Proceedings of the ASME 2014 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2014
August 17-20, 2014, Buffalo, New York, USA

DETC2014-35314

SOLVING THE RECONFIGURABLE DESIGN PROBLEM FOR MULTIABILITY WITH APPLICATION TO ROBOTIC SYSTEMS

Jeffrey D. Arena[*], **James T. Allison** [†]
Engineering System Design Laboratory
Department of Industrial and Enterprise Systems Engineering
University of Illinois at Urbana-Champaign
Urbana, IL 61801
Email: {jdarena2, jtalliso}@illinois.edu

ABSTRACT

Systems that can be reconfigured are valuable in situations where a single artifact must perform several different functions well, and are especially important in cases where system demands are not known a priori. Design of reconfigurable systems present unique challenges compared to fixed system design. Increasing reconfigurable capability improves system utility, but also increases system complexity and cost. In this article a new design strategy is presented for the design of reconfigurable systems for multiability. This study is limited to systems where all system functions are known a priori, and only continuous means of reconfiguration are considered. Designing such a system requires determination of (1) what system features should be reconfigurable, and (2) what should the range of reconfigurability of these features be. The new design strategy is illustrated using a reconfigurable delta robot, which is a parallel manipulator that can be adapted to perform a variety of manufacturing operations. In this case study the tradeoff between end effector stiffness and speed is considered over two separate manipulation tasks.

1 MOTIVATION

Reconfigurable systems have been defined in literature as "those systems that can reversibly achieve distinct configurations (or states), through alteration of system form or function, in order to achieve a desired outcome within acceptable reconfiguration time and cost [1]." (Systems that are able to change their states or configurations have also been called "adaptable" [2], or "flexible" [3,4]. We will continue using the term "reconfigurable" for consistency, but note in passing that the use of the adjective "flexible" to describe reconfigurable systems could cause confusion, especially when designing systems with elastic compliance, i.e. mechanical flexibility.)

Many practical applications benefit from the implementation of reconfigurability in design. A prime example is manufacturing. According to Koren, high responsiveness is among the three principal goals of modern manufacturing systems [5]. In 1996, the state of Michigan, the National Science Foundation, and several major manufacturing companies provided $47 million in seed money to found the Engineering Research Center for Reconfigurable Manufacturing Systems at the University of Michigan. This funding resulted in over 1700 papers published about reconfigurable manufacturing systems by 2010. Reconfigurable manufacturing systems offer firms the ability to change manufacturing capacity and functionality quickly, allowing fast response to unpredictable changes either externally (e.g. in market demand) or internally (e.g. in necessary machine maintenance) [6].

[*]Graduate Student, Department of Industrial and Systems Engineering, ASME Student Member

[†]Assistant Professor, Department of Industrial and Systems Engineering, ASME Member

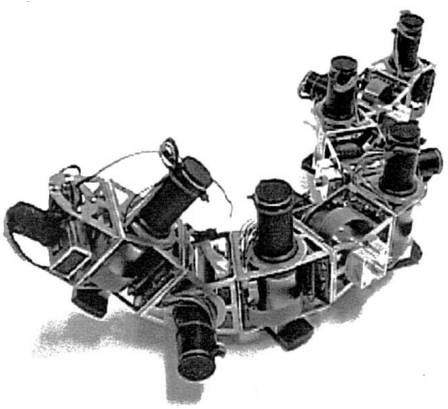


FIGURE 1: POLYBOT ROBOT DEVELOPED AT XEROX PALO ALTO RESEARCH CENTER [8]

Reconfigurability also finds application in aircraft design. Morphable airfoils are a reconfigurable system with the potential to increase the efficiency and maneuverability. Because a single aircraft may assume many distinct operating states over a typical mission (loitering, climbing, cruising, etc.), and because a fixed (non-reconfigurable) system is not capable of providing optimal performance over multiple different operating states, the system is a good candidate for reconfigurability. The performance benefits of changeable wing shape have been explored at least since the beginning of manned flight (the Wright Flyer in 1903), and have seen continued application in many other commercial aircraft to date (e.g., the Grumman F-14 Tomcat, and the Rockwell B-1 Lancer) [7].

Another application of reconfigurability is modular robotics. The goal of the work in this domain is to develop a versatile, cheap, and controllable robot made up of self-similar modules. PolyBot (Fig. 1) is one example of research aimed at achieving this goal. Developed at the Xerox Palo Alto Research Center (PARC), Polybot is a composed of many robotic units that function as actuated hinges in the assembled design. By altering the connections between units, the robot reconfigures itself into different structures, each capable of a different gait. Attainable structures include a four-legged walking robot, a snake-like chain capable of "slithering," and a circular structure capable of moving like the tread on a tank [8].

These examples only scratch the surface of the applications of reconfigurability. Many more systems in industry and academia serve as excellent examples of exploitation of reconfigurability. The interested reader is referred to [1] for a more exhaustive list.

2 LITERATURE REVIEW AND BACKGROUND

This section is divided into two parts. First, we focus on a review of studies that target design for reconfigurability. Second, we review the basic problem structure in product family design and the insight it offers into the reconfigurability problem.

2.1 Classification of Reconfigurable Systems

In the design research community, reconfigurable system design has received considerable attention in recent years. Common issues that are discussed in the literature include: (1) generation/selection of reconfigurable design concepts, (2) estimation of the cost of reconfigurability, (3) classifications of reconfigurable systems, and (4) selection of reconfigurable variables.

Research in reconfigurable design concept generation and selection aims to provide quantitative methods to discount inferior design concepts as early as possible. Mattson and Messac distinguish clearly between the terms "concept generation" and "alternative generation"—the prior referring to generation of topologically different designs, and the latter referring to different embodiment within a particular system architecture. In the same work, Mattson et al. introduced the idea of s-Pareto optimality, a set-level parallel of Pareto optimality. The authors show that a design concept can be represented by an attainable Pareto set in the objective space. When the Pareto sets of multiple concepts are combined in the performance space, the non-dominated points on this combined surface form the s-Pareto frontier [9]. Literman, Cormier, and Lewis expand on existing concept analysis methods to target design for reconfigurability, adapting and applying them to generate promising design concepts for reconfigurable systems early in the design process [10].

Classifying the types reconfigurable systems and the underlying motivations for reconfigurability helps to break down design problems into more manageable parts. For example, the designer's approach in a *modularly* reconfigurable system being designed for *robustness* should be very different from the approach used in a *continuously* reconfigurable system being designed for *adaptability*. To sharpen the meanings associated with terms like "robustness", "adaptability", and "modular reconfigurability", there was a need for clear classifications. In this vein, Olewnik et al. introduced a hierarchical classification of reconfigurable systems [3], which is illustrated in Fig. 2. The most general class of systems was termed "open", which is a generalized term used originally by Simpson to describe "systems of industrial products, services, and/or processes that are readily adaptable to changes in their environment..." [11]. The hierarchy of Olewnik et al. differentiates between "flexible" and "modular" systems; system adaptability versus system robustness; and "passive" versus "active" adaptability. Definitions from the paper are included here for completeness. They are taken directly from [3].

Flexible systems: Systems designed to maintain a high level

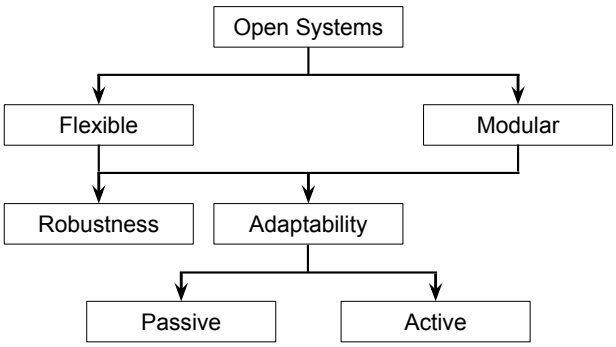


FIGURE 2: Heirarchy of open systems proposed by Olewnik et al. [3]

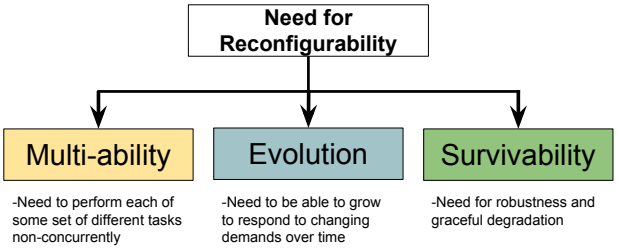


FIGURE 3: The underlying motivations for reconfigurable design [12]

of performance through real-time adaptations in their configuration and/or through robust parameter settings when operation conditions or requirements change in a predictable or unpredictable way. This definition implies that flexibility can be obtained through two modes: adaptability and robustness.

Adaptability: Mode of achieving flexible systems where system parameters (design variables) that can be changed and their range of change are identified to enchance performance of the system in *predictable* changes in the operating environment; they can be changed when the system is not in use (passive) or in real time (active)

Robustness: Mode of achieving flexible systems where system parameters (design variables) are set constant to minimize the effect of *unpredictable* changes in the operating environment on the performance of the system without eliminating the cause of the changes themselves.

Another important classification was made by Siddiqi, De Weck, and Iagnemma. They identified three system requirements that motivate incorporation of reconfigurability [12]: (1) multiability, (the ability of a single system to perform different tasks non-concurrently) (2) evolvability, (the ability of a system to adapt to known or unknown changes in its future operating environment) and (3) survivability (the ability of a system to continue functioning despite failure in one or more components). Figure 3 diagrams these ideas.

Total cost of reconfigurability is an essential part of the decision of whether or not to incorporate it in a design. Further, the relative cost of reconfigurable system components often drives the reconfigurable design variable selection problem. Patterson, Pate, and German detailed many of the benefits associated with designing a modularly reconfigurable family of UAVs [13]. The chief motivating factors for this consideration were both fleet evolvability (ability to adapt to known and unknown future field conditions) and cost (because a modular fleet might also be less expensive to maintain). Ferguson and Lewis considered the performance cost associated with the potential added mass required to incorporate reconfigurability [14]. Ferguson, Kasprzak, and Lewis went on to use this performance cost to help decide which system variables should be made reconfigurable in the design of a family of formula race cars [15]. Khire and Messac identified the similarity between the variable selection problem in reconfigurable system design, and the platform selection problem in product family design, exploiting it in their solution to the variable selection problem for the design of an air-conditioning system [2].

2.2 Relationship Between Reconfigurable Systems and Product Families

At this point, the parallels between product family design and reconfigurable system design may have already occurred to the reader. This connection has been made by more than a few authors in literature [2,4,13,15], especially those who contribute to both reconfigurable design and product family design communities. We outline the premise of product family design below and then describe the similarities it bears to the reconfigurable system design problem. The interested reader is referred to [16] for a more in-depth review of product family design.

At a high level, product family design research aims to find design approaches that maximize profit over all the market segments they serve. This single objective drives firms to simultaneously maximize commonality over their product portfolio (because it costs less to produce a product portfolio with more commonality), while meeting necessary performance metrics in all their target market segments. This dichotomy is commonly referred to in literature as the "commonality vs. performance" tradeoff. Firms respond to this dichotomy by designing core components and technologies that can be shared between many of their products. Black & Decker® power tools are a well-known example of a product line leveraging common core technologies. The set of elements that are shared among multiple products is called a product platform. Collectively, the products that share a platform are known as a product family.

Product family design approaches are classified broadly into

two groups: (1) module-based and (2) scale-based. In module-based product family design, product platforms are functional modules shared over the products of a firm. For example, the chassis and suspension of a car company's luxury sedan might be identical to those of its lower-end models. In this case, the chassis and suspension module would be part of the product platform. In a scale-based product family design, elements of a product platform are technologies or components which designers can "stretch" or "shrink" with minimal redesign.

Product family design requires designers to consider questions that extend beyond single product design in order to remain competitive in all market segments. How many product families should a firm offer? What components should make up each product platform? How can designers optimize the design of a single platform that is used in multiple end products? Justifying design decisions requires information about customer preferences, manufacturing costs, and performance capabilities. Research work in this field has sought to provide systematic approaches to simplify the many decisions necessary in product family design.

The "commonality vs. performance" tradeoff in product family design is analogous to the "reconfigurability vs. performance" tradeoff in reconfigurable system design. Increasing reconfigurable capability supports better performance for each system function, but results in greater system cost. Reducing reconfigurable capability results in a simpler, less-expensive system, but sacrifices the ability to tailor system configuration to optimize performance for each system function. Reducing reconfigurable capability increases commonality between system configurations (modes of operation), similar to increasing commonality in a product family at the cost of reduced performance across market segments.

In this work, we develop the reconfigurable design variable selection problem for a particular class of reconfigurable systems: continuously reconfigurable systems designed for multiability through offline transformation. We show that the reconfigurable system designed for multiability bears many similarities to a formulation originating from product family design, and adapt this formulation to our problem. Further, we show that for this specific class of reconfigurable systems, the designer does not need to compare the attainable Pareto fronts of different reconfigurable candidates, and can instead formulate a larger multiobjective problem whose Pareto optimal points will each encode fully reconfigurable designs.

2.3 Reconfigurable System Design and Product Family Design Comparison

Consider the design of a continuously reconfigurable system for multiability. We use the term "continuously reconfigurable" to mean that all reconfigurable variables can vary continuously along a prescribed range. Suppose that we seek to improve the performance of the system in m discrete tasks. Suppose the system architecture has been specified a priori, and that our goal is to determine optimal ranges for reconfigurable design variables. (It is worth noting that an "optimal" reconfigurable range may be 0.) Solving this variable selection problem means selecting not only which variables are reconfigurable, but also to what extent. Let the number of design variables necessary to specify a fixed (i.e. non-reconfigurable) design be n and the number of configurations of the reconfigurable system be m. For configuration i, let $f^i(\cdot)$ be the negative of the utility function for that operating mode. Let the scalar function $c(\cdot)$ represent the cost of a reconfigurable system. We can formulate the following multiobjective optimization problem:

$$\begin{aligned} \min_{\overline{\mathbf{x}}} \quad & \{f^1(\mathbf{x}^1), ..., f^m(\mathbf{x}^m), c(\overline{\mathbf{x}})\} \\ \text{subject to} \quad & \mathbf{g}_i(\overline{\mathbf{x}}^i) \leq 0 \quad \forall i, \ 1 \leq i \leq m \\ & \mathbf{h}_i(\overline{\mathbf{x}}^i) = 0 \quad \forall i, \ 1 \leq i \leq m \end{aligned} \tag{1}$$

where:

$$\overline{\mathbf{x}} := \begin{bmatrix} \mathbf{x}^1 \\ \vdots \\ \mathbf{x}^m \end{bmatrix}, \quad \mathbf{x}^i := \begin{bmatrix} x_1^i \\ \vdots \\ x_n^i \end{bmatrix}.$$

$\overline{\mathbf{x}}$ is a vector of design variables composed of individual design vectors, where $\mathbf{x}^i$ is a design vector that describes the ith configuration. Thus, the design vector for the full reconfigurable system design problem has length mn, and specifies the values of all n (possibly reconfigurable) design variables at all m configurations. The inequality and equality design constraints (if present) are represented by $\mathbf{g}(\cdot)$ and $\mathbf{h}(\cdot)$, respectively. If reconfigurability cost nothing, the solution to this multiobjective problem would simply be a concatenation of the solutions to each single-objective problem. When realistic cost function c is used, however, the tradeoff between reconfigurability and cost presents itself, forcing the designer to choose between cost and performance.

Compare the above optimization formulation with the following product family design formulation, adapted from Fellini's product platform selection problem [17].

$$\max_{\eta,\overline{\mathbf{x}}}\{f^1(\mathbf{x}^1),...,f^m(\mathbf{x}^m),\sum_{(i,j)pq}\eta_{ij}^{pq}\}$$
$$\forall p,q\in\mathcal{P},(i,j)\in\mathcal{S}^{pq},p<q$$

$$\begin{aligned}\text{subject to}\quad & \mathbf{g}^p(\mathbf{x}^p)\leq\mathbf{0}\\ & \mathbf{h}^p(\mathbf{x}^p)=\mathbf{0}\\ & \eta_{ij}^{pq}(x_i^p-x_j^q)=0\\ & \eta_{ij}^{pq}\in\{0,1\}\end{aligned}\tag{2}$$

where:

$\overline{\mathbf{x}}$	defined as in formulation 1
f^p	performance function for product p
$\mathbf{x}^p$	design vector for product p
$\mathcal{P}$	set of indices, each referring to a product
m	number of products and the cardinality of $\mathcal{P}$
S^{pq}	index pairs of elements that are candidates for being shared between two products p and q
$\mathbf{h}^p$	equality constraints in product p
$\mathbf{g}^p$	inequality constraints in product p
η_{ij}^{pq}	binary variable indicating if components i and j are shared between products p and q

Notice that in both the reconfigurable system optimization formulation and the product platform design formulation, the commonality vs. cost dichotomy arises. At an abstract level the two problems are nearly identical. The implementation of the optimization formulation given in Eqn. (1) will be explored further in the next section.

3 VARIABLE SELECTION PROBLEM FOR CONTINUOUSLY RECONFIGURABLE SYSTEMS FOR MULTIABILITY

Because so many systems fit under the umbrella of "reconfigurable systems", it can be difficult to arrive at specific design strategy capable of treating all of them. The nature of the reconfigurable design problem depends on the goal sought and the method of reconfigurability. For example, systems that are modularly reconfigurable require a different approach than those that are continuously reconfigurable. This work treats the design of a particular type of reconfigurable system: "offline continously reconfigurable," for a particular purpose: "multiability." Further, we assume that the system architecture has already been selected, and that we are looking to solve the variable selection problem. That is, we are looking to decide which variables to make reconfigurable and over what ranges. Having specified this class of reconfigurable systems as the scope of our study, we can proceed with a more in-depth analysis.

We can now discuss the design variables of the reconfigurable system and, because we have targeted multiability as our goal, we can talk about set of tasks that we are interested in having the system perform. Let m be the number of tasks the reconfigurable system is to perform. The reconfigurable system, therefore, will have at most m different configurations, i.e. at most m distinct n-dimensional design vectors, each specifying a nonreconfigurable design.

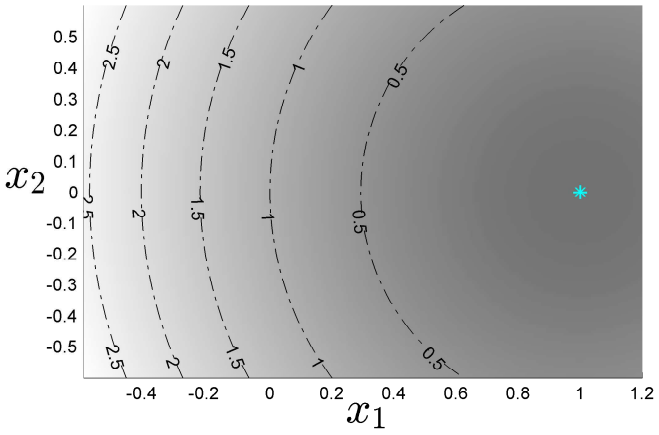
Considering m different tasks for the system, we should be able to establish utility functions for each task, which depend on the values of the static design variables in each task configuration. (In the case of a manufacturing system being designed for multiple machining operations, these utility functions might correspond to the times required to complete reference tasks in each configuration.) Call the set of m utility (objective) functions $f^1(\cdot)$, $f^2(\cdot)$, etc. such that $f^i(\cdot)$ corresponds to task i.

To better describe this variant of the reconfigurable system design problem, we introduce a simple multiobjective optimization example problem, and then recast it as a simultaneous variable selection and optimization problem for reconfigurable system design. Consider the following simple multiobjective optimization problem:

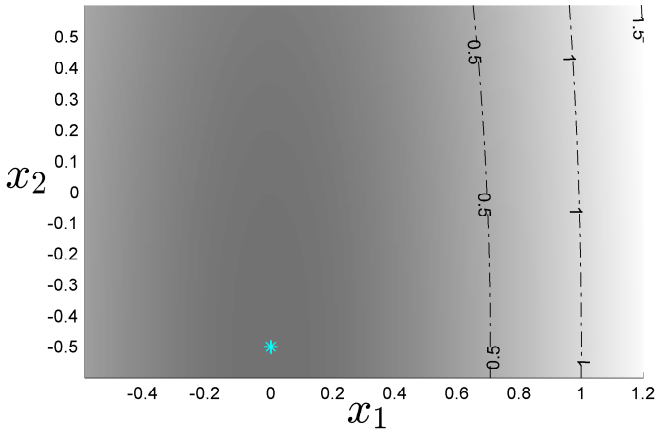
$$\min_{\mathbf{x}=[x_1,x_2]^T}\ \{f^1(\mathbf{x}),f^2(\mathbf{x})\}\tag{3}$$
$$\text{where: } f^1(\mathbf{x})=\left(\frac{x_1-1}{1^2}\right)^2+(x_2)^2$$
$$f^2(\mathbf{x})=(x_1)^2+\left(\frac{x_2+0.5}{2^2}\right)^2$$

Because this multiobjective optimization problem happens to have 2 design variables and 2 objective functions, we can plot points in the design space, along with their corresponding images in the function space. These plots are shown in Fig. 4.

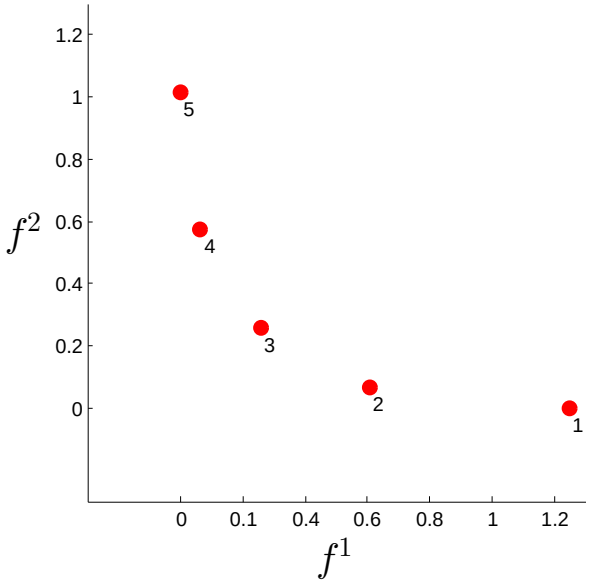
Selecting ranges of reconfigurability for design variables usually means attempting to "value" subsets of the Pareto set in the performance space, and choose optimum ranges for each reconfigurable variable such that this value is maximized. Consider Figures 4c and 4d. The points shown in these figures lie in the Pareto set of the nonreconfigurable design problem—each point corresponding to a fixed design. In this example, the variable selection problem amounts to choosing two ranges, (one each for x_1 and x_2) which capture the "best" subset of the Pareto set in the nonreconfigurable design problem performance space. For example, a designer might choose bounds for reconfigurable variables in the design space that capture points 2, 3, and 4 in Fig. 4c. Valuing subsets of this Pareto set can be difficult and requires prior determination of the Pareto frontier before decisions of reconfigurability can be made. In the continuously reconfigurable design problem for multiability, we are only interested in



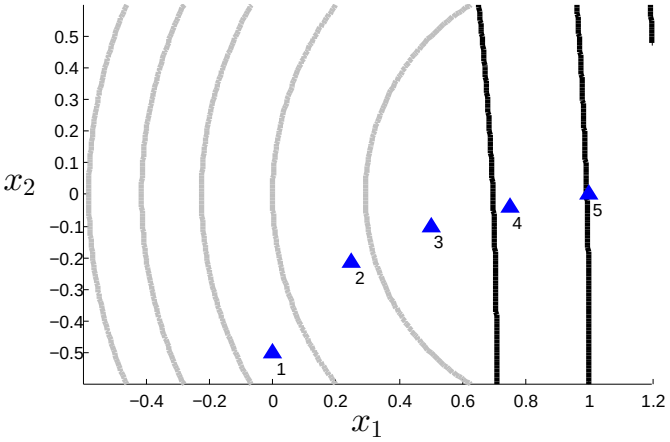
(a) f^1 objective surface with minimum value shown at starred point



(b) f^2 objective surface with minimum value shown at starred point



(c) 5 points along the Pareto frontier



(d) Preimage of the 5 points shown in Fig. 4c

FIGURE 4: MULTIOBJECTIVE DESIGN PROBLEM WITH 2 OBJECTIVES AND 2 VARIABLES

the system's individual performance in each of the m distinct tasks. Since each task has its own objective function, the "value" of a subset of the nonreconfigurable Pareto front is captured only by the m points on the subset of the Pareto front that individually minimize each of the objective functions $f^i(\cdot)$ individually. (i.e., the points that minimize the $f^i(\cdot)$ value achievable by the given reconfigurable system design $\forall i = 1, 2, \ldots m$).

Provided that cost can be quantified as a function of the extent of reconfigurability, the variable selection problem can be recast into the following simple form, whose performance space is only one dimension higher than the performance space of the fixed problem.

Consider the following reformulation:

$$\min_{\mathbf{x^1},\mathbf{x^2}} \quad \{f^1(\mathbf{x^1}), f^2(\mathbf{x^2}), c(\mathbf{x^1},\mathbf{x^2})\} \tag{4}$$

$$\text{where: } f^1(\mathbf{x}) = \left(\frac{x_1 - 1}{1^2}\right)^2 + (x_2)^2$$

$$f^2(\mathbf{x}) = (x_1)^2 + \left(\frac{x_2 + 0.5}{2^2}\right)^2$$

In this formulation, the two vectors $\mathbf{x^1}$ and $\mathbf{x^2}$ specify a reconfigurable design by specifying the system configurations for each task. Based on these two (in general m) vectors, the ranges of reconfigurability of each design variable can be determined. The optimal design vector for the reconfigurable design problem contains more information than just the required ranges for each design variable. It also specifies an optimal system configurations for each task.

Note the changes from Eqn. (3) to Eqn. (4). In Eqn. (4) we add one design vector to the original problem and a new function $c(\cdot)$ that quantifies the cost of reconfigurability. The Pareto set generated by this formulation expresses the tradeoff between reconfigurable system performance in individual tasks and the cost of increasing the level of reconfigurability. The goal of the designer is to select a point along this Pareto set, thereby selecting a design which is non-dominated in the context of the multiability problem.

The performance space of this multiability problem consists of three dimensions ($m + 1$ in general), one for cost and one for each of the m performance functions. In the case of the multiability problem, it is very important to note that the designer is only concerned with the extreme points on any candidate subset of the fixed system's Pareto frontier. This simplification of the design problem stems from the assumption that we are only concerned with m separate tasks individually.

Consider a given point in the Pareto front of Eqn. (3) and how it compares to a point on the Pareto front in Eqn. (4) in both the objective and design space. Points in the Eqn. (3) Pareto set fall along a continuous curve that goes from point 1 to point 5 in Figs. 4c and 4d. The points in the design space (Fig. 4d) and objective space (Fig. 4c) each represent a given design with fixed design variable values. In contrast, points in the Eqn. (4) Pareto set each correspond to a reconfigurable system on a larger Pareto front. In the objective space, the Pareto set of this second formulation must include the Pareto set of the original nonreconfigurable formulation; because cost of all points on the first Pareto set is zero, none can be dominated by a reconfigurable system design on the expanded Pareto set. This expanded multiobjective Pareto set provides valuable information during early stage design, making the price of reconfigurability and its tradeoff clear to the designer. This Pareto set surface in the objective space is depicted in Fig. 5.

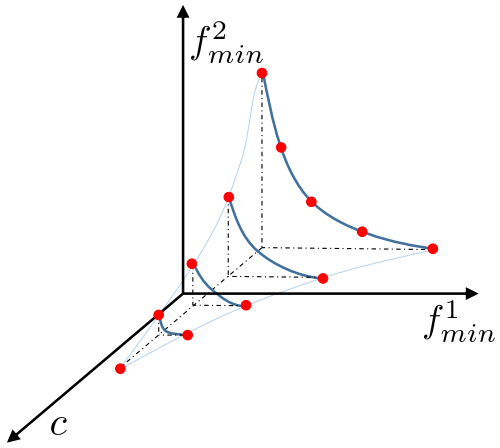


FIGURE 5: EXPANDED PARETO SURFACE FOR A CONTINUOUSLY RECONFIGURABLE SYSTEM DESIGNED FOR MULTIABILITY

A few features are important to note. As was already mentioned, the lowest cost cross-section of the surface corresponds to the Pareto set of the multiobjective optimization of the fixed system. Secondly, the utopia point of the lowest cost Pareto set (in fact the utopia point of Pareto sets at all cost levels) is the only coordinate of the Pareto surface at the highest cost level. This must be true, since allowing the system to reconfigure completely should allow it to achieve the performance extremes of the fixed design. Finally, consider that, as cost increases, the Pareto set cross-sections must perform at least as well as cross sections at lower cost levels.

The important point here is that for continuously reconfigurable systems designed for multiability, this surface can be a useful early stage design tool that demonstrates the cost of reconfigurability from a fixed design to a fully reconfigurable one. Rather than calculate the full Pareto sets of each reconfigurable system and compare them, (in the case of multiability) a designer can focus on only m important points on each Pareto set (i.e. those which minimize performance functions individually). When plotted against the cost objective function (e.g., Fig. 5), the designer is presented with a Pareto front where each point refers to a Pareto optimal reconfigurable system. Aside from the additional cost axis, Note the change in axis labels from f^1 and f^2 in the non-reconfigurable problem to $f^1_{\min}$ and $f^2_{\min}$ in the reconfigurable one.

The careful reader might observe one caveat of this multiability formulation; it assumes the only coupling between vectors $\mathbf{x}^i$ and $\mathbf{x}^j$ occurs through the cost function, $c(\overline{\mathbf{x}})$. Thus, the formulation given in Eqn. (1) of Section 2.3 does not allow for "hard" (i.e., inequality or equality) constraints that couple design variables across two tasks of the reconfigurable system, (what we

FIGURE 6: The ABB Flexpicker®

will call system-level constraints). This is a valid and valuable observation of the given formulation. In practice, the formulation could still be applied to problems with these "system-level" hard constraints by assigning a prohibitively high costs to their violation as a form of penalty. For an appropriately chosen cost penalty, designs that are infeasible because of system-level constraints would be made to costly for further consideration. Incorporating system-level constraints into the given reconfigurability formulation would be an interesting topic for future research.

In the next section, we attempt to find points along this surface in the case of a parallel machine tool being designed for multiability. This case study will help demonstrate the usefulness of the expanded Pareto surface introduced in this paper.

3.1 Overview

The Delta robot is a type of parallel robot which is customarily used for high-speed pick-and-place operations. Kinematically, the Delta is specially designed so that the orientation of its end effector is constant, i.e. all end effector motions are pure translations. Originally developed by Clavel and his research team at EFPL (École Polytechnique Fédérale de Lausanne) in the early 1980s [18], the robot is now used at an industrial level for high-speed pick-and-place operations. Examples of commercial and industrial application of the robot include the FlexPicker® (Fig. 6) and the SurgiScope®.

3.2 Framing The Design Problem

The current case study treats the design of a continuously reconfigurable delta robot for two tasks. The first task is a high

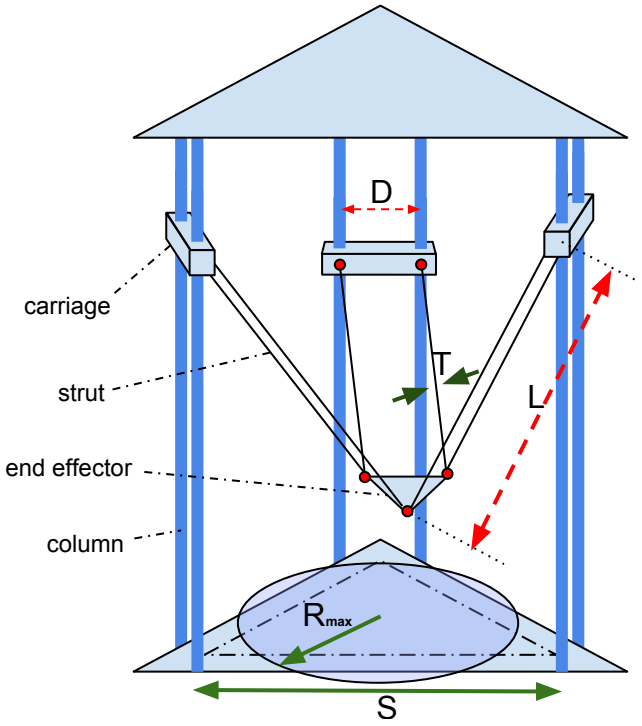


FIGURE 7: DESIGN ARCHITECTURE IN CASE STUDY. TWO DESIGN VARIABLES ARE MARKED WITH DASHED LINES

speed pick-and-place operation of payloads with negligible size. The second task is a milling operation, requiring system rigidity rather than speed. We compare designs by evaluating stiffness and speed characteristics of each point in the build area along a polar grid. Pointwise performance is then used to calculate performance functions for stiffness and speed that apply to the whole design.

The goal of the case study is to determine reconfigurable designs which make up the reconfigurable system Pareto set described in Section 3. The study will illustrate one method for finding this Pareto frontier, as well as its value as an early stage design tool.

The proposed design architecture is shown in Fig. 7. All joints in the design are spherical. A summary of all design variables and parameters in the figure is included in Table 1. Most variables listed in Table 1 are self-explanatory. We would however like to clarify the meaning of the variables S and $R_{\max}$.

The 'S' Parameter Solving the forward kinematics problem in this Delta architecture, i.e., solving for the end effector location given the location of the carriages, amounts to finding the intersection of three spheres as they move in the z direction. These spheres are not centered exactly on the carriages

Variable	Type	Units	Nominal Value or Range
D	design var.	m	$0.05 - 0.35$
L	design var.	m	$(0.6 + R_{\max}) - 3.5$
S	parameter	m	1
$R_{\max}$	parameter	m	$0.7\frac{S}{\sqrt{(3)}}$
T	parameter	m	0.05
E	parameter	Pa	2×10^9

TABLE 1: VARIABLES USED IN RECONFIGURABILITY CASE STUDY

as one might expect, however. Instead, their center locations are offset toward the end effector by $\frac{D}{2\sqrt{3}}$. This is to account for the size of the end effector itself. The distance S, which is held constant in this case study, is the distance between these kinematic spheres as measured in the xy plane. Holding this distance constant allows the kinematics of the Delta to be treated independently of the separation distance D. In this reconfigurable problem it is assumed that reconfigurability in the D variable is accompanied by changes in column separation necessary to preserve the distance S.

The '$\mathbf{R}_{\max}$' Parameter The $R_{\max}$ parameter measures radius of the build area of interest. This value is necessary to specify a priori so that Delta robot designs can be compared using local stiffness and speed metrics at a standard set of points in the build envelope. Calculating rigidity metrics over the reachable envelopes of each design and comparing them would not be an apples-to-apples comparison. Thus, there is a need for a standard build area over which we can compare all designs fairly. For the purposes of this case study we happened to choose the radius of our desired build envelope as $R_{\max} = 0.7\frac{S}{\sqrt{3}}$.

3.3 Performance Functions

In order to frame the reconfigurable variable selection problem as was done in Section 3, we need to provide performance functions for each task. In the current problem, we need two performance functions capable of quantifying the speed and rigidity of a given delta robot geometry, respectively.

Rigidity Performance Function Our approach in defining a stiffness performance function is an adaptation of that used by Courteille, Deblaise, and Maurine [19]. In their work, Courteille et al. design a similar parallel robot for rigidity. Courtielle et al. use the alternative eigenvalue problem proposed

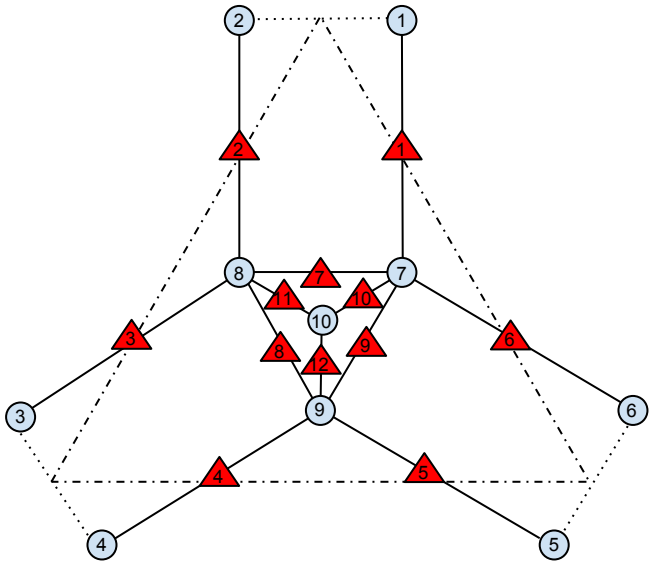


FIGURE 8: NODE AND ELEMENT NUMBERING USED IN DELTA STIFFNESS CALCULATION. NODES ARE LABLED USING A CIRCLE AND ELEMENTS ARE LABELED USING A TRIANGLE.

by Lipkin and Patterson [20] to calculate the principle angular stiffnesses, and principle linear compliance directions in a Delta-like robot. We now provide a general overview of this approach and discuss its implementation in our system. For more details on the theory behind the method, the interested reader is referred to the original articles [19, 20].

The performance function used in this design problem comes from a linear finite element model of the Delta robot. For a set of n regularly-spaced positions within the desired build envelope, we measure the stiffness of the robot end effector as if the Delta were a static structure comprised of beams. This gives us the ability to quantify the stiffness of the robot at any position by using the direct stiffness method. The result is a 6×6 stiffness matrix for each point where we perform the calculation. In this static model of our Delta, we assume that compliance occurring in the columns of the robot are negligible. A diagram of our node and element numbering scheme is included in Fig. 8 for reference. (Note that the nodes have circular labels, and the elements have triangular labels.)

Lipkin and Patterson proposed the "alternative eigenproblem" to decompose the 6×6 stiffness matrix at each point into principle rotational stiffnesses and linear compliances. The alternative eigenproblem is stated as follows:

$$c_{f,i}\mathbf{\Gamma}\overline{F}_i = \mathbf{C}\overline{F}_i \quad (5)$$
$$k_{\gamma,i}\mathbf{\Omega}\overline{x}_i = \mathbf{K}\overline{x}_i \quad (6)$$

where:

$$\mathbf{\Gamma} = \begin{bmatrix}\mathbf{I}_3 & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{0}_3\end{bmatrix} \quad \mathbf{\Omega} = \begin{bmatrix}\mathbf{0}_3 & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{I}_3\end{bmatrix}$$

In the alternative eigenproblem $\overline{F}$ is a generalized force vector made up of three translational forces and three torques, termed a wrench. Similarly, $\overline{x}$ is a generalized displacement vector made up of three linear displacements and three rotational displacements, termed a twist. Generalized force vectors that satisfy Eqn. (5) produce pure translations in the end effector parallel to the applied translational force and are termed eigenwrenches. Similarly, generalized displacement vectors that satisfy Eqn. (6), result in purely rotational reaction force parallel to the applied displacement and are termed eigentwists.

By comparing the eigenvalues of the alternative eigenproblem, a designer is capable of assessing the level of translational and rotational stiffness in three principal directions. These eigenvalues can be used to form local stiffness metrics which indicate the magnitude and uniformity of stiffness at a given location in the build envelope.

While Lipkin and Patterson's approach can be applied to each point in the build envelope, Courteille et al. suggest global metrics, which combine stiffness information from a collection of n points from throughout the build envelope. They combine local stiffness/compliance matrices to form global stiffness/compliance matrices, and then use a singular value decomposition to determine global indices. The global stiffness and compliance matrices are calculated as follows:

$$\widetilde{\mathbf{K}}^G = \begin{bmatrix}\widetilde{\mathbf{K}}_1\\ \widetilde{\mathbf{K}}_2\\ \vdots\\ \widetilde{\mathbf{K}}_n\end{bmatrix}_{(6n\times 6)} \qquad \widetilde{\mathbf{C}}^G = \begin{bmatrix}\widetilde{\mathbf{C}}_1\\ \widetilde{\mathbf{C}}_2\\ \vdots\\ \widetilde{\mathbf{C}}_n\end{bmatrix}_{(6n\times 6)}$$

where for the ith point in the build area we define:

$$\widetilde{\mathbf{K}}_i := \mathbf{K}_i\mathbf{\Gamma} \ , \widetilde{\mathbf{C}}_i := \mathbf{C}_i\mathbf{\Omega}$$

We take the singular value decompositions of both $\widetilde{\mathbf{K}}^G$ and $\widetilde{\mathbf{C}}^G$ to obtain:

$$\widetilde{\mathbf{K}}^G = \mathbf{U}_f^G\mathbf{\Sigma}_f^G\mathbf{V}_f^{G^T}, \ \widetilde{\mathbf{C}}^G = \mathbf{U}_\gamma^G\mathbf{\Sigma}_\gamma^G\mathbf{V}_\gamma^{G^T}$$

The first three values on the diagonal of the middle term, $\mathbf{\Sigma}^G$, provide three scalars which quantify the overall linear stiffness or angular compliance of all n points in the build area. Let the three scalars on the diagonal of $\mathbf{\Sigma}_f^G$ be, in descending order of magnitude, $\sigma_{1,f}$, $\sigma_{2,f}$, and $\sigma_{3,f}$. Define $\sigma_{1,\gamma}$, $\sigma_{2,\gamma}$, and $\sigma_{3,\gamma}$ simmilarly for $\mathbf{\Sigma}_\gamma^G$. We can then state the four stiffness metrics suggested by Courtielle et al. and used as the basis of our stiffness performance function in this work. They are:

$$S_{k1} = \frac{\sigma_{1,f}}{\sigma_{3,f}}, \qquad S_{k2} = \sigma_{3,f}$$
$$S_{c1} = \frac{\sigma_{1,\gamma}}{\sigma_{3,\gamma}}, \qquad S_{c2} = \sigma_{3,\gamma}$$

In this work, our task is to combine these stiffness metrics into a single performance function that corresponds to the milling task in the reconfigurable design problem. We use a simple weighted sum of all four terms to provide this performance function. In future work, this function might be replaced by a higher-fidelity computer simulation of a milling job, or a utility function validated using simulation. For the purposes of the current work, the simple weighted sum is sufficient to demonstrate the design method. In the sum, metrics S_{k2} and S_{c2} are normalized by nominal global values to bring them on the same order as S_{k1} and S_{c1}. Thus we can write our stiffness performance function as:

$$f^1 = S_{k1} + S_{c1} - \frac{S_{k2}}{S_{k2,nom}} + \frac{S_{c2}}{S_{c2,nom}}$$

Note that the best performance comes from minimizing this performance function. By minimizing the first two terms, S_{k1} and S_{c1}, we attempt to obtain a more uniform stiffness in the build area. The second two terms, are meant to maximize the overall level of linear stiffness and minimize the overall level of angular compliance, respectively.

Speed Performance Function The speed performance function of a given design is calculated using the kinematic Jacobian at a set of points in the build area. This is a new metric introduced here. The kinematic Jacobian can be written

as:

$$J = \begin{bmatrix} \frac{\partial z_1}{\partial x_e} & \frac{\partial z_1}{\partial y_e} & \frac{\partial z_1}{\partial z_e} \\ \frac{\partial z_2}{\partial x_e} & \frac{\partial z_2}{\partial y_e} & \frac{\partial z_2}{\partial z_e} \\ \frac{\partial z_3}{\partial x_e} & \frac{\partial z_3}{\partial y_e} & \frac{\partial z_3}{\partial z_e} \end{bmatrix}, \tag{7}$$

where x_e, y_e, and z_e are the Cartesian coordinates of the end effector, and z_1, z_2, and z_3 are the coordinates of the three carriages. The analytical expressions of each entry in the Jacobian expression can be easily obtained by differentiating the following inverse kinematic equation:

$$z_i = \sqrt{L^2 - (x_i - x_e)^2 - (y_i - y_e)^2} + z_e, \tag{8}$$

where x_i, y_i and z_i are the coordinates of the carriage, and L is the strut length for the design. This Jacobian relates input velocities at the carriages into end effector velocity by the following relation:

$$J \begin{bmatrix} \frac{\partial x_e}{\partial t} \\ \frac{\partial y_e}{\partial t} \\ \frac{\partial z_e}{\partial t} \end{bmatrix} = \begin{bmatrix} \frac{\partial z_1}{\partial t} \\ \frac{\partial z_2}{\partial t} \\ \frac{\partial z_3}{\partial t} \end{bmatrix} \tag{9}$$

Provided the Jacobian is invertible, we can multiply both sides by J^{-1} to obtain:

$$\begin{bmatrix} \frac{\partial x_e}{\partial t} \\ \frac{\partial y_e}{\partial t} \\ \frac{\partial z_e}{\partial t} \end{bmatrix} = J^{-1} \begin{bmatrix} \frac{\partial z_1}{\partial t} \\ \frac{\partial z_2}{\partial t} \\ \frac{\partial z_3}{\partial t} \end{bmatrix} \tag{10}$$

Thus, at each point in the build envelope, the inverse Jacobian maps carriage velocity to end effector velocity. At a given point, if we let carriage velocity vary along all possible directions, the resulting end effector velocity vectors will fall along a 3-dimensionsional ellipse. The minimum principal axis of this ellipse shows the limiting velocity direction and magnitude.

This limiting velocity forms the basis of our speed metric. Over a set of points in the build area, we observe the minimum limiting velocity. For point i in the build envelope let the limiting velocity magnitude be λ_i. The performance function to be minimized is:

$$f^2 = -(\min_i \lambda_i)$$

Cost Performance Function Incorporation of cost of reconfigurability is a critical part to this case study, as it establishes the expanded Pareto set introduced in Section 3. Olenik et al. described a few possible sources of cost in a reconfigurable system [3]. We consider two sources of costs, fixed and variable. Fixed costs are associated with making a variable in the design reconfigurable, while variable costs are associated with the extent of changeability. Thus, the cost of reconfigurability of design variable i can be written as:

$$c_i = \begin{cases} 0 & \Delta x_i = 0 \\ FC + VC(\Delta x) & \Delta x_i \neq 0 \end{cases}$$

where we define Δx_i as:

$$\Delta x_i = \max_k(x_i^k) - \min_k(x_i^k)$$

This nonlinear function is a fair representation of the cost of reconfigurability, but is discontinuous. This type of discontinuity also arises in product family design research in the variable selection problem (e.g., see the product family design problem formulation given in Section 1). In an effort to model this discontinuity in an objective function, we will use a continuous mapping function as an approximation. The details of this mapping implementation are left for Section 4.2.

4 CASE STUDY IMPLEMENTATION AND RESULTS

For this study, a polar grid with 10 equally spaced radial steps (from 0 to $R_{\max}$ radians) and 36 equally spaced angular steps (from 0 to 2π) was used to determine global metrics. Symmetry of the build area was exploited to reduce the number of points necessary in the grid by a factor of 6. Using this set of points in the build area, we used a multiobjective genetic algorithm to generate an extended Pareto front. This extended Pareto front was generated for varying values of fixed and variable cost of reconfigurability. Details of the multiobjective genetic algorithm and cost parameter sweep are given next.

4.1 Multiobjective Genetic Algorithm Parameters

In order to determine the Pareto set of the reconfigurable system design problem, a multiobjective genetic algorithm was used. This implementation provides reasonable means to determine the Pareto set. The values used within the genetic algorithm are included for completeness, but we emphasize that they are only a means to the objective of this work, and not central to its contribution. The genetic algorithm was carried out in the MATLAB® programming environment using the built-in genetic algorithm function with the options specified in Table 2.

TABLE 2: Genetic Algorithm Parameters Used

Parameter	Value
Population Size	300
Initial Population Dist.	Uniform
Crossover Fraction	0.4
Pareto Fraction	0.35
Elite Count	30

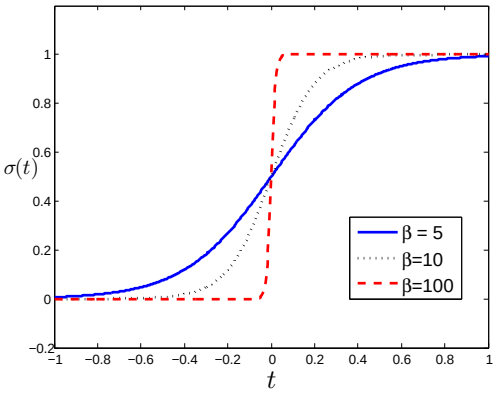


FIGURE 9: Sigmoid curves over varying values of β

4.2 Modeling Nonlinearity in Cost Objective

Modeling the non-linearity of the cost objective was accomplished using a sigmoid function. This function provides a convenient method to approximate a discontinuous function with a continuous one. The sigmoid function is an s-shaped curve that transitions continuously from 0 to 1. The location of that transition and its steepness is controlled by a shifting parameter, α, and scaling parameter, β, respectively.

$$\sigma(t, \alpha, \beta) = \frac{1}{1 + e^{\beta(x-\alpha)}}$$

Figure 9 displays several sigmoid curves for $\alpha = 0$ and varying values of β.

4.3 Numerical Experiments

The set of points used to determine the global performance metrics were taken from a polar grid of points sampled for 10 evenly spaced radii at 10 degree increments. (This polar grid of sample points was used for all subsequential numerical experiments as well.) The first attempt to find the Pareto front of this reconfigurable design problem was performed using the values of fixed cost (FC), variable cost (VC), α, and β that are given in Table 3. The fixed costs for each variable was taken to be 1 in this base case, and the variable costs were set such that the cost of varying either variable along the full range given in Table 1, would be 1.

TABLE 3: Baseline costs used in generation of extended Pareto front

D		L			
FC	VC	FC	VC	α	β
1	0.501	1	3.33	0.05	100

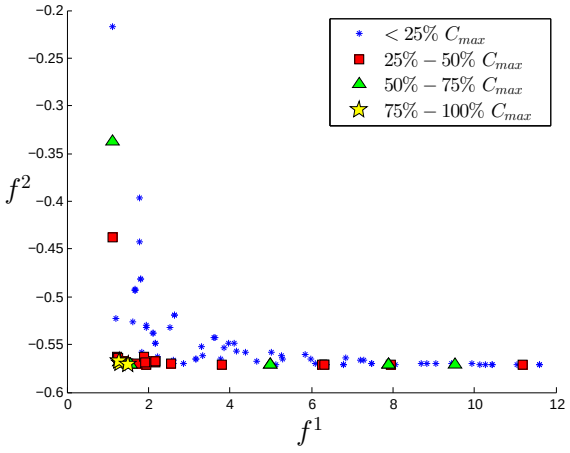
After obtaining a Pareto front from this case study, a numerical experiment was performed to measure the effect of fixed and variable cost values on the determined Pareto front. In this parameter sweep the ratio of fixed to variable cost was varied while the values of sigmoid parameters were held constant to those in the base case. In the 10 runs of the genetic algorithm, variable costs were held constant at the base level while variable costs were increased up to 10 times its base value. In another 10 runs, the experiment was reversed. Fixed cost levels were held constant as variable costs were increased to up to 10 times their level in the base case.

5 CASE STUDY RESULTS

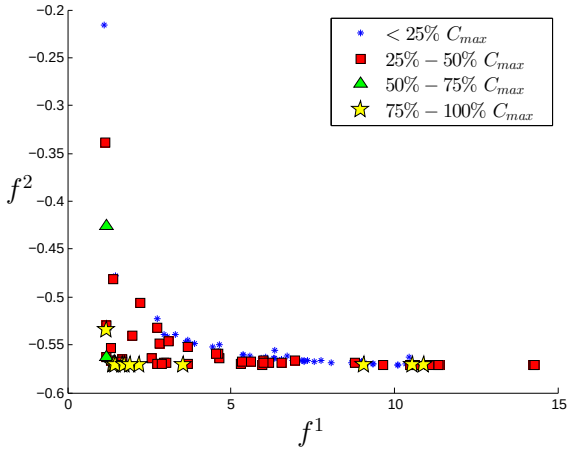
The case study resulted in the determination of over 100 points distributed along the expanded Pareto surface for each genetic algorithm run. Selected expanded Pareto surfaces that were obtained are included in Fig. 10.

Notably, these four expanded Pareto sets bear strong resemblance to the surface shown in Fig. 5. The lowest cost Pareto set shows the attainable performance of designs with little or no reconfigurability. As solutions become more costly, they approach the performance of the fixed system's utopia point. These figures show how the expanded Pareto surface of the same reconfigurable design problem changes as the relative magnitudes of fixed and variable cost of reconfigurability are altered.

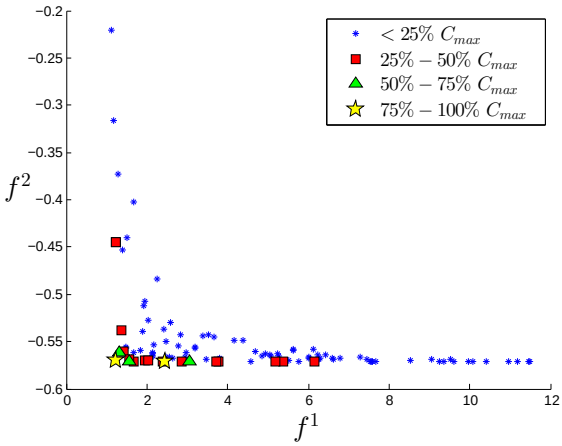
At an early stage in design, having access to points in this performance space helps to demonstrate the tradeoff between performance and cost present in the system. Being able to select from a set of near-Pareto optimal reconfigurable designs circumvents the difficulty of comparing the full Pareto sets associated



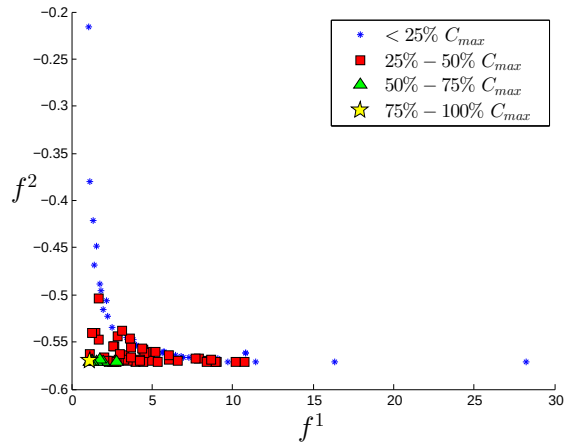
(a) Expanded Pareto surface: $VC = VC_{base}$, $FC = 1$



(b) Expanded Pareto surface: $VC = VC_{base}$, $FC = 9$



(c) Expanded Pareto surface: $VC = 10VC_{base}$, $FC = 1$



(d) Expanded Pareto surface: $VC = 1VC_{base}$, $FC = 8$

FIGURE 10: Selection of extended Pareto fronts found in case study. Different markers show attainable performance at different levels of cost. C_{max} is the highest cost of a system in the study.

with each reconfigurable design. This helps to streamline the reconfigurable design problem in the case of offline continuously reconfigurable systems designed for multiability.

6 CONCLUSIONS AND FUTURE WORK

We have shown that the variable selection problem in a particular subset of reconfigurable system design can be framed as a variant of the product family design variable selection problem. Further, this formulation allows the Pareto set of the reconfigurable problem to be obtained with the addition of only one dimension onto the performance space.

At early stages of design, having access to the expanded Pareto surface would be a great asset to any designer. For the addition of a single dimension to the performance space, this surface contains significantly more information than the static system's Pareto front. For the continuously reconfigurable systems designed for multiability, a designer can distill the pertinent information from all attainable m-dimensional Pareto sets into a single expanded Pareto surface of dimension $m + 1$, where each point in the surface represents a Pareto optimal design along the spectrum of the commonality-performance tradeoff.

While this design formulation is useful for this class of reconfigurable system, the method used to determine the Pareto set in this paper was prepackaged and not specifically tuned for its purpose. One potential area for future development of this work is the selection of a more targeted Pareto discovery method.

The case study used in this work is also an important area of future development. It is a stepping stone toward designing a fully reconfigurable Delta robot capable of functioning as a ver-

satile manufacturing tool. This initial stage of research provides a useful method to obtain non-dominated designs in reconfigurable design problem, but the cost and utility models used will be improved in future work.

Future plans to improve utility models involve the use of simulation to better capture the design requirements of the selected manufacturing operations. These simulations could serve as performance functions directly, or might form the basis of lower fidelity, higher-speed surrogate performance functions. Also, the equations used in this study for the cost of reconfigurability were not based on any industrial cost model. These cost functions were used only to demonstrate the expanded Pareto front concept. Future development of realistic cost models would give the case study results greater weight in the actual fabrication of a reconfigurable Delta robot.

REFERENCES

[1] Siddiqi, A., and de Weck, O., 2008. "Modeling methods and conceptual design principles for reconfigurable systems". *Journal of Mechanical Design, Transactions of the ASME,* ***130***(10), pp. 1011021–10110215.

[2] Khire, R., and Messac, A., 2008. "Selection-integrated optimization (sio) methodology for optimal design of adaptive systems". *Journal of Mechanical Design, Transactions of the ASME,* ***130***(10), pp. 1014011–10140113.

[3] Olewnik, A., Brauen, T., Ferguson, S., and Lewis, K., 2004. "A framework for flexible systems and its implementation in multiattribute decision making". *Journal of Mechanical Design, Transactions of the ASME,* ***126***(3), pp. 412–419.

[4] Ferguson, S., Lewis, K., Siddiqi, A., and de Weck, O., 2007. "Flexible and reconfigurable systems: Nomenclature and review". In Proceedings of the ASME 2007 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Vol. 6, pp. 249–263.

[5] Koren, Y., 2013. "The rapid responsiveness of RMS". *International Journal of Production Research,* ***51***(23-24), pp. 6817–6827.

[6] Mehrabi, M., Ulsoy, A., and Koren, Y., 2000. "Reconfigurable manufacturing systems: Key to future manufacturing". *Journal of Intelligent Manufacturing,* ***11***, pp. 403–419.

[7] Barbarino, S., Bilgen, O., Ajaj, R., Friswell, M., and Inman, D., 2011. "A review of morphing aircraft". *Journal of Intelligent Material Systems and Structures,* ***22***(9), pp. 823–877.

[8] Yim, M., Duff, D. G., and Roufas, K., 2000. "Polybot: A modular reconfigurable robot". In Proceedings of the IEEE International Conference on Robotics and Automation.

[9] Mattson, C. A., and Messac, A., 2003. "Concept selection using s-pareto frontiers". *AIAA Journal,* ***41***, pp. 1190–1198.

[10] Literman, B., Cormier, P., and Lewis, K., 2012. "Concept analysis for reconfigurable products". In Proceedings of the ASME 2012 International Design Engineering Technical Conferences & Computer Information in Engineering Conference, Vol. 3, pp. 209–222.

[11] Simpson, T., Maier, J., and Mistree, F., 1998. *The Information Revolution: Present and Future*. Ablex Publications, Greenwich, Connecticut, ch. Mass Customization in the Age of Information: The Case for Open Engineering Systems.

[12] Siddiqi, A., de Weck, O., and Iagnemma, K., 2006. "Reconfigurability in planetary surface vehicles: Modelling approaches and case study". *JBIS - Journal of the British Interplanetary Society,* ***59***(12), pp. 450–460.

[13] Patterson, M. D., Pate, D. J., and German, B. J., 2011. "Performance flexibility of a reconfigurable family of UAVs". In AIAA Aviation, Technology, Integration and Operations (ATIO) Conference.

[14] Ferguson, S., and Lewis, K., 2008. "Investigating the interaction between reconfigurability and system mass using multidisciplinary design optimization". In AIAA/ASME/ASCE/AHS/ACE Structures, Structural Dynamics, and Materials Conference.

[15] Ferguson, S., Kasprzak, E., and Lewis, K., 2009. "Designing a family of reconfigurable vehicles using multilevel multidisciplinary design optimization". *Structural and Multidisciplinary Optimization,* ***39***(2), pp. 171–186.

[16] Simpson, T. W., 2004. "Product platform design and customization: Status and promise". *Artificial Intelligence for Engineering Design, Analysis and Manufacturing,* ***18***, pp. 3–20.

[17] Fellini, R., Kokkolaras, M., Papalambros, P., and Perez-Duarte, A., 2005. "Platform selection under performance bounds in optimal design of product families". *Journal of Mechanical Design,* ***127***, pp. 524–535.

[18] Rey, L., and Clavel, R., 1999. *The Delta Parallel Robot*. Springer, ch. 29, pp. 401–417.

[19] Courteille, E., Deblaise, D., and Maurine, P., 2009. "Design optimization of a delta-like parallel robot through global stiffness performance evaluation". In International Conference on Intelligent Robots and Systems, pp. 5159–5166.

[20] Lipkin, H., and Patterson, T., 1992. "Geometrical properties of modelled robot elasticity: Part I - decomposition". In 22nd Biennial Mechanisms Conference, Vol. 45, pp. 179–185.