: An Introductory Survey*, 2nd ed. New York City, NY, USA: McGraw-Hill, 1996.

[28] SuperDroid Robots – Robot Kits, Parts, and Custom Solutions. [Online]. Available: https://superdroidrobots.com

Eelco P. van Horssen received the M.Sc. degree (cum laude) in electrical engineering and the Ph.D. degree in mechanical engineering from the Eindhoven University of Technology (TU/e), Eindhoven, The Netherlands, in 2013 and 2018, respectively.

He is currently a dynamics, systems, and control engineer in industry.

Jeroen A. A. van Hooijdonk was born in Breda, The Netherlands. He received the M.Sc. degree in mechanical engineering from the Eindhoven University of Technology, Eindhoven, The Netherlands, in 2016.

Currently, he works as a Systems and Control Technologist within the Mechatronic Equipment Department of Philips Innovation Services, The Netherlands. His research interests include the theoretical and experimental analysis and design of servo control systems with the application to high-precision mechatronic and medical equipment.

Duarte Antunes (M'13) was born in Viseu, Portugal. He received the Licenciatura in electrical and computer engineering and the Ph.D. degree (cum laude) in automatic control from the Instituto Superior Técnico (IST), Lisbon, Portugal, in 2005 and 2011, respectively.

From 2011 to 2013 he held a postdoctoral position at the Eindhoven University of Technology (TU/e). He is currently an Assistant Professor with the Department of Mechanical Engineering of TU/e. His research interests include networked control systems, stochastic control, dynamic programming, distributed optimization, robotics, autonomous systems, and systems biology.

W. P. M. H. Heemels (M'07-SM'10-F'16) received the M.Sc. degree in mathematics and the Ph.D. degree in control theory (both summa cum laude) from the Eindhoven University of Technology (TU/e), Eindhoven, The Netherlands, in 1995 and 1999, respectively.

After being an Assistant Professor at the Electrical Engineering Department, TU/e, and a Research Fellow at the Embedded Systems Institute (ESI), he is currently a Full Professor in the Control Systems Technology Group, Mechanical Engineering Department, TU/e. He held visiting research positions at the Swiss Federal Institute of Technology (ETH), Zurich, Switzerland (2001), at Océ, Venlo, The Netherlands (2004), and at the University of California at Santa Barbara, USA (2008). His current research interests include general system and control theory, hybrid and cyber-physical systems, networked and event-triggered control, and constrained systems including model predictive control.

Professor Dr. Heemels serves/d on the editorial boards of Automatica, NAHS, IEEE TAC, and Annual Reviews in Control, general/program (co)chair of many conferences including ADHS'12, NECSYS'13, ECC'13, ADHS'18, and ECC'21. He is a VICI Laureate of the Netherlands Organisation for Scientific Research (NWO), Chair of the International Federation of Automatic Control (IFAC) Technical Committee on Networked Systems, and a Fellow of IEEE.