Event-based Estimation and Control for Remote Robot Operation with Reduced Communication

Sebastian Trimpe[1] and Jonas Buchli[2]

***Abstract*— An event-based communication framework for remote operation of a robot via a bandwidth-limited network is proposed. The robot sends state and environment estimation data to the operator, and the operator transmits updated control commands or policies to the robot. Event-based communication protocols are designed to ensure that data is transmitted only when required: the robot sends new estimation data only if this yields a significant information gain at the operator, and the operator transmits an updated control policy only if this comes with a significant improvement in control performance. The developed framework is modular and can be used with any standard estimation and control algorithms. Simulation results of a robotic arm highlight its potential for an efficient use of limited communication resources, for example, in disaster-response scenarios such as the DARPA Robotics Challenge.**

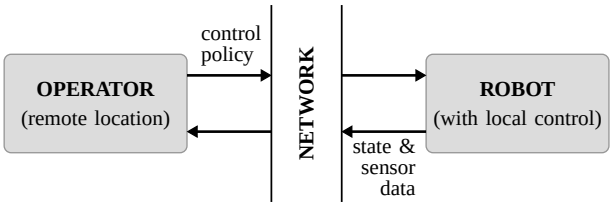


Fig. 1. An operator supervises the mission of a robot from a remote site by monitoring real-time state and sensor data received from the robot, and by sending updated control policies to the robot. Communication between the operator and the robot is via a resource-constraint network, hence data exchange shall be limited to instants when new data is necessary.

I. Introduction

Autonomous robots remotely operated by a human from a remote location are essential assets in current and future disaster-response scenarios; see, for example, the ongoing decommissioning of the Fukushima nuclear power plant [1]. Harsh conditions that are persisting in such scenarios often lead to degraded communication with limited bandwidth between the robot and the operator. For an efficient operation, communication must be managed such that only necessary data is transmitted in order to avoid the congestion of the communication network with irrelevant data. Because of the importance of resource-constraint communication in real-world applications, the ongoing DARPA Robotics Challenge (DRC) [2] requires the participating teams to control their robots under limited communication [3].

In this paper, we consider the remote operation scenario shown in Fig. 1. For monitoring purposes, the robot sends state estimation data, which it accumulates from its local sensors, to the operator. The operator influences the robot's action by sending new input trajectories or control policies. Typically, communication between the components occurs periodically at fixed rates. While periodic communication allows for comparably straightforward analysis and design of the remote control system, it comes with a fundamental limitation: communication instants are predetermined and not chosen in relation to the system's current operating conditions, or the information content of the data. Therefore, this can lead to congestion of the network with irrelevant data and an inefficient use of the available communication resources. Herein, we develop a framework by which the communication from operator to robot, and vice versa, is managed automatically by event-triggering mechanisms ensuring that data exchange happens *only when necessary*.

Recently, event-based communication has received a lot of attention in the controls community as an alternative to periodic (i.e. time-based) communication (see recent surveys [4]–[6]). The underlying idea of event-based methods for estimation and control is to trigger the communication of data between the components of a control system if, and only if, an update is required to meet a certain specification (e.g. stability, estimation or control performance).

We apply event-based estimation and control to the remote robot operation problem in Fig. 1. For the robot-to-operator communication, we follow ideas from event-based state estimation [7], [8]. According to these, the receiver (here, the operator) relies on model-based state predictions at times when the sender (the robot) does not transmit new data. The sender implements a copy of this state predictor, and compares the predictor's information to its own estimator information (which exploits all sensor data). An updated state estimate is sent to the operator only if the prediction is not "good enough" already. The triggering decision is made by comparing the difference of the two state probability density functions (PDFs) as suggested in [9].

For the operator-to-robot link, we implement an event-based control scheme. While most work on event-based control considers the transmission of input commands (see e.g. [5]), we consider a more general framework, where we can also communicate control policies. The triggering decisions are based on the difference of the policy that is currently executed by the robot and the updated policy by

This work was supported by the Max Planck Society and a Swiss National Science Foundation Professorship award to Jonas Buchli.

[1]Sebastian Trimpe is with the Autonomous Motion Department at the Max Planck Institute for Intelligent Systems, 72076 Tübingen, Germany strimpe@tuebingen.mpg.de

[2]Jonas Buchli is with the Agile & Dexterous Robotics Lab at the Institute of Robotics and Intelligent Systems, ETH Zurich, Switzerland buchlij@ethz.ch

Accepted final version. To appear in *2015 IEEE International Conference on Robotics and Automation*.

the operator. This difference is measured in terms of an associated control cost.

The problem of limited communication specifically for the DRC scenario is also addressed in [10]. Therein, the authors consider a manipulation task and develop a framework, in which the human operator can directly control various communication rates via a user interface. In striving for greater autonomy of the human-robot operation, we seek to automate communication decisions using event-based methods herein. The operator specifies a desired accuracy level for control and estimation, and algorithms at the operator and the robot site automatically determine when to transmit data.

A related problem to the one herein is that of teleoperation (see [11]–[13], for example). The topic encompasses a vast body of literature, which can only be touched upon here. In teleoperation, a machine is operated by a human through a remote communication link with special emphasis on the objectives of stability and telepresence [12]. Therefore, compensating for communication delays is a main focus in control for teleoperation. In the application considered herein, the robot is assumed to posses a certain level of autonomy; in particular, it can still operate safely even at low operator communication rates. The focus of this work is on reducing communication, rather than compensating for delays. Nonetheless, the two problems are related. For example, using an architecture where a partial copy of the robot information is maintained at the operator (state prediction), and a partial copy of the operator information is kept at the robot (control policy) is a common principle also in teleoperation [11]. Event-based approaches have also been suggested in the context of teleoperation. For example, in [13], deadband control is used, where new data is sent if the difference of the current and the last communicated value exceeds a threshold. Here, we use more sophisticated triggers: for estimation, triggers based on the state PDFs measuring their information content; and for control, triggers comparing control policies in terms of their expected performance.

In summary, this paper makes the following contributions:

- Development of a framework for remote operation of autonomous robots with limited communication by applying ideas from event-based estimation and control. The framework is modular in that any standard estimation and control method can be used.
- Generalization of typical event-triggering mechanisms: for estimation, transmit decisions are based on the conditional state PDFs, and for control, they are based on costs associated with control policies.
- Performance guarantees are given in terms of upper bounds on the control and estimation degradation due to the reduction in communication (Facts 1 and 2). These bounds are induced by the event-based architecture and thus independent of the choice of estimation and control algorithms, and they can be adjusted by the user.
- Demonstration of the method's effectiveness in reducing communication between operator and robot through simulations of a robotic arm reaching task.

Outline of the paper: After introducing the individual system components (robot, estimation algorithm, control algorithm) in Sec. II, the main framework for event-based communication in remote robot operation is developed in Sec. III. Section IV presents simulation results of a reaching task highlighting the achievable communication reduction. The paper concludes with remarks in Sec. V.

Notation: We use k as the general time index. For a discrete-time signal s, we use s_k to denote s evaluated at index k, and we write $s_{k_1:k_2}$ with $k_1 \leq k_2$ to refer to the sequence $\{s_{k_1}, s_{k_1+1}, \ldots, s_{k_2}\}$. $\mathrm{E}\left[\cdot|\cdot\right]$ denotes the conditional mean, $\mathrm{Var}\left[\cdot|\cdot\right]$ the conditional variance, and $p(\cdot|\cdot)$ a conditional PDF. $\mathcal{N}(\xi; \mu, V)$ denotes the PDF with argument ξ of a (multivariable) normal distribution with mean μ and variance V.

II. Preliminaries

In this section, we introduce the robot model, the state estimator and the controller to set the stage for the discussion of the event-based architecture in Sec. III. No specific choice of estimation and control algorithms is made to emphasize the modularity of the approach, which is independent of the concrete algorithms. In the simulations in Sec. IV, a specific example is presented.

A. System description

The robot dynamics are described by a general nonlinear, discrete-time model

$$x_{k+1} = f_k(x_k, u_k, v_k) \tag{1}$$

with time index $k \geq 0$, state $x_k \in \mathbb{R}^{n_x}$, control input $u_k \in \mathbb{R}^{n_u}$, and process noise v_k. Measurements y_k by the robot's sensors are described by

$$y_k = h_k(x_k, w_k) \tag{2}$$

where w_k is sensor noise. The PDFs of the noise variables are assumed to be known.

Depending on the application, the dynamics (1) can capture the robot's internal states (e.g. positions, velocities, forces), as well as relevant environment states (e.g. contacts, obstacles). The robot dynamics may include local controllers guaranteeing a certain level of autonomy. In particular, the dynamics (1) are typically stable in our framework, such that safe operation can be guaranteed without requiring input at a high rate from the operator. However, stability of (1) is not strictly necessary for the derivations herein.

We consider an architecture as in Fig. 1, where the robot has an embedded computer, which runs local control and estimation algorithms, and which is connected via a network link to a remote operator (bi-directional communication). For the purpose of this work, we assume that network communication is without data loss. This may be ensured by low-level protocols using acknowledgments. We further abstract communication to be instantaneous without delays.[1]

[1]We remark that delays are not as critical in the scenario considered herein as in, for example, teleoperation [11]–[13], since we assume the robot to have a certain level of autonomy and not require operator inputs at high rates for safe operation.

B. State estimation

Let $p(x_{k+\ell}|y_{0:k}, u_{0:k})$ with $\ell \geq 0$ denote the conditional PDF of the state $x_{k+\ell}$ given all measurements and inputs up to and including time k. For simplicity, we also write $p_{k+\ell|k}$ to denote $p(x_{k+\ell}|y_{0:k}, u_{0:k})$.

We assume that there is some state estimation algorithm $\mathcal{E}$, which computes the conditional PDF p, or an approximation $\hat{p}$, from data $y_{0:k}$, $u_{0:k}$ and the model (1), (2):

$$\begin{aligned} \mathcal{E}: \quad & \textbf{Input:} && \text{data } y_{0:k},\ u_{0:k},\ \text{prediction horizon } \ell \\ & \textbf{Output:} && \text{(approximate) PDF} \\ & && \hat{p}_{k+\ell|k} = \hat{p}(x_{k+\ell}|y_{0:k}, u_{0:k}) \end{aligned} \tag{3}$$

Furthermore, we assume that there is a method defined to extract an estimate $\hat{x}_{k+\ell}$ from the PDF $\hat{p}_{k+\ell|k}$ (to be used for state-feedback control, for example). Typical choices are the mean, or maximum likelihood estimate.

Notice that the algorithm (3) can be used for state estimation (i.e. to compute $\hat{p}_{k|k}$) and state prediction ($\hat{p}_{k+\ell|k}$, $\ell \geq 1$). Standard examples for $\mathcal{E}$ include the following, [14]:

- Kalman filter (KF): for a linear time-invariant system (1), (2) with Gaussian noise, the conditional state PDF is Gaussian; that is, $p(x_{k+\ell}|y_{0:k}, u_{0:k}) = \mathcal{N}(x_{k+\ell}; \hat{x}_{k+\ell|k}, P_{k+\ell|k})$. The KF recursively computes conditional mean and variance, $\hat{x}_{k+\ell|k} = \mathrm{E}\left[x_{k+\ell}|y_{0:k}, u_{0:k}\right]$ and $P_{k+\ell|k} = \mathrm{Var}\left[x_{k+\ell}|y_{0:k}, u_{0:k}\right]$. In this case, $\mathcal{E}$ is exact, i.e. $\hat{p}_{k+\ell|k} = p_{k+\ell|k}$.
- Extended Kalman filter (EKF): recursively computes approximations $\hat{x}_{k+\ell|k} \approx \mathrm{E}\left[x_{k+\ell}|y_{0:k}, u_{0:k}\right]$ and $P_{k+\ell|k} \approx \mathrm{Var}\left[x_{k+\ell}|y_{0:k}, u_{0:k}\right]$ by linearizing (1), (2) about the current estimate. The corresponding Gaussian PDF approximates the true PDF: $\hat{p}(x_{k+\ell}|y_{0:k}, u_{0:k}) = \mathcal{N}(x_{k+\ell}; \hat{x}_{k+\ell|k}, P_{k+\ell|k}) \approx p(x_{k+\ell}|y_{0:k}, u_{0:k})$.

C. Control

We consider a suitable control design method $\mathcal{C}$ that generates a control policy over a time horizon M, given the current robot state $x_{\underline{k}}$ at time $k = \underline{k}$ (or a state estimate) and control objectives (e.g. reference, goal state):

$$\begin{aligned} \mathcal{C}: \quad & \textbf{Input:} && \text{time } \underline{k},\ \text{state } x_{\underline{k}},\ \text{control objectives} \\ & \textbf{Output:} && \text{policy } \pi_{\underline{k}} \end{aligned} \tag{4}$$

where $\pi_{\underline{k}}$ is a time-varying control policy mapping the next M states to inputs:

$$u_k = \pi_{\underline{k}}(k, x_k), \quad k = \underline{k}, \ldots, \underline{k}+M. \tag{5}$$

We emphasize that the index '$\underline{k}$' in $\pi_{\underline{k}}$ stands for the time when the control policy was computed, while the arguments k and x_k are the current time and state, at which the policy is evaluated to compute input u_k.

Notice that $\mathcal{C}$ may either be a computer algorithm, or it may represent the direct input of a human operator. For example, (5) can represent an open-loop input trajectory, which is determined by a human operator placing suitable waypoints for the robot on an interactive map of the environment, given the current PDF of the robot state.

A reasonable and common choice for automatic control design are optimal control methods (see e.g. [15]). With these, a policy $\pi_{\underline{k}}$ is found by minimizing a cost function. Let $J_\pi(\underline{k}, x_{\underline{k}})$ be the cost of applying policy π when starting at state $x_{\underline{k}}$ at time $\underline{k}$, and Π be the set of admissible policies. Then

$$\pi_{\underline{k}} = \arg\min_{\pi \in \Pi} J_\pi(\underline{k}, x_{\underline{k}}) \tag{6}$$

is the corresponding optimal control policy.

We give some examples for $\mathcal{C}$:

- Open-loop input trajectory:

  $$\pi_{\underline{k}}(k, x_k) = r_k \tag{7}$$

  with $r_{\underline{k}:\underline{k}+M}$ a constant sequence.
- Linear quadratic regulator (LQR) [16]: For linear (1), $y_k = x_k$, and an infinite-horizon cost ($M \to \infty$)

  $$J_\pi(\underline{k}, x_{\underline{k}}) = \lim_{M\to\infty} \frac{1}{M} \mathrm{E}\left[\sum_{k=\underline{k}}^{\underline{k}+M} x_k^{\mathrm{T}} Q x_k + (\pi(x_k))^{\mathrm{T}} R \pi(x_k)\right]$$

  with Q and R positive definite weights, the solution to (6) is

  $$\pi_{\underline{k}}(k, x_k) = F x_k \tag{8}$$

  where F is a constant feedback gain, which is independent of $x_{\underline{k}}$ and $\underline{k}$, and can readily be computed from the linear model and weights Q and R (see [16]).
- Iterative linear quadratic Gaussian (iLQG) [17]: For nonlinear (1), $y_k = x_k$, and finite-horizon cost

  $$J_\pi(\underline{k}, x_{\underline{k}}) = \mathrm{E}\left[g_{\underline{k}+M}(x_{\underline{k}+M}) + \sum_{k=\underline{k}}^{\underline{k}+M} g_k(x_k, u_k)\right] \tag{9}$$

  where g_k and $g_{\underline{k}+M}$ denote stage and terminal cost, an *approximate* solution to (6) is given by

  $$\pi_{\underline{k}}(k, x_k) = F_k x_k + r_k \tag{10}$$

  where the sequences for r_k and F_k are obtained by iteratively solving linear-quadratic approximations of the nonlinear control problem (see [17] for details). The solution depends on the starting state $x_{\underline{k}}$.

Notice that the above are examples, and many other approaches for control design are conceivable (such as other iterative, approximate optimal control methods [18]–[20]). The architecture proposed herein is not specific to the choice of control design method $\mathcal{C}$, and neither to the choice of estimation algorithm $\mathcal{E}$.

D. Problem formulation

Figure 2 depicts a straightforward implementation of a supervisory control architecture for an autonomous robot. The robot implements a state estimator $\mathcal{E}$ and controller π_k onboard; that is, under normal conditions, the robot can operate autonomously. For monitoring purposes, the robot communicates its state estimate to the operator at a predetermined, fixed update rate (such as every step k). Due to possibly large computational requirements and to allow for

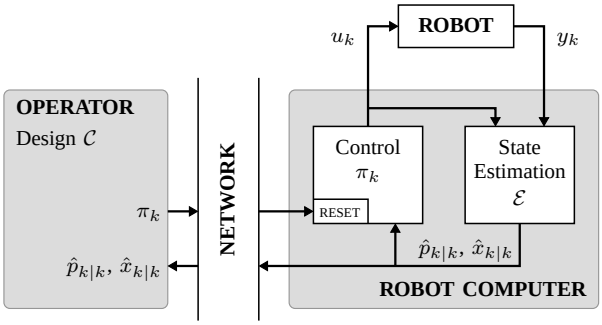


Fig. 2. Standard implementation of a supervisory architecture for controlling a robot. By running state estimator $\mathcal{E}$ and control policy π_k on its embedded computer, the robot has a certain level of autonomy. The operator may send an updated policy over the network link, for example, when the robot task changes. The robot continuously communicates the state PDF $\hat{p}_{k|k}$ (or state estimate $\hat{x}_{k|k}$) to the operator for the purpose of monitoring. Communication between robot and operator is typically periodic, or determined by the operator. In Sec. III, we propose a modified architecture that automatically controls *when* data needs to be send over the network and thus saves communication resources.

human interventions, new control policies $\mathcal{C}$ are computed at the remote operator site. If reference inputs are sent to the robot as in (7), this typically occurs also at a fixed rate. If the operator sends updated policy parameters as in (10), this may happen at a lower rate or on occasion.

The object of this paper is to develop a framework for automating the communication between robot and operator, and this way ensuring an efficient use of the communication resource. Each unit shall transmit data to the other side, only when the other side is in need of new data: the robot communicates new estimation data only if this significantly increases the operator's information, and the operator sends a new policy only if this yields significantly better control performance.

III. Event-Based Communication

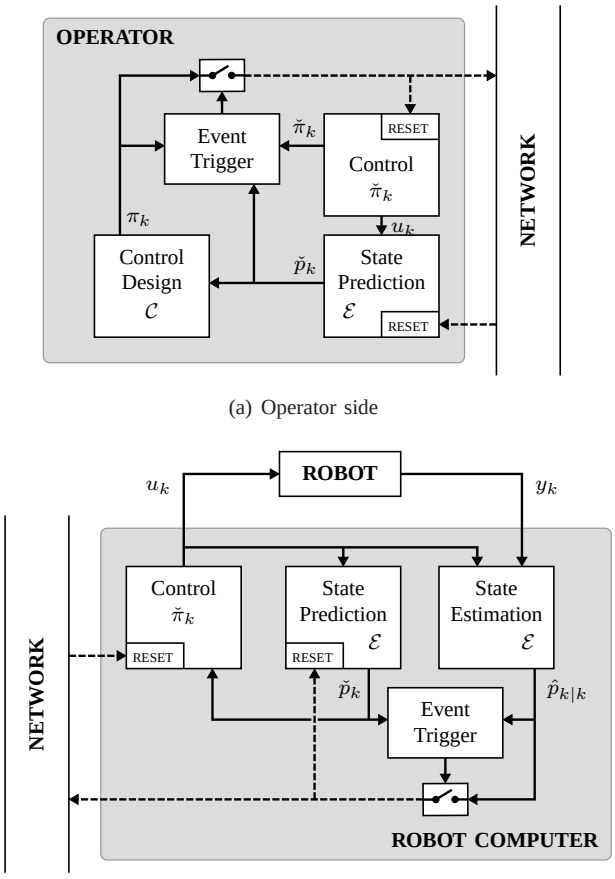
Event-based communication schemes are introduced in this section to address the objective of automatic and efficient communication between robot and operator. The proposed architecture is depicted in Fig. 3 and explained in the following subsections.

A. Robot-to-operator link: Event-based state estimation

The robot computes the current state PDF $\hat{p}_{k|k}$ at every k from the lastest sensor measurements using an estimation algorithm (3) (denoted by the block *State Estimation* in Fig. 3(b)). Given the sensor equipment and estimation algorithm $\mathcal{E}$, $\hat{p}_{k|k}$ is the optimal estimate. The question addressed in this subsection is, when $\hat{p}_{k|k}$ is to be communicated to the operator[2], and how the operator compensates for not always receiving the latest estimate.

The approach for managing the robot-to-operator communication follows the key ideas of event-based state estimation

[2]If instead of the full PDF $\hat{p}_{k|k}$, a lower-dimensional quantity related to the PDF (such as the state estimate $\hat{x}_{k|k}$) is communicated, the following applies with slight modifications.



(a) Operator side

(b) Robot side

Fig. 3. Proposed architecture for remote operation of a robot, using event-based communication for both the robot-to-operator link (explained in Sec. III-A) and the operator-to-robot link (Sec. III-B). Event-based communication is indicated by dashed arrows, while continuous flow of data (periodic communication) is shown by solid arrows.

developed in [7], [8] for reducing communication in networked systems with multiple distributed sensor-estimator-agents. The receiving side (here, the operator) implements the same estimation algorithm $\mathcal{E}$ as the sending side (the robot) in order to compute a prediction of the state PDF (block *State Prediction* in Fig. 3(a)). That is, the operator compensates for not knowing $\hat{p}_{k|k}$ by computing predictions $\hat{p}_{k|\underline{k}^{\text{est}}}$, where $\underline{k}^{\text{est}} < k$ is the last time an update was received. The robot implements a copy of the operator's state predictor (*State Prediction* in Fig. 3(b)), and sends an update $\hat{p}_{k|k}$ only if the prediction is not "good enough."

Let $\gamma_k^{\text{est}} \in \{0, 1\}$ denote the event that the robot communicates $\hat{p}_{k|k}$ at time k ($\gamma_k^{\text{est}} = 1$), or not ($\gamma_k^{\text{est}} = 0$). The operator's knowledge of the state can then be formalized as

$$\check{p}_k = \begin{cases} \hat{p}_{k|k} & \text{if } \gamma_k^{\text{est}} = 1 \\ \hat{p}_{k|\underline{k}^{\text{est}}} & \text{if } \gamma_k^{\text{est}} = 0. \end{cases} \tag{11}$$

The prediction $\hat{p}_{k|\underline{k}^{\text{est}}}$ is computed using the estimation algorithm (3); typically it is obtained by predicting forward the last estimate $\check{p}_{k-1}$ using the process model (1).

The robot also computes the state prediction $\hat{p}_{k|\underline{k}^{\text{est}}}$ and uses the following rule for triggering the communication of

the current estimate $\hat{p}_{k|k}$ to the operator (*Event Trigger* in Fig. 3(b)):

$$\text{transmit } \hat{p}_{k|k} \ (\gamma_k^{\text{est}} = 1) \ \Leftrightarrow \ d^{\text{est}}(\hat{p}_{k|k}, \hat{p}_{k|\underline{k}^{\text{est}}}) \geq \delta^{\text{est}} \quad (12)$$

where $d^{\text{est}}(p_1, p_2) \geq 0$ is some measure of the difference between the PDFs p_1 and p_2 such that $d^{\text{est}}(p_1, p_2) = 0 \Leftrightarrow p_1 = p_2$, and $\delta^{\text{est}} \geq 0$ is a design parameter capturing the tolerable deviation of $\check{p}_k$ from $\hat{p}_{k|k}$. A suitable choice for d^{est} is the Kullback-Leibler (KL) divergence $d^{\text{KL}}(\hat{p}_{k|k}, \hat{p}_{k|\underline{k}^{\text{est}}})$, which can be interpreted as the information loss when the approximation $\hat{p}_{k|\underline{k}^{\text{est}}}$ is used instead of $\hat{p}_{k|k}$, [21]. For n-dimensional normal random variables with $p_1(\xi) = \mathcal{N}(\xi; \mu_1, \Sigma_1)$ and $p_2(\xi) = \mathcal{N}(\xi; \mu_2, \Sigma_2)$, the KL-divergence is

$$d^{\text{KL}}(p_1, p_2) = \frac{1}{2}\big(\operatorname{tr}(\Sigma_2^{-1}\Sigma_1) + (\mu_2 - \mu_1)^{\text{T}}\Sigma_2^{-1}(\mu_2 - \mu_1) - n - \log(\det(\Sigma_1)/\det(\Sigma_2))\big). \quad (13)$$

In (12), the full PDFs are used for making the transmit decision. This can be seen as a generalization of triggers that are based on the mean of a distribution such as in [7], and triggers on the variance [8]. The KL-divergence is an information-theoretic measure, which was also proposed for event-based state estimation in [9].

The following result is immediate from the choice of the event trigger.

Fact 1: The difference between the operator's state prediction $\check{p}_k$ and the robot's state estimate $\hat{p}_{k|k}$ is bounded by δ^{est}; that is,

$$d^{\text{est}}(\hat{p}_{k|k}, \check{p}_k) \leq \delta^{\text{est}} \quad \forall k. \quad (14)$$

Proof: By contradiction. Assume there is a k such that $d^{\text{est}}(\hat{p}_{k|k}, \check{p}_k) > \delta^{\text{est}}$. This implies that $\hat{p}_{k|k} \neq \check{p}_k$, and by (11) that $\check{p}_k = \hat{p}_{k|\underline{k}^{\text{est}}}$. From (12), it then follows $\gamma_k^{\text{est}} = 1$; and from (11), $\check{p}_k = \hat{p}_{k|k}$, which is a contradiction with the assumption. ■

Fact 1 guarantees that the approximate estimation on the operator deviates from the optimal estimation on the robot by no more than δ^{est} (measured by d^{est}). By allowing the estimation on the operator to differ from the optimal estimation on the robot, communication can be saved. The threshold parameter δ^{est} parametrizes this trade-off and is thus used as a tuning parameter.

Remark 1: Fact 1 is independent of the concrete implementation of the state estimation algorithm (3). The meaning of the bound (14) depends on the specific choice of d^{est}. If $d^{\text{est}} = d^{\text{KL}}$ is chosen (as in the simulation example in Sec. IV), the KL-divergence of $\hat{p}_{k|k}$ and $\check{p}_k$ is guaranteed to be bounded. However, other choices can be meaningful. For example, if the mean estimates $\hat{x}_{k|k}$ and $\check{x}_k$ shall be bounded, then $d^{\text{est}}(\hat{p}_{k|k}, \check{p}_k) = \|\hat{x}_{k|k} - \check{x}_k\|$ is a suitable choice.

B. Operator-to-robot link: Event-based control

Using the control design method (4), the operator can recompute the control policy π_k at a step k from its current state prediction $\check{x}_k$, which is extracted from the prediction PDF $\check{p}_k$, and possibly changed objectives. In this section, we address the question when an updated π_k shall be sent to the robot.

Clearly, the need for new control inputs on the robot depends on the application and, in particular, on the level of autonomy of the robot. For example, if π_k represents an M-step open-loop input trajectory as in (7), a new update must happen at least every M steps. For an autonomous robot executing a long-horizon policy, however, it may suffice to send a new task or mission occasionally. We assume that π_k is an optimal policy for an associated cost J_{π_k} as in (6), which provides a common and fairly flexible way of expressing a control objective. The need for a new policy on the robot is measured in terms of the expected cost improvement.

Let $\gamma_k^{\text{ctrl}} \in \{0, 1\}$ denote the event whether or not the operator communicates π_k at time k, and let $\underline{k}^{\text{ctrl}}$ denote the last time k when $\gamma_k^{\text{ctrl}} = 1$. Denote the policy currently used on the robot by $\check{\pi}_k$ (block *Control* in Fig. 3(b)). Then,

$$\check{\pi}_k = \begin{cases} \pi_k & \text{if } \gamma_k^{\text{ctrl}} = 1 \\ \pi_{\underline{k}^{\text{ctrl}}} & \text{if } \gamma_k^{\text{ctrl}} = 0; \end{cases} \quad (15)$$

that is, the robot uses its current policy until it receives a new one from the operator.

The operator keeps track of the current policy $\check{\pi}_k$ (*Control* in Fig. 3(a)) and makes the following transmit decision (*Event Trigger* in Fig. 3(a)):

$$\text{transmit } \pi_k \ (\gamma_k^{\text{ctrl}} = 1) \ \Leftrightarrow \ d^{\text{ctrl}}(\pi_k, \pi_{\underline{k}^{\text{ctrl}}}) \geq \delta^{\text{ctrl}} \quad (16)$$

where $d^{\text{ctrl}}(\pi_1, \pi_2) \geq 0$ is a measure of the difference between policies π_1 and π_2, and $\delta^{\text{ctrl}} \geq 0$ is a design parameter. In the context of optimal control, a reasonable choice for d^{ctrl} is in terms of the associated costs:

$$d^{\text{ctrl}}(\pi_k, \pi_{\underline{k}^{\text{ctrl}}}) = |J_{\pi_k}(k, \check{x}_k) - J_{\pi_{\underline{k}^{\text{ctrl}}}}(k, \check{x}_k)|. \quad (17)$$

Notice that the two policies π_k and $\pi_{\underline{k}^{\text{ctrl}}}$ are typically computed at different times ($k > \underline{k}^{\text{ctrl}}$ for $\gamma_k^{\text{ctrl}} = 0$), yet, they are both evaluated given the latest time and state as known by the operator, namely k and $\check{x}_k$.

The following result is analogous to the estimation case:

Fact 2: The difference between the current control policy $\check{\pi}_k$ employed by the robot and the optimal policy π_k as determined by the operator is bounded by δ^{ctrl}; that is,

$$d^{\text{ctrl}}(\pi_k, \check{\pi}_k) \leq \delta^{\text{ctrl}} \quad \forall k. \quad (18)$$

Again, δ^{ctrl} has the role of trading off communication for performance; in this case, communication from operator to robot, and performance in terms of control. Remark 1 applies analogously to Fact 2.

C. Control on the robot

To complete the explanation of the proposed architecture as depicted in Fig. 3, we emphasize that the robot uses the state prediction $\check{x}_k$ rather than its state estimate $\hat{x}_{k|k}$ to evaluate the current control policy; that is,

$$u_k = \check{\pi}_k(k, \check{x}_k) \quad (19)$$

(cf. block *Control* in Fig. 3(b)). While the robot clearly has access to the improved estimate $\hat{x}_{k|k}$ and could implement

$$u_k = \check{\pi}_k(k, \hat{x}_{k|k}), \quad (20)$$

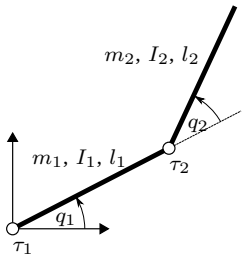


Fig. 4. Two-link robot arm moving in a horizontal plane. The robot is to reach from $q_{\text{init}} = (0, \pi)$ to $q_{\text{goal}} = (\pi/2, 0)$. Physical parameters are given in Table I.

the operator does not know $\hat{x}_{k|k}$ for $k \neq \underline{k}^{\text{est}}$, and thus cannot evaluate the corresponding policy cost $J_{\check{\pi}_k}(k, \hat{x}_{k|k})$. Hence, if (20) is used, a bound as in Fact 2 on the difference of current and optimal control cannot be guaranteed without further communication. Nonetheless, one may consider (20) as an alternative to (19) if local performance is more important than being able to exactly represent this performance at the operator.

IV. Simulation Example

We illustrate the proposed method through simulations of a reaching task for a robotic arm in two dimensions, see Fig. 4. The robot has two links and is to move its end-effector from the starting position ($q_1 = 0$, $q_2 = \pi$) to the goal position ($q_1 = \pi/2$, $q_2 = 0$) in 20 s. Reaching is a basic component of several tasks in the DRC scenario, such as opening a door, removing debris, and closing a valve, [2].

The nonlinear robot model (1) has the states $x = (q_1, q_2, \dot{q}_2, \dot{q}_2)$, joint torque inputs $u = (\tau_1, \tau_2)$, and a sampling time $T_{\text{s}} = 0.01$s. Process noise v_k is modeled as additive Gaussian noise with zero mean and standard deviation (STD) $2 \cdot 10^{-4}$ (independent for each dimension). The robot makes noisy measurements y_k of the joint angles (zero-mean, Gaussian measurement noise w_k with STD 0.01).

A. State estimation

The robot runs an EKF as the state estimation algorithm $\mathcal{E}$. That is, it approximates the state PDF $\hat{p}_{k|k}$ by mean $\hat{x}_{k|k}$ and variance $P_{k|k}$. Analogously, the state predictors on the robot and the operator implement the state prediction step of the EKF using the process model (1) to recursively compute approximate mean $\check{x}_k$ and variance $\check{P}_k$. These are reset to $\hat{x}_{k|k}$ and $P_{k|k}$ at triggering instants $\gamma_k^{\text{est}} = 1$. All estimators are initialized with the known starting configuration.

TABLE I
Essential parameters of the 2D robot arm (cf. Fig. 4).

Link 1		**Link 2**	
mass m_1	3.1 kg	mass m_2	1.8 kg
inertia I_1	0.4 kg m^2	inertia I_2	0.032 kg m^2
length l_1	0.31 m	length l_2	0.34 m

For the triggering rule (12), the KL-divergence (13) is used with $\mu_1 = \hat{x}_{k|k}$, $\mu_2 = \check{x}_k$, $\Sigma_1 = P_{k|k}$, and $\Sigma_2 = \check{P}_k$. As triggering threshold, we chose $\delta^{\text{est}} = 10$.

B. Control

As the control design algorithm $\mathcal{C}$, we take iLQG [17], which provides an approximately optimal solution for the nonlinear control problem. An advantage of iLQG over open-loop optimal control is that it yields a *feedback policy*, which can be expected to be more robust in presence of disturbances, and hence require less communication from the operator. Stage and terminal cost in (9) are chosen as

$$g_k(x_k, u_k) = 10^{-4}\, u_k^{\text{T}} u_k$$
$$g_{\underline{k}+M}(\xi) = (\xi - x_{\text{g}})^{\text{T}}\, \text{diag}(10^4, 10^4, 10^3, 10^3)\, (\xi - x_{\text{g}})$$

where $x_{\text{g}} = (\pi/2, 0, 0, 0)$ is the goal state. The recursions of the iLQG algorithm are terminated if sufficient accuracy is achieved, or after a maximum of 20 iterations.

The approximately optimal iLQG policy (10) is recomputed every 1 s, and correspondingly, the transmit decision (16) is made at the same rate. A new iLQG policy is computed given the current state prediction $\check{x}_k$, and the horizon M in (9) is adjusted to capture the remaining time for task completion. While the EKF computes updates at every step k (i.e. every 0.01 s), the control policy is updated at a lower rate. This is reasonable, since the feedback control system on the robot can be expected to compensate for short-term disturbances. Clearly, if computational performance at the operator permits, the control policy and transmit decisions could also occur at every step. For the control triggering decision (16), the cost difference (17) is used, and $\delta^{\text{ctrl}} = 10^{-6}$ is chosen as the threshold.

C. Simulation results

The system trajectories of an example simulation run over the task horizon of 20 s are shown in Fig. 5, together with the communication decisions for the robot-to-operator link (12) and the operator-to-robot link (16). As can be seen, communication of estimates (γ^{est}) and control policies (γ^{ctrl}) is triggered whenever the corresponding signal exceeds the threshold. Between 6 s and 12 s, the additive state noise was increased by a factor 50 representing, for example, external perturbations, an actuation problem, or a moving platform. We assume that the increased variance is known by the robot estimator, but not at the operator.

Figure 6 shows the average of the triggering signals γ^{est} and γ^{ctrl} over 50 simulation runs. Clearly, for the first six seconds, there is very little communication of estimates and control policies. Communication rates go up in the period from 6 s to 12 s with increased uncertainty (process noise), as expected. Superimposed is the tendency that control decisions are taken toward the end of the horizon: in the beginning, there is still significant time left to reach the goal and the robot can basically be left to its random perturbations. Only toward the end, control action needs to ramp up to ensure reaching the goal state. This phenomenon of delayed decision making is typical for stochastic optimal

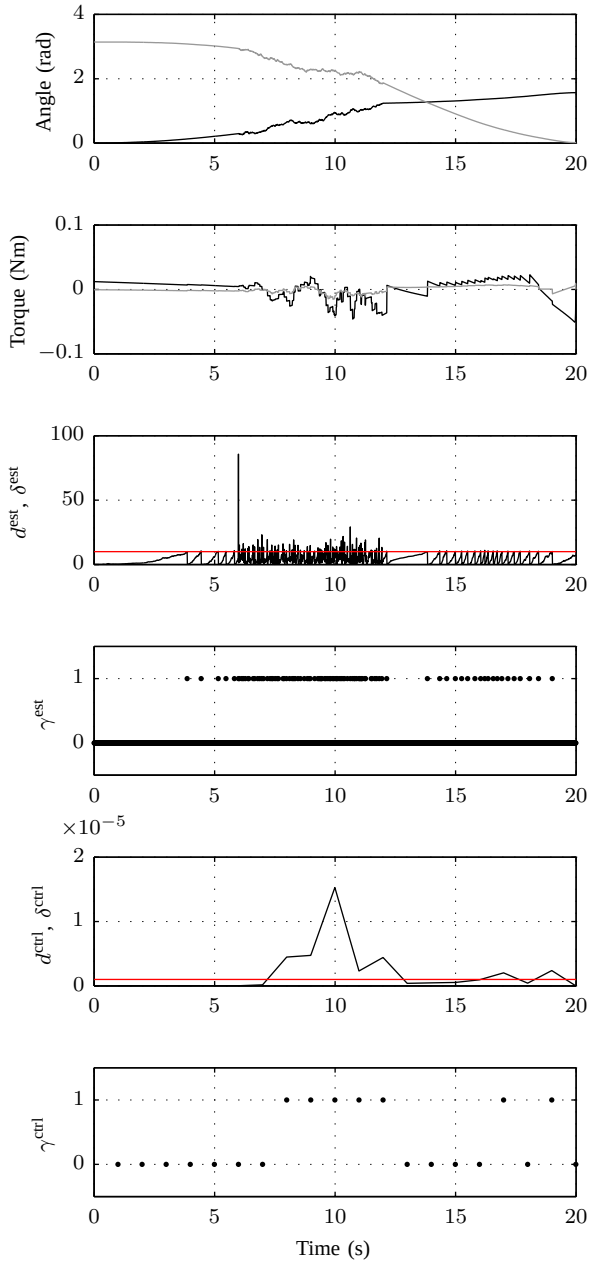


Fig. 5. System trajectories and communication decisions for the 2D arm simulation example. TOP TWO GRAPHS: angles and input torques for arm 1 (q_1, u_1 in black) and arm 2 (q_2, u_2 in gray). MIDDLE TWO GRAPHS: estimation triggering decision (12) with $d^{\text{est}}(\hat{p}_{k|k}, \hat{p}_{k|\underline{k}^{\text{est}}})$ in black and the threshold δ^{est} in red, as well as the corresponding triggers γ^{est}. BOTTOM TWO GRAPHS: analogous signals for the control decision (16). Notice that control communication decisions γ^{ctrl} are taken every 1 s, while estimation decisions γ^{est} are made every 0.01 s. Discontinuities in the control input signals are due to control or estimation communication updates.

control [22], and can also be observed in the communication patterns here.

D. Performance-communication trade-off

Next, we discuss how control and estimation performance depend on the thresholds δ^{est} and δ^{ctrl}. Estimation performance of the remote estimation problem is measured as the difference d^{est} of the operator's estimation to the robot's

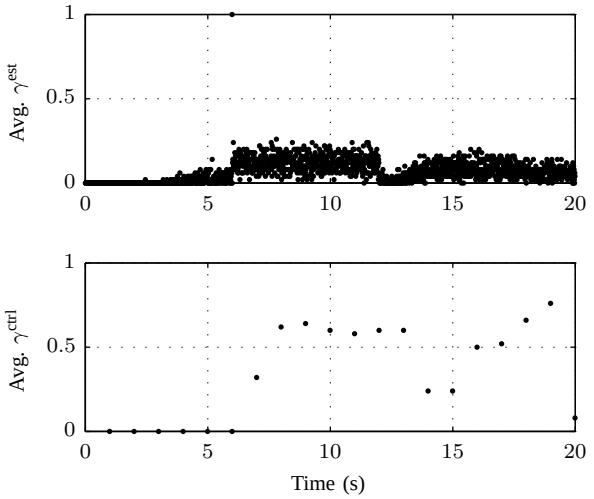


Fig. 6. Communication decisions γ^{est} and γ^{ctrl} averaged over 50 simulation runs.

estimation (which is optimal in the sense that it exploits all sensor data). We define $\bar{d}^{\text{est}}$ as the time-average of the signal $d^{\text{est}}(\hat{p}_{k|k}, \check{p}_k)$ over the simulation horizon, and the average estimation error $\mathcal{P}^{\text{est}}$ as the average of $\bar{d}^{\text{est}}$ over 50 simulation runs. Similarly, we measure the average control cost $\mathcal{P}^{\text{ctrl}}$ as the average of $\bar{d}^{\text{ctrl}}$ over 50 simulations, where $\bar{d}^{\text{ctrl}}$ is the time-average of $d^{\text{ctrl}}(\pi_k, \check{\pi}_k)$.

For fixed $\delta^{\text{ctrl}} = 10^{-6}$ and different δ^{est} in the range from 0 to 100, we ran 50 simulations each, and computed $\mathcal{P}^{\text{est}}$, as well as the average estimation communication rate $\mathcal{R}^{\text{est}}$. The average rate $\mathcal{R}^{\text{est}}$ is normalized such that $\mathcal{R}^{\text{est}} = 1$ corresponds to full robot-to-operator communication ($\gamma_k^{\text{est}} = 1$ for all k) and $\mathcal{R}^{\text{est}} = 0$ corresponds to no communication. Note that $\delta^{\text{est}} = 0$ implies full communication ($\mathcal{R}^{\text{est}} = 1$) and optimal performance ($\mathcal{P}^{\text{est}} = 0$) and, hence, corresponds to the reference scenario with periodic communication in Fig. 2.

The obtained average estimation errors $\mathcal{P}^{\text{est}}$ are plotted versus the average communication rates $\mathcal{R}^{\text{est}}$ in Fig. 7. As expected, the estimation error increases with reduced communication. Remarkably, a significant reduction of communication can be achieved at only a mild increase in estimation error (for example, $\mathcal{R}^{\text{est}} = 0.19$ and $\mathcal{P}^{\text{est}} = 0.65$ for $\delta^{\text{est}} = 2$). Fact 1 implies that the estimation error is bounded: $\mathcal{P}^{\text{est}} \leq \delta^{\text{est}}$. The upper bound is shown in red in Fig. 7. Notice that this bound is not only guaranteed for the average error, but for the actual error at every step k ($d^{\text{est}}(\hat{p}_{k|k}, \check{p}_k) \leq \delta^{\text{est}}$).

The corresponding results for the average control cost $\mathcal{P}^{\text{ctrl}}$ versus average control communication $\mathcal{R}^{\text{ctrl}}$ are shown in Fig. 8. For these, $\delta^{\text{est}} = 10$ was kept constant, and δ^{ctrl} was varied in the range from 0 to 10^{-4}.

V. Concluding Remarks

The proposed framework for remote robot operation with reduced communication was demonstrated in a simulation example of a 2D robot arm, using an EKF and iLQG as the implementations of the estimation and control algorithms. However, the framework is modular and other

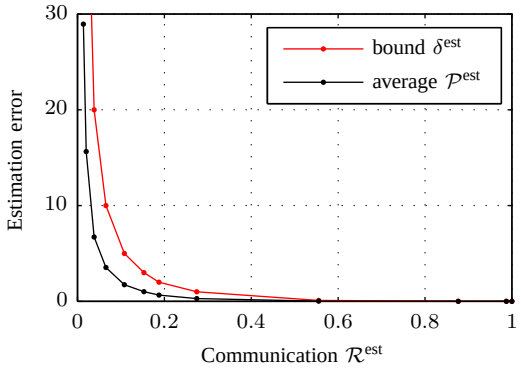


Fig. 7. Average estimation error versus average estimation communication for fixed $\delta^{\text{ctrl}} = 10^{-6}$ and varying δ^{est}. Each data point was averaged over 50 simulations. The threshold δ^{est} (shown in red) is a guaranteed upper bound on the error.

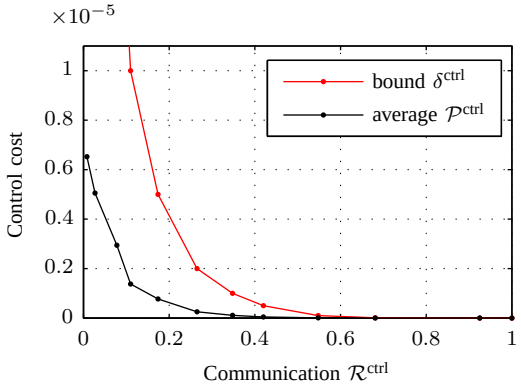


Fig. 8. Average control cost versus average control communication for fixed $\delta^{\text{est}} = 10$ and varying δ^{ctrl}. Each data point was averaged over 50 simulations. The threshold δ^{ctrl} (shown in red) is a guaranteed upper bound on the cost.

algorithms can readily be used. Similarly, the measures for the triggering decisions (KL-divergence for estimation and quadratic cost for control) are specific choices, and others may be meaningful for different applications (cf. Remark 1). Among other aspects, the simulation example demonstrates how asymmetric communication schemes can be obtained (communication of control policies occurs at significantly lower rates than estimation updates).

Encouraged by the simulation results herein, we intend to experimentally validate the proposed framework on a robotic platform in future work. The underlying ideas for event-based state estimation were already successfully applied in previous work [7], where they were used to stabilize the Balancing Cube [23], a 3D multi-body inverted pendulum.

Theoretical aspects interesting for future work concern robustness to model uncertainties and network imperfections (delays and packet drops), extensions to remote operation of multiple robots, and establishing hard bounds on the communication rates.

Acknowledgment

We thank Farbod Farshidian and Michael Neunert for providing the implementation of the iLQG algorithm and the simulation of the 2D arm.

References

[1] H. Asama, "Robot & remote controlled machine technology for accident response and decommissioning of the Fukushima Daiichi nuclear power plant," *Plenary presentation at the 19th IFAC World Congress*, Cape Town, South Africa, Aug. 2014. [Online]. Available: http://www.ifac2014.org/assets/pdf/plenary/Asama.pdf

[2] DARPA robotics challenge. [Online]. Available: http://www.theroboticschallenge.org

[3] E. Ackerman, "DARPA robotics challenge finals: Dates, location, and everything else you need to know," IEEE Spectrum, Automaton Blog, Jun. 2014. [Online]. Available: http://spectrum.ieee.org/automaton/robotics/military-robots/darpa-robotics-challenge-finals-details

[4] M. Lemmon, "Event-triggered feedback in control, estimation, and optimization," in *Networked Control Systems*, ser. Lecture Notes in Control and Information Sciences. Springer, 2010, vol. 406, pp. 293–358.

[5] W. P. M. H. Heemels, K. H. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control," in *Proc. of the IEEE 51st Conference on Decision and Control*, 2012, pp. 3270–3285.

[6] C. G. Cassandras, "The event-driven paradigm for control, communication and optimization," *Journal of Control and Decision*, vol. 1, no. 1, pp. 3–17, 2014.

[7] S. Trimpe and R. D'Andrea, "An experimental demonstration of a distributed and event-based state estimation algorithm," in *Proc. of the 18th IFAC World Congress*, Milano, Italy, Aug. 2011, pp. 8811–8818.

[8] ——, "Event-based state estimation with variance-based triggering," *IEEE Transaction on Automatic Control, Special Issue on Control of Cyber-Physical Systems*, vol. 59, no. 12, pp. 3266–3281, 2014.

[9] J. Marck and J. Sijs, "Relevant sampling applied to event-based state-estimation," in *4th International Conference on Sensor Technologies and Applications*, Jul. 2010, pp. 618–624.

[10] C. Phillips-Grafflin, N. Alunni, H. Suay, J. Mainprice, D. Lofaro, D. Berenson, S. Chernova, R. Lindeman, and P. Oh, "Toward a user-guided manipulation framework for high-DOF robots with limited communication," *Intelligent Service Robotics*, vol. 7, no. 3, pp. 121–131, 2014.

[11] C. Sayers, *Remote control robotics*. Springer, 1999.

[12] P. F. Hokayem and M. W. Spong, "Bilateral teleoperation: An historical survey," *Automatica*, vol. 42, no. 12, pp. 2035–2057, 2006.

[13] S. Hirche and M. Buss, "Human-oriented control for haptic teleoperation," *Proceedings of the IEEE*, vol. 100, no. 3, pp. 623–647, 2012.

[14] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. Wiley-Interscience, 2006.

[15] D. B. Bertsekas, *Dynamic Programming & Optimal Control, Vol. I & II*. Athena Scientific, 2007.

[16] B. D. O. Anderson and J. B. Moore, *Optimal Control: Linear Quadratic Methods*. Mineola, New York: Dover Publications, 2007.

[17] E. Todorov and W. Li, "A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems," in *Proc. of the American Control Conference*, Jun. 2005, pp. 300–306.

[18] D. Mayne, "A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems," *International Journal of Control*, vol. 3, no. 1, pp. 85–95, 1966.

[19] M. Diehl, H. Bock, H. Diedam, and P.-B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control," in *Fast Motions in Biomechanics and Robotics*. Springer Berlin Heidelberg, 2006, vol. 340, pp. 65–93.

[20] E. Theodorou, J. Buchli, and S. Schaal, "A generalized path integral control approach to reinforcement learning," *The Journal of Machine Learning Research*, vol. 11, pp. 3137–3181, 2010.

[21] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Wiley, 2006.

[22] H. J. Kappen, "Path integrals and symmetry breaking for optimal control theory," *Journal of statistical mechanics: theory and experiment*, Nov. 2005.

[23] S. Trimpe and R. D'Andrea, "The Balancing Cube: A dynamic sculpture as test bed for distributed estimation and control," *IEEE Control Systems Magazine*, vol. 32, no. 6, pp. 48–75, 2012.