# *Notes from the RSS'16 Workshop on Minimality & Design Automation*

*Dylan A. Shell*

*September 4, 2016*

## 1 Introduction and Scope

The workshop held in Ann Arbor on June 18th as part of Robotics Science & Systems (RSS) 2016 interwove two broad research themes: **minimality** — which involves questions about resources, such as, how to represent and reason about them, and what are the fewest resources needed for a robot to carry out a task successfully — and **design automation** — encompassing questions of how computation can be used to ease the robot design process, including both hardware and software elements, often with particular attention paid to correctness and techniques for avoiding *ad hoc* trial-and-error processes, and all this encompasses. The workshop was organized around the idea of bringing together researchers who, each having given considerable thought to some subset of topics, were interested in the connections between, and interplay of, ideas. Prior efforts to bring about cross-pollination between the respective topic areas were deemed by the organizers to have been only piecemeal at best.

Interest in the topic of the workshop should be taken as a positive sign, it is indicative of progress. As was expressed at the event: 'Before we've been so happy about a robot doing anything at all.'

In order to make substantial progress with design problems we need computers to help. It isn't exactly clear how, yet. Decisions that a practitioner might make at design-time center around a few concerns: we desire an understanding of the interplay between problem requirements, task constraints, relationships and trade-offs between resources and capabilities. A question of primary importance is: How do we use computers and software to help us arrive at this understanding? For example, are there algorithms that can efficiently establish the relationships between a particular sensor's fidelity and the time to task completion? While this is undoubtedly feasible if the setting is sufficiently narrow, a secondary question is how computers to imbue computers with broader knowledge about such trade-offs so they can use it to productively. Thus, there if concerns how computers can *produce* such insights, and the second is how computers might *make use* of such knowledge.

### 1.1 Guiding questions

At the workshop an initial list of questions were posed to those in attendance in an attempt to define some sort of circumcenter for the discussion. They were written on four boards and remained visible for the entirety of the event:

1. Which parts of the design process are ready to be, or have already been, distilled into crisp formalisms? Which parts of the process are not quite there yet, but are within striking distance?
2. Which parts are amenable to automated reasoning?
3. What algorithmic tools and representations are most suitable?
4. What will future design tools for robots look like, and how will they interact?

There was considerable variability in the amount of discussion devoted to these four. This is likely, at least in part, because they deal with questions at different levels of fundamental detail. Items 1 and 3, for example, pose questions assuming the idea of some formalism as representation—this hypothesis was challenged early in the workshop. Also, item 1 is about the present, whereas item 4 is clearly a speculation about the future. As will be clear in the next section, the workshop saw broader questioning and disagreement than would fit within the confines bounded by the preceding questions.

## 2 Ideas

A word of warning to the reader: the organization of these ideas in a totally cohesive fashion still remains allusive at this point.

### 2.1 Modeling, prototyping, and rapid iteration

An interesting and important difference of opinion arose rather quickly and became the scene of some sparring. Broadly speaking there were those who wish to emphasize modeling as part of the design process and those who emphasized building. Good natured discussion arose from such statements as, on the one hand:

> "Don't model things, build robots."
> "There is no need to build good robots."
> "We need to build bad robots."

And on the other:

> "Absolutely model things! Then build robots..."
> "Ability to model is the one advantage we have over biology.'
> "Person *X* was wrong about this."

I briefly summarize the positions.

*Don't model* — The basic premise is that design should aim for fast iteration, and the process itself should seek to achieve evolvability. In some domains it is extremely challenging to identify all the factors that influence robot behavior and, in attempting to tease these factors apart, one discovers something akin to a fully connected graph: a variety of the dimensions are coupled and resist decomposition and modularization. In some extreme instances (like insect scale flying robots) there remain fundamental limits to our understanding. There are too many unknown aspects about, say, the phenomenology of fluid flow around wings to us to begin to pose the inverse problem of optimizing wing design in this space.

Tenets:

– Start with a previously existing design and iterate on it.
– Believe in the designer. Novices are generally bad at saying what they want, but they can succeed even if they don't know the details of what they're doing. Human guidance and intuition help cut through complexity (including overcoming issues for which there are hardness results).
– Design automation is valuable because it decreases latency in getting to the robot you can try out. The vision is to make a much more direct, tangible connection between a robot and a design; the way this currently being realized is in attempting to enable on-the-demand creation of custom printable electomechanical systems. Any automation is better than no automation in this regard, aim to create a design tool for a novice.

'The perfect is the enemy of the good'

'If at first you don't succeed, try, try, try again. . . '

*Absolutely model* — Substantial advances in science have come from separating causes from effects in a way that allows chains of deduction to reach sound conclusions, so that knowledge can be gained without the cost of actual demonstration in the real world. There is a great deal of structure in design problems: we typically use a narrow set of available hardware components, there are units for quantifying functionality provided by components, and there are languages for providing functional specifications in classes of tasks (and these continue to grow in richness). Once knowledge is "symbolized" so that it can be captured, it becomes re-usable through the formation of libraries and tools. Models provide a way to give expression to assumptions and guarantees; they are also the starting point of a language of operations (e.g., such as composition, refinement, compression) to achieve higher-levels of competency whist managing complexity.

Tenets:

- To move beyond once-off solutions for single points in the design space we need to capture the lessons learned and any understanding gained. Formalisms are the appropriate way to do this. When you can express knowledge about something in a way that can be used by a computer, you understand that thing.
- Automation operates on models: algorithms, computer assisted tools, etc., depend on grounding to actual robots. A substantial part of the robotics research enterprise ought to be characterizing which models are useful and under what precise circumstances.

Given that the preceding statements appear to be mainly about general robot design, it is worth considering how these positions pertain to minimality. The 'don't model' perspective has an affinity for simple robots in at least two ways. First, robots still employ simple programming in regions of the design space where extreme resource limitations prevail, typically as a consequence of weight, scale, or similar constraints. Such robotic research projects may explore how compliance, springs, or other mechanics can be used to realize functionality. As such, these robots often inhabit parts of the design space, or even physical regimes, where analytical methodologies remain deficient. Secondly, techniques for fabrication and rapid manufacturing have quite mixed levels of maturity; overall they still result in fairly rudimentary robots.

'Minimal means that I don't think of imposing what I want to do on the world. You can do that, but it'll cost you. So minimalism compels you to go with what the physics wants to do.'

The modeling-based view has its own affinity. Prior work in minimality has attempted to understand the interplay between problem requirements and task constraints. It has also sought to understand relationships and trade-offs between resources and capabilities. Even if the emphasis has been narrow, focused inquiry of some of these aspects, its primary tool has been formalism. The bridge to formalisms suitable for automated reasoning, and hence the purview of those interested in algorithmic exploration of the design space, runs along a road demanding few swerves. It does however, beg the question that serves as the title for the next section.

### 2.2 *What should design formalisms capture?*

It certainly would be desirable to be able to prescribe *necessary* and *sufficient* conditions for a robot to carry out a given task. Such theory would give a scientific foundation for robotics. Though the rhetoric and claims of current modeling approaches and formalisms appear to mirror these ambitions (for example, in talk of the use of effective, principled methods), in reality most are tool- and technique-centric

'What does it mean to say something is *robotable*? Like when we say something is computable, what is the analogue to the Turing machine?'

perspectives for which understanding remains highly localized. These current approaches strongly bias sufficiency over necessity.

*What questions to ask and sorts of answers to expect:*

In standard terminology, a *design problem* is the posing of some problem for which a design is the answer; this is the way this term is used herein. Two other related questions were posed during the workshop:

(1.) The *resource minimization problem* asks, given some desired functionality, what are the fewest resources required?

(2.) Its co-problem is the *functionality maximization problem*, which asks, given a set of resources, what is the maximal functionality that can be provided?

Neither (1.) nor (2.) require explicit specification or visualization of the design-space *per se*; they don't demand constructive means in order to answer the queries, though it remains unclear how they can be met otherwise. As both are questions of an optimization form, answers have a relation (an ordering or partial ordering) with respect to all elements of a feasible set. Thus, implicit, is some universal quantification.

Optimization problems can be posed as decision problems, but this is usually less useful as typically we are interested in the result from '$\arg\min$' or '$\arg\max$' operators.

Interesting and meaningful questions can be posed with an existential quantifier too. For example, (1.) can be rephrased to ask if the desired functionality can be realized with any of the resources available. The question, in this guise, can be posed even if there infinitely powerful (or many) sensing resources. It's not clear that (2.) can be posed in an analogous way.

*How to ask such questions:*

A view shared by multiple attendees is that design questions involve some specification expressed in some choice of language(s) to describe assumptions and guarantees. Additionally, or within this form, is a catalogue or library of components.

A second, distinct view (by attendees in the pro-model camp) was that a design for a robot itself can provide a specification of intended behavior; the idea is that this design is correct but is potentially resource sub-optimal. Operations then produce a better design, perhaps with a smaller (or even optimal) resource footprint.

*Scoping and framing considerations:*

Search, optimization, satisfaction, or similar problems are posed over a set of variables that make up a space. Important work is involved in characterizing robot systems, or components thereof, and identifying the parameters which describe performance, cost, etc. Effort spent thinking about how to search in this space should lead appreciation the value of such characterizations and not to overlook it. It is worth illustrating via examples given at the workshop. For the class of binary manipulators, the important quantity needed to understand their performance is *workspace density*, an dimensionless quantity characterizing that might be seen as an abstract parameter describing such a resource. When such a parameter depends on relations among other variables, it serves to describe an iso-surface of functionality. Thus, it provides regularity to the space of resources, potentially reducing the dimensionality of the problem.

Another example in this vein was the proposal was that "iterability" might be formalized to express a crucial constraint. In such a way, additional constraints induce structure some algorithm could exploit.

*Conjectures, hypotheses, and proto-theories:*

Speakers made a variety of different statements, a few of these stood out as sufficiently distinct from the thrust and parry of standard academic conversation to be listed as claims here.

*Invariant complexities:* For some class of problems, it may be better to think about complexity as 'volume' in a space. One can move that complexity around, by putting more in mechanical design, or in the control law/algorithm, or structure in the workspace, or number of robots. But ultimately it is conserved. (This was illustrated with the example of very a simple robot within a structured environment: an MRI machine.)

*Convex and concave curves, and the design sweet spot:* Performance as a function of mechanical actuators makes a rapid increase for the first few actuators, but then levels off to a moderately flat slope. Computational complexity makes an approximately convex curve: it is challenging to process sensors and command actuators effectively when resources are either seriously impoverished, or when there are so many that it becomes a scaling complication. If we consider the contributions of both of these aspects, this can give a sweet spot for moderate numbers of actuators.

*Operations:* In addition to composition of elementary pieces, consideration of refinement operations that break meta-objects into basic constituents should be studied. With both composition and refinement, one obtains a structure (to aid in algorithmic efficiency) but also flexibility.

## 2.3 *What would an design process look like?*

A discussion early in the workshop concerned the tractability of algorithms that can determine the fewest resources for given specifications. Some precise problems of interest are NP-hard, even to approximate, even for some natural special cases, for example. Several responses and counter-claims were made almost immediately. These included:

'Algorithms for general-purpose robot design problems would be great but all of the interesting questions are NP-hard.'

- We're not compelled to be afraid of such hardness results—nor necessarily undecidability either. Several problems, indeed many, fall into these classes yet they are solved practically every day. Powerful heuristics have been discovered that do well on many instances of interest. And in many cases "optimal" isn't needed.
- We should trust designers to used their intuition and higher-level view to deal with complexity. The problem becomes one of ensuring a design process that uses human input in the right ways.

This last point emerged as an important discussion point for which a variety of opinions were expressed.

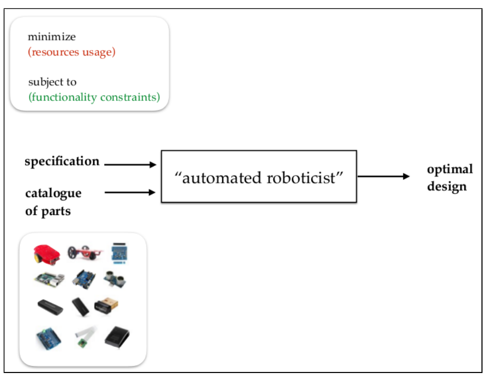


Figure 1: One vision for an automated design process wherein a *catalogue of parts* (also called by some a library of components) forms a store of reusable background knowledge. (Source: Slide of Andrea Censi)

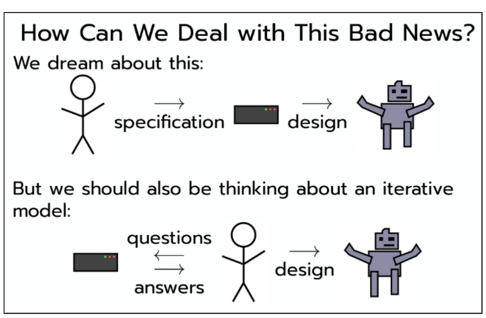


Figure 2: Motivated by 'bad news' in the form of hardness results which show the intractability inherent in automating certain aspects of design, an alternative, iterative process is proposed. (Source: Slide of Jason O'Kane)

Figures 1–3 show hypothetical design processes proposed during the workshop. As should be immediately clear there are a variety of choices. It is noteworthy, though, that despite the discussion about building rather than modeling robots, all figures have arrows that

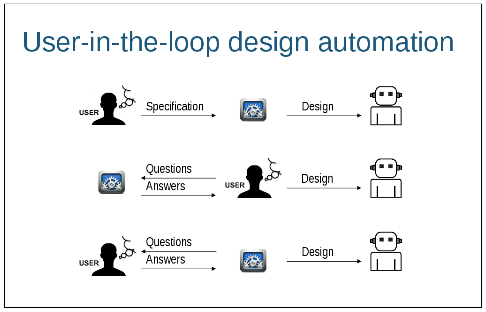


Figure 3: A further alternative has design tools posing questions to the user. (Synthesized from slides of Ankur Mehta, with minimal editing).

terminate in a robot; none incorporate a loop to express feedback or improvements gained from use of a physical robot.

The third option in Figure 3 exchanges the user and software system, which alters the source and destination of the 'Q' and 'A' arrows. This alteration may also be achieved in an additional, distinct way: consider a fourth version which retains the user and software arrangement of the second option, but then exchanges the 'Q' and 'A' labels. Thus fourth version has preserved the source of the arrow labelled 'Design'. How meaningful this combination would be depends on a precise understanding of what Q-ing and A-ing actually is. It probably isn't useful to think of interacting with a spreadsheet as a Q&A process, while in some software for tax-return preparation, for example, such a mode might be quite apt. A richer metaphor might be that of surveying since the person is partly envisioned as using the tools to probe the space of designs, help him or her to see greater distances by clearing away fog of uncertainty in that space. The design tools have aspects of various optical instruments: a telescope gives hyperopic capabilities enabling one to see further, a sextant improves precision in measurements that give relationships between elements, etc.

'Design automation may never produce robot designs, but instead be a Q&A process of design.'

## 3 *Looking forward*

The original workshop proposal had given the intended outcome as reaching a clearer understanding of the questions listed in Section 1.1. It suggested that an ideal answer to each question would include a list of instances (e.g., of problems and known solutions, "folklore results", etc.) along with associated references. This remains as work yet to be done.

More successfully, the work intended to 'get a sense of which those in the minimality and automation communities support the broad vision [of bringing automated techniques to bear on resource questions]'. In this regard, the workshop must be seen as having achieved favorable outcome. Not only was there considerable interest, but it was clear that there was agreement about scientific importance of this vision.

We believe attendees were attracted to workshop on the basis their own scientific motivation. The associated robot design game may also have played a rôle. One may be able to attract more attention by highlighting the applicability the ideas being discussed. Especially minimality is generally poorly motivated in this way, though it does have practical significance for manufacturing costs; for questions of whether a robot with some failed sensors (for example) can still perform a task, perhaps with reduced performance; and for questions about self-replication in robots. Finally, some researchers may be doing work in this space without realizing the relevance. One speaker gave an example of a 'target curve' and how numerical techniques were used to optimize a linkage in order to get it to match the curve. It was the audience who convinced him he was doing a form of synthesis to realize a specification.

## 4 *Directions for further improvement*

Subsequent workshops should be aware of the following weaknesses noted of this workshop:

- Duration: The original website had suggested a half-day workshop. Though the organizers included the Robot Design Game from the outset, it later became apparent that this left too little time for discussion of ideas. The half-day thus grew to become a full-day effort; this 'bait-and-switch' wasn't communicated adequately, and inconvenienced some speakers.

- Wrap-up Discussion: The Robot Design Game was an engaging, entertaining (and marketing) triumph. It did, however, make it difficult to bring the event to a clear and crisp close. One typically envisions a wrap-up discussion or panel that collects, organizes, and summarizes ideas, topics, and threads of discussion. The workshop could have been improved by a more thoroughgoing attempt to synthesize earlier talks, questions, speculations and debate.

- Attendees and Organizers: The diversity of participants, along several dimensions, left considerable room for improvement.