Contents lists available at ScienceDirect

Internet of Things

journal homepage: www.elsevier.com/locate/iot

# Pareto optimal design space exploration of cyber-physical systems

Maral Amir, Tony Givargis[*]

*Department of Computer Science, University of California, Irvine*

A R T I C L E   I N F O

*Article history:*
Received 6 May 2020
Revised 9 October 2020
Accepted 11 October 2020
Available online 15 October 2020

*Keywords:*
Cyber-physical system
Embedded systems
Design space exploration
Pareto-optimal design
Parameterized system design

A B S T R A C T

Cyber-physical systems (CPS) integrate a variety of engineering areas such as control, mechanical, and computer engineering in a holistic design effort. While interdependencies between the different disciplines are key attributes of CPS design science, little is known about the impact of design decisions of the cyber part on the overall performance qualities of the system. To investigate these dependencies, this paper proposes a simulation-based Design Space Exploration (DSE) framework that considers detailed cyber system parameters such as cache size, bus width, and voltage levels in addition to physical and control parameters of the CPS. We propose a DSE algorithm that explores the parameter configurations of the cyber-physical sub-system in order to approximate the Pareto-optimal design points with respect to design objectives such as energy consumption and control stability. For validation, we have successfully applied the proposed framework to an inverted-pendulum application. Here, our holistic evaluation of the Pareto-optimal points reveals the presence of non-trivial trade-offs that are imposed by the control, physical, and detailed cyber parameters. For instance, the identified energy and control optimal design points comprise configurations with a wide range of CPU speeds, sampling rates, and cache configurations following non-trivial zigzag patterns. The proposed framework could identify and manage these trade-offs and, as a result, is an imperative first step to automate the search for superior cyber-physical system configurations.

## 1. Introduction

Cyber-physical systems (CPS) integrate various engineering areas such as control-, computer-, mechanical-, and network engineering [7]. The complex and heterogeneous design aspects of CPS beget methodologies to combine the corresponding disciplines. For example, in automotive industry, it has been investigated that 80% of the innovations in the design of a car are attributed to the computer systems [13].

Sequential and model-based design methodologies [3,8] are well-established techniques to cope with the complexity of designing CPS. The idea is, first, to select a promising physical system, defining the controller, and finally addressing design challenges of the embedded computer system. Such sequential separation of decisions reduces the complexity of the design

[*] Corresponding author.
*E-mail addresses:* mamir@uci.edu (M. Amir), givargis@uci.edu (T. Givargis).

https://doi.org/10.1016/j.iot.2020.100308

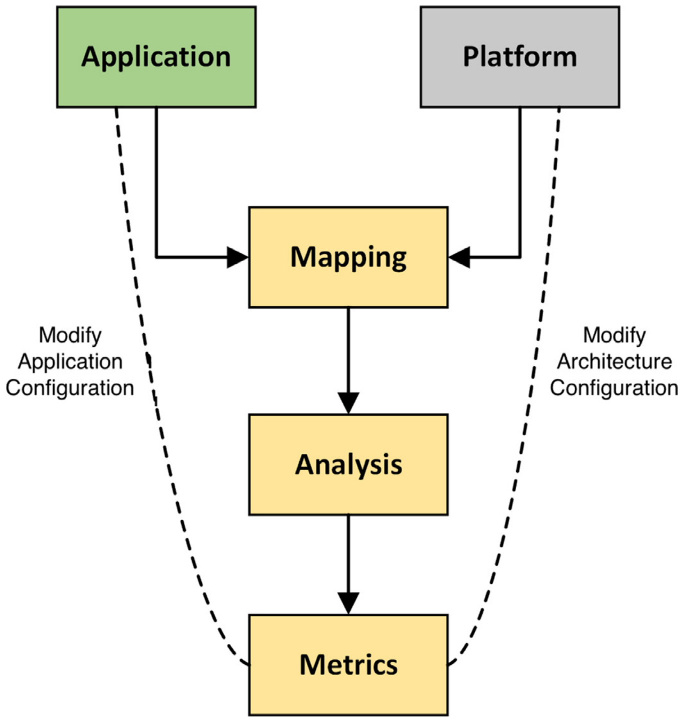

**Fig. 1.** Y-chart methodology models the design representation from a top-down view. The behavioral, structural and layout views are each a branch of the 'Y'.

effort. However, like most greedy approaches, the ultimate product is unlikely to be the best possible design due to missed trade-offs between cyber and physical design knobs.

Recent work shows that holistic design approaches result in superior products and systems when compared to sequential design flows, where the cyber-part is designed independently from the physical-part in a CPS. In this work, the term holistic implies that the physical, the control, and the cyber attributes of a system are evaluated concurrently. Consequently, a holistic optimization of the design space can yield ideal performance qualities such as energy consumption or control stability [1,2].

The large design space, stringent performance objectives, and tight time-to-market requirements create a challenge in designing CPS systems. Typically, Design Space Exploration (DSE) is performed in early development phases to select between multiple high-level design configurations that meet the overall system design requirements. The power to automatically surf the design space and explore viable solution candidates is an essential requirement for many engineering tasks, including systems integration and optimization.

In this work, we explicitly investigate if and how specific properties of the cyber system such as cache size, CPU configuration, memory hierarchy, and peripherals interfere with the overall system performance metrics such as precision, stability, and power consumption. Similar dependencies have been discovered and utilized between physical properties and control parameters, which led to the controller pruning in [2]. Ultimately, the question to be asked is if a connection exist between physical system (PS) and cyber system (CS) in terms of the overall system performance. We answer this question in this paper.

Model-based design provides a flexible environment to identify high-level attributes of the subsystems. Hence, model-based design can be used to specify and analyze the system requirements from a semantic-oriented level rather than an implementation-oriented level [22]. If dependable models from divers disciplines are assembled at the high level without prior heterogeneous analysis and verification, system failures are likely to emerge. Our holistic model of a CPS provides a framework to integrate physical entities, environment settings, and computation resources into a single optimized end product. That is, a codependent methodology is proposed that captures the properties of the system, not only at the subsystem level, but also from a global interoperable point of view.

The proposed framework implements a DSE methodology based on the Y-chart [5], as depicted in Fig. 1. The Y-chart establishes clear distinction between the application and the architecture of the system that is being designed. This methodology follows a top-down flow in the domains of behavior such as structure and layout. A Y-chart based framework enables the exploration and analysis of the system configurations for further alteration in architecture, application settings, and mapping strategies.

Without loss of generality, we have used Simulink as our primary simulation and functional verification tool in developing the networked control application used alongside our proposed approach [25]. Alternatively, any of Scilab [24] or Octave [19] may be used instead of Simulink. In terms of the physical platform, we have used a parameterized System-on-a-Chip (SoC) called Platune [6]. Platune is specifically designed in a tunable way, having settings for processor caches, peripheral configurations, voltage scaling, and so on and so forth. More specifically, this tool facilitates:

- A holistic and comprehensive exploration and analysis of the system design space and its parameters interaction and correlation;
- A compression of the design space into Pareto-optimal design points with regard to system performance metrics; and
- A derivation of a dependency analysis to reduce the search and support tool-based DSE of CPS.

It should be clarified that, while the DSE engine of Platune is universal, the default simulation engine of Platune is based on a fictitious SoC. Given a specific CPS implementation, the SoC (i.e., cyber) model of the Platune would need to be replaced with an appropriate target simulator.

CPS are unique in the way they incorporate interactions between the computing and the physical entities. Therefore, the design foundations, methods, and tools of CPS engineering should accommodate the interdependency between the physical (e.g., mechanical, electrical, etc.) design, the computing hardware (e.g., processors, memories, etc.), and system software (e.g., algorithms for control, application logic, etc.).

Our key contributions in this work are summarized below:

1. We propose a holistic and interdependent framework for the design of a CPS that optimizes the cyber, physical, and control subsystems concurrently;
2. We present a simulation-based framework that integrates the Simulink environment with the Platune SoC for system analysis, testing, and verification; and
3. We present an efficient and concurrent algorithm to prune and explore the design space for CPS, per our holistic design platform.

The proposed framework is applied to a real application of control system, namely an automated inverted pendulum, to analyze and verify the efficiency and necessity of the suggested work.

The rest of this paper is organized as follows. A review of the state-of-the-art DSE frameworks for networked control systems is reported in Section 2. Sections 3 outlines the DSE problem and describes a tool-supported holistic methodology. We demonstrate the workings and effectiveness of our framework for the inverted pendulum example in Section 4. We provide some additional remarks and insight into the proposed methodology in Section 5. Finally, the conclusions and future work directions are summarized in Section 6.

## 2. Related work

DSE techniques evaluate the design points either using a simulation-based approach or an analytical methodology. Tuffin et al. [26] compares these two approaches in order to assist the system designers in choosing an appropriate final configuration. The analytical methodology applies several restrictions and assumptions such as the Markov property [23] to the application model. This class of methodologies which rely on the predictable architectures are appropriate for time critical and safety critical applications. Simulation-based techniques are generally used when the aforementioned assumptions made by the analytical models are inappropriate. Also, simulation approaches are typically needed where the numerical analysis of the system model exceeds the time and space complexity of the development platform. Current state-of-the-art techniques arbitrarily apply either simulation or analytical methodologies during the system design phases. Depending on the application, hybrid approaches that integrate approximate models from the analytical methodology with simulation-based frameworks can be very helpful in reducing the design space complexity.

Holistic and model-based design approaches for CPS have been an active research topic resulting in a range of compelling related works. Canedo et al. [3] presented a top-down design framework which applies reusable blueprints of physical and cyber models to synthesize efficient CPS. Similarly, Vatanparvar et al. [28,29] have benefited from a top-down system-level design approach for CPS in the automotive domain. These works have extracted parameters from the physical systems and modeled their behavior for better optimization of the controller in cyber subsystem. Likewise, they have modeled and estimated the dynamic behavior of the electric vehicle components, e.g. power train, hybrid electrical energy storage, and automotive climate control, in order to improve the performance of the vehicle in terms of driving range and energy consumption. However, we are suggesting that exploration and analysis of the cyber parameters in the controller design of these components may impact the system performance and stability, hence should be considered during the physical subsystem design phase.

Neema et al. [17] proposed a framework for CPS DSE in which the designers can define physical and computational components and include constraints on system parameters and assembly processes. Their work outlines the design space as a set of hierarchical AND-OR elements with many Boolean constrains included in the design decisions. Furthermore, feasible configurations are simulated to analyze the changes of variables-of-interest across different design alternatives. Similarly, Aminifar et al. [1] have developed an automatic adaptation of software components in CPS. While these works provide a good motivation and foundation for the research presented in this paper, they do not specifically address the co-optimization strategy and rather define schemes to capture all the design configurations from the cyber as well as the physical subsystems.

In the work presented by Maasoumy et al. [14], the authors have focused on the control and cyber co-design with automatic control selection and parameterization. However, the presented methodology does not consider parameterization of the physical subsystem and does not evaluate the impact of architectural design decisions on the cyber subsystem. In a similar manner, Muhleis et al. [16] present a simulation-based framework in Jitterbug to analyze the control quality during the DSE phase. The work of these authors is aligned well with our work in terms of including the control quality in the CPS design decisions. However, these works do not consider the mapping process and the computing platform design alternatives for the respective controller applications.

Larsen et al. [11] have introduced a collaborative modeling and simulation tool to design embedded control systems and alongside the physical plant dynamics. Their tool, named Crescendo, incorporates a combination of Discrete-Event (DE) controller models with the Continuous-Time (CT) models to allow multidisciplinary system designs. They suggest techniques to reduce the number of simulations for rapid DSE applications. While relevant to CPS, this work is very specific to DE/CT optimization and does not address physical subsystems such as processors and memories.

Buini et al. [2] have outlined the impact of the physical design decisions in a holistic CPS DSE. However, in their work the cyber subsystem was represented solely by a single parameter, namely the sampling rate. In a similar vein, the automatic adaptation of cyber components is investigated by Otto et al. [21] in the form of an iteration-based exploration for preferable software parameter configuration, under consideration of product and raw material descriptions. In a more practical setting, Michniewicz et al. [15] have proposed a virtual representation of the robot cell, containing its individual physical and cyber components. In our work we combine physical and cyber parameters in one framework that is generic rather than specific to a sub-domain of CPS.

Nuzzo et al. [18] introduce a platform-based design methodology that addresses the complexity and heterogeneity of cyber-physical systems by using assume-guarantee contracts to formalize the design process and enable realization of control protocols in a hierarchical and compositional manner. Given the architecture of the physical plant to be controlled, the design is carried out as a sequence of refinement steps from an initial specification to a final implementation, including synthesis from requirements and mapping of higher-level functional and nonfunctional models into a set of candidate solutions built out of a library of components at the lower level. Initial top-level requirements are captured as contracts and expressed using linear temporal logic (LTL) and signal temporal logic (STL) formulas to enable requirement analysis and early detection of inconsistencies.

Zhang et al. [30] conducts extensive research on DSE with a very specific focus on Automotive Systems. Specifically, the authors analyze the current state of the art on DSE methods focusing on the assumed architecting process and concerns. They further investigate the state of practice in the automotive industry through a literature study and interviews with experienced system architects from five different automotive manufacturers.

Our work is further related to simulation frameworks [4,31] that facilitate collaborative modeling and simulation of embedded control systems and physical plants, incorporating discrete-event controller models with the continuous-time models to allow multidisciplinary systems design. However, our tool-supported methodology extends this idea to combine state-of-the art SoC exploration and model-based simulation tools in a single design methodology.

While DSE has been heavily researched, for example by Vanommeslaeghe et al. [27], we position our work as being one that specifically looks at combining Physical parameters (e.g., the length of a pendulum), Control parameters (e.g., the sample period for reading the system state and computing an output control value) and Cyber parameters (e.g., configuration of various caches, or CPU speed and frequency) in a holistic exploration environment. We explore the interplay between these parameters which are typically optimized in isolation and we propose an integrated DSE environment that leverages existing tools (e.g., Platune for exploring Cyber systems and Simulink for exploring Physical systems). We further propose an algorithm that copes with the DSE state explosion problem via constraint-based pruning.

## 3. The DSE methodology

Our proposed platform integrates Simulink (i.e., to model the networked control application) with Platune (i.e., a system-on-a-chip (SoC) framework) to design the CPS architecture. Further descriptions of the aforementioned tools are mentioned in the next section.

### 3.1. Tools and environments

In this section, we review the tools and technologies employed in the implementation of the proposed DSE framework.

#### 3.1.1. Simulink

The proposed methodology employs the Simulink [25] environment for model-based simulation and analysis. Our choice is mainly based on the rapid design and algorithm exploration capabilities of Simulink combined with its code generation capabilities (i.e. Simulink Coder). However, other simulation environments may be used in place of Simulink without the loss of generality [19,24]. The hierarchical design block representation of the Simulink modeling tool simplifies the design complexity and level/language transition [9]. Simulink applies a set of programs, called solvers, to simulate the system models. In Simulink, a model is represented as a set of ordinary differential equations. Depending on the nature of the system (e.g. continuous, discrete, time complexity, etc.) a solver is handpicked to apply a numerical method to the system model and compute its states at successive time steps over the simulation time.

#### 3.1.2. Platune

System-on-a-chip (SoC) platform manufacturers are increasingly adding configurable features that provide power and performance flexibility in order to increase a platform's applicability. Platune [6] is one such SoC platform simulator featuring detailed performance and power tuning capabilities. Platune is used to simulate an embedded application that is mapped onto the SoC platform and output performance and power metrics for any configuration of the SoC platform. Furthermore, Platune is used to automatically explore the large configuration space of such an SoC platform. The versatility, in terms of accuracy and speed of exploration, of Platune is demonstrated experimentally using a number of large benchmarks [6]. The power estimation techniques for processors, caches, memories, buses, and peripherals combined with the DSE algorithm deployed by Platune form a methodology for design of tuning frameworks for parameterized SoC platforms in general. Platune's parameters include a parametrized microprocessor, parameterized memory/caches, parameterized interconnect buses, and parameterized peripherals (e.g. DCT CODEC, UART, etc.).

### 3.2. DSE problem formulation

A DSE problem is one where the intent is to find the optimal combination of values for each of the many system design parameters (e.g., physical parameters, computing platform parameters, control system parameters, etc.). Exhaustive DSE algorithms search all possible combinations of the parameter space to find the optimal configurations. The Pareto-optimal design points, with regards to the design objectives (e.g., execution time or power consumption) and constraints (e.g., timing constraints) are of particular interest, as they represent the set of optimal tradeoffs that an engineer may select from. Of particular interest are DSE algorithms that search a multidimensional design space for systems with multiple parameters, objectives, and constraints. Such multidimensional searches are challenging simply due to the very large space that must be explored.

If the system's configuration is comprised of $N$ parameters, each having $P_i$ distinct values (i.e., configurable settings), the non-linear search space is defined to have $P_1 \times P_2 \ldots \times P_N$ design points. Note that the $\times$ symbol is a cartesian product operator in this context. We define this as the 'design space' of the system.

Large and complex systems may include billions of design alternatives to be explored in their design space. Manual and inefficient approaches to DSE are considered labor intensive, error prone, and time/space prohibitive. That is, an interrelated algorithm is needed not only to reduce the complex design space but also to provide interactive feedback to the designer during the search process. The power to automatically surf the design space and explore the solution candidates promotes DSE tools for many engineering tasks, including systems integration and optimization. This paper suggests a holistic methodology which combines the subsystem search spaces interactively and structures the global design space in an interactive process rather than a sequential approach.

#### 3.2.1. DSE in networked CPS systems

In embedded CPS design, the global design space is defined as an integration of local design space configurations, in that the local represents the subsystems such as cyber computing platform, physical systems, physical environments, and networked control systems.

We define the parameters of the target system as members of the global design space $S_{\text{CPS}}$. $S_{\text{CPS}}$ is defined as the cartesian product of the local design spaces as:

$$S_{\text{CPS}} = S_{\text{Cyber}} \times S_{\text{Physical}} \times S_{\text{Control}} \tag{1}$$

$S_{\text{CPS}}$ represents the set of global parameters that are available to the designer of the CPS. $S_{\text{Cyber}}$ represent the cyber subsystem parameters. $S_{\text{Physical}}$ represent the physical parameters of the CPS and $S_{\text{Control}}$ represent the control system parameters. Each of the spaces (i.e., $S_{\text{Cyber}}$, $S_{\text{Physical}}$, $S_{\text{Control}}$) is composed of $N$ parameters, called $P_i$, defined as follows:

$$P_i = (name, range) \tag{2}$$

Where *name* is the name or identifier of the parameter and *range* is a numeric interval of assignable values. Alternatively, the range may be a set of discrete values. The proposed methodology provides a holistic design space $S_{\text{CPS}}$, that evaluates multi discipline local design space modules in embedded CPS subsystems interactively. It introduces an abstract design space representation and provides a comprehensive interoperable tool-based search strategy. The methodology uses pruning of the design space and supports automated DSE activities in order to compute the Pareto-optimal design alternatives.

The proposed modular representation of the proposed DSE architecture is illustrated in Fig. 2. The methodology follows the Y-chart philosophy to integrate behavioral, structural, and layout abstraction levels using the application, platform, and mapping views respectively.

The local design spaces $S_{\text{Cyber}}$, $S_{\text{Physical}}$ and $S_{\text{Control}}$ are introduced individually in the following sections.

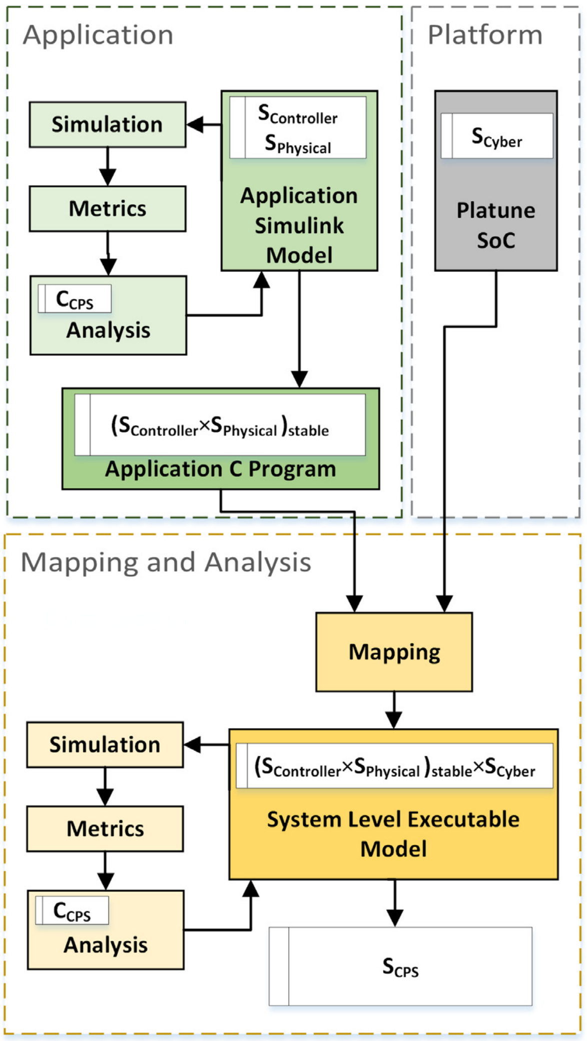

**Fig. 2.** The proposed DSE architecture integrates $S_{\mathbf{Cyber}}$, $S_{\mathbf{Physical}}$ and $S_{\mathbf{Control}}$ sub-spaces interactively using the branches of a Y-chart.

**The Physical and Control Design Spaces:** The proposed methodology employs Simulink as the model-based framework to surf the physical space $S_{\text{Physical}}$ and the networked control space $S_{\text{Control}}$ and optimize the overall system in order to meet a set of predefined performance criteria. The Simulink environment is one of the most popular tools among the researchers in various domains. Simulink provides customizable blocks from different domains, such as motors, sensors, actuators, and controllers that can be integrated into a single product, i.e., the CPS. This process employs a model as an executable system specification. The simulation results from the Simulink models in the $S_{\text{Physical}} \times S_{\text{Control}}$ design space configurations, produce the design objectives such as the controller performance (e.g., stability) and the computed physical energy consumption for the specified design configuration.

Control engineers are chasing after a control system design which demonstrates stable behavior while meeting the timing requirements accommodated by the processing platform. Naturally, computer engineers are obliged to design a computing platform that meets the timing requirements of the implemented product. For example, time delays or dead time (DTs) are ubiquitous in various system domains. Measurement, analysis, processing, and communication lags impose time delays on the control systems [12]. Networked control systems incorporate controllers, sensors, and actuator devices to perform several computational tasks. The control delay $\tau_c^k$, includes the computation delay induced by the computation and processing routines in addition to the communication delay which encompasses the sensor-to-controller and controller-to-actuator delays, as shown below:

$$\tau_k = \tau_c^k + \tau_{sc}^k + \tau_{ca}^k \tag{3}$$

Where $\tau_c^k$ is the computation delay and $\tau_{sc}^k$ and $\tau_{ca}^k$ represent sensor-to-controller and controller-to-actuator delays respectively. Embedded system applications that use microprocessors with bounded CPU performance must account for the processing delay, which can be significant and impactful in the design of the control subsystem. Computational delays can have substantial impact on the performance of a control system. The closed loop feedback of a control system produces unstable and oscillatory behavior if this delay is not compensated. That is, in CPS design, it is crucial to take into account the computational delay during simulation. The proposed framework captures the computational delay of the control system implemented on the processing platform and considers the delay compensation in the overall CPS design. Accordingly, we need a processing platform that implements the controller application and tracks the computation delay of the target control system. We consider a parametrized processing platform in our design framework to implement the controller application as a system-on-a-chip which will be described in the next section.

**The Cyber Design Space**: As mentioned in the Section 3.1.2, we employ Platune as an SoC in order to map the controller algorithm to a physical processor and calculate the cyber power consumption and associated computation delays. The platform architecture that represents the cyber design space includes a MIPS R3000 processor, data and instruction cache, buses, on chip memory, UART, and CODEC peripherals. Platune supports 26 parameters and each of these parameters can be assigned a value from a corresponding range of possibilities, i.e., $S_{\mathbf{Cyber}}$ could exceed over one hundred trillion design points.

Platune is designed to load C program applications, compile and link these applications with the runtime libraries and simulate the SoC at a cycle-accurate granularity. We convert the controller algorithm from the Simulink model into C code to be mapped on the Platune SoC with user defined specifications and parameters from the local space $S_{\mathbf{Cyber}}$. Platune carries a MIPS virtual machine to simulate the application software and generates a report on the power consumption and execution time for the specified configuration of the SoC platform.

**The CPS Design Space**: The integration of the local design spaced is carried out with regards to the interactions and interdependency between all the parameters $P_i$ and objective functions (e.g., time, power consumption, accuracy, etc.). That is, we need to perform a global analysis on the local design space parameters and the output metrics to prune the global design space accordingly and meet the systems requirements and design constrains.

For each set of possible configurations for some input parameter, there are a set of associated evaluation metrics. We consider two metrics that we believe would vary during different design space configurations in networked control CPS applications.

- **Energy Consumption:** One important requirement that is imposed on a CPS is low energy consumption. The total energy consumption $E_{\text{total}}$ of a system can be decomposed into the energy consumption of all the relevant subsystems. We accumulate the cyber energy consumption $E_{\text{Cyber}}$ of the target computing platform and the physical energy consumption $E_{\text{Physical}}$ of the physical model during the simulation run. Specifically, the cyber energy consumption is measured by Platnue on a cycle accurate level. The physical energy consumption of the system model is calculated in the Simulink for each of the respective simulation runs. The number of cycles in each design configuration are calculated by dividing the total simulation time by the respective sample time of simulation.

$$E_{\text{total}} = E_{\text{Cyber}} + E_{\text{Physical}} \tag{4}$$

- **Integral Square Error (ISR):** Integral square error (ISR) is a control quality measure that establishes the deviation from the desired output (expected value) of the CPS. This metric is applied in the control system applications that are intended to filter out large error values instantly. It integrates the square of the system error over the simulation time. The error values are computed as the difference between the desired output (set point) and the actual output of the CPS control systems. For example, the target output could be the angle of the pendulum in the inverted pendulum example.

$$ISR = \sum (ExpectedValue - ActualValue)^2 \tag{5}$$

### *3.3. The DSE algorithm*

Our proposed DSE methodology is presented as pseudocode in Algorithm 1. The algorithm integrates design spaces from various domains in several interactive pruning phases. During each pruning phase, we only discard design points that fail to meet a system constraint. Otherwise, these points are retained as a candidate until the ultimate Pareto optimal selection. This ensures that we take all interdependencies between parameters into account.

Our approach, (1) reduces the design space drastically in comparison with the exhaustive design space constructed by taking a cartesian product of all the parameters, and (2) takes into account the interoperability between systems parameters and objectives from different domains in the CPS system.

In this algorithm, first, the performance metric (e.g. ISR) from the Simulink model is evaluated for the respective design configurations and stable designs are selected from the ($C_{\text{Physical}} \times C_{\text{Control}}$) design spaces. Therefore, the first phase of pruning reduces the design space size from ($C_{\text{Physical}} \times C_{\text{Control}}$) to a set of viable points that meet the desired $ISR_{constraint}$, i.e., the stable set of design points. Next, the stable design points are passed on to the Platune platform for a full SoC simulation using all the available configurations of the SoC, i.e., the cyber parameters. For each simulated design point, we discard those that do not meet the timing requirements, i.e., $Time_{constraint}$. The resulting pruned set of design points are stored in $S_{CPS}$. Next, we sort this pruned space, i.e., $sorted\_list$. Finally, we compute a set of Pareto-optimal design points by discarding inferior points with regards to energy consumption obtaining the desired $S_{Pareto}$ set.

**Algorithm 1** Design Space Exploration Algorithm.

```
Require: $C_{\text{Physical}}, C_{\text{Control}}, ISR_{constraint}, Time_{constraint}$
Ensure: $S_{CPS}, S_{\text{Pareto}}, \text{min\_energy} \leftarrow INF$
 1: for all $s \in C_{\text{Physical}} \times C_{\text{Control}}$ do
 2:     $O_{PC} \leftarrow \text{Simulate\_Simulink}(s)$
 3:     if $O_{PC}.ISR < ISR_{constraint}$ then
 4:         $stable.push(s)$
 5: for all $s' \in Stable$ do
 6:     $temp \leftarrow \text{Simulate\_Platune}(s')$
 7:     if $temp.Time < Time_{constraint}$ then
 8:         $S_{CPS}.\text{push}(s')$
 9: $\text{sorted\_list} \leftarrow \text{Sort}(S_{CPS})$
10: for all $s' \in \text{sorted\_list}$ do
11:     if $s'.Energy \leq \text{min\_energy}$ then
12:         $\text{min\_energy} \leftarrow s'.Energy$
13:         $S_{\text{Pareto}}.\text{push}(s')$
```

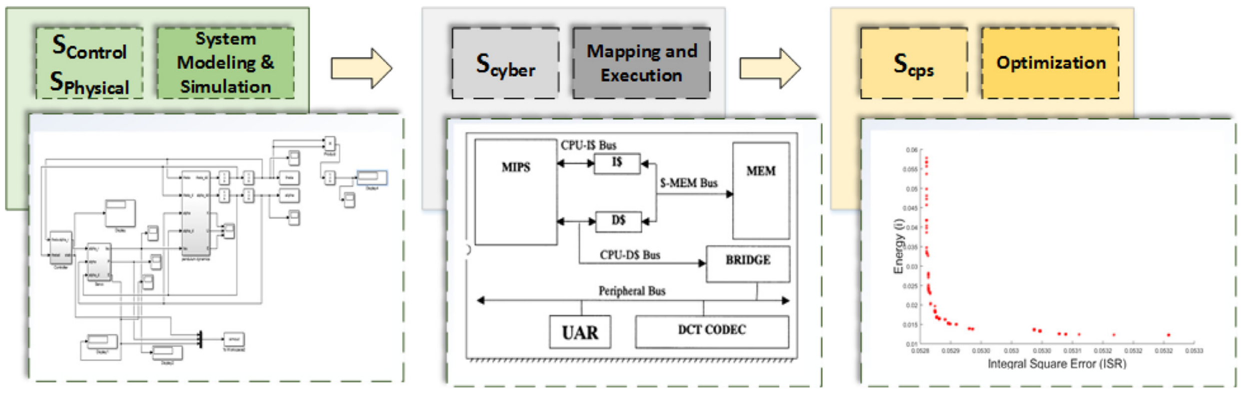

**Fig. 3.** The architectural view of the proposed DSE algorithm is illustrated in three steps. The first block performs system modeling and simulation using Simulink to produce a set of stable design alternatives. The second block performs application mapping on Platune, simulates and performs DSE with pruning. The third block is the optimization stage in which the Pareto optimal design alternatives with regards to energy and ISR objectives are calculated.

Ultimately, the problem is to find a set of parameter configurations for the CPS application in the $S_{Pareto}$ set that optimizes all design objectives, or what we call the multi-objective design optimization problem. Pareto-optimal curves that depict the tradeoff between the design objectives are solutions to multi-objective optimization problems. This design objective can be a vector of system responses that we are trying to maximize or minimize.

Our intention in this paper is to evaluate and analyze the CPS global design space and to demonstrate that the interaction and interdependency between local design space parameters is not trivial and intuitive and should be accounted for during the design process. That is, optimal embedded CPS design is in need of a holistic exploration in all the local design space configurations that inhabits the design constraints imposed by the interdependency between the spaces. Therefore, an exhaustive DSE exploration algorithm is afforded to find the Pareto-optimal points with regards to the design objective metrics. The pruning steps that integrate the local design spaces $C_{\text{Physical}}$, $C_{\text{Control}}$ and $C_{\text{Cyber}}$ into the global design space $C_{\text{CPS}}$ is presented in our algorithm and further illustrated in Fig. 3.

As a final remark and practical matter, the set of Pareto design candidates must include the points that fall within a neighborhood of the Pareto curve defined by the accuracy of the simulation and performance measurement metrics. Points falling within this Pareto band may ultimately prove to be desirable in the final implementation. Moreover, lower order design decisions that were not considered during the DSE could slightly change the location of a design point within the Pareto set, hence making it beneficial to not discard points that are close to the Pareto curve.

## 4. Experimental results

A DSE example using a real application of automated networked control system for a state space inverted pendulum model, with full state feedback controller, has been developed to evaluate the proposed methodologies of this work. A comprehensive description of the inverted pendulum control example is presented by Buini et al. [2] or by Krishna et al. [10]. In our work, an inverted pendulum has a closed loop control system consisting of a controller, actuator, and pendulum dynamics. The local design space parameters that we handpicked to vary for different design configurations are illustrated

**Table 1**
Design parameters in $S_{\text{Control}}$, $S_{\text{Physical}}$ and $S_{\text{Cyber}}$ design spaces.

| Space | $S_{\text{Control}}$ | $S_{\text{Physical}}$ | $S_{\text{Cyber}}$ |
|---|---|---|---|
| Parameters | $P_{\text{Sample Time}}$ | $P_{\text{Length}}$ | $\{P_{\text{i\$-size}}, P_{\text{i\$-line}}, P_{\text{i\$-associate}}, P_{\text{d\$-size}}, P_{\text{d\$-line}}, P_{\text{d\$-associte}}, P_{\text{CPU Speed}}\}$ |
| Tuples | $P_{\text{Sample Time}} = (S_{\text{Control}}$, Sample Time, [1e-4:28]) | $P_{\text{Length}} = (S_{\text{inverted pendulum}}$, Length, [1e-4:20]) | $P_{\text{i\$-size}}=(S_{\text{platform}}$, i\$-size, [128-32k]) <br> $P_{\text{i\$-line}}=(S_{\text{platform}}$, i\$-line, [4:64]) <br> $P_{\text{i\$-associate}}=(S_{\text{platform}}$, i\$-associate, [1:16]) <br> $P_{\text{d\$-size}}=(S_{\text{platform}}$, d\$-size, [128:32k]) <br> $P_{\text{d\$-line}}=(S_{\text{platform}}$, d\$-line, [4:64]) <br> $P_{\text{d\$-associte}}=(S_{\text{platform}}$, d\$-associate, [1:16]) <br> $P_{\text{CPU Speed}}=(S_{\text{platform}}$, CPU Speed, [4:256]) |

**Table 2**
Number of configurations in the design space.

| Design Space | # of Configurations |
|---|---|
| $S_{\text{Physical}} \times S_{\text{Control}}$ | 2,552 |
| $S_{\text{Cyber}}$ | 32,400 |
| $S_{\text{Physical}} \times S_{\text{Control}} \times S_{\text{Cyber}}$ | 82,648,800 |
| $(S_{\text{Physical}} \times S_{\text{Control}})_{\text{Stable}}$ | 655 |
| $(S_{\text{Physical}} \times S_{\text{Control}})_{\text{Stable}} \times S_{\text{Cyber}}$ | 113,975 |
| $S_{\text{CPS}}$ | 95,316 |

in Table 1. The parameters include the sampling period/time for our control systems, the inverted pendulum length for our physical system, and a number of hardware parameters for our Cyber system that include size, associativity, and line-length for each of the instruction and data caches as well as the CPU speed. To summarize, our physical system is the inverted pendulum. Our cyber system is an embedded processor executing the controller code.

The configurations in our experiments are integrated through different filters as described in Section 3. The number of configurations for the integrated design space, after each pruning phase, is depicted in Table 2 (note that entries that are labeled as stable, represent the design space after the pruning phase). As illustrated in the table, the design space for a holistic networked control inverted pendulum example is reduced compared to the sequentially integrated design space from 82 million design points down to approximately 95 thousand design points.

Our experiments were performed on a PC with an Intel Core i5 Quad processor running on a 64 bit Windows 7 operating system. First, the networked control system for the inverted pendulum example was modeled in the Simulink (i.e., MATLAB R2015a). A Dormand-Prince solver with variable step size was selected to perform the simulation of the model for each of the design configurations. Model simulation was iterated for 2252 sets of design alternatives in the $(C_{\text{Physical}} \times C_{\text{Control}})$ spaces with 30 s simulation time per iteration. The sample time of the controller as a control parameter and the length of the inverted pendulum as a physical parameter were selected to vary between different design points.

Our DSE algorithm selected 655 sets of configurations with stable behavior to be integrated with the computing platform design in Platune. The controller C source code was loaded into Platune to be mapped and simulated on the configured SoC. While the cyber design space $C_{\text{Cyber}}$ using Platune could exceed $10^{14}$ design points, for the inverted pendulum example, we limited this space to 32,400 design points by excluding parameters that had no impact on this benchmark (e.g., those associated with unused sub-systems of the SoC). Combining the 32,400 design points with those from the control space, we obtained 113,975 design alternatives, as highlighted in Table 2. The computing platform architecture altered for cache size and CPU speed parameter respectively. Finally, our DSE algorithm applied the final design constraint $C_{\text{CPS}}$ to satisfy the control and software engineers requirements. That is, the algorithm pruned the configurations in which the sampling time from the Simulink model was larger than computation delay obtained from Platune. The proposed pruning algorithm had the following advantages:

1. We reduced search space from 83 million design points to 95,316 design points prior to Pareto-optimal configuration selection.
2. We filtered the design space to meet the design decisions of control engineers and software engineers in an interactive manner.

The exploration algorithm to find the Pareto-optimal points were implemented to process the 95,316 configurations in the global design space of the networked control inverted pendulum application. As mentioned before, the objective metrics in our multi-objective optimization problem was the total energy consumption and the control quality ISR. Fig. 4 depicts the global design space points $C_{\text{CPS}}$ with gray dots and the Pareto-optimal curve as the result of the Pareto optimization algorithm with red dots. The curve represents the tradeoffs between the energy consumption and ISR values for all the design points in the global design space.

The tradeoff between the ISR and total energy consumption is more prominent for low power numbers in the range of 0.015–0.025 J. Tradeoff information is extremely important to identify the most preferred design point along the Pareto-

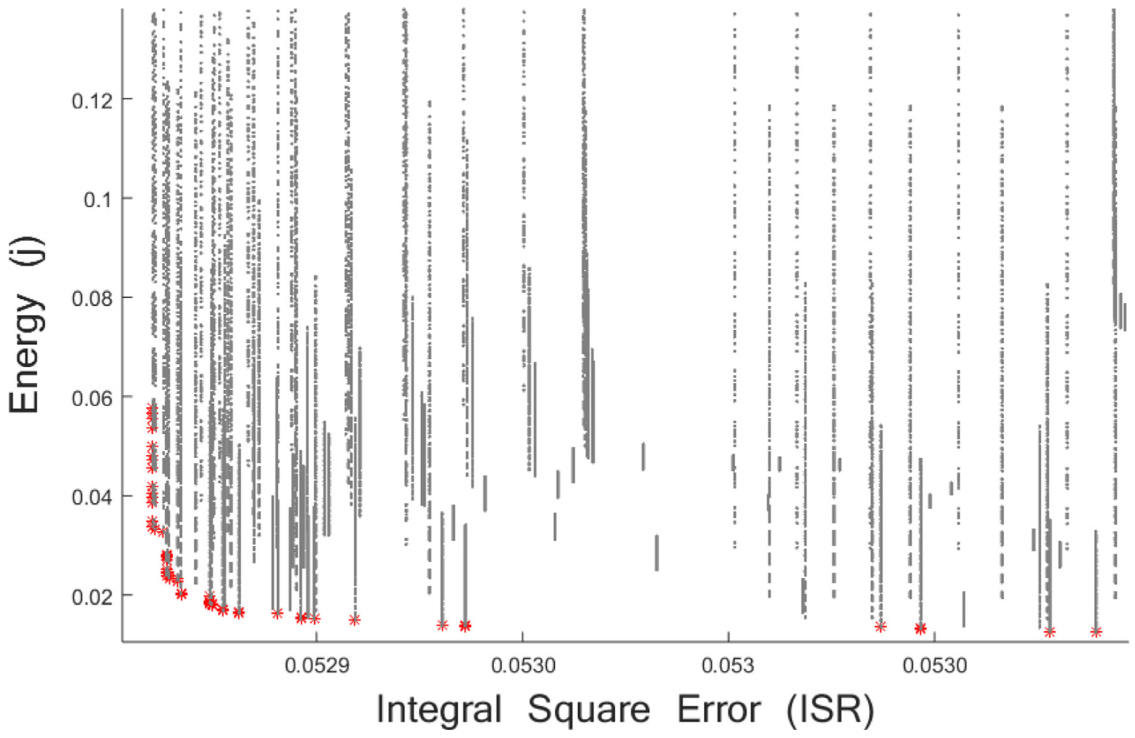

**Fig. 4.** Pareto-optimal points, the red stars, residing in the design space. The grey points, represents the trade-off between the energy and ISR.

curve. It is evident that the application requirements determine the superior Pareto-optimal point which satisfies the design objective accordingly. For example, robotic surgical procedures [20] afford more power consumption in tradeoff for the best accuracy and system reliability. On the other hand, embedded mobile applications are desperate for low power design decisions.

Fig. 5 illustrates the variations in cyber, physical, and controller parameters for energy consumption in the Pareto-optimal configurations. As depicted by the plots, the respective parameters do not follow a steady pattern in tradeoff with the energy consumption metric. The swinging behavior of the parameters before the energy consumption values in the Pareto-optimal configurations, accentuates the need for DSE methodologies and frameworks. A holistic and interactive evaluation on design alternatives for embedded CPS systems design is, hence, needed. For example, the non-intuitive interaction between the local design space parameter, the length of the pendulum, and the design objective metric, total energy consumption, can be obtained from the Length-Total Energy diagram. Plot (a) in the Fig. 5 discloses that increasing the total energy consumption measurements between the range of 0.018–0.021 J corresponds with an increase in the respective length values. On the other hand, the increase in the Pareto-optimal energy consumption measurements in the 0.021 to 0.024 range observes a decrease in the corresponding length values. This non-trivial correlation between the system level design parameters and metrics plays a significant role in design decisions. That is, we are claiming that embedded CPS designers should take into account not only the tradeoff between the design objectives to pick the most preferred Pareto-optimal points, but also the interdependency between the design space parameters (i.e., cyber parameters, physical parameters, control parameters, etc.) and the design metrics (i.e., ISR, Energy, Power, etc.). Accordingly, noticing the interplay between the design parameters and design metrics includes the hardware, inventory, and technology constraints in the design decisions. For example, one Pareto-optimal point may be preferred due to the inventory availability for the corresponding physical parameter. Consequently, the engineering of CPS inherently demands collaboration between diverse domains and a holistic computer-aided design framework.

Revisiting an earlier question around the dependencies in the CS, control, and PS parameters, we conducted six experiments. We set all the parameters of the system to their default (i.e., mid-point) settings and chose a single parameter from each of the CS, control, and PS space, namely pendulum length (PS), sampling period (control) and instruction cache size (CS). For each experiment, we determined a desired setting for one of the control, PS or CS sub-systems and searched for optimal settings for the remaining two parameters such that power consumption would be minimized. We followed up by making a change to the desired setting and repeating the experiment. For example, we arrived at the optimal setting for the sampling period of 500 milliseconds, and instruction cache line size of 40 bytes, for a desired pendulum length of 0.005 m, which yielded a power consumption of 0.025 J. Changing the desired pendulum length to 0.01 m and reevaluated the optimal settings for our two parameters, we observed that the optimal value for the sampling period changed to 750 milliseconds and the optimal value for the cache line size changed to 48 bytes. Evaluating the optimal values for each of

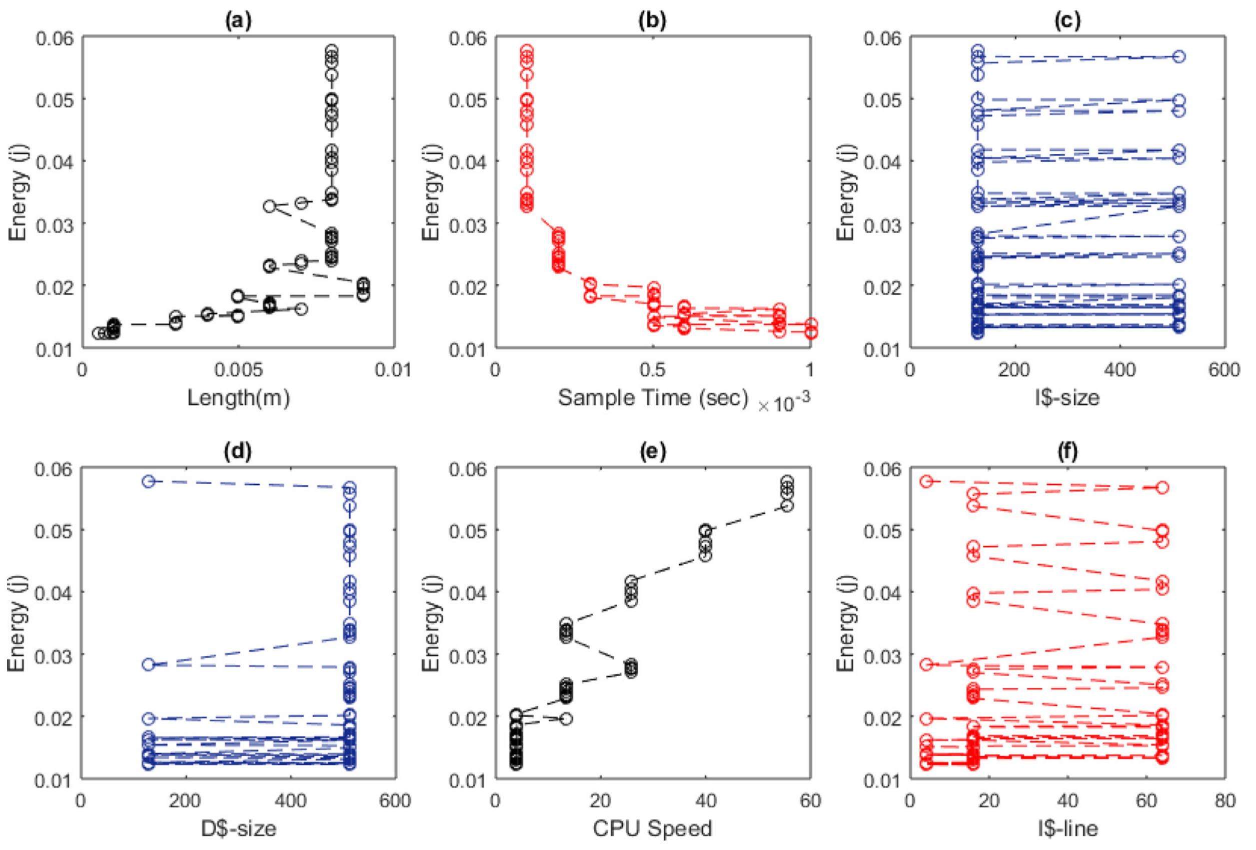

**Fig. 5.** The plots illustrate the behavior of design parameters for the inverted pendulum example with regards to the total Energy consumption. These plots include the parameters from the physical (a), control (b) and cyber (c, d, e, f) design spaces. The zigzag patterns of the plots represent a non linear behavior for successive design points. Specifically, we observed that as we swept a parameter's value from the minimum setting toward its maximum setting, we often measured extreme swings in energy consumption. Some of these variations may be be explained by the unpredictable nature of processor caches while others maybe as a result of simulation inaccuracy and sensitivity. Plot (f) depicts the relation between the instruction cache line and total energy values. It can be observed that for two successive Pareto-optimal design points with total Energy between (0.030 to 0.035), we observe an increase in the corresponding cache line values. On the other hand, in the (0.035–0.040) range, for two successive Pareto-optimal points we observe a decrease in the cache line values.

our three selected parameters across the six experiments, we could not find a single value that would be a priori optimal. The instruction cache line size varied from 20 to 46 bytes as control/PS parameters changed, the sampling period varied from 100 to 800 milliseconds as CS/PS parameters changed, and the pendulum length varied from 0.003 to 0.01 m as the control/CS parameters changed. Hence, per our experiments, and for a simple example, we established that a strong connection exist between these subdomains and the optimality of the overall system can not be guaranteed unless these are co-optimized in a holistic DSE approach.

It should be noted that we modeled the entire system energy consumption. Specifically, we used Platune to obtain the Cyber (i.e., computation) energy and combined that with an inverted pendulum analytical power model (i.e., one that included modeling the power supply, DC motor, and the analog drive circuit). We separately validated the efficiency of our Inverted Pendulum efficiency by comparing it to an actual implementation and found the results to be accurate to within 30%, which would be sufficient for high-level DSE.

For completeness, we should mention that as part of our experiments we pre-pruned the design space to eliminate design points that would overtly violate basic design principles or that they would clearly break various design constraints. This was part of the design space definition work that was conducted prior to the full DSE runs. For example, situations such as changing the length or the mass of the pendulum such that the forces required would become unfeasible for the DC motor or power supply to provide enough torque to keep the pendulum up would never be considered by our DSE. Despite this pre-pruning, in our benchmarking, the effective design space remained prohibitively large for brute force exploration, hence all our results are based on heuristic algorithms that are near-optimal.

## 5. Additional remarks

We would like to note that the DSE methodology stated in this paper is ultimately a brute force approach and may not be suitable in large applications having many parameter dimensions. The presented approach relies on successive pruning in order to lower the ultimate number of simulations and reduce the search time. Despite this, the time complexity of the system, in theoretical terms, remains exponential with respect to the number of Physical and Cyber parameters. However, in practice, this process can be managed by quantizing the parameter values or more aggressively pruning the space using additional performance constraints. This works if these additional constraints represent system requirements, but obviously would be counterproductive if they are added solely for the purpose of limiting the DSE. In our work, we aimed at studying the importance of a combined Physical and Cyber co-design.

While our presentation makes use of multiple individual tools (e.g., Platune and Simulink), we have an integrated end-to-end system that drives the entire process. We are currently looking at coping with the scalability issue by virtue of exploring Deep Reinforcement Learning based approaches combined with a Map-Reduce implementation that can utilize a higher number of servers in the cloud to solve large problems. To that effect, we plan to create a docker image of our DSE toolchain for fast and rapid deployment in Amazon Web Services (AWS).

## 6. Conclusion and future work

In this paper, a holistic and interactive DSE framework is presented for aid in the design of embedded CPS. We first discussed a simulation and exploration tool that combines the state-of-the-art exploration of SoC properties with model-based simulation tools like Simulink. As a result, we are able to analyze the impact of cyber design decisions, such as voltage, processor configurations, or cache sizes on the overall CPS performance and stability.

We employed the proposed framework in the design of a real application of automated networked control system to verify the efficiency and necessity of the suggested work. Our experimental results confirm that the interrelated framework is needed for design of efficient CPS systems due to non linear behavior of the Pareto-optimal design points. That is, successive design point on the Pareto-optimal curve do not follow a linear transfer between the design configurations. For instance, design space configuration parameters, e.g., a Pendulum's length, CPU speed, cache size, or control sampling rate follow a nonlinear (zigzag) behavior with regards to the Pareto-optimal design objectives, e.g., total energy consumption and control stability. Furthermore, we discovered analogous zigzag patterns for some parameters from different local design spaces (e.g., cyber, physical, or control). In our future work, we plan to apply methodologies to automatically mine the interdependency between the parameters from different design spaces in the CPS and employ the correlations to present heuristics for efficient DSE of relevant emerging applications.

## Declaration of Competing Interest

No conflict of interest exists.

## Acknowledgments

This work was supported in part by the National Science Foundation under NSF grant number 1563652.

## References

[1] A. Aminifar, P. Eles, Z. Peng, A. Cervin, Control-quality driven design of cyber-physical systems with robustness guarantees, in: Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013, pp. 1093–1098.
[2] H.-M. Buini, S. Peter, T. Givargis, Including variability of physical models into the design automation of cyber-physical systems, in: Design Automation Conference (DAC), 2015, pp. 1–6.
[3] A. Canedo, E. Schwarzenbach, Al-Faruque, M. Abdullah, Context-sensitive synthesis of executable functional models of cyber-physical systems, in: ACM/IEEE Conference on Cyber-Physical Systems (ICCPS), 2013, pp. 99–108.
[4] C. Gamble, K. Pierce, Design space exploration for embedded systems using co-simulation, in: Collaborative Design for Embedded Systems, Springer, 2014, pp. 199–222.
[5] A. Gerstlauer, C. Haubelt, A. Pimentel, T. Stefanov, D. Gajski, J. Teich, Electronic system-level synthesis methodologies, IEEE Trans. Comput. Aided Des. Integr. Circt. Syst. 28 (10) (2009) 1517–1530.
[6] T. Givargis, F. Vahid, Platune: a tuning framework for system-on-a-chip platforms, IEEE Trans. Comput. Aided Des. Integr. Circt. Syst. 21 (11) (2002) 1317–1327.
[7] V. Gunes, S. Peter, T. Givargis, F. Vahid, A survey on concepts, applications, and challenges in cyber-physical systems, KSII Trans. Internet Inf.Syst. (TIIS) 8 (12) (2014) 4242–4268.
[8] J. Jensen, D. Chang, E. Lee, A model-based design methodology for cyber-physical systems, in: Wireless Communications and Mobile Computing Conference (IWCMC), 2011, pp. 1666–1671.
[9] K. Kant, Computer-based industrial control, Prentice-Hall of India, 2004. ISBN-13: 978-8120339880
[10] C. Krishna, Mani and Koren, Israel. Adaptive fault-tolerance fault-tolerance for cyber-physical systems, in: International Conference on Computing, Networking and Communications (ICNC), 2013, pp. 310–314.
[11] P.G. Larsen, C. Gamble, K. Pierce, A. Ribeiro, K. Lausdahl, Support for Co-modelling and Co-simulation: the Crescendo Tool, Collaborative Design for Embedded Systems, pages 97–114, Springer, 2014.
[12] W.S. Levine, Control system fundamentals, CRC Press, 1999. ISBN-13: 978-0849300530
[13] T. Litman, Autonomous Vehicle Implementation Predictions: Implications for Transport Planning, Victoria Transport Policy Institute, 2014, pp. 36–4201.
[14] Q.-Z. Mehdi Maasoumy, C. Li, F. Meggers, A. Vincentelli, Co-design of control algorithm and embedded platform for building HVAC systems, in: IEEE International Conference on Cyber-Physical Systems (ICCPS), pages 61–70, IEEE, 2013.
[15] J. Michniewicz, G. Reinhart, Cyber-physical robotics–automated analysis, programming and configuration of robot cells based on cyber-physical-systems, Procedia Technol. 15 (2014) 566–575.
[16] N. Muhleis, M. Glass, L. Zhang, J. Teich, A co-simulation approach for control performance analysis during design space exploration of cyber-physical systems, ACM SIGBED Rev. 8 (2) (2011) 23–26.

[17] H. Neema, Z. Lattmann, P. Meijer, J. Klingler, S. Neema, T. Bapty, J. Sztipanovits, G. Karsai, Design space exploration and manipulation for cyber physical systems, in: IFIP Workshop on Design Space Exploration of Cyber-Physical Systems (IDEAL), 2014, pp. 1–8.
[18] P. Nuzzo, J.B. Finn, A. Iannopollo, A.L. Sangiovanni-Vincentelli, Contract-based design of control protocols for safety-critical cyber-physical systems, in: Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014, pp. 1–4.
[19] GNU octave. Www.gnu.org/software/octave/.
[20] T. Ortmaier, M. Gröger, D.H. Boehm, V. Falk, G. Hirzinger, Motion estimation in beating heart surgery, IEEE Trans. Biomed. Eng. 52 (10) (2005) 1729–1740.
[21] J. Otto, O. Niggemann, Automatic parameterization of automation software for plug-and-produce, in: The AAAI-15 Workshop on Algorithm Configuration (AlgoConf), 2015, pp. 1–6.
[22] F. Paterno, Model-based design and evaluation of interactive applications, Springer Science & Business Media, 2012. ISBN-13: 978-1852331559
[23] C. Rokitansky, C. Wietfeld, Markov chain analysis methods and simulation tools for performance evaluation and validation of vehicle-roadside communications, in: Personal, Indoor and Mobile Radio Communications, 1995. (PIMRC), 2, IEEE, 1995, pp. 846–852.
[24] ESI group, scilab. Www.scilab.org.
[25] Mathworks, simulink. Www.mathworks.com/products/simulink.html.
[26] B. Tuffin, P. Choudhary, C. Hirel, K. Trivedi, Simulation versus analytic-numeric methods: illustrative examples, in: Proceedings of the 2nd International Conference on Performance Evaluation Methodologies and Tools, ICST, 2007, pp. 1–10.
[27] Y. Vanommeslaeghe, J. Denil, J. De Viaene, D. Ceulemans, S. Derammelaere, Leveraging domain knowledge for the efficient design-space exploration of advanced cyber-physical systems, in: Euromicro Conference on Digital System Design (DSD), 2019, pp. 351–358.
[28] K. Vatanparvar, M.A. Al-Faruque, OTEM: optimized thermal and energy management for hybrid electrical energy storage in electric vehicles, in: Design, Automation & Test in Europe Conference (DATE), 2016, pp. 19–24.
[29] K. Vatanparvar, M.A. Al-Faruque, Battery lifetime-aware automotive climate control for electric vehicles, in: Design Automation Conference (DAC), 2015, pp. 1–67.
[30] X. Zhang, N. Mohan, M. Torngren, J. Axelsson, D.-J. Chen, Architecture exploration for distributed embedded systems: a gap analysis in automotive domain, in: IEEE international symposium on industrial embedded systems (SIES), 2017, pp. 1–10.
[31] Q. Zhang, et al., Semantic integration platform for cyber-physical system design, in: Design, Automation & Test in Europe Conference & Exhibition (DATE), 2019, pp. 1619–1624.