Sensor Selection for Dynamics-Driven User-Interface Design

Abraham P. Vinod, *Member, IEEE*, Adam J. Thorpe, *Graduate Student Member, IEEE*, Philip A. Olaniyi, Tyler H. Summers, and Meeko M. K. Oishi, *Senior Member, IEEE*

***Abstract*— We present a method for dynamics-driven, user-interface design for a human–automation system via sensor selection. We define the user interface to be the output of a multiple-input–multiple-output (MIMO) linear time-invariant (LTI) system and formulate the design problem as one of selecting an output matrix from a given set of candidate output matrices. Necessary conditions for situation awareness are captured as additional constraints on the selection of the output matrix. These constraints depend on the level of trust the human has in the automation. We show that the resulting user-interface design problem is a combinatorial, set-cardinality minimization problem with set function constraints. We propose tractable algorithms to compute optimal or suboptimal solutions with suboptimality bounds. Our approaches exploit monotonicity and submodularity present in the design problem and rely on constraint programming and submodular maximization. We apply this method to the IEEE 118-bus, to construct correct-by-design interfaces under various operating scenarios.**

***Index Terms*— Human–automation interaction, observability, output synthesis, sensor selection, user-interface design.**

I. Introduction

SITUATION awareness, a state of knowledge that relies upon the ability to deduce the current state of the system and predict the evolution of the state in the short term [1], is essential for effective human–automation interaction. In expensive, high-risk, and safety-critical systems, such as power grid distribution systems, aircraft and other transportation systems, biomedical devices, and nuclear power generation, the user-interface helps the user maintain situation awareness by providing critical information about the system to the user [2], [3]. Indeed, a lack of situation awareness is known to be a contributing factor to operator error in major grid failures [4], [5]. A variety of recommendations and guidelines for "good" user-interface design have been posited [6]–[8]. However, *formal tools* for user-interface design, which explicitly incorporate the underlying dynamics, could help avert potential errors and mishaps, and reduce time consumption and costly design and testing iterations.

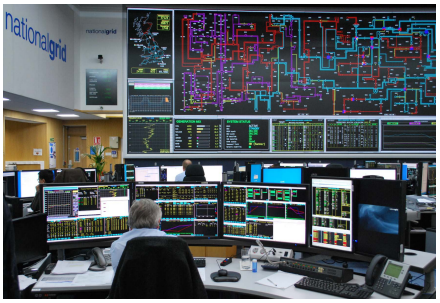


Fig. 1. User interfaces for power grid operators facilitate situation awareness, by providing information from which the power grid operator can estimate the state and predict its evolution. The sheer volume of information warrants the use of constructive tools (as opposed to ad hoc guidelines) to synthesize the information content of the interface. Image licensed under CC BY-ND 2.0.

We consider the user interface to be equivalent to an output map of the dynamical system and pose the question of user-interface design as one of *sensor selection*: among the sensors that could be the elements of the interface, we aim to identify a combination that is minimal [8], yet enables necessary conditions for situation awareness, and dependent upon the user's trust in the automation. We focus solely on the information content and not on the qualitative aspects of *how* that information is provided. The need for minimal interfaces is particularly evident in large systems (see Fig. 1), for which providing too much information can render the interface ineffective because it is overwhelming, and providing too little information can result in perceived nondeterminism [9].

Sensor selection [10]–[14] is typically posed as a combinatorial optimization problem, which becomes intractable even for moderate problem sizes. While some heuristics, such as convex relaxation [15], [16] and combinatorial algorithms that avoid a full exhaustive search [17]–[19], have been employed,

Manuscript received April 14, 2020; revised September 22, 2020; accepted January 18, 2021. Manuscript received in final form January 29, 2021. This work was supported in part by the National Science Foundation and in part by the Army Research Office (ARO). The work of Abraham P. Vinod, Adam J. Thorpe, Philip A. Olaniyi, and Meeko M. K. Oishi were supported by NSF under Grant CMMI-1254990 and Grant OIA-1757207. The work of Tyler H. Summers was supported in part by NSF under Grant CNS-1566127 and Grant CMMI-1728605 and in part ARO under Grant W911NF-17-1-0058. Recommended by Associate Editor T. Hatanaka. *(Corresponding author: Abraham P. Vinod.)*

Abraham P. Vinod was with the Electrical and Computer Engineering Department, The University of New Mexico, Albuquerque, NM 87131 USA. He is now with Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139 USA (e-mail: abraham.p.vinod@ieee.org).

Adam J. Thorpe and Meeko M. K. Oishi are with the Department of Electrical and Computer Engineering, The University of New Mexico, Albuquerque, NM 87131 USA (e-mail: ajthor@unm.edu; oishi@unm.edu).

Philip A. Olaniyi was with the Electrical and Computer Engineering Department, The University of New Mexico, Albuquerque, NM 87131 USA. He is now with Intel Corporation, Beaverton, OR 97007 USA (e-mail: philip.olaniyi@intel.com).

Tyler H. Summers is with the Department of Mechanical Engineering, The University of Texas at Dallas, Richardson, TX 75080 USA (e-mail: tyler.summers@utdallas.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCST.2021.3056242.

Digital Object Identifier 10.1109/TCST.2021.3056242

computational complexity remains a significant challenge. For some problem classes (e.g., cardinality-constrained submodular set function maximization [17], [20]), greedy algorithms and other graph theoretic approaches can yield provably optimal or near-optimal results [17]–[23]. Hence, we focus heavily in this article on characterization of the computational aspects of user-interface design via sensor selection.

Other approaches to user-interface analysis and design have focused on related aspects of human–automation interaction. Model checking has been used to detect mode confusion in discrete-event systems [7], [24], [25], and finite-state machine reduction techniques have been used to synthesize user interfaces of minimal cardinality for discrete-state abstractions of hybrid systems [9], [26]. Interfaces have been designed to assure internal and external awareness [27], to facilitate the transfer of control authority between the human and the automation, and to articulate information related to the role of regret in human decision aids [28]. In [29] and [30], the effect of transparency on workload and trust was evaluated, and a feedback scheme developed that alters transparency of the interface. Other interface design approaches focus on moderating human input [31], [32] despite uncertainty and on mixed-initiative control [33], [34] for human–robot interaction.

Our approach is based on observability conditions that presume the human is a special type of observer, to assess whether the interface provides sufficient information for the human to accomplish a given task [35]–[37]. Hence, in contrast to standard sensor placement problems, additional constraints arise to ensure necessary conditions for situation awareness and to capture the effect of the user's trust in the automation. The main contributions of this article are: 1) assurances of optimality and suboptimality via submodularity and monotonicity properties, specific to the user-interface design problem, and 2) efficient numerical implementations that employ constraint programming, greedy heuristics for submodular maximization, and a novel enumeration framework for large user-interface design problems. The algorithmic advances proposed here enable the application to problems that would be computationally prohibitive with our preliminary approach [38]. Furthermore, the model proposed here captures gradated user trust in the automation, a more subtle characterization than the simplistic, no trust or full trust, characterization that was used in [38].

This article is organized as follows. Section II provides the problem formation. Section III formulates user-interface design as a combinatorial optimization problem. Section IV describes a novel enumeration framework that enables computationally efficient search for feasible user interfaces. Section V demonstrates our approach on user-interface design for a large system, the IEEE 118-bus, and Section VI provides the conclusions.

II. Preliminaries and Problem Statement

A finite set $\mathscr{S}$ has cardinality $|\mathscr{S}|$ and power set $2^{\mathscr{S}}$. A set function $f : 2^{\mathscr{S}} \to \mathbb{R}$ takes as input a subset of $\mathscr{S}$ and returns a real number. For natural numbers $a, b \in \mathbb{N}$ with $a \leq b$, we define the set $\mathbb{N}_{[a,b]} = \{c \in \mathbb{N} : a \leq c \leq b\}$. For a matrix $M \in \mathbb{R}^{p\times q}$, we denote its column rank by $\text{rank}(M)$ and its column space (range) by $\mathcal{R}(M)$. We define a matrix whose column space coincides with a subspace $\mathcal{V}$ as basis($\mathcal{V}$). Recall that basis($\mathcal{R}(M)$) is not unique. Given two vector spaces $\mathcal{V}_1$ and $\mathcal{V}_2$, their sum ($\mathcal{V}_1 + \mathcal{V}_2 = \{v_1 + v_2 : v_1 \in \mathcal{V}_1, v_2 \in \mathcal{V}_2\}$) and their intersection are vector spaces [39, p. 22].

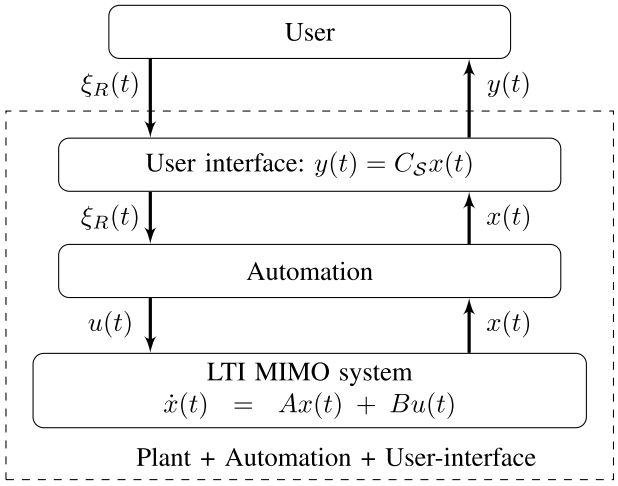


Fig. 2. Human–automation system in which the human provides a reference trajectory, and the automation synthesizes a low-level control to achieve it.

Consider a human–automation system (see Fig. 2) in which the human provides a reference trajectory $\xi_R(t) \in \mathbb{R}^p$, and the automation synthesizes a low-level controller to achieve reference tracking [36]. We presume a multiple-input–multiple-output (MIMO) linear time-invariant (LTI) system

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{1a}$$

$$y(t) = C_{\mathcal{S}}x(t) \tag{1b}$$

with state $x(t) \in \mathcal{X} = \mathbb{R}^n$, input $u(t) \in \mathbb{R}^m$, output $y(t) \in \mathbb{R}^p$, and known matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. The user receives information about the plant via the user interface.

Definition 1 (User Interface and Sensors): We define the output $y(t)$ as the user interface of the system (1), with the candidate rows of $C_{\mathcal{S}}$ referred to as the sensors $s_i \in \mathbb{R}^n$.

By Definition 1, a sensor is a potential element of the user interface. We denote the set of all sensors as $\mathscr{S} = \{s_1, \ldots, s_{|\mathscr{S}|}\}$ for a finite $|\mathscr{S}| \in \mathbb{N}$. For any sensor combination $\mathcal{S} \in 2^{\mathscr{S}}$, the output matrix $C_{\mathcal{S}}$ is a matrix whose rows consist of the elements $s_i \in \mathcal{S}$, and the total number of outputs associated with $C_{\mathcal{S}}$ is $p = |\mathcal{S}|$.

Definition 2 (Task): A task is characterized by the tuple $(\ell, C_{\mathcal{S}_{\text{task}}})$, with a known task matrix $C_{\mathcal{S}_{\text{task}}} \in \mathbb{R}^{|\mathcal{S}_{\text{task}}|\times n}$ associated with $\mathcal{S}_{\text{task}} \in 2^{\mathscr{S}}$, and a known, possibly nonlinear, function $\ell : \mathbb{R}^{|\mathcal{S}_{\text{task}}|} \to \mathbb{R}$. The task $(\ell, C_{\mathcal{S}_{\text{task}}})$ is a specification of the form always $x(t) \in \mathcal{F}(t)$ or eventually $x(t) \in \mathcal{F}(t)$, for $\mathcal{F}(t) = \{x(t) : \ell\big(C_{\mathcal{S}_{\text{task}}}x(t)\big) \geq 0\}$.

The task is defined in terms of safety or liveness specifications, i.e., a desirable phenomenon that should always or eventually happen [36], [40]. The task may also be interpreted as imposing a specification on the output $y_{\text{task}}(t) = C_{\mathcal{S}_{\text{task}}}x(t)$.

Illustrative Example: Consider an LTI model of a jerk-controlled robot constrained to move in a line, which is

tasked with maintaining a velocity above a minimum speed $v_{\min}$. The robot has a mounted camera with independent dynamics. The position dynamics (3-D) and camera heading dynamics (1-D) result in

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \; B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{2}$$

with states that include position, velocity, acceleration, and the camera heading. We consider a suite of sensors based on measurements of each state, i.e., $\mathscr{S} = \{s_p, s_v, s_a, s_h\}$ with $s_p = [1\ 0\ 0\ 0]$, $s_v = [0\ 1\ 0\ 0]$, $s_a = [0\ 0\ 1\ 0]$, and $s_h = [0\ 0\ 0\ 1]$. The task is defined by $(\ell, C_{\mathcal{S}_{\text{task}}})$, with $\mathcal{S}_{\text{task}} = \{s_v\}$, $C_{\mathcal{S}_{\text{task}}} = s_v$, $\ell(z) = z - v_{\min}$, and $\mathcal{F}(t) = \{x(t) : C_{\mathcal{S}_{\text{task}}} x(t) \geq v_{\min}\}$. □

For a given task $(\ell, C_{\mathcal{S}_{\text{task}}})$, we seek to design a user interface $C_{\mathcal{S}}$ that satisfies, in order of importance.

1) *C1:* Necessary conditions for situation awareness.
2) *C2:* Compatibility with the user's trust in the automation.
3) *C3:* Conciseness.

These properties represent human factors that are key for effective human–automation interaction and will be described in detail in Section III. Briefly, constraint C1 considers the limitations of the human operator and the complexity of the task. Constraint C2 requires that more information is provided to the user when the user's trust in the automation is low and vice versa. Constraint C3 prevents high cognitive load associated with excessive data.

We embed these properties as constraints in the sensor selection problem for user-interface design

$$\begin{aligned} \underset{\mathcal{S} \in 2^{\mathscr{S}}}{\text{minimize}} \; & |\mathcal{S}| && \text{(conciseness)} && \text{(3a)} \\ \text{subject to } & \mathcal{S} \in \mathscr{S}_{\text{sit-aware}} && \text{(situation awareness)} && \text{(3b)} \\ & \mathcal{S} \in \mathscr{S}_{\text{trust}} && \text{(trust)} && \text{(3c)} \end{aligned}$$

in which (3a) arises from C3, (3b) arises from C2, and (3c) arises from C1.

Problem 1: Given a task $(\ell, C_{\mathcal{S}_{\text{task}}})$ and a human–automation system (1), find a succinct characterization of the constraint for necessary conditions for situation awareness $\mathscr{S}_{\text{sit-aware}}$ and the constraint for trust compatibility $\mathscr{S}_{\text{trust}}$.

Problem 2: Construct tractable combinatorial optimization algorithms to solve (3), with guarantees of optimality or suboptimality, as appropriate.

Because combinatorial optimization problems are typically hard to solve due to their large feasible solution space, solving (3) directly is a challenging endeavor. Problem 1 provides a structure that we can exploit to address Problem 2 so that (3) can be addressed through tractable reformulation.

III. User-Interface Design as Sensor Selection

A. Necessary Conditions for Situation Awareness via Observability

Situation awareness has three necessary conditions: perception, comprehension, and projection, more formally defined in [1] as "perception of the elements in an environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future." These three elements are related: perception is a necessary condition for comprehension and comprehension is a necessary condition for projection. As in [35], [37], and [41], we interpret these three elements, respectively, as the ability to reconstruct those elements of the state that are relevant for the task at hand, the ability to understand the output and its time derivatives, and the ability to reconstruct those elements of the state derivative relevant to the task at hand. These definitions bear significant resemblance to standard notions of observability, modified to focus on the task, as opposed to the state [35], [36], which motivates our approach.

We focus primarily on information necessary for perception, comprehension, and projection. Perception is not possible if incorrect or inadequate information is provided through the display, and comprehension and perception are not possible if perception is not possible or if the additional information needed for comprehension and perception is not available. While a complex set of factors are known to impact perception, comprehension, and perception, including memory, experience, workload, mental models, and ability to classify information and respond to cues [1], [42], we limit our focus to information content provided to the user, as it is a necessary condition for perception, comprehension, projection, and ultimately, situation awareness. Furthermore, information content is important for systems in which the dynamics are complex and potentially nonintuitive. In addition, although qualitative aspects of user-interface design are key for effective human–automation interaction [3], [43], we focus solely on quantitative aspects and presume that information content will be presented in a human-centric manner.

Assumption 1 (Necessary Conditions for Situation Awareness): For a given user interface, constructed from elements $\mathcal{S} \in 2^{\mathscr{S}}$, the user can reconstruct the output of the system, $y(t) = C_{\mathcal{S}} x(t)$, the unforced higher derivatives of the output, and their linear combinations.

We make the assumption that if the information necessary for perception, comprehension, and projection is available, the user's perception, comprehension, and projection will inevitably follow. While this is a strong assumption for many practical circumstances, we invoke it here solely for the purpose of analyzing the content of the designed interface. This assumption is required to isolate the effect of information content on necessary conditions for situation awareness, that is, in order to focus solely on the impact of information content on the feasibility of situation awareness, we must presume that all other elements and circumstances that enable situation awareness are in place, in order to avoid confounding factors. Finally, we note that while our approach does not in theory limit the size of the dynamical system or task, in practice, limits on workload and attention would constrain the validity of this approach (although experience, automatic processes, and other phenomena may mitigate their impact) [1].

As in [35], [37], and [38], we employ input–output linearization to capture the user's interaction with the system (1). We presume that the user provides a reference trajectory $\xi_R(t)$ that is smooth. Given an output matrix $C_{\mathcal{S}}$

TABLE I

APPLICATION OF VARIOUS DEFINITIONS TO THE ILLUSTRATIVE EXAMPLE GIVEN IN SECTION II. HERE, $\mathscr{S} = \{s_p, s_v, s_a, s_h\}$, $\mathcal{S}_{\text{task}} = \{s_v\}$, AND $\mathscr{S}_{\text{reduced}} = \{s_p, s_v, s_a\}$. INTERFACES THAT SATISFY BOTH SITUATION AWARENESS AND TRUST CONSTRAINTS FOR A GIVEN LEVEL OF USER TRUST IN THE AUTOMATION ARE FEASIBLE FOR (3); INTERFACES THAT ARE OPTIMAL FOR A GIVEN TRUST LEVEL ARE INDICATED IN BOLD

$\mathcal{S}$	$\{s_p\}$	$\{s_v\}$	$\{s_a\}$	$\{s_h\}$	$\{s_p, s_v\}$	$\{s_p, s_a\}$	$\{s_p, s_h\}$	$\{s_v, s_a\}$	$\{s_v, s_h\}$	$\{s_a, s_h\}$	$\{s_p, s_v, s_a\}$	$\{s_p, s_v, s_h\}$	$\{s_p, s_a, s_h\}$	$\{s_v, s_a, s_h\}$	$\{s_p, s_v, s_a, s_h\}$
$\Gamma(\mathcal{S})$	3	2	1	1	3	3	4	2	3	2	3	4	4	3	4
$\Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}})$	3	2	2	3	3	3	4	2	3	3	3	4	4	3	4
$\mathscr{S}_{\text{sit-aware}}$	✓	✓			✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
$2^{\mathscr{S}_{\text{reduced}}}$	✓	✓	✓		✓	✓		✓			✓				
$\mathscr{S}_{\text{sit-aware,reduced}}$	✓	✓			✓	✓		✓			✓				
$\mathscr{S}_{\text{trust}}, k_{\text{trust}} = 1$	**✔**	**✔**	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
$\mathscr{S}_{\text{trust}}, k_{\text{trust}} = 2$	**✔**	**✔**			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
$\mathscr{S}_{\text{trust}}, k_{\text{trust}} = 3$	**✔**				✓	✓	✓		✓		✓	✓	✓	✓	✓
$\mathscr{S}_{\text{trust}}, k_{\text{trust}} = 4$							**✔**					✓	✓		✓

with $\mathcal{S} = \{s_1, s_2, \ldots, s_{|\mathcal{S}|}\} \in 2^{\mathscr{S}}$, we construct a similarity transform $P_{\mathcal{S}} \in \mathbb{R}^{n \times n}$

$$\begin{bmatrix} \xi(t) \\ \eta(t) \end{bmatrix} = P_{\mathcal{S}} x(t) = \begin{bmatrix} T_{\mathcal{S}} \\ T_{\mathcal{S}}^{\perp} \end{bmatrix} x(t) \tag{4}$$

that results in observable states $\xi(t) \in \mathcal{R}(T_{\mathcal{S}})$ and unobservable states $\eta(t) \in \mathcal{R}\big(T_{\mathcal{S}}^{\perp}\big)$. The linear transformation $T_{\mathcal{S}}$ is defined using T_{s_i} for some $s_i \in \mathcal{S}$ as

$$T_{s_i} = \begin{bmatrix} s_i & (s_i^{\top} A)^{\top} & (s_i^{\top} A^2)^{\top} & \ldots & (s_i^{\top} A^{\gamma(s_i)-1})^{\top} \end{bmatrix}^{\top} \tag{5}$$

$$T_{\mathcal{S}} = \text{basis}\left(\mathcal{R}\left(\begin{bmatrix} T_{s_1}^{\top} & T_{s_2}^{\top} & \ldots & T_{s_{|\mathcal{S}|}}^{\top} \end{bmatrix}\right)^{\top}\right) \tag{6}$$

where $\gamma : \mathscr{S} \to \mathbb{N}_{[1,n]}$ is the relative degree of the MISO system with the single output $s_i^{\top} x(t)$. By (5), $\mathcal{R}(T_{\mathcal{S}})$ is the state subspace spanned by the outputs characterized by $y(t) = C_{\mathcal{S}} x(t)$ and their unforced higher derivatives.

Assumption 2: (Correctly Designed Automation) The automation generates $u(t)$ such that $(\xi(t), \dot{\xi}(t))$ tracks the reference trajectory $(\xi_R(t), \dot{\xi}_R(t))$.

The implications of Assumptions 1 and 2 are twofold: 1) the user can reconstruct $\xi(t)$ and predict its evolution (because $\dot{\xi}(t)$ can be reconstructed) and 2) the user delegates control of the internal dynamics $\eta(t)$ to the automation.

To tractably enumerate $\mathscr{S}_{\text{sit-aware}}$, we propose the *user information index*, a set function that measures the dimension of the state subspace the user can reconstruct and predict from the information presented in the user interface.

Definition 3 (User Information Index): The user information index is the set function $\Gamma : 2^{\mathscr{S}} \to \mathbb{N}_{[1,n]}$

$$\Gamma(\mathcal{S}) = \dim(\mathcal{R}(T_{\mathcal{S}})) = \text{rank}(T_{\mathcal{S}}). \tag{7}$$

The user information index $\Gamma(\mathcal{S})$ characterizes the dimensions of $\xi(t)$ and $\eta(t)$ since $\xi(t) \in \mathbb{R}^{\Gamma(\mathcal{S})}$ and $\eta(t) \in \mathbb{R}^{n-\Gamma(\mathcal{S})}$. Table I shows $\Gamma(\mathcal{S})$ for the illustrative example.

Proposition 1 (Sufficient Information for Task Completion): If $\mathcal{R}\big(C_{\mathcal{S}_{\text{task}}}\big) \subseteq \mathcal{R}(T_{\mathcal{S}})$, then the user interface $C_{\mathcal{S}}$ provides sufficient information to complete the task $(\ell, C_{\mathcal{S}_{\text{task}}})$.

Proof: If $\mathcal{R}\big(C_{\mathcal{S}_{\text{task}}}\big) \subseteq \mathcal{R}(T_{\mathcal{S}})$, we can express the task output $y_{\text{task}}(t) = C_{\mathcal{S}_{\text{task}}} x(t) \in \mathcal{R}\big(C_{\mathcal{S}_{\text{task}}}\big)$ as a linear combination of the observable state $\xi(t) \in \mathcal{R}(T_{\mathcal{S}})$. Hence, under Assumptions 1 and 2, the user can estimate $y_{\text{task}}(t)$ and $\dot{y}_{\text{task}}(t)$ from the user-interface output $y(t) = C_{\mathcal{S}} x(t)$. □

Proposition 1 states that a user interface satisfies necessary conditions for situation awareness with the task at hand, provided that $y_{\text{task}}(t)$ is contained in the observable subspace $\mathcal{R}(T_{\mathcal{S}})$. However, the conditions in Proposition 1 are not amenable to tractable computation. Hence, we reframe Proposition 1 in terms of the user information index.

Lemma 1: Given any $\mathcal{P}, \mathcal{Q} \in 2^{\mathscr{S}}$, the following conditions hold.

1) $\mathcal{P} \subseteq \mathcal{Q}$ implies that $\mathcal{R}(T_{\mathcal{P}}) \subseteq \mathcal{R}(T_{\mathcal{Q}})$ and $\Gamma(\mathcal{P}) \leq \Gamma(\mathcal{Q})$.
2) $\Gamma(\mathcal{P} \cup \mathcal{Q}) = \Gamma(\mathcal{P}) + \Gamma(\mathcal{Q}) - \dim(\mathcal{R}(T_{\mathcal{P}}) \cap \mathcal{R}(T_{\mathcal{Q}}))$.
3) $\Gamma(\mathcal{P} \cap \mathcal{Q}) \leq \dim(\mathcal{R}(T_{\mathcal{P}}) \cap \mathcal{R}(T_{\mathcal{Q}}))$.
4) $\Gamma(\mathcal{P} \cup \mathcal{Q}) = \Gamma(\mathcal{P})$ if and only if $\mathcal{R}(T_{\mathcal{P} \cup \mathcal{Q}}) = \mathcal{R}(T_{\mathcal{P}})$.

The proof of Lemma 1 is provided in Appendix B.

Proposition 2 (Necessary Conditions for Situation Awareness via User Information Index): For every $\mathcal{S} \in \mathscr{S}_{\text{sit-aware}}$, defined as

$$\mathscr{S}_{\text{sit-aware}} \triangleq \big\{\mathcal{S} \in 2^{\mathscr{S}} : \Gamma(\mathcal{S}) = \Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}})\big\} \tag{8}$$

the user interface $C_{\mathcal{S}}$ provides sufficient information to complete the task.

Proof: By (6), $\mathcal{R}\big(C_{\mathcal{S}_{\text{task}}}\big) \subseteq \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)$. By Lemma 14, we have $\mathscr{S}_{\text{sit-aware}} = \big\{\mathcal{S} \in 2^{\mathscr{S}} : \mathcal{R}(T_{\mathcal{S}}) = \mathcal{R}\big(T_{\mathcal{S} \cup \mathcal{S}_{\text{task}}}\big)\big\}$. Furthermore, $\mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big) \subseteq \mathcal{R}\big(T_{\mathcal{S} \cup \mathcal{S}_{\text{task}}}\big)$ for any $\mathcal{S} \in \mathscr{S}_{\text{sit-aware}}$ by Lemma 11. Hence, we have $\mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big) \subseteq \mathcal{R}\big(T_{\mathcal{S} \cup \mathcal{S}_{\text{task}}}\big) = \mathcal{R}(T_{\mathcal{S}})$ for any $\mathcal{S} \in \mathscr{S}_{\text{sit-aware}}$. Thus, $\mathcal{R}\big(C_{\mathcal{S}_{\text{task}}}\big) \subseteq \mathcal{R}(T_{\mathcal{S}})$. Applying Proposition 1 completes the proof. □

Table I shows $\mathscr{S}_{\text{sit-aware}}$ for the illustrative example, and two possible interfaces are shown in Fig. 3. As expected, $\{s_h\} \notin \mathscr{S}_{\text{sit-aware}}$ since the heading measurement s_h alone provides no information about velocity (the task), due to the decoupled dynamics (2). Furthermore, $\{s_a\} \notin \mathscr{S}_{\text{sit-aware}}$ since reconstructing velocity from acceleration measurements requires integration. Thus, all sensor combinations in $2^{\mathscr{S}} \setminus \{\{s_a\}, \{s_h\}, \{s_a, s_h\}\}$ provide sufficient information, enabling task completion.

Since enumerating $2^{\mathscr{S}}$ to compute $\mathscr{S}_{\text{sit-aware}}$ is computationally expensive for large $|\mathscr{S}|$, we propose Algorithm 1 for a tractable enumeration of $\mathscr{S}_{\text{sit-aware}}$. We construct a reduced set of admissible sensors $\mathscr{S}_{\text{reduced}}$

$$\mathscr{S}_{\text{reduced}} \triangleq \{s \in \mathscr{S} : \Gamma(s) + \Gamma(\mathcal{S}_{\text{task}}) > \Gamma(s \cup \mathcal{S}_{\text{task}})\} \tag{9}$$

$$= \{s \in \mathscr{S} : \dim\big(\mathcal{R}(T_s) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) > 0\} \tag{10}$$

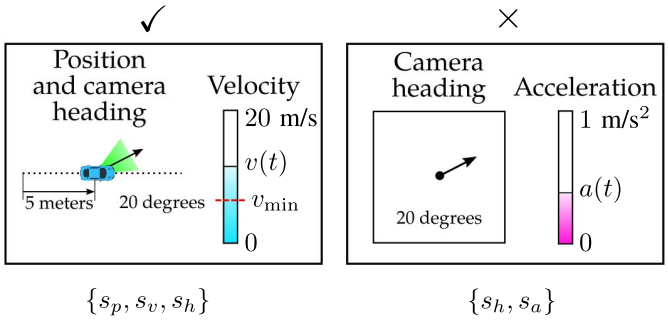


Fig. 3. User interfaces for the illustrative example in Section II. The user interface on the left enables situation awareness for the task of keeping $v(t) \geq v_{\min}$ and is appropriate for all levels of user trust. In contrast, the user interface on the right does not enable situation awareness and meets the trust requirement only for high levels of trust.

Algorithm 1 Efficient Enumeration of $\mathscr{S}_{\text{sit-aware}}$ via Characterization of $\mathscr{S}_{\text{sit-aware,reduced}}$

Input: Set of all sensors $\mathscr{S}$, sensors that define the task $\mathcal{S}_{\text{task}}$, the user information index $\Gamma(\cdot)$

Output: Sensor combinations that enable necessary conditions for situation awareness $\mathscr{S}_{\text{sit-aware}}$, and a reduced set $\mathscr{S}_{\text{sit-aware,reduced}}$

1: $\mathscr{S}_{\text{sit-aware}} \leftarrow \emptyset, \quad \mathscr{S}_{\text{sit-aware,reduced}} \leftarrow \emptyset$
2: Compute $\mathscr{S}_{\text{sit-aware,reduced}}$ using (11)
3: **for** $\mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}$ **do**
4: $\quad \mathscr{S}_{\text{sit-aware}} \leftarrow \mathscr{S}_{\text{sit-aware}} \cup \{\mathcal{P} \times 2^{\mathscr{S} \setminus \mathcal{P}}\}$
5: **end for**
6: **return** $(\mathscr{S}_{\text{sit-aware}}, \mathscr{S}_{\text{sit-aware,reduced}})$

(where (10) follows from (9) and Lemma 12), to construct an easily computable subset of $\mathscr{S}_{\text{sit-aware}}$

$$\mathscr{S}_{\text{sit-aware,reduced}} = \{\mathcal{P} \in 2^{\mathscr{S}_{\text{reduced}}} | \Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}) = \Gamma(\mathcal{P})\}. \quad (11)$$

The set $\mathscr{S}_{\text{sit-aware,reduced}}$ is "minimal," in that removing any sensor from the sensor combinations in $\mathscr{S}_{\text{sit-aware,reduced}}$ will violate the constraint for necessary conditions for situation awareness (8). Additional elements are appended to $\mathscr{S}_{\text{sit-aware,reduced}}$ (line **??**) so that Algorithm 1 provides an exact enumeration of the members of $\mathscr{S}_{\text{sit-aware}}$. Algorithm 1 is computationally tractable since enumeration is done over $2^{\mathscr{S}_{\text{reduced}}}$ and $|2^{\mathscr{S}_{\text{reduced}}}| \ll |2^{\mathscr{S}}|$.

Theorem 1 (Correctness of Algorithm 1): The set $\mathscr{S}_{\text{sit-aware}}$ can be constructed as the union of two sets

$$\mathscr{S}_{\text{sit-aware}} = \left\{ \mathcal{S} \in \mathscr{S} \,\middle|\, \begin{array}{l} \mathcal{P} = \mathcal{S} \cap \mathscr{S}_{\text{reduced}}, \\ \Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}) = \Gamma(\mathcal{P}) \end{array} \right\} \quad (12)$$

$$= \{\mathcal{P} \times 2^{\mathscr{S} \setminus \mathcal{P}} | \mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}\}. \quad (13)$$

Proof: We show that $\Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{S}) = \Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{P})$, which implies that $\Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{S}) = 0$ if and only if $\Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{P}) = 0$. This implies (12) by Proposition 2, and (13) follows from (12). The complete proof is in Appendix C. □

Table I shows $\mathscr{S}_{\text{sit-aware,reduced}}$ for the illustrative example, with $|\mathscr{S}_{\text{sit-aware,reduced}}| = 6$ and $|\mathscr{S}_{\text{sit-aware}}| = 12$. For this problem, $2^{\mathscr{S}_{\text{reduced}}}$ has only seven elements, while $2^{\mathscr{S}}$ has 15. The computational savings become far more dramatic for larger problems, as shown in Section V.

Lemma 2: $\mathcal{S}_{\text{task}}$ is a subset of $\mathscr{S}_{\text{reduced}}$, and $\mathcal{S}_{\text{task}}$ is a member of $\mathscr{S}_{\text{sit-aware,reduced}}$ and $\mathscr{S}_{\text{sit-aware}}$.

Lemma 2 describes the intuitive observation that constructing a user interface using only the sensors that describe the task should also be sufficient to complete the task. The proof of Lemma 2 is given in Appendix D.

B. User Trust in the Automation

User trust in the automation depends on many factors, including the expertise of the user, the performance and reliability of the automation, and the difficulty of the task. While some dimensions of trust may be static (i.e., dispositional trust), other dimensions may be highly dynamic (i.e., situational or learned trust) [44], [45]. Both low and high levels of trust in the automation are known to be problematic, as they are related to disuse of the automation due to underreliance and misuse due to overreliance, respectively [46].

The main principle driving the trust constraint (3b) is that the information presented to the user should be responsive to, and appropriate for, the user's current level of trust in the automation [47]. We focus on the challenges associated with low levels of trust, although extensions to overtrust may be possible. We presume that additional information would be helpful when the user's trust in the automation is relatively low, but that when the user's trust is relatively high, additional information is not warranted and may actually be detrimental, if it is overwhelming to the user [8], [48]. Mathematically, we account for this phenomenon by constraining the user information index by the user's level of trust in the automation.

Definition 4 (Trust Constraint): For a given level of trust in the automation, described by $k_{\text{trust}} \in \mathbb{N}_{[1,\Gamma(\mathscr{S})]}$, we define the set of sensors that are compatible with trust level k_{trust} as those whose user information index is above k_{trust}

$$\mathscr{S}_{\text{trust}} = \{\mathcal{S} \in 2^{\mathscr{S}} : \Gamma(\mathcal{S}) \geq k_{\text{trust}}\}. \quad (14)$$

The trust level k_{trust} could correspond to a variety of trust metrics, depending on the problem at hand [49] (see Fig. 4). Although considerable variability exists among questionnaire-based trust metrics [50]–[52], many seek a summative assessment of trust. For example, in the SHAPE Automation Trust Index (SATI) instrument, the "overall amount of trust in the total" system, which solicits trust as a percentage, would be most relevant to our framework [50]. A quantized, affine transformation from the SATI scale, ranging from 0% (no trust) to 100% (full trust), to our trust level scale, ranging from $\Gamma(\mathscr{S})$ (low trust) to 1 (high trust), respectively, would map the SATI "overall trust" to our trust level k_{trust}, resulting in a static value for a given user. A similar transformation could be applied to recent efforts in dynamic trust sensing via behavioral [33], [53] and psychophysiological data [45] (which feature either real-valued, bounded trust variables, i.e., $T(t) \in [0, 1]$ for some trust value $T(t)$, or discrete-valued trust variables, i.e., "low," "medium," and "high"), which would allow k_{trust} to vary over time.

Because our system model (see Fig. 2) presumes that the user dictates high-level reference tracking, and the automation

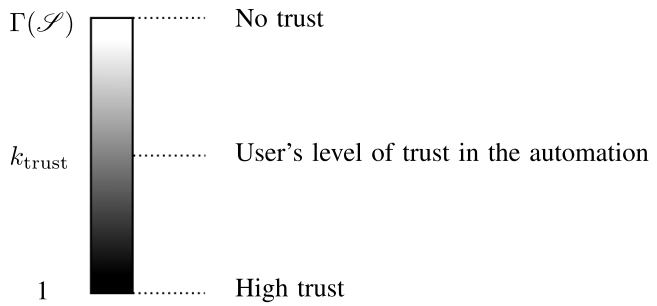


Fig. 4. Parameter k_{trust} indicates the level of trust the user has in the automation, with high values corresponding to low trust, and vice versa.

carries out low-level control, Assumption 2 in effect implies that the the user delegates the control of the unobservable state $\eta(t) \in \mathbb{R}^{n-\Gamma(\mathcal{S})}$ to the automation. Hence, by imposing a lower bound on the user information index $\Gamma(\cdot)$ in (14), we impose an upper bound on the dimension of the unobservable states. In essence, this bound ensures that the unobservable state space does not become so large that it causes further decrease in trust.

For example, in off-nominal operation (i.e., scenarios in which the user may not trust the automation), high values of k_{trust} ensure that the unobservable state is low dimensional, and the user retains a large degree of control. On the other hand, in nominal operation, low values of k_{trust} allow the dimension of the unobservable states to increase, potentially reducing cognitive workload as the user delegates control over these variables to the automation.

Table I shows $\mathscr{S}_{\text{trust}}$ for the illustrative example under various levels of trust. We see that $\mathscr{S}_{\text{trust}} = 2^{\mathscr{S}}$ when $k_{\text{trust}} = 1$, meaning that all possible interfaces satisfy the trust constraint when the user's trust level is high. With higher k_{trust} (i.e., lower trust level), the number of sensor combinations that need to be considered for the user-interface design drastically reduces. For $k_{\text{trust}} = 4$, only four user interfaces are feasible; the observable state is 0-D for these user interfaces.

C. Dynamics-Driven User-Interface Design as Tractable, Combinatorial Optimization Problems

With the constraint for necessary conditions for situation awareness (8) and trust constraint (14) established, we reformulate (3) as the combinatorial optimization problem

$$\begin{aligned} &\underset{\mathcal{S}\in 2^{\mathscr{S}}}{\text{minimize}} && |\mathcal{S}| && \text{(15a)}\\ &\text{subject to} && \Gamma(\mathcal{S}) = \Gamma(\mathcal{S}\cup\mathcal{S}_{\text{task}}) && \text{(15b)}\\ &&& \Gamma(\mathcal{S}) \geq k_{\text{trust}}. && \text{(15c)} \end{aligned}$$

Problem (15) is well-posed since $\mathscr{S}$ is a feasible solution: $\Gamma(\mathscr{S}) = \Gamma(\mathscr{S}\cup\mathcal{S}_{\text{task}})$ and $\Gamma(\mathscr{S}) \geq k_{\text{trust}}$, by definition. In other words, the user interface constructed using all the sensors in $\mathscr{S}$ is always a feasible solution to (15), irrespective of the task $\mathcal{S}_{\text{task}}$ and the value of k_{trust}.

However, solving (15) directly is hard, due to the potentially large number of sensor combinations in consideration $2^{\mathscr{S}}$. We propose different tractable methods to solve (15) using the properties of $\Gamma(\cdot)$. First, using Theorem 1, we reformulate (15) into (16) without introducing any approximation

$$\begin{aligned} &\underset{\mathcal{S}\in 2^{\mathscr{S}},\mathcal{P}\in 2^{\mathscr{S}_{\text{reduced}}}}{\text{minimize}} && |\mathcal{S}| && \text{(16a)}\\ &\text{subject to} && \mathcal{P}\in\mathscr{S}_{\text{sit-aware,reduced}} && \text{(16b)}\\ &&& \mathcal{P} = \mathcal{S}\cap\mathscr{S}_{\text{reduced}} && \text{(16c)}\\ &&& \Gamma(\mathcal{S}) \geq k_{\text{trust}}. && \text{(16d)} \end{aligned}$$

We denote the optimal solution of (16) as $\mathcal{S}^*$ and $\mathcal{P}^*$.

Next, we investigate the submodularity and monotonicity of $\Gamma(\cdot)$ since these properties enable greedy heuristics for computing efficient, near-optimal solutions (see Appendix A). We refer the reader to [19], [21], and [54]–[56] for more details.

Definition 5 (Submodularity): A set function $f(\cdot)$ is submodular if for all sets $\mathcal{P}, \mathcal{Q} \in 2^{\mathscr{S}}$

$$f(\mathcal{P}) + f(\mathcal{Q}) \geq f(\mathcal{P}\cup\mathcal{Q}) + f(\mathcal{P}\cap\mathcal{Q}). \tag{17}$$

Definition 6 (Monotone Increasing): A set function $f(\cdot)$ is monotone increasing if for all sets $\mathcal{P}, \mathcal{Q} \in 2^{\mathscr{S}}$

$$\mathcal{P}\subseteq\mathcal{Q} \Rightarrow f(\mathcal{P}) \leq f(\mathcal{Q}). \tag{18}$$

Submodular functions demonstrate diminishing returns, i.e., adding an element to a smaller set results in a higher gain compared to adding it to a larger set. Monotone increasing functions preserve the inclusion ordering in $2^{\mathscr{S}}$.

Proposition 3: The user information index $\Gamma(\cdot)$ is a submodular monotone increasing function.

Proof: Submodularity: For any $\mathcal{P}, \mathcal{Q} \in 2^{\mathscr{S}}$, we show that $\Gamma(\cdot)$ meets (17) using Lemma 12 and Lemma 13

$$\begin{aligned} \Gamma(\mathcal{P}\cup\mathcal{Q}) &= \Gamma(\mathcal{P}) + \Gamma(\mathcal{Q}) - \dim(\mathcal{R}(T_{\mathcal{P}})\cap\mathcal{R}(T_{\mathcal{Q}}))\\ &\leq \Gamma(\mathcal{P}) + \Gamma(\mathcal{Q}) - \Gamma(\mathcal{P}\cap\mathcal{Q}). \end{aligned} \tag{19}$$

Monotone Increasing Property: It follows from Lemma 11. □

Corollary 1: For any $\mathcal{S} \in \mathscr{S}_{\text{sit-aware}}$, $\Gamma(\mathcal{S}) \geq \Gamma(\mathcal{S}_{\text{task}})$.

Corollary 2: Given $\mathcal{S}_{\text{task}} \in 2^{\mathscr{S}}$, the following conditions hold.

1) If $k_{\text{trust}} \leq \Gamma(\mathcal{S}_{\text{task}})$, then $\mathscr{S}_{\text{sit-aware}} \subseteq \mathscr{S}_{\text{trust}}$.
2) If $k_{\text{trust}} = \Gamma(\mathscr{S})$, then $\mathscr{S}_{\text{trust}} \subseteq \mathscr{S}_{\text{sit-aware}}$.

Corollary 1 provides a lower bound on $\Gamma(\mathcal{S})$ for $\mathcal{S} \in \mathscr{S}_{\text{sit-aware}}$. Corollary 21 states that the trust constraint (14) is trivially satisfied, if the user interface enables necessary conditions for situation awareness and k_{trust} is low enough (i.e., user's trust level is high enough). On other hand, when k_{trust} is as high as possible (i.e., lowest trust level), user-interface design is task agnostic, and trust constraint satisfaction automatically enables necessary conditions for situation awareness.

For the illustrative example given in Section II, note that $\Gamma(\mathcal{S}_{\text{task}}) = \Gamma(\{s_v\}) = 2$ in Table I. As stated in Corollary 2, we see that $\mathscr{S}_{\text{sit-aware}} \subset \mathscr{S}_{\text{trust}}$ when $k_{\text{trust}} \leq 2 = \Gamma(\mathcal{S}_{\text{task}})$. Furthermore, $\mathscr{S}_{\text{trust}} \subset \mathscr{S}_{\text{sit-aware}}$ when $k_{\text{trust}} = 4 = \Gamma(\mathscr{S})$.

We propose three different approaches to compute a solution to (16) under different ranges of k_{trust}.

Algorithm 2 Optimal Solution to (16) When $k_{\text{trust}} \leq \Gamma(\mathcal{S}_{\text{task}})$

Input: Set of all sensors $\mathscr{S}$, trust parameter k_{trust}, sensors that define the task $\mathcal{S}_{\text{task}}$, the user information index $\Gamma(\cdot)$
Output: An optimal solution to (16)
1: Compute $\mathscr{S}_{\text{sit-aware,reduced}}$ using Algorithm 1
2: $\mathscr{S}_{\text{feas,high}-\text{trust}} \triangleq \mathscr{S}_{\text{sit-aware,reduced}} \cap \{\mathcal{S} : |\mathcal{S}| \leq |\mathcal{S}_{\text{task}}|\}$
3: $\mathcal{S}^* \leftarrow \min\{|\mathcal{S}| : \mathcal{S} \in \mathscr{S}_{\text{feas,high}-\text{trust}}\}$
4: **return** $\mathcal{S}^*$

1) Optimal Solution When $k_{\text{trust}} \leq \Gamma(\mathcal{S}_{\text{task}})$: With a high level of trust, by Corollary 21, the trust constraint (16d) is trivially satisfied by any choice of $\mathcal{S} \in \mathscr{S}_{\text{sit-aware}}$. Because we seek sensor combinations with minimum cardinality, we search only in $\mathscr{S}_{\text{sit-aware,reduced}}$. Since $\mathcal{S}_{\text{task}}$ is a feasible solution by Lemma 2, we can reformulate (16) into (20) without introducing any approximation

$$\underset{\mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}}{\text{minimize}} \quad |\mathcal{P}| \tag{20a}$$
$$\text{subject to} \quad |\mathcal{P}| \leq |\mathcal{S}_{\text{task}}|. \tag{20b}$$

The optimal solution to (20) is also the optimal solution to (16). The constraint (20b) requires a brute force search, and hence, the numerical implementation in Algorithm 2 has a worst case computation complexity of $\mathcal{O}\Big(\sum_{i=1}^{|\mathcal{S}_{\text{task}}|} \binom{|\mathscr{S}_{\text{reduced}}|}{i}\Big)$.

2) Greedy Suboptimal Solution When $k_{\text{trust}} = \Gamma(\mathscr{S})$: With the lowest level of trust, the situation awareness constraints (16b) and (16c) are trivially satisfied for any $\mathcal{S} \in \mathscr{S}_{\text{trust}}$ by Corollary 22. By Proposition 3, (16) simplifies to the following submodular optimization problem (Appendix A):

$$\underset{\mathcal{S} \in 2^{\mathscr{S}}}{\text{minimize}} \quad |\mathcal{S}| \tag{21a}$$
$$\text{subject to} \quad \Gamma(\mathcal{S}) \geq \Gamma(\mathscr{S}). \tag{21b}$$

We compute a suboptimal solution with provable suboptimality guarantees via a greedy algorithm (Algorithm 4 in Appendix A).

3) Suboptimal Solution for $\Gamma(\mathcal{S}_{\text{task}}) < k_{\text{trust}} < \Gamma(\mathscr{S})$: For trust values in between, we propose Algorithm 3, which solves a submodular optimization problem [(32), Appendix A] for every $\mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}$

$$\underset{\mathcal{Q}_{\mathcal{P}} \subseteq \mathscr{S} \setminus \mathcal{P}}{\text{minimize}} \quad |\mathcal{Q}_{\mathcal{P}}| \tag{22a}$$
$$\text{subject to} \quad \Gamma(\mathcal{P} \cup \mathcal{Q}_{\mathcal{P}}) \geq k_{\text{trust}}. \tag{22b}$$

By Theorem 1, the optimal solution to (16) is the minimum cardinality set in the following:

$$\mathscr{S}_{\text{subopt}} = \bigcup\nolimits_{\mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}} \{\mathcal{P} \cup \mathcal{Q}_{\mathcal{P}} | \mathcal{Q}_{\mathcal{P}} \text{ solves (22)}\}. \tag{23}$$

However, solving (22) for each $\mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}$ is computationally expensive for large $|\mathscr{S}|$. We know that $\Gamma(\mathcal{P} \cup \mathcal{Q}_{\mathcal{P}})$ is a submodular monotone function in $\mathcal{Q}_{\mathcal{P}}$ for any $\mathcal{P} \in 2^{\mathscr{S}}$ [56, Sec. 1.2]. Therefore, (22) is also a submodular optimization problem. We again use the greedy approach (Algorithm 4 in Appendix A) to compute a suboptimal solution $\mathcal{Q}_{\mathcal{P}}^{\dagger}$. Note that lines 3–6 of Algorithm 3 is trivially parallelizable.

Algorithm 3 Suboptimal Solution to (16) for $\Gamma(\mathcal{S}_{\text{task}}) < k_{\text{trust}} < \Gamma(\mathscr{S})$

Input: Set of all sensors $\mathscr{S}$, trust parameter k_{trust}, sensors that define the task $\mathcal{S}_{\text{task}}$, the user information index $\Gamma(\cdot)$
Output: A suboptimal solution to (16) $\mathcal{S}_{\text{subopt}}$
1: $\mathscr{S}_{\text{subopt}} \leftarrow \emptyset$
2: Compute $\mathscr{S}_{\text{sit-aware,reduced}}$ using Algorithm 1
3: **for** $\mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}$ **do**
4: Compute $\mathcal{Q}_{\mathcal{P}}^{\dagger}$ by solving (22) given $\mathcal{P}$ suboptimally using Algorithm 4 (see Appendix A)
5: Add $\mathcal{P} \cup \mathcal{Q}_{\mathcal{P}}^{\dagger}$ to $\mathscr{S}_{\text{subopt}}$
6: **end for**
7: $\mathcal{S}_{\text{subopt}} \leftarrow \min\{|\mathcal{S}| : \mathcal{S} \in \mathscr{S}_{\text{subopt}}\}$
8: **return** $\mathcal{S}_{\text{subopt}}$

To quantify the suboptimality bound for Algorithm 3, we define a real-valued function $\Delta_{\Gamma} : \mathbb{N}_{[1,\Gamma(\mathscr{S})]} \times 2^{\mathscr{S}} \to \mathbb{R}$ as

$$\Delta_{\Gamma}(k, \mathcal{S}) = \begin{cases} \log\Big(\frac{\Gamma(\mathscr{S})}{k - \Gamma(\mathcal{S})}\Big) & \Gamma(\mathcal{S}) < k \\ \infty & \text{otherwise.} \end{cases} \tag{24}$$

Proposition 4 (Suboptimality Bound for Algorithm 3): For $\Gamma(\mathcal{S}_{\text{task}}) < k_{\text{trust}} < \Gamma(\mathscr{S})$, Algorithm 3 computes a suboptimal solution $\mathcal{S}_{\text{subopt}}$ to (16) that satisfies

$$1 \leq \frac{|\mathcal{S}_{\text{subopt}}|}{|\mathcal{S}^*|} \leq \big(1 + \max\nolimits_{\mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}} \Delta_{\Gamma}(k_{\text{trust}}, \mathcal{P} \cup \mathcal{Q}_{\mathcal{P}}^{-})\big) \tag{25}$$

where $\mathcal{P} \cup \mathcal{Q}_{\mathcal{P}}^{-}$ is the solution prior to the termination step of Algorithm 4 in Line 4 of Algorithm 3.

Proof: Let $\mathcal{S}^* = \mathcal{P}^* \cup \mathcal{Q}_{\mathcal{P}^*}^*$ be the (unknown) optimal solution to (16), where $\mathcal{P}^* \subseteq \mathscr{S}_{\text{sit-aware,reduced}}$. Such a decomposition is guaranteed by Theorem 1. Let $\mathcal{Q}_{\mathcal{P}^*}^{\dagger}$ be the solution of (22) for $\mathcal{P}^*$ computed using Algorithm 4, and $\mathcal{Q}_{\mathcal{P}^*}^{-}$ be the solution prior to the termination step. By Lemma 3 in Appendix A

$$|\mathcal{Q}_{\mathcal{P}^*}^{\dagger}| \leq |\mathcal{Q}_{\mathcal{P}^*}^{*}|\big(1 + \Delta_{\Gamma}(k_{\text{trust}}, \mathcal{P}^* \cup \mathcal{Q}_{\mathcal{P}^*}^{-})\big). \tag{26}$$

Equation (26) uses the observation that $\Gamma(\emptyset) = 0$ and upper bounds the suboptimality bound in Lemma 3 in Appendix A using Δ_{Γ}. The upper bound and the finiteness of Δ_{Γ} follow from the fact that $\Gamma(\mathcal{P}^* \cup \mathcal{Q}_{\mathcal{P}^*}^{\dagger}) \geq k_{\text{trust}} > \Gamma(\mathcal{P}^* \cup \mathcal{Q}_{\mathcal{P}^*}^{-})$ by the termination rule of Algorithm 4.

By line 7 of Algorithm 3, we have

$$|\mathcal{S}^*| = |\mathcal{P}^*| + |\mathcal{Q}_{\mathcal{P}^*}| \leq |\mathcal{S}_{\text{subopt}}| \leq |\mathcal{P}^*| + |\mathcal{Q}_{\mathcal{P}^*}^{\dagger}|. \tag{27}$$

Applying (26) to and rearranging the resulting terms

$$\begin{aligned} 1 \leq \frac{|\mathcal{S}_{\text{subopt}}|}{|\mathcal{S}^*|} &\leq \left(1 + \frac{|\mathcal{Q}_{\mathcal{P}^*}^{*}|}{|\mathcal{S}^*|} \Delta_{\Gamma}(k_{\text{trust}}, \mathcal{P}^* \cup \mathcal{Q}_{\mathcal{P}^*}^{-})\right) \\ &\leq \big(1 + \Delta_{\Gamma}(k_{\text{trust}}, \mathcal{P}^* \cup \mathcal{Q}_{\mathcal{P}^*}^{-})\big) \\ &\leq \left(1 + \max_{\mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}} \Delta_{\Gamma}(k_{\text{trust}}, \mathcal{P} \cup \mathcal{Q}_{\mathcal{P}}^{-})\right) \end{aligned}$$

since $|\mathcal{Q}_{\mathcal{P}^*}^{*}| \leq |\mathcal{S}^*|$ and $\Delta_{\Gamma}(k_{\text{trust}}, \mathcal{P}^* \cup \mathcal{Q}_{\mathcal{P}^*}^{-})$ is bounded from above by $\max_{\mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}} \Delta_{\Gamma}(k_{\text{trust}}, \mathcal{P} \cup \mathcal{Q}_{\mathcal{P}}^{-})$. For every $\mathcal{P} \in$

TABLE II
SOLUTION METHODS TO (16) FOR $k_{\text{trust}} \in [1, \Gamma(\mathscr{S})]$

Range of k_{trust}	Method	Optimality	Worst-case compute complexity
$k_{\text{trust}} \leq \Gamma(\mathcal{S}_{\text{task}})$	Alg. 2	Optimal	$\mathcal{O}\left(\sum_{i=1}^{\lvert\mathcal{S}_{\text{task}}\rvert}\binom{\lvert\mathscr{S}_{\text{reduced}}\rvert}{i}\right)$
$\Gamma(\mathcal{S}_{\text{task}}) < k_{\text{trust}} < \Gamma(\mathscr{S})$	Alg. 3	Suboptimal as in (25)	$\mathcal{O}\left(2^{\lvert\mathscr{S}_{\text{reduced}}\rvert}\lvert\mathscr{S}\rvert^2\right)$
$k_{\text{trust}} = \Gamma(\mathscr{S})$	Solve (21) via Alg. 4	Suboptimal as in (33)	$\mathcal{O}\left(\lvert\mathscr{S}\rvert^2\right)$

$\mathscr{S}_{\text{sit-aware,reduced}}$, $\Delta_\Gamma(k_{\text{trust}}, \mathcal{P}\cup\mathcal{Q}^-_{\mathcal{P}}) < k_{\text{trust}}$ by the termination rule of Algorithm 4. □

We simplify (25) to obtain a weaker upper bound

$$|\mathcal{S}^*| \leq |\mathcal{S}_{\text{subopt}}| \leq |\mathcal{S}^*|(1 + \log(\Gamma(\mathscr{S}))). \tag{28}$$

Equation (28) follows from the observation that $\Delta_\Gamma(k_{\text{trust}}, \mathcal{P} \cup \mathcal{Q}^-_{\mathcal{P}}) \leq \log(\Gamma(\mathscr{S}))$ for every $\mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}$. Therefore, $\max_{\mathcal{P}\in\mathscr{S}_{\text{sit-aware,reduced}}} \Delta_\Gamma(k_{\text{trust}}, \mathcal{P}\cup\mathcal{Q}^-_{\mathcal{P}}) \leq \log(\Gamma(\mathscr{S})))$. Equation (28) shows that the upper bound in (25) can not be arbitrarily loose.

Proposition 5 (Computational Complexity Bound for Algorithm 3): For $\Gamma(\mathcal{S}_{\text{task}}) < k_{\text{trust}} < \Gamma(\mathscr{S})$, Algorithm 3 has a worst case computational complexity of $\mathcal{O}\big(2^{|\mathscr{S}_{\text{reduced}}|}|\mathscr{S}|^2\big)$.

Proof: In Algorithm 3, the evaluation of lines 2, 3–6, and 7 have a worst case computational complexity of $\mathcal{O}(2^{|\mathscr{S}_{\text{reduced}}|})$, $\mathcal{O}(|\mathscr{S}_{\text{sit-aware,reduced}}||\mathscr{S}|^2)$ (from Lemma 3 in Appendix A), and $\mathcal{O}(|\mathscr{S}_{\text{sit-aware,reduced}}|)$, respectively. The worst case computational complexity of Algorithm 3 is $\mathcal{O}(|\mathscr{S}_{\text{sit-aware,reduced}}||\mathscr{S}|^2 + |\mathscr{S}_{\text{sit-aware,reduced}}| + |2^{\mathscr{S}_{\text{reduced}}}|)$. Using the observation that $|\mathscr{S}_{\text{sit-aware,reduced}}| \leq 2^{|\mathscr{S}_{\text{reduced}}|}$, we obtain the simplified worst case complexity bound. □

An alternative heuristic to Algorithm 3 is to use Algorithm 2 to solve (20) to obtain $\mathcal{P}^\dagger \subseteq \mathscr{S}_{\text{sit-aware,reduced}}$ that enables necessary conditions for situation awareness and then solve the associated submodular maximization problem (22) with $\mathcal{P}^\dagger$. This approach may provide a faster solution since the search for $\mathcal{P}^\dagger$ is assisted by the cardinality constraint (20b). Furthermore, it only requires the solution of a single submodular maximization problem, as opposed to a collection of $|\mathscr{S}_{\text{sit-aware,reduced}}|$ problems in Algorithm 3. However, the suboptimality bound of Algorithm 3 no longer holds for this approach since (27) fails to hold.

The approaches proposed in this section are summarized in Table II.

IV. EFFICIENT IMPLEMENTATION OF ALGORITHM 1

We propose a computationally efficient implementation of Algorithm 1 using constraint programming and a novel enumeration framework based on binary number representation. The proposed approach exploits the monotonicity properties of the user information index function.

A. Enumerating $\mathscr{S}_{\text{sit-aware,reduced}}$ via Constraint Programming

Constraint programming exploits transitivity properties in set functions to reduce the search space [57] (for example, for a monotone increasing constraint $f(\mathcal{S}) \leq k$ for some $k \in \mathbb{N}$, infeasibility of $\mathcal{P} \in 2^{\mathscr{S}}$ implies infeasibility of all $\mathcal{Q} \in 2^{\mathscr{S}}$ such that $\mathcal{P} \subseteq \mathcal{Q}$). To avoid enumeration in Algorithm 1 of the set $\mathscr{S}_{\text{sit-aware,reduced}}$ (11) using Theorem 1, we construct the feasibility problem corresponding to (16)

$$\text{find all} \quad \mathcal{P} \in 2^{\mathscr{S}_{\text{reduced}}}, \; t \in \mathbb{N}_{[\Gamma(\mathcal{S}_{\text{task}}),\Gamma(\mathscr{S}_{\text{reduced}})]} \tag{29a}$$

$$\text{subject to} \quad \Gamma(\mathcal{P}) \geq t \tag{29b}$$

$$\Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}) \leq t. \tag{29c}$$

Since $\Gamma(\cdot)$ is monotone increasing (Proposition 3), we can prune the search space when a tuple $(\mathcal{P}, t)$ that does not satisfy (29c) is encountered. Specifically, given $\mathcal{P}_1 \in \mathscr{S}_{\text{reduced}}$ such that $\Gamma(\mathcal{P}_1 \cup \mathcal{S}_{\text{task}}) \not\leq t_0$ for some $t_0 \in \mathbb{N}_{[\Gamma(\mathcal{S}_{\text{task}}),\Gamma(\mathscr{S}_{\text{reduced}})]}$, then for every superset $\mathcal{P}_2 \in \mathscr{S}_{\text{reduced}}$, $\mathcal{P}_1 \subseteq \mathcal{P}_2$, and $t \leq t_0$, we know $\Gamma(\mathcal{P}_2 \cup \mathcal{S}_{\text{task}}) \not\leq t$. We can also incorporate the cardinality constraint (20b) to further restrict the search space in Algorithm 2.

Proposition 6: A set $\mathcal{P} \subseteq \mathscr{S}_{\text{reduced}}$ is feasible for (29) for some $t \in \mathbb{N}_{[\Gamma(\mathcal{S}_{\text{task}}),\Gamma(\mathscr{S}_{\text{reduced}})]}$ if and only if $\mathcal{P} \in \mathscr{S}_{\text{sit-aware,reduced}}$.

Proof: The constraints (29b) and (29c) together are equivalent to the following equality constraint [identical to (11)]

$$t = \Gamma(\mathcal{P}) = \Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}).$$

We have $t \in \mathbb{N}_{[\Gamma(\mathcal{S}_{\text{task}}),\Gamma(\mathscr{S}_{\text{reduced}})]}$ since $\Gamma(\cdot)$ is monotone increasing (Proposition 3) and Corollary 1. □

B. Computationally Efficient Enumeration of the Search Space

We employ constraint propagation to enumerate the search space, which, for ease of discussion, we presume is $2^{\mathscr{S}}$. We desire to create an oracle, referred to as a *generator*, that provides the next sensor combination in $2^{\mathscr{S}}$ which needs to be evaluated. The generator must satisfy three requirements.

1) *R1:* Produce sensor combinations within $2^{\mathscr{S}}$ in an exhaustive manner.
2) *R2:* Eliminate sensor combinations that are a superset of a given set.
3) *R3:* Enforce cardinality constraints.

Any sensor combination $\mathcal{S} \in 2^{\mathscr{S}}$ can be associated with a unique $|\mathscr{S}|$-bit long binary number representation, with the bit values set to one at the respective positions of every selected sensor. We also use a bijection of this representation to the corresponding decimal number $N_{\mathcal{S}} \in \mathbb{N}_{[0,2^{|\mathscr{S}|}-1]}$. Therefore, any generator over $\mathbb{N}_{[0,2^{|\mathscr{S}|}-1]}$ exhaustively enumerates $2^{\mathscr{S}}$.

A naive approach to enumerate $2^{\mathscr{S}}$ is to use a linear generator, which enumerates $\mathbb{N}_{[0,2^{|\mathscr{S}|}-1]}$ by incrementing $N_{\mathcal{S}}$ by 1. However, satisfying R2 and R3 with a linear generator is difficult. We propose a generator that satisfies all three requirements by enumerating over a tabular representation of $2^{\mathscr{S}}$. We associate a unique column number $\text{Col}_{\mathcal{S}} \in \mathbb{N}_{[0,|\mathscr{S}|-1]}$ and row number $\text{Row}_{\mathcal{S}} \in \mathbb{N}_{[0,2^{|\mathscr{S}|-1}-1]}$ with every sensor combination $\mathcal{S} \in 2^{\mathscr{S}}$

$$\text{Col}_{\mathcal{S}} = \lfloor \log_2(N_{\mathcal{S}}) \rfloor \tag{30a}$$

$$\text{Row}_{\mathcal{S}} = N_{\mathcal{S}} - 2^{\text{Col}_{\mathcal{S}}} \tag{30b}$$

TABLE III
BINARY ITERATION TABLE FOR $\mathscr{S} = \{s_0, s_1, s_2, s_3, s_4\}$

Row	s_0	s_1	s_2	s_3	s_4
0	1	2	4	8	16
1		3	5	9	17
2			6	10	18
3			7	11	19
4				12	20
5				13	21
6				14	22
7				15	23
8					24
9					25
10					26
11					27
12					28
13					29
14					30
15					31

where $\lfloor a \rfloor$ is the floor of $a \in \mathbb{R}$, the largest integer below a. The column number is the position of the most significant bit of the binary representation of $N_{\mathcal{S}}$, and the row number is the decimal representation of the number defined by the remaining bits. The number of nonzero bits in the binary representation of $\mathcal{S}$ is equal to $|\mathcal{S}|$.

We demonstrate this approach on $\mathscr{S} = \{s_0, s_1, s_2, s_3, s_4\}$ in Table III. For illustration, consider $\mathcal{S} = \{s_0, s_4\}$. We associate with $\mathcal{S}$ a binary representation, 10001, based on the selection of sensors. The decimal number representation of 10001 is $N_{\mathcal{S}} = 17$. Note that $|\mathcal{S}| = 2$ is the number of nonzero bits in the binary representation 10001. By (30a), $\mathrm{Col}_{\mathcal{S}} = 4$, which is the position (count starts from zero) of the most significant nonzero bit. By (30b), $\mathrm{Row}_{\mathcal{S}} = 1 = 17 - 2^4$.

1) Satisfaction of R1: All numbers in $\mathbb{N}_{[0,2^{|\mathscr{S}|}-1]}$ have a unique position in the tabular representation of $2^{\mathscr{S}}$, which follows from the unique binary representation of $N_{\mathcal{S}}$ by (30). Thus, any enumeration of the proposed table satisfies R1.

2) Satisfaction of R2: Due to (30b), each row contains sensor combinations with a similar pattern in the lower significant bits. Specifically, the binary representation of the row number coincides with the binary representation of $N_{\mathcal{S}}$ *without* its most significant bit. For example, row 3 of Table III contains numbers 7 (select s_0, s_1, s_2), 11 (select s_0, s_1, and s_3) and 19 (select s_0, s_1, and s_4). All these numbers have the elements s_0 and s_1 in common since their row number, 3, has the binary representation 00011.

Using this observation, we skip enumeration of the supersets of infeasible sets, by maintaining a collection of rows to skip. For example, suppose that we wish to skip enumeration of all supersets of $\mathcal{P} = \{s_0, s_1\}$. This is the case when $\mathcal{P}$ violates (29c). We must skip rows 3, 7, 11, and 15 of Table III, as they are the row numbers with the pattern $XX11$ where X indicates "don't care" bits. The generator then produces $\mathcal{S}$, with $N_{\mathcal{S}} \notin \{7, 11, 19, 15, 23, 27, 31\} = \{00111, 01011, 10011, 01111, 10111, 11011, 11111\}$. Note that each of the skipped numbers has the bits set at their zeroth and first positions, i.e., they are supersets of $\mathcal{P}$.

3) Satisfaction of R3: Recall that the binary representation of $\mathrm{Row}_{\mathcal{S}}$ provides an accurate characterization of $\mathcal{S}$, except for one sensor element. Therefore, the number of nonzero bits in the binary representation of $\mathrm{Row}_{\mathcal{S}}$ is equal to $|\mathcal{S}| - 1$ since the most significant bit is excluded. Thus, by restricting the number of bits in the binary representation of the enumerated row numbers, we can enforce cardinality constraints like (20b) and satisfy R3.

The proposed generator provides an efficient way to enumerate the search space and incorporate constraint programming. Specifically, the row-wise enumeration permits the enforcement of cardinality constraints as well as the elimination of supersets of an infeasible sensor combination. We use this framework for computations involving Algorithm 1, including computation of the set $\mathscr{S}_{\text{subopt}}$ in Algorithm 2 and the set $\mathscr{S}_{\text{sit-aware,reduced}}$ in Algorithm 3.

V. APPLICATION: USER-INTERFACE DESIGN FOR IEEE 118-BUS POWER GRID

The IEEE 118-bus model is a power network composed of 118 buses, 54 synchronous machines (generators), 186 transmission lines, nine transformers, and 99 loads [58]. We use linearized swing dynamics to describe the interconnected generator dynamics [59], [60]. As is typically done in large networks [61], [62], we used Kron reduction to reduce the network to a generator-only network with LTI dynamics

$$\dot{x}(t) = A_i x(t) + B_j u(t). \tag{31}$$

Here, the state $x(t) \in \mathbb{R}^{108}$ denotes the phase and phase rate for each of the 54 generator buses, and the input $u(t) \in \mathbb{R}^m$ denotes the power injection provided at each generator bus. We construct the system and input matrices, $A_i \in \mathbb{R}^{108\times 108}$ and $B_j \in \mathbb{R}^{108\times m_j}$, under four different network configurations.

1) Normal operation (A_1, B_1) with $m_1 = 54$.
2) Load bus 38 is down (A_2, B_1) with $m_1 = 54$.
3) Lines 65 and 66 is down (A_3, B_1) with $m_1 = 54$.
4) Alternate generators are down (A_1, B_2) with $m_2 = 27$.

The admittance values for the interconnections in the reduced network were obtained using `MATPOWER` [63]. We considered all the generators to be homogenous. We chose the moment of inertia and damping coefficients as $H = 2.656\ s$ and $D = 2$ [64, Table I].

We presume that the user (a power grid operator) is tasked with the maintaining the power flow to a predetermined substation (generator bus 28) under each of the four network configurations. The power grid operator therefore requires information about the power flowing from all neighboring nodes, which can be described as nonlinear functions of the difference in phase measurements [59], [60]. Therefore, the task is defined in terms of the phase measurements of the generator buses that have a direct connection to bus 28 in the Kron reduced network, consisting of only generator buses.

We define $\mathscr{S}$ to be all phase measurements of the generators on the Kron reduced network, $\mathscr{S} = \{e_i : i \in \mathbb{N}_{[1,54]}\}$, where e_i is a column vector of zeros with one at the ith component. For the first three configurations, we have task matrices $C_{\mathcal{S}_{\text{task},1}}$, $C_{\mathcal{S}_{\text{task},2}}$, and $C_{\mathcal{S}_{\text{task},3}}$ due to differences in the neighbors to bus 28. Since the network configuration is the same in the first and the fourth configurations, the task matrix for the fourth configuration is also $C_{\mathcal{S}_{\text{task},1}}$.

TABLE IV
OPTIMAL USER-INTERFACE SOLUTIONS AND COMPUTATION TIME FOR IEEE 118-BUS POWER GRID PROBLEM

Network configuration	Trust level (k_{trust})	User-interface design				Compute Time
		Approach	$\lvert\mathscr{S}_{\text{sit-aware,reduced}}\rvert$	$\lvert\mathcal{S}_{\text{soln}}\rvert \leq \Delta\lvert\mathcal{S}^*\rvert$	Solution	
Normal operation $(A_1, B_1, C_{\mathcal{S}_{\text{task},1}})$ $\Gamma(\mathcal{S}_{\text{task},1}) = 34$	High ($k_{\text{trust}} = 24$)	Alg. 2	1	$\Delta = 1$	$\mathcal{S}_{\text{task},1}$	0.23 s
	Moderate ($k_{\text{trust}} = 44$)	Alg. 3	1	$\Delta = 4.09$	$\mathcal{S}_{\text{task},1} \cup \{1, 2, 3, 5, 9\}$	0.91 s
	None ($k_{\text{trust}} = 108$)	Alg. 4	–	$\Delta = 4.99$	$\mathscr{S}$	9.62 s
Bus 38 down $(A_2, B_1, C_{\mathcal{S}_{\text{task},2}})$ $\Gamma(\mathcal{S}_{\text{task},2}) = 14$	High ($k_{\text{trust}} = 4$)	Alg. 2	1	$\Delta = 1$	$\mathcal{S}_{\text{task},1}$	0.11 s
	Moderate ($k_{\text{trust}} = 24$)	Alg. 3	1	$\Delta = 3.49$	$\mathcal{S}_{\text{task},2} \cup \{1, 2, 3, 4, 5\}$	0.49 s
	None ($k_{\text{trust}} = 108$)	Alg. 4	–	$\Delta = 4.99$	$\mathscr{S}$	9.47 s
Line 65, 66 down $(A_3, B_1, C_{\mathcal{S}_{\text{task},3}})$ $\Gamma(\mathcal{S}_{\text{task},3}) = 30$	High ($k_{\text{trust}} = 20$)	Alg. 2	1	$\Delta = 1$	$\mathcal{S}_{\text{task},3}$	0.20 s
	Moderate ($k_{\text{trust}} = 40$)	Alg. 3	1	$\Delta = 4.00$	$\mathcal{S}_{\text{task},3} \cup \{1, 2, 3, 5, 9\}$	0.77 s
	None ($k_{\text{trust}} = 108$)	Alg. 4	–	$\Delta = 4.99$	$\mathscr{S}$	9.26 s
Alternate generators down $(A_1, B_2, C_{\mathcal{S}_{\text{task},1}})$ $\Gamma(\mathcal{S}_{\text{task},1}) = 52$	High ($k_{\text{trust}} = 42$)	Alg. 2	2306	$\Delta = 1$	$\mathcal{S}_{\text{task},1} \setminus \{37, 53\}$	$\sim 10^5$ s
	Moderate ($k_{\text{trust}} = 62$)	Alg. 3	2306	$\Delta = 4.43$	$\mathcal{S}_{\text{task},1} \setminus \{37, 53\} \cup \{2, 52\}$	$\sim 10^5$ s
	None ($k_{\text{trust}} = 108$)	Alg. 4	–	$\Delta = 4.99$	$\mathscr{S} \setminus \{5, 7, 9, 11, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 50, 51, 53, 54\}$	3.23 s

We consider three different trust levels $k_{\text{trust}} \in \{\Gamma(\mathcal{S}_{\text{task}}) - 10, \Gamma(\mathcal{S}_{\text{task}}) + 10, \Gamma(\mathscr{S})\}$. Informally, this may be interpreted as designing the user interface under the following conditions.

1) *High Trust:* Normal operating conditions, in which the user trusts the automation to a high degree ($k_{\text{trust}} = \Gamma(\mathcal{S}_{\text{task}}) - 10$).
2) *Moderate Trust:* Off-nominal operating conditions, in which the user has some distrust of the automation, but not excessive distrust ($k_{\text{trust}} = \Gamma(\mathcal{S}_{\text{task}}) + 10$).
3) *No Trust:* Extreme, off-nominal operating conditions, in which the user totally distrusts the automation ($k_{\text{trust}} = \Gamma(\mathscr{S})$).

Our results for the four configurations, under each of the three trust levels, is shown in Table IV. In comparison, typical power grid transmission operators would have the entire grid topology available to them, with all phase angles, voltages, and currents, irrespective of the user's trust level or the task.

In configurations 1–3, the relative degree $\gamma(s_i) = 2$ for every $s_i \in \mathscr{S}$. This means that given the phase measurement of generator bus $i \in \mathbb{N}_{[1,54]}$, the user can only infer the phase and the phase rate measurement of bus i, but not of the other buses.

1) *High Trust:* Due to this decoupling, $\mathcal{S}_{\text{task},1}$ is the only user interface that enables necessary conditions for situation awareness for configurations 1–3 under high trust. In other words, we need to monitor all the buses that are involved in the task specification.
2) *Moderate Trust:* Additional sensors are required; due to the decoupling of the generator dynamics in the network, any combination of five previously unselected generators can satisfy the trust constraint.
3) *No Trust:* Phase measurements from all the buses must be displayed to attain a user information index of $\Gamma(\mathscr{S}) = n = 108$. Note that even though this is the optimal solution to (16) for $k_{\text{trust}} = 108$, the conservative suboptimality bound for Algorithm 4 is $\Delta = 4.99$.

For the fourth configuration (shown in Fig. 5), in which only alternate generators are operational, phase measurements of bus i let the user infer information about the network beyond bus i. Algorithm 1 returned a nontrivial $\mathscr{S}_{\text{sit-aware,reduced}}$, with 2306 elements, each of which could enable necessary conditions for situation awareness.

1) *High Trust:* Algorithm 2 identified $\mathcal{S}_{\text{task},1} \setminus \{37, 53\}$ as an optimal user interface, which in contrast to configurations 1–3, provides sensors other than those associated with the task. This interface exploits the underlying dynamics and the user's perception, comprehension, and projection so that phases of the task-relevant generators can be reconstructed based on less information than would be provided merely by duplicating the sensors associated with the task. Specifically, paths between nodes 28, 37, and 53 in the network topology of the 118-bus grid, which appear in a block of the dynamics matrix in the generator swing equations, allow the user to reconstruct relevant states with fewer sensors at this level of trust.
2) *Moderate Trust:* Algorithm 3 determined that two additional sensors were required. Since the user has some distrust in the automation in this case, the additional sensors reveal additional paths in the network topology that can be observed through the dynamics, which allows the user to reconstruct supplemental states and their derivatives relevant to monitoring bus 28 to meet the trust constraint.
3) *No Trust:* Algorithm 4 yields a set of 28 sensors that are needed to monitor power flow to bus 28. When the user fully distrusts the automation, a relatively large number of sensors are required to allow the user to reconstruct and understand states relevant to monitoring bus 28, where previously were entrusted to the automation. These sensors correspond to generators spread across the network, in order to improve observability over the entire system.

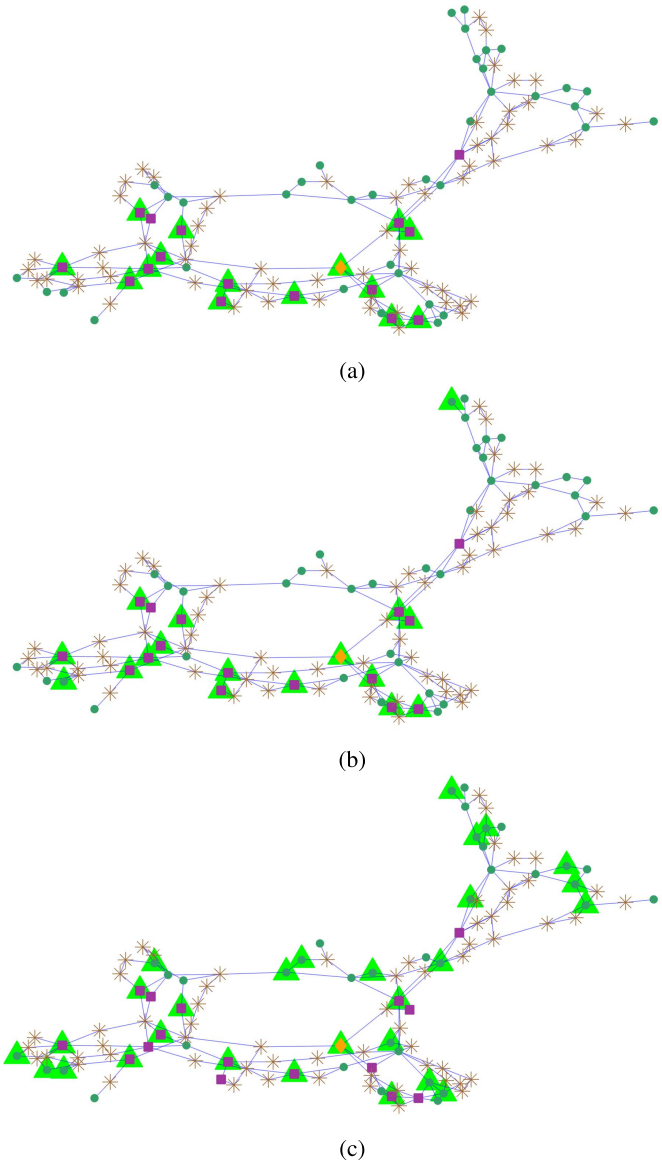


Fig. 5. Three interfaces are shown for the IEEE 118-bus under configuration 4, in which alternate generators are operational. The power grid operator's task is to maintain power flow at bus 28 (♦). The user interface consists of selected generator phase angles (▲); neighboring generators (■), load buses ($*$), and generator buses (•) are shown for clarity. As expected, more generators must be monitored with lower levels of trust. (a) High trust ($k_{\text{trust}} = 42$). (b) Moderate trust ($k_{\text{trust}} = 62$). (c) No trust ($k_{\text{trust}} = 118$).

Using efficient enumeration techniques described in Section IV, enumerating $2^{\mathscr{S}_{\text{reduced}}}$ with $|2^{22}| \approx 4 \times 10^6$ elements took only approximately 10^5 s (about 27 h) to compute. In contrast, a naive approach using linear search over $2^{\mathscr{S}}$ for the minimum cardinality set that satisfies the constraints would require checking $|2^{\mathscr{S}}| = 2^{108} \approx 3 \times 10^{32}$ elements, resulting in approximately 10^{13} billion hours computation time (presuming each evaluation takes 10^{-4} s). All computations were performed using MATLAB on an Intel i7-4600U CPU with four cores, 2.1-GHz clock rate, and 7.5-GB RAM.

VI. Conclusion

This article presents a method for user-interface design via sensor selection. Unlike many UI-based approaches, our method is driven by the underlying dynamics of the human–automation system and constrained by necessary conditions for situation awareness, as well as the user's trust in the automation. We use submodular maximization and constraint programming to solve the sensor selection problem as a constrained combinatorial optimization, and exploit submodularity and monotonicity properties to identify optimal or suboptimal solutions. We applied our approach to a large human–automation system, consisting of a power grid operator for the IEEE 118-bus model and constructed correct-by-design interfaces for a variety of trust levels and operating scenarios.

Algorithm 4 Greedy Algorithm to Solve (32)

Input: Submodular monotone increasing function $f(\cdot)$, power set $2^{\mathscr{S}}$, submodular function lower bound k

Output: Optimal greedy solution $\mathcal{S}^*_{\text{greedy}}$ and the solution prior to termination step $\mathcal{S}^-_{\text{greedy}}$ (see (33))

procedure GREEDYALGORITHM
  $\mathcal{S}^*_{\text{greedy}} \leftarrow \emptyset$
  while $f(\mathcal{S}^*_{\text{greedy}}) < k$ **do**
    $s^* \leftarrow \underset{s \in \mathscr{S} \setminus \mathcal{S}^*_{\text{greedy}}}{\operatorname{argmax}} \; f(\mathcal{S}^*_{\text{greedy}} \cup \{s\}) - f(\mathcal{S}^*_{\text{greedy}})$
    $\mathcal{S}^-_{\text{greedy}} \leftarrow \mathcal{S}^*_{\text{greedy}}$, $\mathcal{S}^*_{\text{greedy}} \leftarrow \mathcal{S}^*_{\text{greedy}} \cup \{s^*\}$
  end while
  return $\mathcal{S}^*_{\text{greedy}}, \mathcal{S}^-_{\text{greedy}}$
end procedure

Appendix

A. Submodularity in Combinatorial Optimization Problems

Let $\mathscr{S}$ denote a finite set. Consider the following combinatorial optimization problem with a submodular, monotone increasing set function $f : 2^{\mathscr{S}} \to \mathbb{N}_{[0, f(\mathscr{S})]}$:

$$\begin{aligned} &\underset{\mathcal{S} \in 2^{\mathscr{S}}}{\text{minimize}} \; |\mathcal{S}| \\ &\text{subject to } f(\mathcal{S}) \geq k \end{aligned} \tag{32}$$

for some problem parameter $k \in \mathbb{N}_{[1, f(\mathscr{S})]}$.

Lemma 3 (Suboptimality Bound for the Greedy Solution to (32) *[21], [22]):* Submodular maximization problem (32) admits a $\mathcal{O}(|\mathscr{S}|^2)$ greedy algorithm (Algorithm 4) such that its solution $\mathcal{S}^*_{\text{greedy}}$ satisfies the property

$$1 \leq \frac{|\mathcal{S}^*_{\text{greedy}}|}{|\mathcal{S}^*|} \leq 1 + \log\left(\frac{f(\mathscr{S}) - f(\emptyset)}{f(\mathcal{S}^*_{\text{greedy}}) - f(\mathcal{S}^-_{\text{greedy}})}\right) \tag{33}$$

with $\mathcal{S}^-_{\text{greedy}}$ is the solution at the iteration prior to termination of Algorithm 4.

Algorithm 4 is a greedy approach to solve the submodular optimization problem (32) with provable worst case suboptimality bounds (Lemma 3). The suboptimality bound given by Lemma 3 is the best bound available by any polynomial-time algorithm [65], assuming that $P \neq NP$. The bound in (33) is a worst case bound; Algorithm 4 often performs significantly better in practice [21].

B. Proof of Lemma 1

We have 1) from (5), (6), and (7).

We have 2) by (34) and [39, Sec. 1.6, Ex. 29],

$$\mathcal{R}(T_{\mathcal{P}\cup\mathcal{Q}}) = \mathcal{R}(T_{\mathcal{P}}) + \mathcal{R}(T_{\mathcal{Q}}). \tag{34}$$

We have 3) from (35) [39, Sec. 1.4, Ex. 15] and 1),

$$\mathcal{R}(T_{\mathcal{P}\cap\mathcal{Q}}) \subseteq \mathcal{R}(T_{\mathcal{P}}) \cap \mathcal{R}(T_{\mathcal{Q}}). \tag{35}$$

We have 4) from 1) and 2). □

C. Proof of Theorem 1

We have to show that

$$\mathscr{S}_{\text{sit-aware}} = \left\{ \mathcal{S} \in \mathscr{S} \,\middle|\, \begin{array}{c} \mathcal{P} = \mathcal{S} \cap \mathscr{S}_{\text{reduced}}, \\ \Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}) = \Gamma(\mathcal{P}) \end{array} \right\}. \tag{36}$$

We show that $\Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{S}) = \Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{P})$, which implies that $\Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{S}) = 0$ if and only if $\Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{P}) = 0$. This implies (36) by Proposition 2.

For any $\mathcal{S} \in 2^{\mathscr{S}}$, we use (9) and (36) to define

$$\mathcal{P} = \mathcal{S} \cap \mathscr{S}_{\text{reduced}} \in 2^{\mathscr{S}_{\text{reduced}}} \text{ and } \mathcal{Q} = \mathcal{S} \setminus \mathcal{P}. \tag{37}$$

Proof for $\Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{S}) = \Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{P})$*:* We will use the claims.

1) $\dim\big(\mathcal{R}(T_{\mathcal{Q}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) = 0.$

2) $\dim\big(\mathcal{R}(T_{\mathcal{P}}) \cap \mathcal{R}(T_{\mathcal{Q}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) = 0.$

On applying Lemma 12 twice

$$\begin{aligned} \Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}}) &= \Gamma(\mathcal{P} \cup \mathcal{Q} \cup \mathcal{S}_{\text{task}}) \\ &= \Gamma(\mathcal{P}) + \Gamma(\mathcal{Q}) + \Gamma(\mathcal{S}_{\text{task}}) \\ &\quad -\dim(\mathcal{R}(T_{\mathcal{P}}) \cap \mathcal{R}(T_{\mathcal{Q}})) \\ &\quad -\dim\big(\mathcal{R}(T_{\mathcal{P}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) \\ &\quad -\dim\big(\mathcal{R}(T_{\mathcal{Q}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) \\ &\quad +\dim\big(\mathcal{R}(T_{\mathcal{P}}) \cap \mathcal{R}(T_{\mathcal{Q}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big). \end{aligned} \tag{38}$$

By our claims above, the last two terms in (38) are zero. By adding and subtracting an additional $\Gamma(\mathcal{P})$ to (38), we have $\Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}}) = \Gamma(\mathcal{P} \cup \mathcal{Q}) + \Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{P})$. Since $\mathcal{S} = \mathcal{P} \cup \mathcal{Q}$, we have $\Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{S}) = \Gamma(\mathcal{P} \cup \mathcal{S}_{\text{task}}) - \Gamma(\mathcal{P})$.

Proof of claim 1): By (10), for any $s \in \mathscr{S} \setminus \mathscr{S}_{\text{reduced}}$, $\dim\big(\mathcal{R}(T_s) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) = 0$. Therefore, for any $s_1, s_2 \in \mathscr{S} \setminus \mathscr{S}_{\text{reduced}}$, $\dim\big(\mathcal{R}\big(T_{s_1}\big) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) = \dim\big(\mathcal{R}\big(T_{s_2}\big) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) = \dim\big(\mathcal{R}\big(T_{s_1}\big) \cap \mathcal{R}\big(T_{s_2}\big) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) = 0$. By applying Lemma 12 twice, we have $\Gamma(\{s_1\} \cup \{s_2\} \cup \mathcal{S}_{\text{task}}) = \Gamma(\{s_1\} \cup \{s_2\}) + \Gamma(\mathcal{S}_{\text{task}})$. This also implies that $\dim\big(\mathcal{R}\big(T_{\{s_1\}\cup\{s_2\}}\big) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) = 0$. Using similar arguments inductively, we conclude that $\dim\big(\mathcal{R}(T_{\mathcal{Q}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) = 0$ since $\mathcal{Q} \subseteq 2^{\mathscr{S}\setminus\mathscr{S}_{\text{reduced}}}$.

Proof of claim 2): Clearly, $\mathcal{R}(T_{\mathcal{P}}) \cap \mathcal{R}(T_{\mathcal{Q}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)$ is a subset of $\mathcal{R}(T_{\mathcal{Q}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)$, which implies that $\dim\big(\mathcal{R}(T_{\mathcal{P}}) \cap \mathcal{R}(T_{\mathcal{Q}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big)$ is smaller than $\dim\big(\mathcal{R}(T_{\mathcal{Q}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big)$, by definition. Since $\dim\big(\mathcal{R}(T_{\mathcal{Q}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) = 0$ (shown in claim 1), we have $\dim\big(\mathcal{R}(T_{\mathcal{P}}) \cap \mathcal{R}(T_{\mathcal{Q}}) \cap \mathcal{R}\big(T_{\mathcal{S}_{\text{task}}}\big)\big) = 0$. □

D. Proof of Lemma 2

By (9), $\Gamma(s \cup \mathcal{S}_{\text{task}}) = \Gamma(\mathcal{S}_{\text{task}}) < \Gamma(s) + \Gamma(\mathcal{S}_{\text{task}})$ for each $s \in \mathcal{S}_{\text{task}}$. Thus, $\mathcal{S}_{\text{task}} \in \mathscr{S}_{\text{reduced}}$.

By construction, $\Gamma(\mathcal{S}_{\text{task}} \cup \mathcal{S}_{\text{task}}) = \Gamma(\mathcal{S}_{\text{task}})$. Thus, $\mathcal{S}_{\text{task}} \in \mathscr{S}_{\text{sit-aware,reduced}}$. □

E. Proof of Corollary 1

Since $\mathcal{S}_{\text{task}} \subseteq (\mathcal{S} \cup \mathcal{S}_{\text{task}})$, we have $\Gamma(\mathcal{S}) = \Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}}) \geq \Gamma(\mathcal{S}_{\text{task}})$ for every $\mathcal{S} \in \mathscr{S}_{\text{sit-aware}}$ (8), due to the monotone increasing property of $\Gamma(\cdot)$. □

F. Proof of Corollary 2

To prove 1), we note that for any $\mathcal{S} \in \mathscr{S}_{\text{sit-aware}}$ with $k_{\text{trust}} \leq \Gamma(\mathcal{S}_{\text{task}})$, we have $k_{\text{trust}} \leq \Gamma(\mathcal{S}_{\text{task}}) \leq \Gamma(\mathcal{S})$ by Corollary 1. Thus, $\mathcal{S} \in \mathscr{S}_{\text{trust}}$.

To prove 2), we note that for any $\mathcal{S} \in \mathscr{S}_{\text{trust}}$, $k_{\text{trust}} \leq \Gamma(\mathcal{S}) \leq \Gamma(\mathscr{S})$. Since $k_{\text{trust}} = \Gamma(\mathscr{S})$, $\Gamma(\mathcal{S}) = \Gamma(\mathscr{S})$ for any $\mathcal{S} \in \mathscr{S}_{\text{trust}}$. By the monotone increasing property of $\Gamma(\cdot)$ (Proposition 3), we have $\Gamma(\mathscr{S}) = \Gamma(\mathcal{S}) \leq \Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}}) \leq \Gamma(\mathscr{S})$, which implies $\Gamma(\mathcal{S}) = \Gamma(\mathcal{S} \cup \mathcal{S}_{\text{task}})$. Thus, $\mathcal{S} \in \mathscr{S}_{\text{trust}}$ with $k_{\text{trust}} = \Gamma(\mathscr{S})$ implies $\mathcal{S} \in \mathscr{S}_{\text{sit-aware}}$ by (8). □

ACKNOWLEDGMENT

The authors thank Hasan Poonawala for his input in the formulation of the combinatorial optimization problem (29). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. Fig. 1 is licensed from the Department of Energy and Climate Change under Creative Commons Attribution-NoDerivs 2.0 Generic (CC BY-ND 2.0). This photo appeared under the title "Energy Minister Michael Fallon visits the National Grid Control Centre in Wokingham" at https://www.flickr.com/photos/deccgovuk/8725424647/in/photostream/.

REFERENCES

[1] M. R. Endsley, "Toward a theory of situation awareness in dynamic systems," *Hum. Factors, J. Hum. Factors Ergonom. Soc.*, vol. 37, no. 1, pp. 32–64, Mar. 1995.

[2] T. Sheridan, *Telerobotics, Automation, and Human Supervisory Control*. Cambridge, MA, USA: MIT Press, 1992.

[3] A. Dix, *Formal Methods for Interactive Systems*. New York, NY, USA: Academic, 1991.

[4] M. Panteli, P. A. Crossley, D. S. Kirschen, and D. J. Sobajic, "Assessing the impact of insufficient situation awareness on power system operation," *IEEE Trans. Power Syst.*, vol. 28, no. 3, pp. 2967–2977, Aug. 2013.

[5] M. R. Endsley and E. S. Connors, "Situation awareness: State of the art," in *Proc. IEEE Power Energy Soc. Gen. Meeting Convers. Del. Electr. Energy 21st Century*, Jul. 2008, pp. 1–4.

[6] A. Steinfeld, "Interface lessons for fully and semi-autonomous mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom. ICRA*, Apr. 2004, pp. 2752–2757.

[7] A. Pritchett and M. Feary, "Designing human-automation interaction," in *Handbook of Human-Machine Interaction*. Farnham, U.K.: Ashgate Publishing, 2011, pp. 267–282.

[8] C. Billings, *Aviation Automation: The Search for a Human-Centered Approach*. Trenton, NJ, USA: Erlbaum, 1997.

[9] A. Degani and M. Heymann, "Formal verification of human-automation interaction," *Hum. Factors, J. Hum. Factors Ergonom. Soc.*, vol. 44, no. 1, pp. 28–43, Mar. 2002.

[10] M. Vitus and C. Tomlin, "Sensor placement for improved robotic navigation," in *Robotics, Science and Systems VI*. Cambridge, MA, USA: MIT Press, 2010.

[11] A. I. Mourikis and S. I. Roumeliotis, "Optimal sensor scheduling for resource-constrained localization of mobile robot formations," *IEEE Trans. Robot.*, vol. 22, no. 5, pp. 917–931, Oct. 2006.

[12] H. Rowaihy *et al.*, "A survey of sensor selection schemes in wireless sensor networks," *Proc. SPIE*, vol. 6562, May 2007, Art. no. 65621A.

[13] J. Qi, K. Sun, and W. Kang, "Optimal PMU placement for power system dynamic state estimation by using empirical observability gramian," *IEEE Trans. Power Syst.*, vol. 30, no. 4, pp. 2041–2054, Jul. 2015.

[14] B. Gou, "Generalized integer linear programming formulation for optimal PMU placement," *IEEE Trans. Power Syst.*, vol. 23, no. 3, pp. 1099–1104, Aug. 2008.

[15] S. Joshi and S. Boyd, "Sensor selection via convex optimization," *IEEE Trans. Signal Process.*, vol. 57, no. 2, pp. 451–462, Feb. 2009.

[16] B. Polyak, M. Khlebnikov, and P. Shcherbakov, "An LMI approach to structured sparse feedback design in linear control systems," in *Proc. Eur. Control Conf. (ECC)*, Jul. 2013, pp. 833–838.

[17] A. Krause and C. Guestrin, "Near-optimal observation selection using submodular functions," in *Proc. AAAI*, vol. 7, 2007, pp. 1650–1654.

[18] M. Shamaiah, S. Banerjee, and H. Vikalo, "Greedy sensor selection: Leveraging submodularity," in *Proc. 49th IEEE Conf. Decis. Control (CDC)*, Dec. 2010, pp. 2572–2577.

[19] T. Summers, F. L. Cortesi, and J. Lygeros, "On submodularity and controllability in complex dynamical networks," *IEEE Trans. Control Netw. Syst.*, vol. 3, no. 1, pp. 91–101, Mar. 2016.

[20] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions—I," *Math. Program.*, vol. 14, no. 1, pp. 265–294, Dec. 1978.

[21] A. Clark, B. Alomair, L. Bushnell, and R. Poovendran, "Submodularity in input node selection for networked linear systems: Efficient algorithms for performance and controllability," *IEEE Control Syst.*, vol. 37, no. 6, pp. 52–74, Dec. 2017.

[22] L. A. Wolsey, "An analysis of the greedy algorithm for the submodular set covering problem," *Combinatorica*, vol. 2, no. 4, pp. 385–393, Dec. 1982.

[23] T. Y. Berger-Wolf, W. E. Hart, and J. Saia, "Discrete sensor placement problems in distribution networks," *Math. Comput. Model.*, vol. 42, no. 13, pp. 1385–1396, Dec. 2005.

[24] G. Gelman, K. M. Feigh, and J. Rushby, "Example of a complementary use of model checking and human performance simulation," *IEEE Trans. Human-Mach. Syst.*, vol. 44, no. 5, pp. 576–590, Oct. 2014.

[25] M. L. Bolton, E. J. Bass, and R. I. Siminiceanu, "Using formal verification to evaluate human-automation interaction: A review," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 43, no. 3, pp. 488–503, May 2013.

[26] M. Oishi, I. Mitchell, A. M. Bayen, and C. J. Tomlin, "Invariance-preserving abstractions of hybrid systems: Application to user interface design," *IEEE Trans. Control Syst. Technol.*, vol. 16, no. 2, pp. 229–244, Mar. 2008.

[27] T. Rezvani, K. Driggs-Campbell, D. Sadigh, S. S. Sastry, S. A. Seshia, and R. Bajcsy, "Towards trustworthy automation: User interfaces that convey internal and external awareness," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2016, pp. 682–688.

[28] L. Jiang and Y. Wang, "A human-computer interface design for quantitative measure of regret theory," *IFAC-PapersOnLine*, vol. 51, no. 34, pp. 15–20, 2019.

[29] K. Akash, K. Polson, T. Reid, and N. Jain, "Improving human-machine collaboration through transparency-based feedback—Part I: Human trust and workload model," *IFAC-PapersOnLine*, vol. 51, no. 34, pp. 315–321, 2019.

[30] K. Akash, T. Reid, and N. Jain, "Improving human-machine collaboration through transparency-based feedback—Part II: Control design and synthesis," *IFAC-PapersOnLine*, vol. 51, no. 34, pp. 322–328, 2019.

[31] J. Baillieul, N. E. Leonard, and K. A. Morgansen, "Interaction dynamics: The interface of humans and smart machines [scanning the issue]," *Proc. IEEE*, vol. 100, no. 3, pp. 567–570, Mar. 2012.

[32] K. Fitzsimons, E. Tzorakoleftherakis, and T. Murphey, "Optimal human-in-the-loop interfaces based on Maxwell's Demon," in *Proc. Amer. Control Conf.*, Jul. 2016, pp. 4397–4402.

[33] B. Sadrfaridpour, H. Saeidi, J. Burke, K. Madathil, and Y. Wang, "Modeling and control of trust in human-robot collaborative manufacturing," in *Robust Intelligence and Trust in Autonomous System*. Boston, MA, USA: Springer, 2016, pp. 115–141.

[34] H. Saeidi, J. R. Wagner, and Y. Wang, "A mixed-initiative haptic teleoperation strategy for mobile robotic systems based on bidirectional computational trust analysis," *IEEE Trans. Robot.*, vol. 33, no. 6, pp. 1500–1507, Dec. 2017.

[35] N. Eskandari and M. Oishi, "Computing observable and predictable subspaces to evaluate user-interfaces of LTI systems under shared control," in *Proc. IEEE Conf. Syst., Man Cybern.*, Oct. 2011, pp. 2803–2808.

[36] M. M. K. Oishi, "Assessing information availability for user-interfaces of shared control systems under reference tracking," in *Proc. Amer. Control Conf.*, Jun. 2014, pp. 3474–3481.

[37] T. M. Hammond, N. Eskandari, and M. M. K. Oishi, "Observability of user-interfaces for hybrid LTI systems under collaborative control: Application to aircraft flight management systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 1, pp. 78–84, Jan. 2016.

[38] A. P. Vinod, T. H. Summers, and M. M. K. Oishi, "User-interface design for MIMO LTI human-automation systems through sensor placement," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2016, pp. 5276–5283.

[39] S. Friedberg, A. Insel, and L. Spence, *Linear Algebra*, 4th ed. London, U.K.: Pearson, 2014.

[40] C. Baier and J.-P. Katoen, *Principles of Model Checking*. Cambridge, MA, USA: MIT Press, 2008.

[41] N. Eskandari, G. A. Dumont, and Z. J. Wang, "An observer/predictor-based model of the user for attaining situation awareness," *IEEE Trans. Human-Mach. Syst.*, vol. 46, no. 2, pp. 279–290, Apr. 2016.

[42] C. D. Wickens, "Situation awareness: Review of mica Endsley's 1995 articles on situation awareness theory and measurement," *Hum. Factors, J. Hum. Factors Ergonom. Soc.*, vol. 50, no. 3, pp. 397–403, Jun. 2008.

[43] A. Oulasvirta, N. R. Dayama, M. Shiripour, M. John, and A. Karrenbauer, "Combinatorial optimization of graphical user interface designs," *Proc. IEEE*, vol. 108, no. 3, pp. 434–464, Mar. 2020.

[44] K. A. Hoff and M. Bashir, "Trust in automation: Integrating empirical evidence on factors that influence trust," *Hum. Factors, J. Hum. Factors Ergonom. Soc.*, vol. 57, no. 3, pp. 407–434, May 2015.

[45] W.-L. Hu, K. Akash, T. Reid, and N. Jain, "Computational modeling of the dynamics of human trust during human–machine interactions," *IEEE Trans. Human-Machine Syst.*, vol. 49, no. 6, pp. 485–497, Dec. 2019.

[46] R. Parasuraman and V. Riley, "Humans and automation: Use, misuse, disuse, abuse," *Hum. Factors, J. Hum. Factors Ergonom. Soc.*, vol. 39, no. 2, pp. 230–253, Jun. 1997.

[47] J. D. Lee and K. A. See, "Trust in automation: Designing for appropriate reliance," *Hum. Factors, J. Hum. Factors Ergonom. Soc.*, vol. 46, no. 1, pp. 50–80, 2004.

[48] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "Situation awareness, mental workload, and trust in automation: Viable, empirically supported cognitive engineering constructs," *J. Cognit. Eng. Decis. Making*, vol. 2, no. 2, pp. 140–160, Jun. 2008.

[49] M. Lewis, K. Sycara, and P. Walker, *The Role of Trust in Human-Robot Interaction*. Cham, Switzerland: Springer, 2018, pp. 135–159.

[50] P. Goillau, C. Kelly, M. Boardman, and E. Jeannot, "Guidelines for trust in future ATM systems: Measures," in *Eurocontrol, the European Organisation for the Safety of Air Navigation*. Brussels, Belgium: EUROCONTROL, 2003.

[51] M. Madsen and S. Gregor, "Measuring human-computer trust," in *Proc. 11th Australas. Conf. Inf. Syst.*, vol. 53. Brisbane, QLD, Australia: Citeseer, 2000, pp. 6–8.

[52] J.-Y. Jian, A. M. Bisantz, and C. G. Drury, "Foundations for an empirically determined scale of trust in automated systems," *Int. J. Cognit. Ergonom.*, vol. 4, no. 1, pp. 53–71, Mar. 2000.

[53] T. Setter, A. Gasparri, and M. Egerstedt, "Trust-based interactions in teams of mobile agents," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2016, pp. 6158–6163.

[54] S. Fujishige, *Submodular Functions and Optimization*, vol. 58. Amsterdam, The Netherlands: Elsevier, 2005.

[55] L. Lovász, "Submodular functions and convexity," in *Mathematical Programming The State of the Art*. Berlin, Germany: Springer, 1983, pp. 235–257.

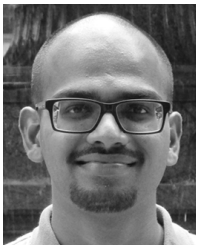
[56] A. Krause and D. Golovin, "Submodular function maximization," in *Tractability: Practical Approaches to Hard Problems*. Cambridge, U.K.: Cambridge Univ. Press, 2014, pp. 71–104.

[57] F. Rossi, P. Van Beek, and T. Walsh, *Handbook of Constraint Programming*. Amsterdam, The Netherlands: Elsevier, 2006.

[58] *IEEE-118 Power Network*. Accessed: Feb. 4, 2021. [Online]. Available: http://www.ee.washington.edu/research/pstca/pf118/pg_tca118bus.htm

[59] D. Van Hertem, "Usefulness of DC power flow for active power flow analysis with flow controlling devices," in *Proc. 8th IEE Int. Conf. AC DC Power Transmiss. (ACDC)*, 2006, pp. 58–62.

[60] J. Machowski, J. Bialek, and J. Bumby, *Power System Dynamics: Stability and Control*. Hoboken, NJ, USA: Wiley, 2011.
[61] A. Bergen and V. Vittal, *Power System Analysis*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1999.
[62] F. Dorfler and F. Bullo, "Kron reduction of graphs with applications to electrical networks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 1, pp. 150–163, Jan. 2013.
[63] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas, "MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Trans. Power Syst.*, vol. 26, no. 1, pp. 12–19, Feb. 2011.
[64] P. Demetriou, M. Asprou, J. Quiros-Tortos, and E. Kyriakides, "Dynamic IEEE test systems for transient analysis," *IEEE Syst. J.*, vol. 11, no. 4, pp. 2108–2117, Dec. 2017.
[65] U. Feige, "A threshold of ln n for approximating set cover," *J. ACM*, vol. 45, no. 4, pp. 634–652, Jul. 1998.



Abraham P. Vinod (Member, IEEE) received the B.Tech. and the M.Tech. degree in electrical engineering from IIT Madras (IITM), Chennai, India, both in 2014, and the Ph.D. degree in electrical engineering from The University of New Mexico, Albuquerque, NM, USA, in 2018.

His research interests are in the areas of optimization, stochastic control, and learning.

Dr. Vinod received the Best Student Paper Award at the 2017 ACM Hybrid Systems: Computation and Control Conference and the IITM Prof. Achim Bopp Prize.

Adam J. Thorpe (Graduate Student Member, IEEE) received the B.S.E.E. and M.S. degrees in electrical engineering from The University of New Mexico, Albuquerque, NM, USA, in 2017 and 2019, respectively, with a focus in systems and controls. He is currently pursuing the Ph.D. degree with The University of New Mexico.

His research interests are in the areas of stochastic optimal control, autonomous systems, and statistical learning, with an emphasis on data-driven control techniques.

Philip A. Olaniyi received the B.Eng. degree in electrical engineering from the University of Ilorin, Ilorin, Nigeria, in 2010, and the M.S. degree in electrical engineering from The University of New Mexico, Albuquerque, NM, USA, in 2017, with a research focus on the control of power systems.

He joined Intel Corporation, Beaverton, OR, USA, in 2017, where he currently works as a Facility Technology Development Electrical Engineer.

Tyler H. Summers received the B.S. degree in mechanical engineering from Texas Christian University, Fort Worth, TX, USA, in 2004, and the M.S. and Ph.D. degrees in aerospace engineering from The University of Texas at Austin, Austin, TX, USA, in 2007 and 2010, respectively, with emphasis on feedback control theory.

He was a Fulbright Postgraduate Scholar at the Australian National University, Canberra, ACT, Australia, from 2007 to 2008. He is currently an Assistant Professor of mechanical engineering with an affiliate appointment in electrical engineering with The University of Texas at Dallas, Richardson, TX, USA. Prior to joining UT Dallas, he was an ETH Post-Doctoral Fellow with the Automatic Control Laboratory, ETH Zürich, Zürich, Switzerland, from 2011 to 2015. His research interests are in control, optimization, and learning in complex dynamical networks, with applications to electric power networks and distributed robotics.

Meeko M. K. Oishi (Senior Member, IEEE) received the B.S.E. degree in mechanical engineering from Princeton University, Princeton, NJ, USA, in 1998, and the M.S. and Ph.D. degrees in mechanical engineering (Ph.D. minor, electrical engineering) from Stanford University, Stanford, CA, USA, in 2000 and 2004, respectively.

She was a Visiting Researcher with the AFRL Space Vehicles Directorate and a Science and Technology Policy Fellow with the National Academies, Washington, DC, USA. She previously held a faculty position at The University of British Columbia, Vancouver, BC, USA, and post-doctoral positions at Sandia National Laboratories, Albuquerque, and the National Ecological Observatory Network, Boulder, CO, USA. She is currently a Professor of electrical and computer engineering at The University of New Mexico, Albuquerque, NM, USA. Her research interests include hybrid dynamical systems, human-in-the-loop control, stochastic optimal control, and autonomous systems.

Dr. Oishi was a recipient of the UNM Regents' Lectureship, the NSF CAREER Award, the UNM Teaching Fellowship, the Peter Wall Institute Early Career Scholar Award, the Truman Postdoctoral Fellowship in National Security Science and Engineering, and the George Bienkowski Memorial Prize, Princeton University.