Evolutionary Fabrication: The Emergence of Novel Assembly Methods in Artificial Ontogenies

John Rieffel
DEMO Lab, Brandeis University
Waltham, MA
jrieffel@cs.brandeis.edu

Jordan Pollack
DEMO Lab, Brandeis University
Waltham, MA
pollack@cs.brandeis.edu

ABSTRACT

Evolutionary Design Systems (EDSs) have demonstrated the ability to generate a wide array of novel objects, including robots, tables, and antennas. Often, the novelty of these evolved designs is due to their ability to discover and exploit important principles of the design space, such as the truss and the ratchet. One current obstacle to the real-world application of such EDSs is that they often create purely *descriptive* representations, and are therefore capable of generating designs whose specific assembly is difficult, if not impossible, to infer. One solution that we offer is to evolve *how* to build, rather than *what* to build. When evolution occurs in *assembly space* rather than design space, only buildable objects are produced. Furthermore, as we demonstrate in this paper, doing so allows for the emergence not just of novel designs, but of novel *means of assembly.*

Categories and Subject Descriptors

I.2.9 [**Artificial Intelligence**]: Robotics

General Terms

Design,Algorithms

Keywords

Artificial Ontogeny, Evolutionary Design, Assembly, Fabrication

1. INTRODUCTION

Evolutionary Design Systems (EDS) have been used to create a wide range of unique and novel objects, ranging from tables and robots to satellite antennas. One notable advantage of Evolutionary Design Systems (EDS) is their ability to novel, elegant, and unexpected solutions to the design task. Notable elegant solutions to evolutionary design problems include the emergence of the counterweight and the truss [4], and the discovery of the ratchet as a means for locomotion [10].

The continuing success of EDS means that more evolved designs are beginning to be transferred into the physical world for real applications. Perhaps the most notable example of this is Lohn *et al*'s evolved satellite antennas [9].

An obstacle to the physical assembly of evolved designs is that the end result of EDS is usually a direct representation, such as a blueprint. Direct representations specify only what to build, but contain no information on how, or even if, the evolved design can be physically assembled. Further work, often requiring significant human insight, is then needed to infer some sequential physical assembly process which can yield the specified design.

One way to avoid this extra work is to evolve *prescriptive* representations rather than descriptive ones. Prescriptive representations, such as recipes, or assembly plans, describe *how* to build rather than what to build. This bottom-up approach to design avoids the task of assembly inference. As an added benefit, EDSs which evolve prescriptive representations, by definition, only produce *buildable* designs. Furthermore, since the end result is a sequence of assembly instructions, physical assembly of evolved assembly plans can be fully automated.

The process by which a physical structure is assembled can often bear similarities to organic growth. Artificial Ontogenies, a form of Evolutionary Algorithm which takes inspiration from the biological processes of growth and development, are therefore a useful lens through which to explore the evolution of assembly.

The evolution of prescriptive representations requires simulating an object's entire assembly, rather than only its final shape, a process which we call *Situated Development.* In this context, the environment in which evolution occurs is meant to be equivalent to some physical assembly mechanism, such as a rapid prototyping machine.

In this paper we explore how EDSs which operate in the domain of assembly design (rather than the domain of direct design) are capable of generating not only novel assemblies, but novel *means* of assembly as well.

2. BUILDABILITY

In this work, since our interest is in *automating* real-world assembly, by saying a structure is *buildable* we mean it can be produced by a *sequential*, *situated*, assembly process. By

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
GECCO'05, June 25–29, 2005, Washington, DC, USA.
Copyright 2005 ACM 1-59593-097-3/05/0006 ...$5.00.

sequential, we mean that the process of assembly must be composed of discrete steps - we discuss sequentiality in more detail in Section 3. By situated, we mean that assembly occurs in a real-world environment, not *in utero*, as we discuss in Section 5.

3. DESCRIPTIVE REPRESENTATIONS AND THE ASSEMBLY INFERENCE PROBLEM

Although the field of EDS has been around for quite awhile, only recently have evolved designs begun to be transferred to the real world. As we have discussed in the past [12, 13], one obstacle to this transition to the real world is that most EDSs produce *direct* representations, such as blueprints, of the evolved design. As such, further work is required in inferring an assembly sequence to construct the object. In most cases neither the inference of assembly nor the subsequent execution of the assembly is performed automatically - rather, both tasks fall on the shoulders of a human. Thus, while the evolution of descriptive representations removes human effort from the design task, it fails to remove human effort from the assembly task - and may in fact increase it.

Our interest is in automating not just design, but assembly as well. Doing so will therefore require either automating this inference task or, alternatively, circumventing it entirely.

3.1 Assembly Sequence Planning

In the field of engineering, the task of inferring a sequence of assembly instructions given a particular structure *a priori* is known as *Assembly Sequence Planning*, or *Assembly Sequencing* and is a rather richly studied topic. Although the process of determining an assembly sequence may come readily to humans, it often much harder to computationally solve, and has in fact been demonstrated to be NP-complete in the general case [7].

Computational approaches to sequence planning for a given object usually involve much easier inverse problem of *disassembly planning* - that is, exploring all the ways of removing parts one at a time [5]. Doing so, however, makes the critical assumption that every stage of assembly is both *reversible* and *symmetric* [5].

In the context of EDS, in order for an evolutionary system to produce a descriptive representation of an object whose assembly sequence can be reasonably inferred, the design search space must be limited to objects whose assembly contains the properties of reversibility and symmetry. And even within those constraints, the problem is demonstrably rather complex [5, 7]. Needless to say, without those constraints descriptive EDSs are capable of generating designs of objects whose assembly sequence is, optimistically, extremely difficult if not impossible to produce automatically.

Anyone who has taken apart a home appliance and then put it back together, only to be left with a solitary remaining mysterious screw, knows that assembly and disassembly are rarely symmetric, reversible processes in the real world.

4. DIRECTLY EVOLVING ASSEMBLY SEQUENCES

Faced with such difficulties in inferring an assembly sequence from an evolved descriptive representation, it might be better to avoid the problem entirely - by evolving assembly sequences (that is, *prescriptive* representations) directly. Every structure produced by an evolved assembly sequence is, by nature, and by our definition of the term in Section 2, *buildable*

Moreover, the language of assembly no longer needs to be constrained to reversible and symmetric operations. As a result, the domain of *buildable* evolved structures is greatly expanded - because they no longer need to be *unbuildable* as well. It is this last aspect that we would like to explore the consequences of in this paper. By directly evolving assembly sequences, and unconstrained to reversible and symmetric assembly methods, EDS can arrive at not just novel objects, but novel means of assembling those objects.

Of course, there is no "free lunch" in this process. By allowing evolutionary design to range over the entire space of assembly sequences, we greatly increase the search domain, and other methods may be needed to intelligently constrain the search.

As a final advantage, however, consider that evolved assembly sequences can be directly interpreted by a compatible manufacturing system, and so allow for fully automated design *and* assembly, offering the prospect of removing the human from the loop entirely.

5. ARTIFICIAL ONTOGENIES AND SITUATED DEVELOPMENT

The process by which an object is manufactured, particularly those assembled automatically, can often bear similarities to organic growth. Evolutionary Design based upon Artificial Ontogenies [8], which use biological growth and development as metaphors for physical assembly, are therefore a suitable lens through which to explore these issues of buildability and sequence planning. (Since we are using biological growth as a metaphor for manufacturing, we will use the terms *assembly* and *development* interchangeably.)

However, most EDSs based on Artificial Ontogenies take the actual assembly process for granted, either by allowing virtual structures to appear *ex nihilo* - that is, out of thin air - or else *in utero* - in a very simplified environment, significantly less complex than the real world environment that their physical counterparts are to be assembled in. Hornby's tables, for instance, were originally drawn as OpenGL voxels in 3-D space, and were not subject to gravity as they were assembled, only once they were completed [6].

By contrast, we use the term *Situated Development* to refer to an Artificial Ontogeny which simulates the entire process of assembly, not just the final evaluation. The most notable example of Situated Development in recent literature is Bongard's Gene-Regulatory Networks [1], which slowly "grew" a robotic morphology piece-by-piece in a realistic physics environment.

Complex representations such as Bongard's Gene Regulatory networks have been demonstrated to be quite effective for design, but do not easily present themselves as a means of describing automated manufacture. Since our interest is in producing assembly sequences which can be automatically interpreted by a manufacturing system, we will evolve a simple type of assembly sequence directly.

Our Situated Artificial Ontogeny is therefore analogous to an automated design and manufacturing system. The genotypes of our system will be linear assembly sequences, and our development environment will be analogous to a manufacturing plant which interprets and executes those assembly

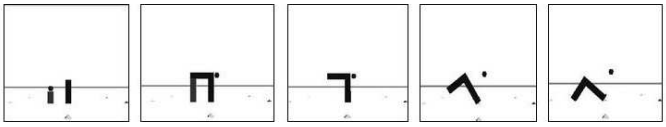

Figure 1: Assembly has three stages. In the first, both permanent and temporary bricks are placed. In the second, adjacent permanent bricks are glued together, and scaffolding is removed. Finally, the remaining structure settles.

sequences.

6. A FRAMEWORK FOR EXPLORING ASSEMBLY SEQUENCE DESIGN

Our Situated Development environment is based upon the Open Dynamics Engine (ODE) [1] the widely used open-source physics engine, which provides high-performance simulations of 3D rigid body dynamics.

Assembly is performed by a LOGO-like turtle, acting as a print head, capable of movement in the X-Z plane, and of depositing 2x1x1 bricks in the environment. When strung into a sequence, commands to the turtle (move, rotate, put brick, take brick) form an *assembly plan*. Commands which would cause the turtle to move outside the target area, or place a brick where a brick already exists, are ignored. The speed of an ODE simulation is heavily influenced by the number of objects being simulated. Consequently, the maximum number of objects placed by any assembly plan was limited to 25.

Since our recent work has demonstrated the ability of similar systems to "discover" scaffolding implicitly during the course of evolution [11, 12], we chose to allow for the *explicit* placement of scaffolding. The turtle is capable of placing two kinds of bricks: permanent ones (shown as black in the animation frames), and temporary ones(shown in gray), which are removed once the assembly is completed. This aspect is analogous to similar features of modern rapid prototyping machines, which can lay thin water-soluble support structures that are dissolved once manufacture is complete.

Our simulated assembly falls into three stages (Figure 1). In the first, the turtle interprets the assembly plan, moving and placing bricks as directed. In this stage, each brick is a separate component in the environment, subject to gravity and interactions (such as collisions) with other objects. Once assembly is complete and the structure is stable, the scaffolding is removed and adjacent bricks are glued together (but not to the floor). Finally, once the scaffolding is gone, the final structure is allowed to come to a rest before being evaluated.

7. EXPERIMENTS

We begin by describing the emergence of novel assembly methods in earlier results of ours from [13], and then explore a variation of our setup, which allows for an increase in these phenomena.

7.1 Set Up

[1] www.ode.org

Figure 2: The Goal Arch. Each legs consists of two vertical bricks, whereas the center section consists of three horizontal bricks. Note, therefore, that the center bricks are not resting on top of the legs, but are instead cantilevered off their side - as a consequence, until the glue phase they cannot remain in place without scaffolding.

The genotypes of our Evolutionary Algorithm were assembly plans, consisting of a sequential set of parameterized instructions to the situated development system described above.

Rather than using a single fitness function, we used Evolutionary Multi Objective Optimization (EMOO) [2] over a set of objectives, which are described in more detail for each experiment. (For the sake of brevity, since we have previously discussed the particulars of the algorithm used in more detail[13], and since the topic of this paper is independent of the particular EA used, we will omit the algorithmic details here.)

7.2 Novel Assembly Sequences for a Goal Structure

As a preliminary example, we first consider an EDS in which the goal is to find a suitably efficient assembly plan for a pre-specified goal structure. This is, in a sense, automating the task of Assembly Sequence planning, but from the bottom-up. We begin, therefore, by evolving assembly plans capable of building a pre-determined goal structure, in this case an arch (Figure 2).

In order to compare each resulting structure to the goal structure, a simple bitmap was generated by sampling the central 100×100 region (bricks are $10 \times 10 \times 20$) in the X-Z plane. This bitmap was then compared to a corresponding bitmap of the goal structure.

The specific objectives used were as follows (in each case, smaller values are considered more fit)

- Length of Assembly Plan
- Mass - number of bricks in the entire world, not just the sample region.
- Number of bits missing from goal structure
- Number of "wrong" bits - i.e. either extraneous or missing bits.

The first two objectives exist for more than just deterrence against bloat, per [3]. Rather, they also reward assembly

plans for efficiency in terms of time (the length objective) and in terms of material (the mass objective). Physical prototyping machines are slow, and require expensive material - therefore any reduction in print time or print material is highly valuable.

7.2.1 Results

Figure 3 shows animation frames from a representative evolved solution. (Full color images of all results, as well as animations, are available at the author's web page [2]). Discovered after roughly 2000 generations and with a length of 22 instructions, it is able to perfectly generate the goal structure. By comparison, we were unable to produce a hand-built assembly plan shorter than 29 instructions.

This efficiency is due largely to the novel placement of the vertical scaffolding used to hold up the center section of the arch. Each vertical scaffolding brick is placed directly under the center of mass of the brick it supports. This placement location exists between two of the discrete print-head positions, and so could not have been placed directly. Rather, it is dropped horizontally onto the leg sections and subsequently topples vertically into its final location. In fact, if it had been placed directly into one of the adjacent positions, it wouldn't have been under the supported brick's center of mass, and the supported brick might have tilted sideways.

This toppling of the central scaffolding is a good example of a non-reversible assembly sequence - the scaffolding's final location is a result of the material's interaction with its environment, rather than purely a direct result of an assembly instruction. Any disassembly sequence of the arch would not be able to produce a matching inverse of the action.

7.3 Novel Assembly in Open Ended Design

While the task of discovering an assembly sequence for a goal structure is important, the true strength of Evolutionary Design lies in more open-ended domains.

Our open-ended design task in [13] was to create a structure which maximized the total *open* volume beneath a structure, thereby encouraging structures which both maximize height and maximize the number of empty spaces beneath the structure.(Figure 4)

In this particular setup, the range of the turtle was limited to the same box as fitness was measured in (box A in the Figure 4).

The length and mass objectives are retained from the experiment is Section 7.2, and the fitness function above replaces the two goal-based objectives from 7.2.

- Length of Assembly Plan(minimizing)
- Mass of Structure(minimizing)
- Shaded Area (maximizing)

7.3.1 Results

As discussed in [13], the majority of structures evolved in this context were stable, static arches, which did not demonstrate particularly novel assembly methods. However, figures 5 and 6 show some novel assembly mechanisms which did arise in this setup, and so caught our eye.

In Figure 5, although the initial structure(Frame 1) is not particularly fit (with a fitness value of 10%), after the assembly phase, once scaffolding is removed and the remaining structure is glued together, the larger section is no longer stable, and topples sideways (Frames 2 and 3), ultimately coming to a rest balanced on the smaller section (Frame 4). This resulting "T" shape is much more fit (a fitness of 49%), and is produced more efficiently this way than by using scaffolding to prop the cross-piece of the "T" into place.

[2]www.cs.brandeis.edu/~jrieffel/situated-development/

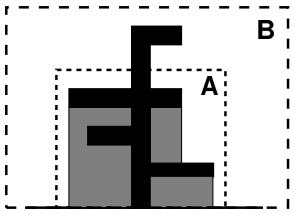

Figure 4: Illustration of the two Development Environments. In each case the fitness function is measured over the smaller box (A) and within that box the gray regions under the black structure is considered "shaded". Note that the uppermost overhang does not contribute any shade, because it exists outside of the fitness bounds. In the first environment (Section 7.3), the turtle is limited to the same box (A) as the fitness measure, whereas in the second (Section 7.4), the turtle may range in the larger box (B).

Figure 6 shows a similar result. In this case, the original structure has a fitness of 22%. Once the permanent bricks are glued and scaffolding is removed, the larger structure on the left tips to the side, knocking over the smaller structures in the process. Once it comes to a rest, the smaller structures help prop it into this cantilevered shape with a fitness of 52%.

These are both examples of novel assembly: capable only of placing bricks one at a time, and so lacking any formal ability for modular assembly of larger components, the assembly plans, evolved in the context of Situated Assembly, have nonetheless "discovered" how to construct two separate modules, and then join them in the final phase of assembly. We conjecture that in each case the process used in creating the final structure is more efficient than a purely sequential process evolved in a simpler environment without gravity or momentum.

7.4 Encouraging Novel Assembly

We then made one small change to the environment. Rather than limiting the turtle to the same box that fitness was evaluated over, it was allowed to range over a larger, 200×200 box (labeled B Figure 4). Otherwise, all of the objectives, and the 25 brick limit, remain unchanged.

This slight adjustment allows the turtle to place bricks outside of the fitness box which then fall into it during the final settle phase of development. In this case, the maximum possible fitness increases to 2250, because the although the very top row of the fitness box needs to be covered by a row of bricks, that row can itself be supported by legs which are outside of the fitness range.

7.4.1 Results

Unlike the setup in Section 7.3, which produced relatively few examples of such dynamic assembly, this small change

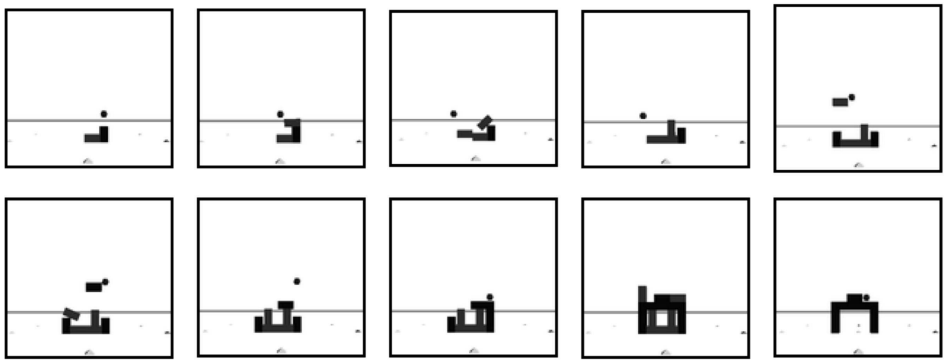

Figure 3: Building the Goal Arch. Note how the horizontal scaffolding placed in frame 3 tumbles into a vertical position to support the top of the arch. This is repeated with the piece of scaffolding placed in frame 5. (frames are read left to right, top to bottom - solid bricks are black, scaffolding is grey, the small sphere is the location of the print-head. The horizontal line is the horizon)

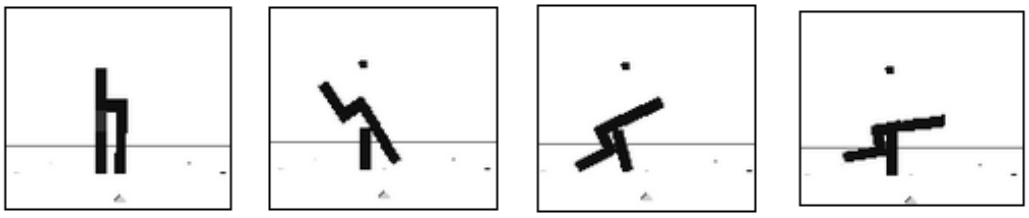

Figure 5: A novel assembly method discovered in the first open ended setup (Section 7.3) . Once the assembly is complete (Frame 1), scaffolding is removed and remaining bricks glued together (Frame 2), the larger section topples onto the smaller section, balancing there to form a T. This resulting shape has significantly higher fitness (49%) than the original structure (10%)(Frame 1)

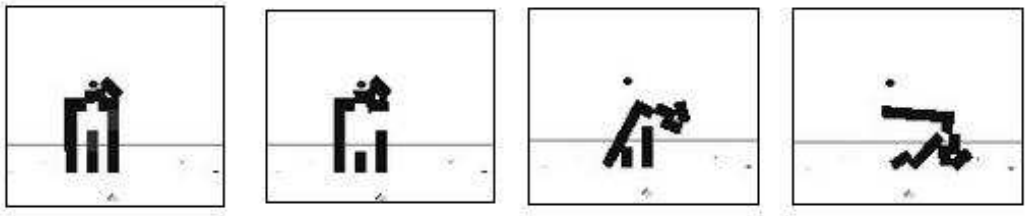

Figure 6: Another novel assembly process from the setup in Section 7.3. The original structure has a fitness of only 22%. Once scaffolding is removed and remaining bricks glued, the leftmost portion tumbles rightward, and the smaller segments below are knocked sideways, ultimately serving to prop up the larger shape. This final structure has a fitness of 52%)

resulted in a significantly higher number of such phenomena. Figures 7 thru 10 all show examples of assembly methods which arise under this variation.

As can be inferred by the increased number of structures which make use of this dynamic assembly, the evolutionary algorithm quickly discovers solutions which can take advantage of this small change - that is, solutions in which bricks placed outside of the fitness box subsequently fall into it during the "settle" phase.

Note that in each of the examples given, there is a significant increase in fitness between when scaffolding is removed and when the final structure settles. We will focus on the last of these examples, Figure 10. Small black spheres have been added to the figure to denote the top corners of the fitness bounds.

As can be seen, the initial structure has near zero fitness, because the overhanging brick is outside of the fitness bounds, and so does not contribute any "shade"(per Figure 4). Once scaffolding is removed, the leftmost structure becomes unbalanced, and so topples onto the smaller section. Once the larger section comes to rest, it forms an almost perfect "T" shape. This particular process is similar to the one shown in Figure 6, but the larger range of the turtle allows it to construct much larger pieces, which can then topple into a much higher position.

8. DISCUSSION

In the context of our situated development system, the evolutionary incentives for novel assembly methods arise largely from the formulation of the fitness objectives, which reward for efficiency. The length objective, for instance, encourages assembly plans to be short (efficient in time), whereas the mass objective encourages assembly to be light (efficient in material). These, combined with the built-in absolute maximum of 25 bricks, necessitate a certain parsimony in order to generate highly fit solutions.

This emphasis on efficiency leads to the discovery of several novel means of assembly, in several different contexts.

Consider the placement of scaffolding in the assembly process shown Figure 3, when building the goal structure in Section 7.2, for instance. The toppling lands the scaffolding in a position where it couldn't be directly placed by the turtle, and so allows the assembly plan to support the central arch bricks with fewer bricks, and fewer instructions that it might otherwise need.

Also consider the assembly procedure in Figure 10. With a shade percentage of 80%, is not maximally fit - and yet, it manages to generate this fitness using only 17 instructions and 14 bricks (including the solitary piece of scaffolding). All of our hand designed solutions with higher fitness contained many more than the maximum allowable 25, and were longer that 40 instructions.

Both of these phenomena - the toppling placement of the scaffolding and the large-scale dynamic assembly, are examples of *non-reversible* assembly steps. They are, therefore, the exact kind of process that is unavailable to traditional top-down Assembly Sequence Planning.

9. CONCLUSION

The evolution of descriptive represention may be the most widely used in the field of Evolutionary Design Systems, but from the perspective of real-world assembly, it carries with it the burden of requring subsequent effort to generate a suitable assembly sequence. This assembly sequence inference task, if automated via Assembly Sequence Planning, occurs backwards, from the perspective of *disassembly*, and so is constrained to assembly techniques which are reversible and symmetrical.

Directly evolving assembly sequences offers a means around this subsequent effort. Furthermore, as we have shown, since evolved assembly sequences aren't constrained to reversibility and symmetry, the process allows for the generation of novel and efficient means of assembly.

In other words, evolving *what* to build runs the risk of designing objects which you might not be able to build, whereas evolving *how* to build allows you to build objects which you might not otherwise design.

The other advantage of evolving assembly sequences is that the results can be directly interpreted by an autonomous assembly mechanism, and so the process lends itself to the full automation of both design and assembly. Such Integrated Automated Design and Assembly, as it might be called, offers itself greatly to the realms of both industrial design and extraplanetary exploration.

10. REFERENCES

[1] J. Bongard and R. Pfeifer. *Morpho-functional Machines: The New Species (Designing Embodied Intelligence)*, chapter Evolving complete agents using artificial ontogeny, pages 237–258. Springer-Verlag, Berlin, 2003.

[2] C. A. C. Coello. An updated survey of evolutionary multiobjective optimization techniques: State of the art and future trends. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, editors, *Proceedings of the Congress on Evolutionary Computation*, volume 1, pages 3–13, Mayflower Hotel, Washington D.C., USA, 6-9 1999. IEEE Press.

[3] E. D. De Jong, R. A. Watson, and J. B. Pollack. Reducing bloat and promoting diversity using multi-objective methods. In L. Spector, E. Goodman, A. Wu, W. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. Garzon, and E. Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2001*, pages 11–18, San Francisco, CA, 2001. Morgan Kaufmann Publishers.

[4] P. Funes. *Evolution of Complexity in Real-World Domains*. PhD thesis, Brandeis University, Dept. of Computer Science, Boston, MA, USA, 2001.

[5] M. Goldwasser, , J. Latombe, and R. Motwani. Complexity measures for assembly sequences. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 1581–1587, Minneapolis, MN, Apr. 1996.

[6] G. S. Hornby and J. B. Pollack. The advantages of generative grammatical encodings for physical design. In *Proceedings of the 2001 Congress on Evolutionary Computation CEC2001*, pages 600–607, COEX, World Trade Center, 159 Samseong-dong, Gangnam-gu, Seoul, Korea, 27-30 2001. IEEE Press.

[7] L. E. Kavraki, J.-C. Latombe, and R. H. Wilson. On the complexity of assembly partitioning. *Information Processing Letters*, 48(5):229–235, 1993.

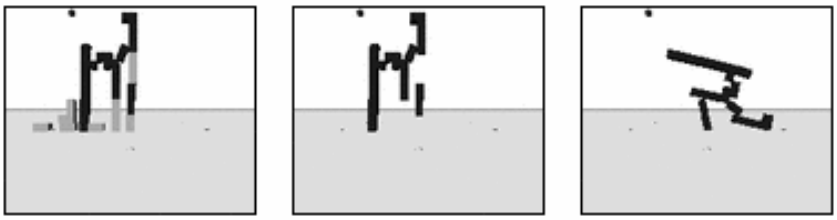

Figure 7: Dynamic assembly from Setup B (Section 7.4. This assembly plan contains 38 instructions and results in a structure with a mass of 17. Initial fitness is: 44%. Final fitness: 77% - a 70% increase.

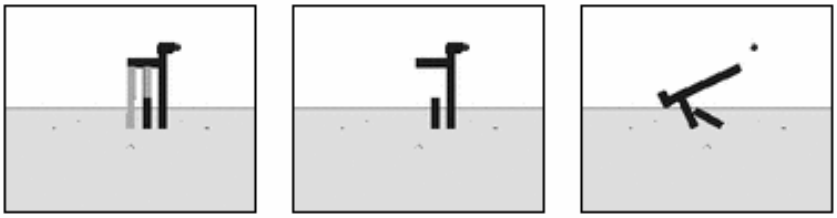

Figure 8: Another dynamic assembly sequence from Setup B. This is 22 instructions long, with a final mass of 10. Initial fitness is 31%, final fitness is 47%, a 52% increase.

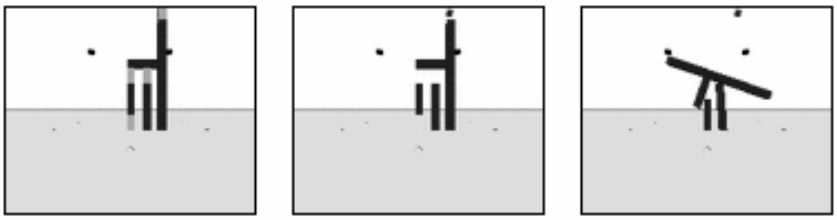

Figure 9: Another dynamic assembly sequence from Setup B. 20 instructions, 14 mass. Initial fitness: 27%, Final Fitness: 62%, a 129% increase. The black spheres mark the upper corners of the fitness bounds.

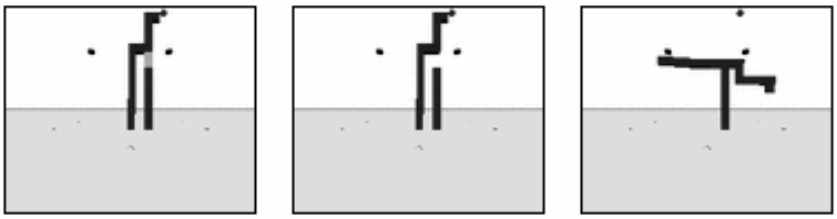

Figure 10: The most extreme example of dynamic assembly from Setup B. With only 17 instructions, and a mass of 13. Initial fitness is 0.4% (no typo), Final fitness is 80%. The black spheres mark the upper corners of the fitness bounds. The near-zero initial fitness is due to the fact that the overhanging brick is outside of the fitness bounds, and therefore contributing no shade.

[8] S. Kumar and P. J. Bentley. *Advances in evolutionary computing: theory and applications*, chapter Computational embryology: past, present and future, pages 461–477. Springer-Verlag New York, Inc., 2003.

[9] J. D. Lohn, G. S. Hornby, and D. S. Linden. An Evolved Antenna for Deployment on NASA's Space Technology 5 Mission. In U.-M. O'Reilly, R. L. Riolo, T. Yu, and B. Worzel, editors, *Genetic Programming Theory and Practice II*. Kluwer, in press.

[10] J. B. Pollack, H. Lipson, P. Funes, S. G. Ficici, and G. Hornby. Coevolutionary robotics. In A. Stoica, J. Lohn, and D. Keymeulen, editors, *The First NASA/DoD Workshop on Evolvable Hardware*, pages 208–216, Pasadena, California, 19-21 July 1999. Jet Propulsion Laboratory, California Institute of Technology, IEEE Computer Society.

[11] J. Rieffel and J. Pollack. The Emergence of Ontogenic Scaffolding in a Stochastic Development Environment. In K. D. et al., editor, *Genetic and Evolutionary Computation–GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part I*, pages 804–815, Seattle, Washington, USA, June 2004. Springer-Verlag, Lecture Notes in Computer Science Vol. 3102.

[12] J. Rieffel and J. B. Pollack. Artificial ontogenies for real world design and assembly. In M. B. et al., editor, *Ninth International Conference on the Simulation and Synthesis of Living Systems (ALIFE9) Workshop: Self-Organization and Development in Artificial and Natural Systems (SODANS)*, pages 37–41. MIT Press, 2004.

[13] J. Rieffel and J. B. Pollack. Situated development: Using artificial ontogenies to evolve buildable 3-d objects. In *Genetic and Evolutionary Computation–GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. (to appear)*, 2005.