WebAL Comes of Age: A Review of the First 21 Years of Artificial Life on the Web

Tim Taylor*,**
University of York

Joshua E. Auerbach†
Ecole Polytechnique Fédérale de Lausanne

Josh Bongard‡
University of Vermont

Jeff Clune§
University of Wyoming

Simon Hickinbotham**
University of York

Charles Ofria††
Michigan State University

Mizuki Oka‡‡
University of Tsukuba

Sebastian Risi§§
IT University of Copenhagen

Kenneth O. Stanley***
University of Central Florida

Jason Yosinski†††
Cornell University

Abstract We present a survey of the first 21 years of web-based artificial life (WebAL) research and applications, broadly construed to include the many different ways in which artificial life and web technologies might intersect. Our survey covers the period from 1994—when the first WebAL work appeared—up to the present day, together with a brief discussion of relevant precursors. We examine recent projects, from 2010–2015, in greater detail in order to highlight the current state of the art. We follow the survey with a discussion of common themes and methodologies that can be observed in recent work and identify a number of likely directions for future work in this exciting area.

Keywords
WebAL, web technologies, HTML5, open science, crowdsourcing, crowdfunding, public outreach

* Contact author.
** University of York, UK. E-mail: tim@tim-taylor.com (T.T.); simon.hickinbotham@york.ac.uk (S.H.)
† Ecole Polytechnique Fédérale de Lausanne (EPFL), Laussane, Switzerland. E-mail: joshua.auerbach@epfl.ch
‡ University of Vermont, USA. E-mail: jbongard@uvm.edu
§ University of Wyoming, USA. E-mail: jeffclune@uwyo.edu
†† Michigan State University, USA. E-mail: ofria@msu.edu
‡‡ Department of Computer Science, Graduate School of SIE, University of Tsukuba, Japan. E-mail: mizuki@cs.tsukuba.ac.jp
§§ IT University of Copenhagen, Denmark. E-mail: sebr@itu.dk
*** University of Central Florida, USA. E-mail: kstanley@cs.ucf.edu
††† Cornell University, Ithaca, NY, USA. E-mail: yosinski@cs.cornell.edu

 Artificial Life 22: 364–407 (2016) doi:10.1162/ARTL_a_00211

1 Introduction

In recent years there has been a growing body of work in artificial life (ALife) that makes use of web technology in one way or another.

Over the last five years or so, web technologies have shifted away from proprietary browser plug-ins and towards standardized, native application programming interfaces (APIs) for providing graphics, animation, multimedia, and other advanced features. This progress is due to the development and adoption of the HTML5 language and associated API specifications introduced by the World Wide Web Consortium (W3C).[1]

This movement has made it much easier to develop and deploy rich web-based applications that work reliably and consistently on any browser, across multiple platforms and devices. It is therefore unsurprising that a number of high-profile ALife projects have emerged over this period that utilize web technology in various different ways. We refer to such work as *WebAL*, and broadly construe the field to include the multitude of ways in which ALife and the web might intersect. Examples include the creation of massively distributed user-guided evolutionary systems, the creation of open science platforms, the use of web-based applications for public outreach and engagement, and the use of crowdfunding platforms for supporting the development of ALife systems. As we demonstrate in this review and summarize in Section 7, there are many other points of intersection in addition to these.

In light of this emerging trend, an inaugural Workshop on Artificial Life and the Web (WebAL-1)[2] was held at the 14th International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14) in New York City on 31 July 2014 [128]. Inspired by the success of the workshop, a number of its participants decided to collaborate on writing a comprehensive review of the field—the result of which is the current article.[3]

Although recent years have witnessed a rapid growth in this area, the first web-based ALife systems date back 21 years to the mid-1990s, with various antecedents employing alternative forms of network technology dating back to the 1970s. In Figure 1 we present a timeline showing some of the main WebAL projects discussed in this article, plotted against developments in the underlying web technology. We will refer to this figure throughout the review.

The organization of the rest of the article is as follows. In Section 2 we define the scope of this review and describe the methodology adopted to collect the materials upon which the review is based. In Section 3 we review various non-web or non-ALife projects that have heavily influenced the state of the art of modern-day WebAL. In Section 4 we cover the first true WebAL systems that appeared in the 1990s soon after the birth of the web itself, and in Section 5 we explore developments in the following decade, the 2000s. In Section 6 we look in slightly more detail at WebAL systems that have appeared since 2010, in order to present the current state of the art. On the basis of the work reviewed, we identify some of the important emerging themes in Section 7, and comment on likely directions for future research as well as projects in active development. Finally, we present our conclusions in Section 8.

2 Review Scope and Methodology

As described in Section 1, we have approached this review with an open-minded perspective on what constitutes WebAL, in order to examine the many different meeting points of ALife and the web. Although such an approach allows us to address a variety of topics that might not have

1 See http://www.w3.org/TR/html5/ and https://en.wikipedia.org/wiki/HTML5.

2 http://alife.org/ws/weba11

3 This article is a vastly expanded and more detailed revision of a previous review [127], with additional contributions from many of the participants and organizers of the WebAL-1 Workshop.

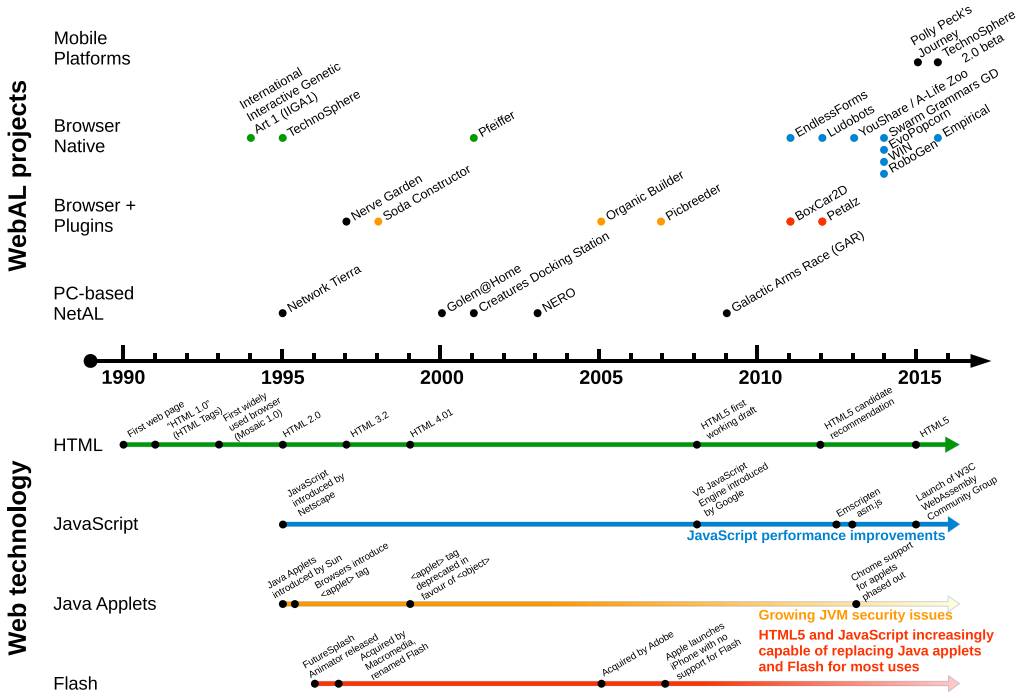


Figure 1. A timeline of selected WebAL projects in relation to developments in web technology. The color of the dot for each project indicates the main technology used (green: basic HTML and CGI scripts; blue: JavaScript; orange: Java; red: Flash; black: other).

been covered in a more narrowly focused review, there is a danger that it might lead us into vast areas of research where there is no hope of providing a comprehensive, coherent review that would be of interest to readers of this journal. This danger is compounded by the fact that the boundaries of ALife research are indeterminate, and the distinction between web applications and other forms of Internet application is also becoming increasingly blurred (particularly in the case of mobile apps and social media). To guard against these dangers, we have taken a number of measures to enhance the focus of the review and to provide a clear methodology.

To determine what counts as ALife in this context, we considered work that describes itself as ALife, or is published in the ALife literature, as a de facto definition of the field. Specifically, we searched for relevant work among all articles ever published in publications affiliated with the International Society for Artificial Life—that is, every issue of the *Artificial Life* journal, and every proceedings volume of the International Conference on Artificial Life and the European Conference on Artificial Life.[4] We acknowledge that other ALife-related journals and conferences exist, but we needed to draw a line somewhere in order to keep the task manageable. The initial search yielded approximately 450 potentially relevant articles, but many of these were spurious results. These articles were then investigated in more detail in order to determine which were of genuine relevance. In addition to this systematic search, we also tapped into the collective knowledge of the coauthors of this article (all of whom are actively working in WebAL), as well as asking a number of other colleagues associated with the field for feedback on an early draft.

Furthermore, we have attempted to draw a distinction between work relating to the web and work related more generally to the Internet. We use the terms *WebAL* and *NetAL*, respectively, for

4 Where electronic copies of articles were available, searches were performed using the following keywords: (*web, internet, Java, applet, HTML, Flash, browser, Android, iOS, screensaver, JavaScript*). Where no electronic copies were available, abstracts were read to determine potential relevance.

the narrower and broader fields. The major focus of this review is on WebAL, although it has not always been sensible to draw a clear distinction here. In cases where we feel that specific NetAL work is especially relevant, we have included it in our review. In particular, we highlight some of the most relevant NetAL work in Box 1 and Box 2. For completeness, any other NetAL work that was uncovered in our systematic search of the *Artificial Life* journal and conference proceedings is listed in a bibliographical appendix.

Having established our criteria for what work to review, we chose particular aspects of this body of work to focus on. In terms of current work, we are primarily concerned with modern HTML5 and associated APIs as a *platform*, an *environment*, and an *enabler* for ALife. We are especially interested in how the modern web enables new ways of working that were previously not possible or feasible—at the end of the review we highlight the most important of these in Section 7. In our review of earlier work, we are primarily interested in pioneering work that used ALife and the web in novel ways, and in tracing the important antecedents to today's WebAL projects. There are other areas of work that we have chosen not to focus on because they are large fields that would have taken us too far from our core topics, and because comprehensive reviews of those areas are available elsewhere. These include the general areas of web crawling and information retrieval, distributed multi-agent systems, and frameworks for distributed evolutionary algorithms. We do, however, discuss specific work in these fields that have a particular ALife-oriented focus, and we provide pointers to broader reviews of these areas where appropriate.

3 The Precursors to WebAL

Many of the WebAL projects discussed in later sections involve the use of network technology and distributed human interaction to produce some kind of emergent artefact. Seen in this general light, a wide variety of potentially relevant antecedents can be found. We cannot provide an exhaustive review of such a broad body of prior work, but instead highlight a few significant landmarks and waypoints.

The arts world provides many interesting studies of the combination of network technology, communication, emergent behavior, and inhabited virtual worlds. In 1977, the artists Kit Galloway and Sherrie Rabinowitz exhibited the *Satellite Arts* project, with funding and technical support from NASA [26, 84]. The project was a live performance piece involving two sets of dancers, located on opposite coasts of the USA, connected by a satellite-linked video feed that blended the images of both sets of dancers to produce a live, shared virtual space in which the performance happened. The project was one of the first examples of the use of network (in this case, satellite) technology to explore distinctions such as real versus virtual, subject versus object, and here versus there. In the years since the *Satellite Arts* project, the arts world has continued to push the boundaries of technology to explore issues of networks, communication, virtuality, and emergent behavior.[5]

Besides the arts world, computer games have an equally long history of development of shared virtual worlds. Real-time, shared virtual worlds date back to the development of the original *Multi User Dungeon* (*MUD*) by Roy Trubshaw in 1978 [83]. *MUD* initially allowed multi-user play via ARPANet before being licensed to CompuServe,[6] where it ran until 1999. The *MUD* source code has recently been acquired by Stanford University Library as part of an ongoing effort to preserve virtual worlds [58]. *MUD* spawned a diverse and distinguished lineage of massively multiplayer online games (MMOs) that still thrives today [83].

The 1970s witnessed the emergence of computer viruses on mainframes, and by the early 1980s viruses and worms were also appearing on microcomputers [125]. One of the first examples of a worm that spread via the Internet, causing widespread damage and attracting the attention of the mainstream

5 To briefly mention just two more recent examples: Karlheinz Stockhausen's *Helikopter-streichquartett* (Helicopter String Quartet), first performed in 1995, involved a coordinated live performance of four musicians each flying in separate helicopters [122]. Elsewhere, Matthew Fuller's group performance project *The Human Cellular Automaton*, first performed in 2000, implemented cellular automata rules (such as Conway's *Game of Life*) in a crowd of human participants [133, pp. 173–174], thereby creating a distributed human-powered computer.
6 http://british-legends.com/CMS/index.php/about-mud1-bl/history

media, was Robert Morris' Internet worm of November 1988 [23]. Referring to the growth of computer viruses, Christopher Langton warned in his introduction to the proceedings of the Second Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems (*Artificial Life II*, 1990):[7]

> There is a caution here that we all must attend to. Attempts to create Artificial Life may be pursued for the highest scientific and intellectual goals, but they may have devastating consequences in the real world, if researchers do not take care to insure that the products of their research cannot 'escape,' either into computer networks or into the biosphere itself. [50, p. 18]

Although the general history of computer viruses is beyond the scope of this review, we will refer back to Langton's warning when we discuss recent developments with client-centric WebAL in Section 7.7. For further discussion of the broader topic of computer viruses and their relation to ALife, we refer the interested reader to Eugene Spafford's review article [114].

Relevant antecedents to WebAL can also be found in the development of distributed evolutionary systems, with theoretical work on parallel genetic algorithms starting in the 1960s and implementations in the 1980s—see [14] for a good review. One of the most ambitious examples of a distributed evolutionary system is Karl Sims' *Evolved Virtual Creatures* [110], released in 1994, which ran on a Connection Machine CM-5 supercomputer across 1024 cores. In this simulation, Sims fully evolved the morphology and behavior of a population of virtual organisms in a 3D world, inspiring many subsequent systems. Furthermore, Sims' gallery installations of interactive evolutionary art, such as *Genetic Images* (1993)[8] and *Galápagos* (1997),[9] were important precursors for WebAL evolutionary art systems such as *Picbreeder* (discussed in Section 5.1.1) and later projects (Section 6.1).

Huge advances have been made in the more general field of distributed computing over the last couple of decades. The WebAL projects discussed in later sections have developed in the context of noteworthy successes in several major scientific projects using distributed computing and Internet crowdsourcing, including *SETI@Home* [5], *Foldit* [46], and *Galaxy Zoo* [56]. The continued advances in large-scale distributed systems over this period have also stimulated the growth of the new field of autonomic computing, which aims to use bio-inspired techniques to produce large-scale, self-managing distributed IT systems that are substantially more robust than traditional complex computing systems [45].

In this section we have only briefly touched upon some of the most relevant precursors to WebAL. By the early 1990s, the technological milieu was pregnant with many of the ideas outlined above. As the Internet matured and the World Wide Web was born, such ideas were invigorated by the development of easy-to-use, standardized network technology and the mushrooming uptake of Internet and Web technology not just by special interest groups but by the general public the world over. In the next section, we look at the earliest examples of work that truly deserves the label WebAL.

4 The 1990s: Early WebAL

One of the first examples of WebAL was Michael Witbrock and Scott Neil-Reilly's evolutionary art project *International Interactive Genetic Art 1* (*IIGA1*), developed in 1994, along with the subsequent *IIGA2* system developed by John Mount [134].[10] In this work, the fitness of images generated by genetic programming representations was determined by taking the average ratings for each image as scored by users accessing the system via the web. Both *IIGA1* and *IIGA2* attracted tens of

7 Similar concerns were also raised by Harold Thimbleby, who presented at the Artificial Life II workshop but did not appear in the proceedings; his ideas were later published elsewhere [129].

8 http://www/karlsims.com/genetic-images.html. Like Evolved Virtual Creatures, this work also ran on a massively parallel Connection Machine supercomputer.

9 http://www/karlsims.com/galapagos/

10 See also http://www.win-vector.com/blog/2009/06/what-is-genetic-art/.

Box 1. Highlights of other NetAL work from the 1990s.

The *Network Tierra* project, initiated around 1995, was a networked version of Tom Ray's well-known system *Tierra*, in which the digital organisms were native, self-reproducing code structures running in a virtual computer. The goal was to use the Internet to create a world-wide, distributed, complex environment in which the digital organisms could roam and freely evolve. Project participants ran specially designed servers that ran *Tierra* locally and managed network communications with other *Tierra* servers in a secure and contained manner.[a] Over a period of 5 years, Ray and coworkers used *Network Tierra* to investigate the evolution of complexity in parallel programs (their analogy to multi-cellular organisms). Results were mixed, but the project ultimately failed to achieve an evolutionary increase in the number of differentiated parallel processes (cell types) [96].[b]

A more applied variation of the ideas behind *Network Tierra* can be seen in Filippo Menczer and colleagues' 1995 article on evolving agents for information retrieval on the web [64]. This work proposed an algorithm where agents searched for documents driven by a user's query, gaining energy and reproducing according to their success. The algorithm was tested on a small set of 116 connected web pages, although this early work did not employ actual on-line agents. A full review of the field of online information retrieval agents is beyond the scope of this article, but a summary of work in this field in the 1990s can be found in [47].

Elsewhere, in 1996 David Ackley presented a description of his *ccr* system [1], which has been under development intermittently since 1989.[c] *ccr* is an Internet-based software system that aimed to provide a large-scale, distributed common environment in which humans and artificial agents could communicate and interact. Some further analysis of the system was published in 2000 [2], and the system's key server, launched in 1996, is still running today.[d] Although according to Ackley many of the lessons learned from the system have not yet been presented in the literature, the project has influenced his more recent research on indefinitely scalable computing [3, 4].

Another early example of the use of the Internet for distributed communication is provided by Luc Steels and colleagues' long-running *Talking Heads* project—initiated in 1999—which studied the development of language in communicating robots. In these experiments, physical robots were situated in various locations around the world, and virtual agents (i.e., the controllers for the robots) could "teleport" over the Internet to inhabit different physical bodies. The public could also interact with the agents via the project's website [120, 121].

A final example from this period is the work of Johnson and colleagues in 1998 [42], which presented an early discussion of the potential of the Internet to facilitate large-scale collaborative problem solving and self-organization of knowledge. As discussed in Section 7, these concepts form a core thread of a significant portion of more recent WebAL work.

a http://life.ou.edu/tierra/netfaq.html
b Subsequent work with the *Avida* system was able to generate the spontaneous evolution of differentiated cell types, but these experiments were run on local clusters, not on the web [33].
c http://keys.ccrcentral.net/ccr/history/
d http://keys.ccrcentral.net/ccr/data/central-stats-v0.3+.html

thousands of users, and the work was mentioned in *Wired* magazine [115]. The *IIGA2* site, built upon CGI scripts and HTML forms,[11] racked up nearly 4000 generations (each of 10 images) on its original server, with over 115,000 page views of the images. The use of the web as a means of collecting distributed aesthetic evaluation of 2D and 3D images in evolutionary art systems remains a strong component of WebAL work to this day, as will be highlighted in later sections.[12]

Moving from 2D to 3D, the year 1995 saw the launch of the web-based artificial life virtual world *TechnoSphere*, created by Jane Prophet and Gordon Shelley [87]. The front end of the system

11 http://www.mzlabs.com/MZLabsJM/page4/page4.html
12 The work on web-based evolutionary art systems reported here is part of a larger field of work on evolutionary art, much of which is not web-based. A good review of the wider field can be found in [52].

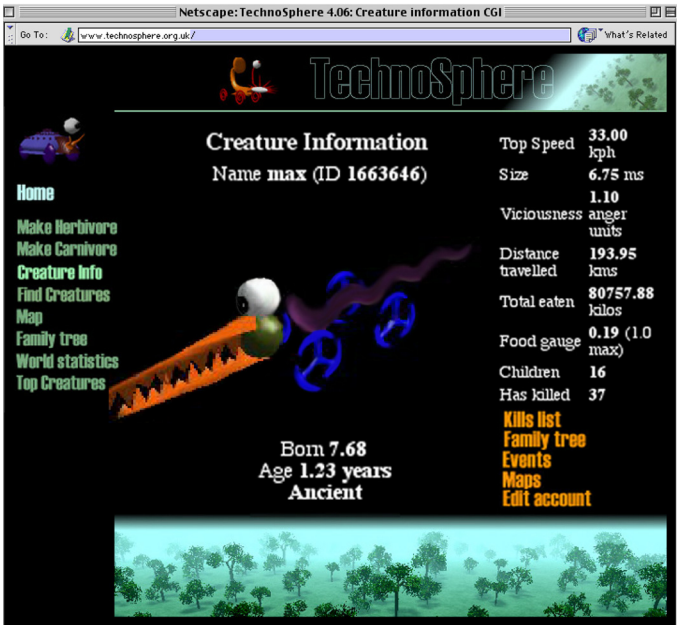


Figure 2. The *TechnoSphere* web interface.

was a website where users could design their own creatures by selecting from a limited range of predesigned body parts (see Figure 2). Once created, the user submitted their creature to the web server, and it was tagged with the user's email address and a unique ID. Submitted creatures were released into a 3D virtual world (which was not rendered live on the website), featuring a fractally generated landscape and other creatures (many of which were designed by other users), subject to ecological rules that governed their interactions. At key events, the users would receive email updates. For example, when creatures interacted with each other, the email addresses of the two authors were shared, to facilitate discussion. Users could even request "postcards" of their creatures, which were generated by rendering a scene showing the creature in its current location. In 1996 the *TechnoSphere* world reached a peak population of 90,000 creatures. In 1998, work started on a version with real-time 3D rendering [88], which was exhibited at a number of art galleries and museums over the period 1999–2001; this version, however, ran on a local network of PCs rather than on the web.[13]

An early example of using the web to mix real and virtual worlds—and thereby addressing some of the same topics investigated by the pre-web *Satellite Arts* project described in Section 3—was *Telegarden*.[14] Building upon their work on web-based tele-operated robotics [32], Ken Goldberg and colleagues created a system in which web users could tend to a remote real-world garden, using a tele-operated robotic arm to plant and water seedlings. *Telegarden* ran successfully for 10 years (1995–2004), amassing 9000 active users by August 1996[15] and attracting coverage from many mainstream media channels. A sociological study of the community of users was reported in [62].

Bruce Damer and colleagues' *Nerve Garden*, launched in 1997, shared some of the ideas of *Telegarden*, but swapped the mixed-reality aspect for a shared 3D virtual environment [19]. The system allowed users to grow 3D plant models, generated by L-systems, on a client-side Java application. Users could then select their favorite plants, name them, and submit them to a central server, where

13 At the time of writing, a new version of *TechnoSphere*, in the form of an augmented-reality mobile app, is currently under development (see Section 6.2.2).

14 http://goldberg.berkley.edu/garden/

15 http://web.archive.org/web/19980203215817/http://telegarden.aec.at/html/nyt.html

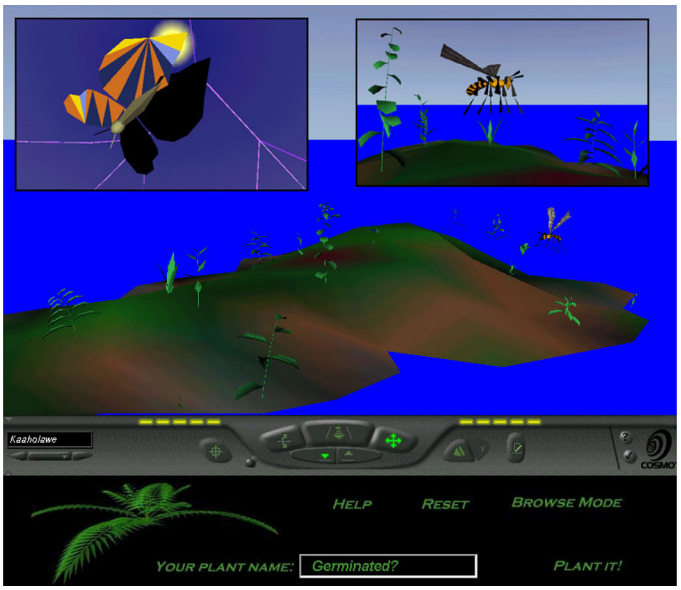


Figure 3. *Nerve Garden*. Composite image of the *Nerve Garden* Island with interface, butterfly in the sunshine, and lightning storm and bee in flight over the island.

they would be planted in a shared VRML-based[16] 3D environment that could be viewed by anyone with a browser suitably equipped with a VRML viewer plug-in (see Figure 3).

By the mid-1990s, projects such as *Telegarden* were attracting growing attention in the popular media. In some cases, the rapidly expanding media interest in cyberculture even led to funding for new WebAL projects. Following the publication of Kevin Kelly's book *Out of Control* [44], Absolut Vodka (sponsors of Kelly's website) funded the development of a web-based genetic art system called *Absolut Kelly*, built upon the ideas of distributed intelligence and the "hive mind" discussed in the book. The system, developed by Jeffrey Ventrella, used an interactive genetic algorithm to allow users to evolve images featuring the distinctive outline of Absolut's vodka bottles.[17]

Extending the hive mind approach to evolving art even further, and taking its name from the sci-fi novel *Do Androids Dream of Electric Sheep?* [24], Scott Draves' *Electric Sheep* is a distributed, interactive artwork system launched in 1999 [25].[18] The software exists in the form of a screensaver that can be downloaded by users and used to render frames of abstract artworks. Once complete, these frames are sent back to a central server, where they are used to generate animations (known as "sheep"). The animations are compressed by the server and sent back to the client-side screensavers for display (the system is therefore not web-based as such, but relies on Internet communications between the server and client-side screensavers). Users can vote for their favorite animations, their votes being used by a genetic algorithm to guide the future evolution of new sheep. The sheep are therefore the result of massively distributed computation and human participation from tens of thousands of users.

While genetic art projects such as *Electric Sheep* and the *Absolut Kelly* employed web users purely for the aesthetic selection step of the evolutionary process,[19] the shared virtual environment projects such as *TechnoSphere* and *Nerve Garden* provided users with tools to directly manipulate the design of

16 http://www.w3.org/MarkUp/VRML/

17 Example images can be seen at http://www.ventrella.com/Tweaks/Absolut/absolut.html. The work is also described in [133, 45] and on Kevin Kelly's website: http://kk.org/ct2/2007/10/17/mutating-art-from-chaos/.

18 http://electricsheep.org. See also [133, pp. 154–157].

19 Another early example of this kind was the *Artificial Painter* project, a genetic art system that included a limited web-based interface allowing user-guided aesthetic selection (archived at http://web.archive.org/web/19980503002715/http://gracco.irmkant.rm.cnr.it/luigi/lupa_ap.html) [81].

their contributed virtual inhabitants. In *TechnoSphere* this manipulation was accomplished by the composition of standard elementary parts, and in *Nerve Garden* by providing an interface to adjust various parameters of the L-system.

A somewhat more abstract approach, in which users used a web interface to enter written text that was then treated as a genetic code and mapped to a 3D creature, was developed by Laurent Mignonneau and Christa Sommerer in a series of projects they developed in the late 1990s [112]. The first of these projects was *Life Spacies*, introduced in 1997 and followed by *Life Spacies II* in 1999 [69]. This project was an interaction environment installed in a museum in Tokyo and connected to a website through which users from all over the world could design virtual creatures by entering text messages that would then be mapped into 3D creatures and introduced into the environment displayed at the museum. Once uploaded to the shared environment, the creatures could interact with each other there. Furthermore, visitors to the museum could interact with the creatures by touching them,[20] thereby blurring the boundary between real and virtual worlds (a recurring theme that has come up in various of the projects already discussed). A related web-based system, *Verbarium*, was also introduced in 1999, and allowed users to create shapes and forms in real time using the same idea of a text-to-form encoding and an online interactive text editor [112, 113].

In addition to arts projects, the mid-1990s saw the birth of many WebAL-related projects in the fields of computer games and animation.

1996 saw the release of the ALife focused game *Creatures*, designed by Steve Grand.[21] The characters in the game were digital life forms, called *Norns*, that were capable of lifetime learning, and possessed a physiology, drives, and communication abilities, all of which could evolve over generations. Although the first version of the game ran on standalone PCs, a growing online community of players soon started exchanging their *Norns* via enthusiast websites [16, 41]. By 2001, *Creatures Docking Station* was released, an Internet-based add-on to *Creatures 3* that allowed *Norns* to travel between different online worlds.[22]

The growing popularity of the Java programming language in the mid-1990s (Figure 1), and the ease with which it allowed programs to be distributed via applets embedded in web pages, led to a flourishing of websites hosting ALife-related applets. In 1996, Craig Reynolds ported his well-known *Boids* work, which had originally been published in 1987 [97], to a Java applet on his website.[23] Various other *Boids* applets also appeared around the same time.[24] One particularly noteworthy example was *Floys* by Ariel Dolan,[25] which extended the basic *Boids* design by adding territorial behaviors to the creatures [80]. *Floys* (along with *Boids*) was described in a popular science article in *Scientific American* in 2000, which highlighted the potential for amateur scientists to modify the code and use it for their own investigations [15].

An example of a more extensive ALife-related Java application was developed by the British design group Soda Creative in 1998. Their system, *Soda Constructor*,[26] employed a 2D physics engine and presented users with an online editor with which they could construct creatures based upon mass-spring systems with oscillating muscles. By mid-2000, the popularity of the game had soared through "word of email," and an online forum enabled users to share their creations.[27] Soda Creative won an Interactive Arts BAFTA Award in 2001 for their work.[28] In 2002, they teamed up with

20 This interaction was achieved using a camera tracking system that captured a visitor's image and projected it into the display, thereby integrating them into the virtual environment [69].

21 http://en.wikipedia.org/wiki/Creatures_(artificial_life_series)

22 See http://creatures.wikia.com/wiki/Docking_Station. The information in this paragraph has been verified by personal communication with Steve Grand.

23 See http://www.red3d.com/cwr/boids/applet/. Date of original publication confirmed by personal communication with Craig Reynolds.

24 For example, Conrad Parker's implementation http://www.vergenet.net/~conrad/boids/. Various other examples are listed at http://www.aridolan.com/ad/Alife.html.

25 http://www.aridolan.com/ofiles/JavaFloys.html

26 http://soda.co.uk/work/sodaconstructor

27 http://www.acmi.net.au/soda.htm

28 http://awards.bafta.org/award/2001/interactive/interactive-arts

Queen Mary University London to develop *Sodarace*, a shared online environment where users from around the world could pit their creations against each other in competitions [63].[29] The development of *Sodarace* was supported by the UK's Engineering and Physical Sciences Research Council, and had a strong public outreach and educational flavor.[30]

Elsewhere, an early example of work employing a distributed web-based genetic algorithm was reported in 1998 by Jens Astor and Christoph Adami [6] (with further experiments reported a couple of years later [7]). The work investigated the evolution of developmental artificial neural networks (ANNs), and utilized a server program written in Java that farmed out evaluations of individual ANNs to heterogeneous clients running a browser-based Java applet (or a local desktop Java program).[31] The authors saw the potential of web-based distributed architectures for running computational tasks at a massive scale. However, their publications only reported results from experiments run over a local network; they cited Java's lack of speed as a problem, but they foresaw that improvements in Java compiler technology would make this kind of architecture more feasible in the future.[32]

As demonstrated in the preceding discussion, by the late 1990s WebAL was already a fertile and vibrant field of research. A short review article entitled "ALife Meets Web: Lessons Learned," published in 1998, summarized some of the work discussed in this section and drew some conclusions about what might be the most productive areas for WebAL research going forward [80]. Areas highlighted included using the web as a shared testing ground—a global laboratory—for ALife experiments, and using the close parallels between the web and natural biological environments (e.g., both are large, dynamic, heterogeneous, noisy, and distributed) to inspire the design of complex WebAL worlds. As we will see later on (Section 7), both of these areas have become important components of current WebAL research.

5 The 2000s: WebAL Develops

We begin this section by describing a rather different kind of intersection between ALife and the web that took place at the Artificial Life VII conference in 2000, held in Portland, Oregon. One aspect of the conference theme of "Looking backwards, looking forwards" was a drive to better understand the interests and opinions of the community regarding the status and direction of ALife as a field of study. In addition, the organizers wished to canvass the community on the establishment of a professional society. In order to achieve these tasks, Steen Rasmussen and colleagues set up a web-based survey that allowed open-ended responses to a number of questions formulated by a group of community members [93]. The survey was implemented using Active Server Pages[33] technology, and stored responses in a database for further analysis. The survey authors produced histograms and mind maps from the results in order to better understand the interests and concerns of the community, and also to investigate differences in opinion between respondents from computer science and biology. In addition, the positive responses to the questions regarding the establishment of a professional society led to the formation of the International Society for Artificial Life. In reporting the survey design and analysis, Rasmussen et al. explained that they wished to use the web to harness the collective intelligence of the ALife community. In addition to reporting the specific results of their survey, they also described a more general methodology for performing this kind of web-based collective intelligence gathering and self-organization of knowledge from a community [93]. This is an early example of WebAL *crowd creativity*—we will come across other examples later in the article, and briefly summarize these projects in Section 7.1.

29 See also http://sodarace.net/, http://soda.co.uk/work/sodarace-online-olympics.

30 In 2013, Szerlip and Stanley developed an open-source browser-based version of *Sodarace*, called *IESoR* [123]. It features a developmental encoding of creatures suitable for evolutionary experiments, and is designed to be an accessible platform that other researchers can easily use.

31 An archived copy of the project's website is available at http://web.archive.org/web/20050126162950/http://norgev.alife.org/.

32 They were right in that this kind of architecture is now more feasible, although in modern systems this can be accomplished using native HTML5 technology rather than Java—see Section 7.2.

33 https://msdn.microsoft.com/en-us/library/aa286483.aspx

Box 2. Highlights of other NetAL work from the 2000s.

The 2000s also saw the continued development of various ALife-related desktop (non-browser-based) systems that made use of Internet connections for collaborative functionality.

A prominent example of this type of project was *Golem@Home*, developed by Hod Lipson and Jordan B. Pollack at Brandeis University and launched in the summer of 2000.[a] *Golem@Home* was an extension to the *Golem* project that sought to combine the evolution of simulated robotic systems with 3D printing technology to create a highly automated system for the design and manufacture of real-world robots [57]. *Golem@Home* allowed the general public to download a screensaver that harnessed idle CPU cycles to evolve simulated robots and animate some of them on the user's screen. If a network connection was available, evolved robots would occasionally migrate between different users' machines. Each screensaver kept a list of the IP addresses of other users' machines, allowing a highly distributed system without the need of a central server to manage the process. Over the course of approximately 12 months, *Golem@Home* attracted over 30,000 participants, who jointly contributed several million CPU hours. From a technical perspective, the project was highly successful, although from a scientific perspective the authors did not observe the increase in evolved robot complexity they had hoped for.[b]

Other examples of desktop-based systems with collaborative features implemented with Internet connections also continued to come from the area of ALife-related computer games research. In 2003, Stanley and colleagues at the University of Texas at Austin started developing the computer game *NERO*, which allowed users to train a team of in-game agents using a real-time version of the NEAT architecture [117]. Once trained, the team could compete against an opposing team designed by another (often remote) user. The battle mode ran on a server such that both users could watch the battle while running the program on separate Internet-connected machines.[c]

A few years later, Stanley and colleagues at the University of Central Florida produced *Galactic Arms Race* (*GAR*), another desktop-based video game using Internet connectivity for collaborative game play [34, 35].[d] *GAR* includes a genetic algorithm to evolve new weapons (based upon particle systems) according to the weapons preferred by current users. In single-player mode, the weapons evolve according to the single user, but in full multi-player Internet mode the weapons evolve in separate lineages based upon the aggregate usage of all players. The end result is the continual introduction of new in-game content based upon the players' tastes.

a http://www.demo.cs.brandeis.edu/golem/download.html
b Results as reported on the cited web page.
c *NERO* was originally distributed as a binary file running on Mac or Windows. In 2009 work commenced an an open-source version called *OpenNERO* (http://nn.cs.utexas.edu/?opennero).
d http://gar.eecs.ucf.edu

Returning to more familiar flavors of WebAL, the general increase in the computing power of desktop PCs in the 2000s over the 1990s allowed the introduction of more sophisticated web browsers, and meant that it was feasible to run more powerful programs on the client side (within the browser). Hence, the 2000s witnessed an increase in complexity of WebAL projects, made possible by the parallel improvements in hardware and web technology in addition to the scientific progress of the field itself. In the remainder of this section we look at how the changing landscape of web technology affected the development of WebAL during the 2000s.

5.1 The Changing Technology Landscape of WebAL

5.1.1 Java Applications and Applets of Growing Sophistication

Countless Java applets appeared in the late 1990s and early 2000s that simulated ALife-related concepts, some of which were already described in Section 4. In addition, various cellular automata (CA) were popular.[34] While many of the earlier examples were fairly simple, some serious academic

34 For lists of many of these, see http://cell-auto.com/links/ and http://www.aridolan.com/ad/CA.html.

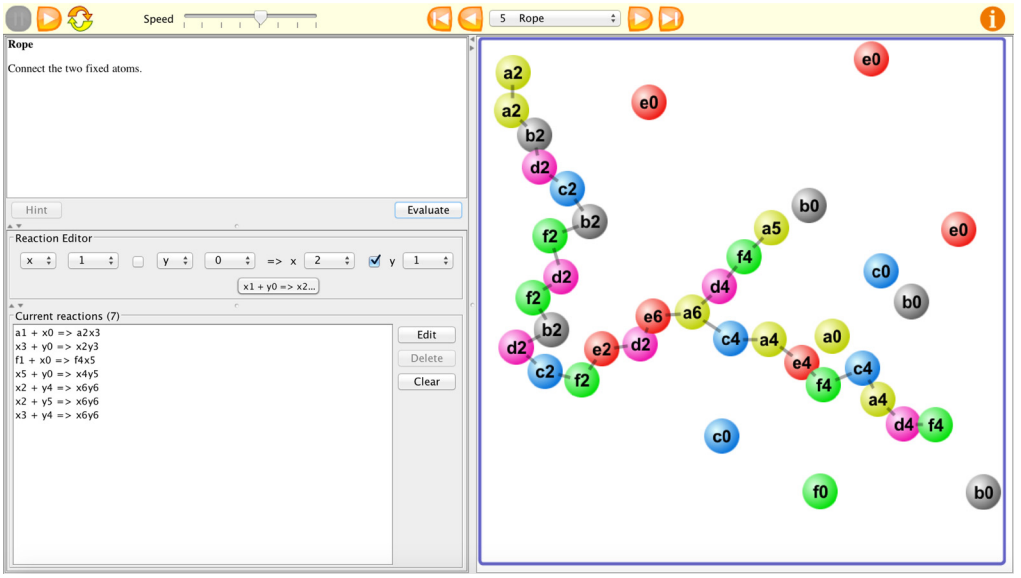


Figure 4. The *Organic Builder* user interface.

projects made their code available as applets as a means of distribution to encourage experimentation by other researchers (e.g., [68]).[35]

In addition, some more elaborate projects were developed as interactive educational tools, including *Soda Constructor* (described in Section 4 but further developed in the 2000s). Another example is *Organic Builder*,[36] an accessible tool for experimenting with artificial chemistries, developed by Tim Hutton and first launched in 2005. The system allowed users to edit the reaction rules of an artificial chemistry and immediately observe the results in a browser-based graphical simulation of interacting particles (see Figure 4). A series of progressively harder challenges was presented, to test a user's skill in devising reaction rules to achieve particular behaviors. All 19 challenges set by Hutton were solved by users, sometimes in very unexpected and ingenious ways. As well as solving the challenges individually, some users also discussed and shared their results on an online discussion forum.[37] Reflecting on the experience of running the system for a number of years, Hutton commented: "Though it was initially intended as a set of challenges to be tackled as a game, the users experimented with the system far beyond this and discovered several novel forms of self-replicators. When searching for a system with certain properties such as self-replication, making the system accessible to the public through a Web site is an unusual but effective way of making scientific discoveries, credit for which must go to the users themselves for their tireless experimentation and innovation" [39, p. 21]. Thus, in addition to the education and outreach goals, *Organic Builder* also provides an early example of web-mediated crowdsourced human computation for solving complex tasks.

Another important WebAL project in the late 2000s was the evolutionary art system *Picbreeder* [109, 108]. The system, launched in 2007 and still running today, allows users to evolve two-dimensional images using a Java applet that sends results back to a central server. The evolved images and their lineages can then be viewed via the project's website.[38] The images are encoded by a neural network variant called *compositional pattern-producing networks* (CPPNs) [116] and evolved through an

35 As the decade developed, the growing popularity of agent-based modeling toolkits such as NetLogo [111] made it easier than ever to package simulation models as Java applets.

36 http://organicbuilder.sourceforge.net/

37 http://groups.google.com/group/OrganicBuilder/

38 http://picbreeder.org

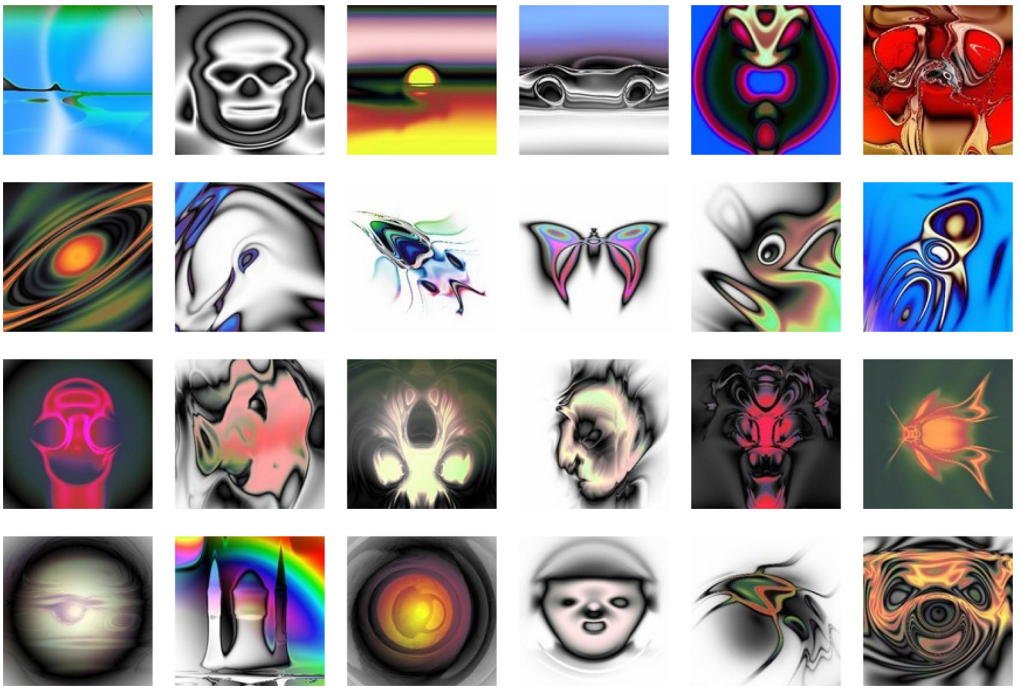


Figure 5. Examples of evolved images from *Picbreeder*.

implementation of the *neuroevolution of augmenting topologies* (NEAT) algorithm [118, 119].[39] A major innovation of *Picbreeder* was its support for a process called *branching*, which enables genuine collaborative interactive evolution; users are allowed not only to evolve their own images, but also to select and continue evolving images produced by other users. As branch accumulates upon branch, the system facilitates the interactive evolution of deep lineages of evolved pictures encompassing the contributions of many users, and accordingly also enables the collective exploration of a vast search space of images. Results from *Picbreeder* demonstrate that it is possible to effectively harness the input of many unrelated users through branching. Some examples of images evolved with *Picbreeder* are shown in Figure 5.

Java-based projects such as *Organic Builder* and *Picbreeder* capitalized on the opportunities for distributed interaction and crowdsourced computation afforded by the web. Elsewhere, other projects continued to utilize Java and applets purely as a means for easy distribution, dissemination, and engagement. A prominent example is Hiroki Sayama's *Swarm Chemistry* project[40] [104], introduced in 2006 and still being actively developed.[41]

5.1.2 Adobe Flash

In addition to Java, Adobe Flash[42] was another widely used technology in the 2000s for delivering animated multimedia web content (see Figure 1). There are various examples of Flash-based WebAL

39 An earlier, but much more limited project in web-based evolutionary art based upon the NEAT algorithm was the *Living Image Project* (http://w-shadow.com/li/), written by Jānis Elsts and operational over 2006–2007. The server-side application was partially based upon Mattias Fagerlund's Delphi NEAT package (http://nn.cs.utexas.edu/?neatdelphi), and generated PNG image files of evolved images to be displayed on the client-side web page.

40 http://bingweb.binghamton.edu/~sayama/SwarmChemistry/

41 *Swarm Chemistry* has inspired various more recent independent projects, including an iOS app (https://itunes.apple.com/us/app/emergent/id965513030) and an Adobe Flash-based web version (http://flexmonkey.blogspot.co.uk/2013/08/advection-swarm-chemistry-with.html).

42 https://en.wikipedia.org/wiki/Adobe_Flash

projects, one of the most notable being a web-based genetic algorithm for evolving cars to run over uneven terrain in a 2D simulated physics environment, which first appeared in 2008.[43] This simulation served as the inspiration for the more recent and very popular reimplementation of the idea called *BoxCar2D*.[44] Developed in 2011 by the UC Santa Cruz graduate Ryan Weber, *BoxCar2D* was also written in Flash and used a Flash port[45] of the popular *Box2D* physics engine.[46] Also in 2011, shortly after the arrival of *BoxCar2D*, Rafael Matsunaga released a reimplementation of the system using native HTML5 technologies rather than Flash.[47]

5.1.3 WebAL in Online Virtual Worlds

Beyond the realms of academic research, Linden Lab's online virtual world *Second Life* was launched in 2003 and gained massive worldwide popularity over the following years.[48] A number of ALife-related projects were developed within this platform, two of the most notable being *Svarga* and *Terminus*, which both came to prominence in 2006. *Svarga*, created by *Second Life* user Laukosargas Svarog, was an island with a fully functioning ecosystem comprising a weather system and various types of plants and animals.[49] The island can still be visited in *Second Life* today,[50] but it was purchased from Svarog by Linden Lab in 2010[51] and some of its original ecosystem features may no longer be present. Shortly after the release of *Svarga*, a separate effort was launched by the Ecosystem Working Group and associated with the in-game location *Terminus*.[52] The group's aim was to develop an open-source programming language that would not only allow developers to freely create their own creatures, but also allow the creatures in *Terminus* to interact and evolve using a shared language. However, the project apparently ran into funding and resource problems, and is no longer available.[53]

5.1.4 Native HTML and HTTP Technologies

The 2000s also saw the beginnings of a drive for native technologies and APIs in preference to Java applets and other techniques that required browser plug-ins. This drive accelerated towards the end of the decade, boosted by developments such as the beginnings of work on the HTML5 specification in the mid-2000s,[54] and the emergence of fast JavaScript engines offering substantial speed-up for client-side code, led by the introduction in 2008 of the V8 JavaScript engine by Google[55] (see Figure 1).

Even before these developments, there are examples of WebAL work that chose to focus on native technologies. An interesting early WebAL project that explored the potential of distributed computation and native client-side storage was William Langdon's *Pfeiffer* website, released in late 2001 and still running today[56] [49]. This browser-based system allowed users to evolve 2D patterns described by L-systems (see Figure 6). A user was presented with a variety of patterns on screen, and could select those they thought were good and bad, which directly influenced their evolutionary fitness. The patterns were presented as GIF files that had been generated by the server, based upon

43 The original site is now defunct, but it can still be seen on the Internet Archive at https://web.archive.org/web/20100729074214/http://www.wreck.devisland.net/ga/. We have been unable to trace the author of this work, beyond his anonymous announcement on Reddit (https://www.reddit.com/r/programming/comments/7i22c/genetic_programming_evolution_of_mona_lisa/c06pt65).

44 http://boxcar2d.com/

45 http://sourceforge.net/projects/box2dflash/

46 http://box2d.org/

47 See http://rednuht.org/test/simulator/. Matsunaga has more recently also written other web-based evolutionary systems using 2D physics, including one for biped walkers (http://rednuht.org/genetic_walkers/) and another for a quadruped (http://rednuht.org/geneticat/).

48 http://secondlife.com/

49 http://nwn.blogs.com/nwn/2006/05/god_game.html, http://nwn.blogs.com/nwn/2010/03/svarga-returns.html

50 http://secondlife.com/destination/svarga

51 http://nwn.blogs.com/nwn/2010/03/svarga-returns.html

52 http://news.nationalgeographic.com/news/2007/03/070308-second-life.html

53 http://forums-archive.secondlife.com/191/83/133314/1.html

54 http://www.w3.org/TR/html5/introduction.html

55 https://en.wikipedia.org/wiki/JavaScript_engine

56 http://www0.cs.ucl.ac.uk/staff/W.Langdon/pfeiffer.html

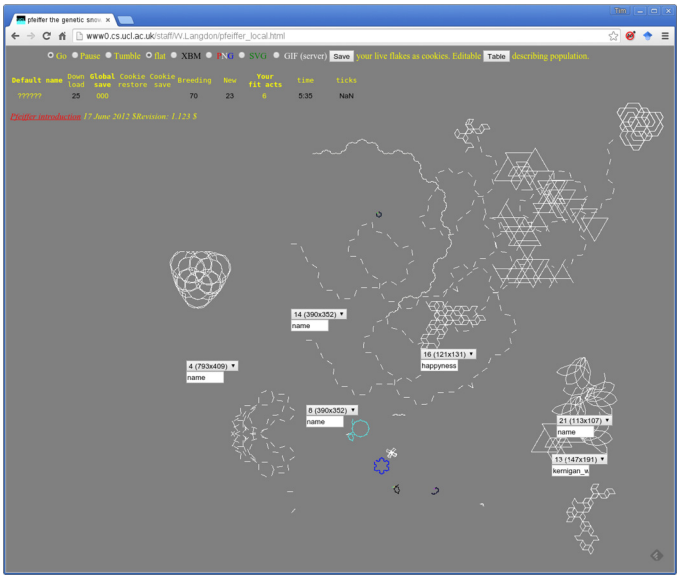


Figure 6. The *Pfeiffer* user interface.

the genetic description of an L-system, and then sent to the user's browser. The user could also select patterns to be parents for a new offspring. Surviving patterns were made persistent on the client side using HTTP cookies.[57] Users could name their favorite patterns and save them, in which case they were not only stored locally but also uploaded to the system's global server, where they would become available to be sent to other users. *Pfeiffer* therefore implemented distributed web-based evolution with aesthetic selection.

Also noteworthy is Alexander Wait's project *Quantum Coreworld*, first reported in 2004 [132]. The basic system was inspired by Rasmussen et al.'s early ALife system *Coreworld* [92], with the addition of "physics" inspired by quantum mechanics. The program was written in the C language and ran continually on a web server. The novel WebAL aspect of *Quantum Coreworld* was that it allowed users to inject "biotic" or "abiotic" changes into the system by uploading an instruction file via the system's web interface.[58]

An early example of a WebAL project using native HTML5 technologies is AlteredQualia's *Image Evolution* website,[59] developed in 2008. This was inspired by Roger Johansson's earlier (non-web-based) work on evolving a polygon-based representation of the *Mona Lisa*.[60] At the time, the required HTML5 canvas element for drawing graphics was not widely supported by browsers, but the situation has now greatly improved, as we'll see in the next section.

6 The 2010s: Current WebAL

The movement away from Java applets and proprietary plug-ins towards native code and standardized, open APIs have continued to gain momentum during the 2010s. Along with the development of HTML5 (see Figure 1), the possibilities for creating native browser-based applications have been greatly expanded by the appearance of many open JavaScript APIs. These include, among others,

57 https://en.wikipedia.org/wiki/HTTP_cookie
58 It appears that this system was not developed much further after its initial publication.
59 http://alteredqualia.com/visualization/evolve/
60 http://rogeralsing.com/2008/12/07/genetic-programming-evolution-of-mona-lisa/

Web Storage[61] for persistent data storage on client machines, WebSocket[62] for two-way communication between server and client machines, Web Workers[63] for running background scripts on the client machine, Web Audio[64] for processing and synthesizing audio, and WebGL[65] for hardware-accelerated graphics. This movement to open standards and technologies can be expected to continue now that the World Wide Web Consortium (W3C) has published the final, complete specification for HTML5.[66]

In 2010, the Swedish creative studio B-Reel Creative, in collaboration with Google Creative Labs and the American music video director Chris Milk, designed a groundbreaking web-based video called *The Wilderness Downtown*.[67] The project, a Grand Prix winner at the 2011 Cannes Advertising Awards and recipient of a host of other awards,[68] was a trailblazer for many of the possibilities opened up by open web technologies, including HTML5 video, audio, and canvas. It included several ALife-related techniques, such as an interactive bird flocking simulation that reacted both to the audio and to mouse interaction, as well as procedural drawing and generative typefaces.[69]

The Wilderness Downtown graphically illustrated the potential of the modern web as a platform for ALife applications. In the years since it was released, the reach, ambition, and volume of WebAL research has greatly expanded, as other projects begin to realize the potential offered by modern, native web technology. In this section, we discuss a number of recent and current projects in a little more detail than in previous sections, in order to reflect the current state of the art. We will look at developments in the fields of evolutionary art and design (Section 6.1), games (Section 6.2), science and education (Section 6.3), frameworks for WebAL (Section 6.4), and, finally, some new directions for WebAL (Section 6.5).

6.1 Evolutionary Art and Design

6.1.1 EndlessForms

The *Picbreeder* system's approach to collaborative interactive evolution through branching, described in Section 5.1.1, has inspired various subsequent projects. The foremost example is *EndlessForms*,[70] created in 2011 by Jeff Clune, Jason Yosinski, Eugene Doan, Hod Lipson, and colleagues at Cornell University [17]. The design and goals of the project were influenced by *Picbreeder*, but *EndlessForms* focuses on the design of 3D shapes rather than 2D images. The practical purpose of the project is to allow people to create unique physical objects and see the power of evolution in action, and the scientific purpose is to explore what complex morphologies can be created with a computational implementation of developmental biology. A screenshot of the *EndlessForms* home page is shown in Figure 7.

Besides demonstrating the scientifically interesting coupled power of evolution and human interaction, *EndlessForms* also addresses an important real-world problem. While 3D printing technology is rapidly advancing, most people do not know how to design their own 3D objects, even though they may have strong opinions and preferences. That is, many people are naturally skilled critics who will know what they like when they see it, but they are not skilled designers who can

61 http://www.w3.org/TR/webstorage/
62 http://www.w3.org/TR/websockets/
63 http://www.w3.org/TR/workers/
64 http://www.w3.org/TR/webaudio/
65 https://www.khronos.org/webgl/
66 The final specification was published in October 2014. See http://www.w3.org/blog/news/archives/4167
67 http://thewildernessdowntown.com/. The project was an interactive interpretation of the band Arcade Fire's song "We Used to Wait." For further information, see http://b-reel.com/projects/digital/case/57/the-wilderness-downtown/.
68 http://docubase.mit.edu/project/the-wilderness-downtown/
69 https://www.chromeexperiments.com/arcadefire/
70 http://endlessforms.com

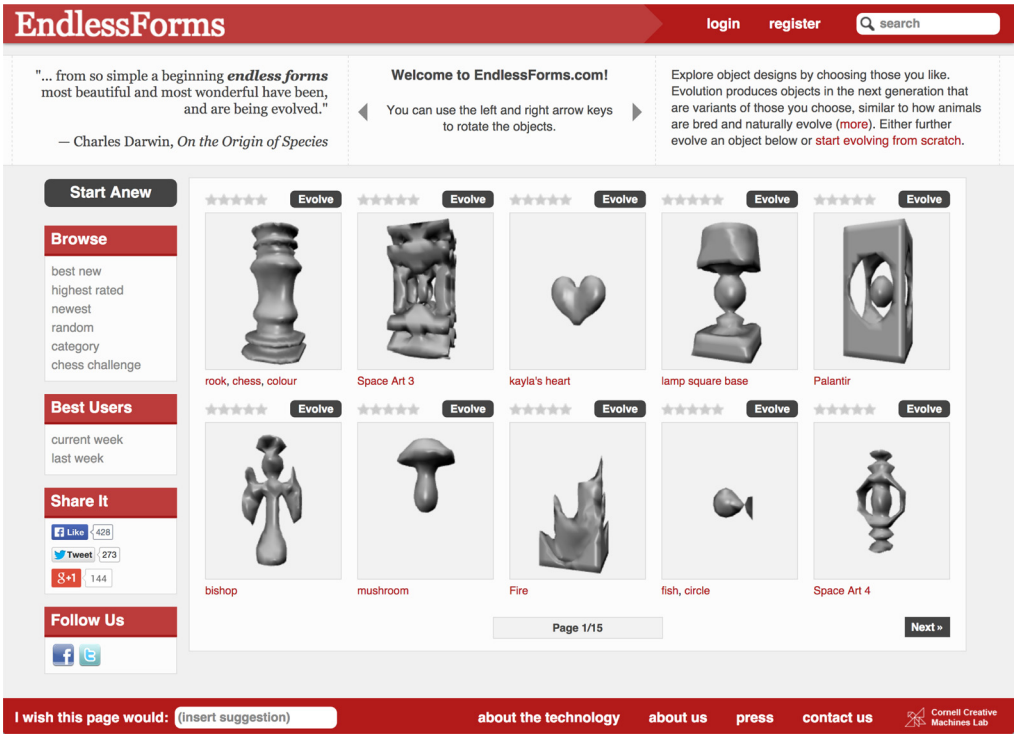


Figure 7. The *EndlessForms* home page.

create what they desire from scratch. *EndlessForms* enables people to create and design interesting physical objects—such as jewelry, doorknobs, candlesticks, and sculptures—without any technical knowledge. This ability will be a key component to the shift from mass manufacturing to small-scale custom manufacturing fueled by the 3D printer revolution, as well as the sharing and easy modification of such information in the rapidly forming "Internet of things."

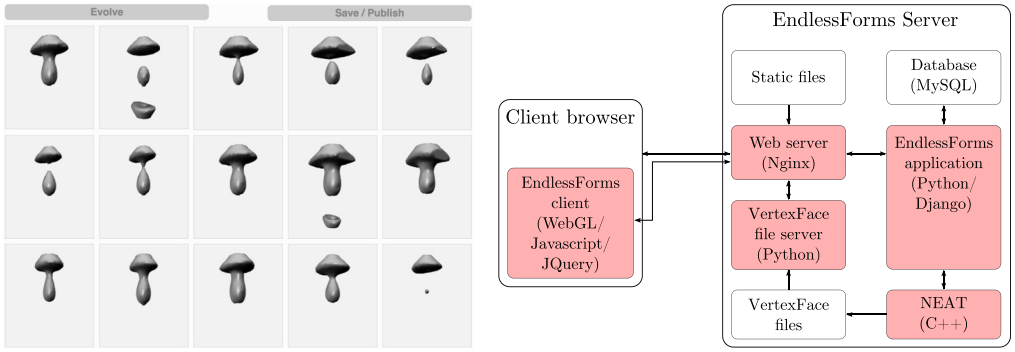
6.1.1.1 Mechanism and User Interface In a similar vein to *Picbreeder*, users are presented with a population of 15 shapes on a screen and are asked to select the one or several that they prefer (see Figure 8a). The shapes rotate slowly in the user's browser window, allowing them to see all sides of the 3D object.

Once users have created a shape to their liking, they are able to save it to their personal account, tag it with a string for later reference, and download the shape in the stereolithography (STL) file format to enable printing on 3D printers. For users who wish to have a printed copy of their shape but do not possess a 3D printer, they click through to have the shape printed and mailed to them by a 3D printing company, Shapeways.[71]

Critically, users are also afforded the option to publish the shapes they evolve. Upon publication, the object appears on the home page and becomes available for other users to download or further evolve. The authors have found that many of the most interesting shapes have been collaboratively evolved starting from previously published objects.

6.1.1.2 Technology *EndlessForms* was created by combining several different web technologies (Figure 8b). On the server side, the main *EndlessForms* application is written in Python and uses

71 http://www.shapeways.com



(a) User selection interface (b) System architecture

Figure 8. Example screenshot of the *EndlessForms* user selection interface (a), and architecture diagram showing the system's main components and their interaction.

the Django web framework.[72] It handles all user interaction, stores important states—such as login information, user e-mails, publication records, organism fitness, and so on—in a database, and spawns and manages NEAT processes (as used in *Picbreeder*) to generate shapes. The client-side code, written in JavaScript using the JQuery framework,[73] handles fetching each shape and drawing it using WebGL.[74] The site was initially constructed using shapes drawn in pure JavaScript, but this implementation was found to be slow and could not handle high-resolution shapes. Thus, the site has become technologically useful only with the recent widespread adoption of WebGL in browsers to take advantage of the client machine's graphics card for rendering.

6.1.1.3 Results As of 2015, users on *EndlessForms* have clicked through over 350,000 generations (screens of 15 individuals) to evolve over 5.3 million organisms. Many users have been driven to the site by coverage in the popular press, including *New Scientist*, MSNBC.com, Slashdot, *MIT Technology Review*, KurzweiAI.net, *Y-combinator Hacker News*, and the *Communications of the ACM*. It is only due to the large volume of users, eyes, and clicks that many interesting shapes have been able to be created. Some of these objects are shown in Figure 9, together with examples of physical objects created by 3D printing of some of the evolved forms.

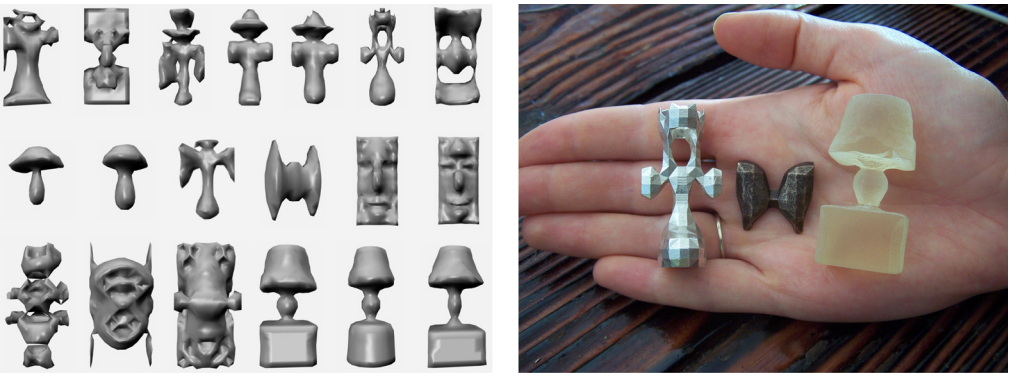
6.1.2 Other Recent Work

A recent project called *DrawCompileEvolve*,[75] created in 2013 by Rasmus Taarnby and Jinhong Zhang in Sebastian Risi's lab at the IT University of Copenhagen, builds on some of *Picbreeder's* web technology while adding the novel idea of a genotype-to-phenotype compiler [98]. This system allows users to draw a sketch of a particular image annotated with regularities (e.g., a butterfly with two symmetric wings) and then further evolve a CPPN representation of the image interactively [138].

Another new project with a strong *Picbreeder* flavor, but implemented using HTML5 scalable vector graphics (SVG) support, is Craig Mandsager's *Genolve* system (2014).[76]

Elsewhere, a novel variety of WebAL was reported by Joshua Auerbach in 2012 while working at the University of Vermont [9]. This work evolved 2D images with a similar representation to that used in *Picbreeder*. The key difference was that the fitness of each image avoided the time-consuming process of user selection. To accomplish this automation, the fitness function

72 https://www.djangoproject.com/
73 https://jquery.com/
74 https://www.khronos.org/webgl/
75 http://rasmustaarnby.dk/drawcompileevolve/
76 http://www.genolve.com/. For further details, see http://wildwebwidget.com/.



(a) Highly rated evolved shapes (b) Example 3D printed objects

Figure 9. Highly rated 3D shapes evolved in *EndlessForms* (a), and 3D-printed physical objects generated from evolved forms, in silver, bronze, and plastic (b).

included a call to Google Search by Image[77]—a Google service that performs an image search based upon a query image instead of (or in addition to) query text. Auerbach reasoned that, since images on the web should be of interest to humans (otherwise they would not have been uploaded), interesting images should return many hits. Hence, the number of returned hits of a query image was used as a component of its fitness. Some example results are shown in Figure 10.

This work was novel in that it leveraged a third-party web service for fitness function evaluations, and presented one possibility for leveraging existing "big data" stores concerning human preferences. Additionally, the work presented several lessons for future research in this vein. Primary among these lessons were the constraints imposed by utilizing a service not meant for automated interaction. Calls to the service needed to be rate-limited in order to avoid being blocked by Google, which turned out to be a major setback to performing larger-scale experimentation. This obstacle highlights the need to involve the owners of web services in future research, so that such problems can be avoided.

An alternative approach to automating the fitness evaluation of evolutionary design systems was described in 2015 by Anh Nguyen, Jason Yosinski, and Jeff Clune at the University of Wyoming, in their work on *Innovation Engines* [73]. Although they employed a deep neural network (DNN) rather than a web service to implement the evaluation function, their system relied upon web services in an indirect but interesting way. The DNN was trained on labeled images from the ImageNet data set.[78] The evolutionary component of the system (based upon the CPPN-NEAT approach) was then challenged to generate images that the DNN would classify as representing a previously trained label with high confidence. Although the system itself was not web-based, the ImageNet data set upon which the DNN was trained, which contained 1.3 million labeled images, was created with the help of Amazon's Mechanical Turk service[79] [22]. Mechanical Turk is a web-based crowdsourcing marketplace that allows "requesters" (individuals or businesses) to farm out tasks that require human intelligence (in this case, labeling images according to various categories) to a crowdsourced group of workers who accept the work and get paid for completing it. Hence, crowdsourcing has played an important part in the *Innovation Engines* project.

Beyond two- and three-dimensional forms, WebAL art applications have also been extended to sound generation. One such application is the *Breedesizer* system,[80] developed in 2015 by Björn Þór

77 http://www.google.com/imghp?sbi=1
78 http://www.image-net.org
79 https://www.mturk.com/mturk/welcome
80 http://bthj.is/breedesizer/

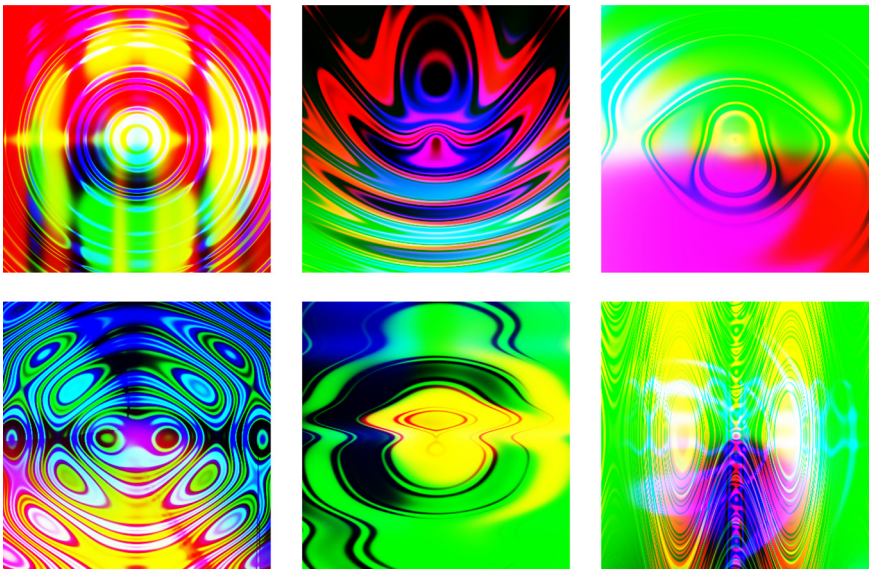


Figure 10. Images from Auerbach's work on automated image evolution using a fitness measure related to the number of hits returned by Google Search by Image [9].

Jónsson, Amy Hoover, and Sebastian Risi, which allows users to interactively evolve novel timbres in a manner similar to *Picbreeder* [43]. *Breedesizer* builds on the recently introduced Web Audio API,[81] along with the Worldwide Infrastructure for Neuroevolution (WIN) framework (which we discuss in more detail in Section 6.4.3).

6.2 Games

A variety of WebAL-related games have appeared in recent years, making use of several different technologies and platforms.

6.2.1 Desktop and Multiplatform WebAL Games

An example of using Facebook as a platform for WebAL is provided by *Petalz*,[82] developed by Risi and colleagues first at the University of Central Florida and more recently at the IT University of Copenhagen. *Petalz* is a social game that allows users to breed and share evolved virtual flowers [99]. Released in 2012, the game is written using the Adobe Flash platform and delivered as a Facebook app. Facebook's Graph API[83] allows players to share their flower creations on other people's walls or sell them through an in-game marketplace. *Petalz* enables players to breed new flowers from purchased market seeds, thereby facilitating meaningful collaborations between users. Like *Endless-Forms*, *Petalz* also allows players to transfer their evolved flowers to the real world via 3D printing through the Shapeways API [101, 100].

81 http://webaudioapi.com/, http://www.w3.org/TR/webaudio/
82 https://apps.facebook.com/petalzgame
83 https://developers.facebook.com/docs/graph-api

Another platform that has been used for developing WebAL games is the Unity Game Engine,[84] which facilitates (among other things) the use of Internet connections for collaborative gaming.[85] An example of such a game is *EvoCommander*[86] [40], which was inspired by ideas introduced in *GAR* and *NERO* (discussed in Box 2), and released in 2014. *EvoCommander* allows players to incrementally evolve arsenals of ANN-controlled behaviors such as ranged attack or flee. Players can then battle other players' robots online and, through the novel game mechanic of "brain switching," select which evolved neural network is active at any point during a battle.

Unity has proven a powerful tool for a broad range of web game development. Another recent game that builds on its network capabilities is *FPSEvolver* [79], in which a group of players iteratively generate, play, and improve multiplayer FPS levels to fit their particular preferences by voting on a selection of evolving levels. Other new games are in active development, such as *Evolve and Conquer*,[87] a StarCraft-style game focused on teaching evolutionary principles.

An interesting prospect for WebAL is to allow collaborative problem solving through a crowdsourced web-based approach. Online video games such as *Foldit*,[88] in which users have to discover protein structures, hint at the power of crowdsourcing the brain's natural ability for certain tasks that involve pattern matching or spatial reasoning. A recently introduced WebAL system that similarly tries to harness human intuition is *BrainCrafter*,[89] which allows users to collaboratively build artificial neural networks to control a simulated robot in an ALife setting [86]. The aim of this project is to ultimately facilitate a mixed initiative process, in which a human and a computational creator take turns and propose changes to an evolving neural network [102].

6.2.2 Mobile and Augmented-Reality WebAL

At the time of writing there has been little in the way of WebAL projects designed specifically for mobile platforms. However, we are aware of two significant projects of this type that are currently under development. Wiggle Planet,[90] a company founded by the ALife veteran Jeffrey Ventrella (whose earlier work on the *Absolut Kelly* website was described in Section 4), is currently developing an augmented-reality, mixed media game called *Polly Peck's Journey*. This is an interactive puzzle game, comprising a physical book and a tablet application, that aims to encourage children to interact, learn, and play in their natural environments. Children will be able to adopt the animated augmented-reality characters in the game, called *wiglets*, and there are plans for the wiglets to be stored and shared in the Cloud. Another web-oriented aspect of *Polly Peck's Journey* is that its development was partially funded through a Kickstarter crowdfunding campaign that raised over US $15,000 of development funding.[91] Elsewhere, Jane Prophet and Mark Hurry are working on a new version of *TechnoSphere* (the original version of which, discussed in Section 4, was one of the earliest WebAL systems). *TechnoSphere 2.0*[92] will be an augmented-reality mobile application [89]. An Android app version of the program is currently undergoing beta testing.

A somewhat different example of the combination of ALife research and mobile apps is provided by AppEco,[93] an agent-based simulation model of mobile app ecosystems developed by Soo Ling Lim and Peter Bentley. The authors calibrated the model with real data about numbers of developers, apps, and users collected from Apple's iOS ecosystem over three years. In a series of studies,

84 https://unity3d.com/

85 Unity Personal Edition is free but proprietary software. It runs on many different platforms—with the notable exception of Linux, although an experimental Linux build is currently under development (see http://blogs.unity3d.com/2015/08/26/unity-comes-to-linux-experimental-build-now-available/).

86 http://jallov.com/thesis/

87 http://adamilab.msu.edu/evolve-and-conquer/

88 http://fold.it/portal/

89 http://braincrafter.dk/

90 http://www.wiggleplanet.com/

91 https://www.kickstarter.com/projects/1582488758/peck-pecks-journey-a-picture-book-that-spawns-virt

92 http://technosphe.re/

93 http://www.appeco.co.uk/

they used it to investigate the effectiveness of various design strategies for developers [54], publicity strategies for new apps [53], and app store organization strategies [55].

6.3 Science and Education

Many of the projects discussed in Sections 6.1 and 6.2 implicitly introduce users to ALife concepts such as evolutionary algorithms. In this section, we review a number of recent projects that have more explicit pedagogical aims, and others developed as platforms for open science.

6.3.1 Evolutionary Robotics

The first three projects discussed below all focus on using the web for teaching and outreach in the field of evolutionary robotics.

6.3.1.1 *Ludobots* *Ludobots*[94] is an educational WebAL system developed by Josh Bongard and colleagues at the University of Vermont, and launched in 2012. The system serves as an infrastructure for those who wish to explore the design and evolution of robots. It is designed for students with a wide range of experience, from no programming expertise up to students who create and program new projects for other students to learn from. Students are gradually guided through a series of increasingly challenging projects, during which they learn about various concepts such as evolutionary algorithms, artificial neural networks, robotics, and embodied cognition.

The first project requires no programming: The student uses a web interface to create a robot by simply "connecting the dots" and observing the resulting robot behave in a web-embedded physics engine[95] (see Figure 11b) [131]. Later projects switch from the Web-based simulation to a more powerful C++-based simulation running on a student's computer (see Figure 11c). Importantly, *Ludobots* does not restrict students to using prespecified robots or environments: Students are free to modify their growing code base as they progress, and even modify the instructions of the assignments themselves. Figure 11a outlines the system's curriculum flow.

Ludobots was inspired by open-ended web projects such as *Reddit* and *Wikipedia*: As the site grows, the growing user base continuously expands and improves the site's content. Students may improve existing assignments, annotate those assignments with educational material, and create new projects of their own. The creators of *Ludobots* intend for this positive feedback of material to help overcome one of the major limitations of ALife projects: Once an investigator publishes an article describing their project, the project falls into disuse and is rarely replicated or extended by others.

6.3.1.2 *Evolve-A-Robot* and WebGL Visualizer In 2014, Jared Moore and colleagues at Michigan State University reported work on *Evolve-A-Robot*, a web-based evolutionary robotics simulation[96] [71]. Their work pushes the boundary of what can currently be achieved in web-based 3D rigid body simulation and evolutionary robotics: It embeds a whole evolutionary robotics system and 3D physics simulation in a JavaScript client-side web application.[97]

In the same article, the authors also report work on a WebGL-based visualizer[98] that provides a tool for the interactive 3D visualization of previously recorded evolutionary robotics experiments. In

94 http://www.reddit.com/r/ludobots/

95 Implemented using ammo.js (https://github.com/kripken/ammo.js/), a JavaScript port of the popular Bullet Physics engine (http://bulletphysics.org), and the Three.js WebGL programming library (http://threejs.org/).

96 http://evolve-a-robot.github.io/

97 Like *Ludobots*, *Evolve-A-Robot* also uses a JavaScript port of the Bullet Physics engine: specifically, PhysiJS (http://chandlerprall.github.io/Physijs/), which is actually built on top of ammo.js but runs the physics simulation in a separate thread using Web Workers (http://www.w3.org/TR/workers/) to improve performance. The web-based simulation component of *Ludobots* does not (yet) incorporate an evolutionary algorithm; hence *Evolve-A-Robot* is novel is this respect.

98 https://github.com/jaredmoore/WebGLVisualizer

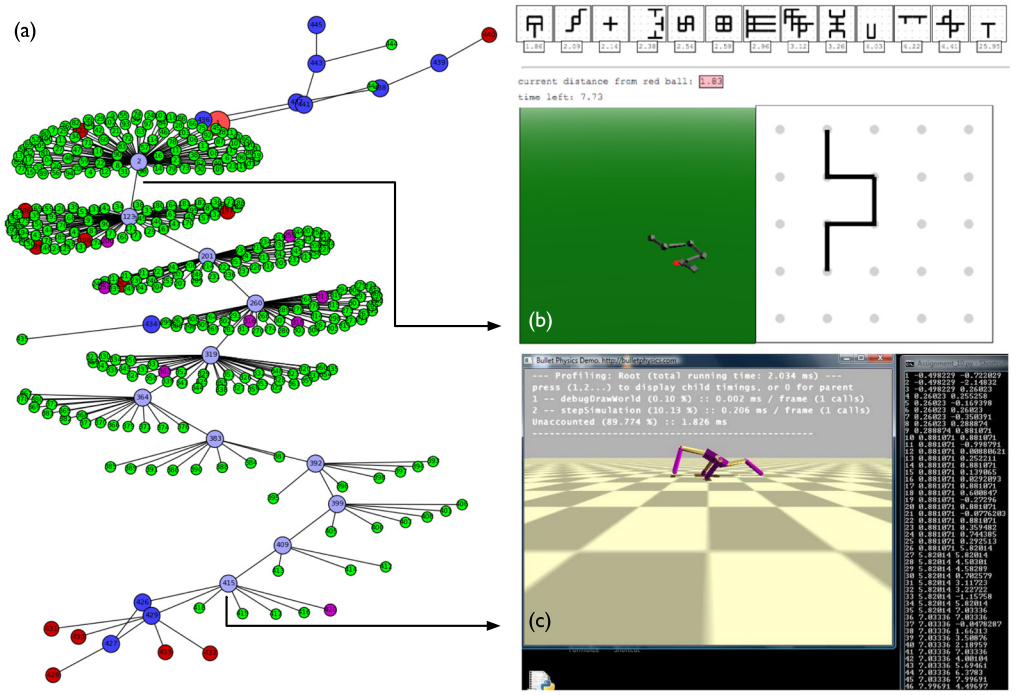


Figure 11. The *Ludobots* educational platform. The course tree (a) depicts all activity on the platform visually: Light blue nodes represent programming assignments; green nodes represent student submissions; purple and red nodes represent resources and questions, respectively; dark blue nodes represent student-generated projects. The first project (b) requires no programming: The student simply creates a robot by "connecting the dots" and observing the resulting robot behave in a web-embedded physics engine. They can also see robots created by other students at the top of the page. Students progress through a series of programming assignments to the final assignment (c), which results in a basic evolutionary robotics test bed. Students can then extend their system by following projects made by other students (the three dark blue dots at the bottom of (a)), or create new projects of their own and advertise them to their fellow students.

contrast to pre-recorded results videos, the system allows the user to watch the system from different angles and focal points.

The authors state that their motivation for both of these projects is to "facilitate the exchange of ideas with other researchers as well as outreach to K–12 students and the general public." [71, p. 1]

6.3.1.3 *RoboGen* Another ongoing project that uses WebAL in an educational context is *RoboGen*™, introduced in 2014 by Joshua Auerbach, Dario Floreano, and colleagues at EPFL [8]. *RoboGen* is a platform for the coevolution of robot morphologies and controllers, which focuses on the evolution of real (rather than virtual) robots. The goal of the project is to provide a simple and cost-effective means of evolving robots in simulation and rapidly fabricating them in reality. This goal is accomplished through the use of inexpensive 3D printers[99] and the use of simple, open-source, low-cost, off-the-shelf electronic components.

RoboGen has already been used by more than 100 students for course projects in EPFL's Bio-Inspired Artificial Intelligence class, and there are plans for it to be part of a future Massive Open Online Course (MOOC) on this topic. Like *Ludobots*, *RoboGen* has been designed to be flexible in order to accommodate users with a diverse set of skill levels and backgrounds. Those with little familiarity with evolutionary computation or programming can familiarize themselves with new concepts in a

99 Such as the MakerBot Replicator 2x: http://store.makerbot.com/replicator2x.

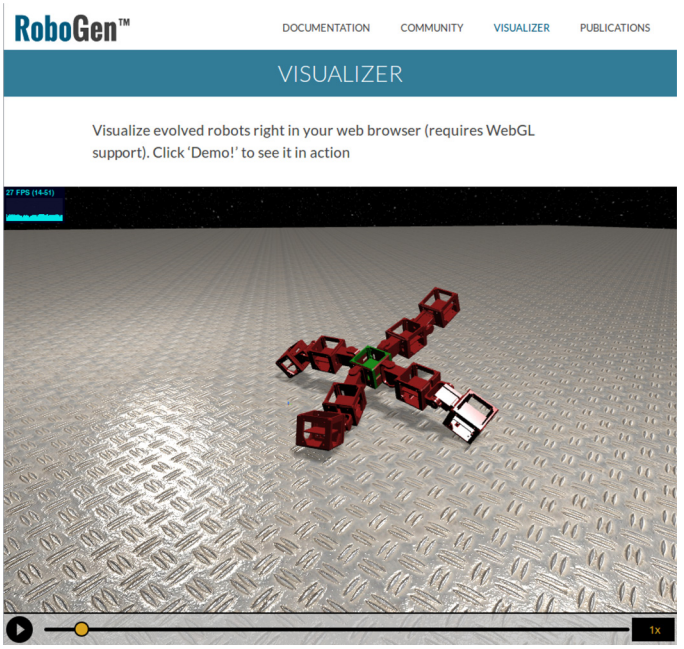


Figure 12. Screenshot of *RoboGen's* WebGL-based visualization interface.

controlled way without needing to write any code, whereas advanced users can dive more deeply by customizing the evolutionary algorithm or simulator or even by introducing new morphological building blocks. This open-endedness is a major advantage of the platform and addresses current concerns regarding science laboratory education [61].

The project has developed rapidly since its introduction. Initially, its only web-oriented aspects were the *RoboGen* website and wiki[100] that provided access to the necessary utilities for evolving and manufacturing robots.[101] However, the project's website now features a WebGL visualization engine—inspired by Moore's work described in Section 6.3.1.2—allowing for rich client-side 3D rendering of evolved robots within the browser (see Figure 12). Recently, a port has been released of the complete *RoboGen* software stack that runs natively in the browser. This was accomplished by compiling the original *RoboGen* C++ code[102] with Emscripten,[103] a source-to-source compiler that generates highly efficient JavaScript from C/C++ source code. The ability to run the complete system directly in the browser without the need to download and install additional software drastically lowers the barrier to entry of using *RoboGen* for students, researchers, and hobbyists. While similar to, and inspired by, the projects mentioned above, *RoboGen* on the web is distinct in its concern for evolving and simulating robots that can be easily manufactured.

There are plans to more fully capitalize on the potential offered by these web-based aspects of the system in future work. These include adding the facility for collaborative robot evolution, where users may upload new morphological building blocks and/or complete evolved robots for others to fabricate themselves, modify, or use as seeds for their own further evolutionary runs, and even to allow for distributing computational resources among users of the system. The creators of *RoboGen* believe that it will be through this collaborative exchange of ideas that the platform can truly

100 http://www.robogen.org

101 Including an evolution engine complete with a physics simulator, as well as utilities both for generating design files of body components for 3D printing, and for compiling neural network controllers to run on an Arduino microcontroller board (http://www.arduino.cc).

102 Including the Open Dynamics Engine physics simulator (http://www.ode.org).

103 http://kripken.github.io/emscripten-site/

blossom, a diversity of morphologies can be fabricated and tested, and users can benefit from each other's experience.

6.3.2 Other Science and Education Projects

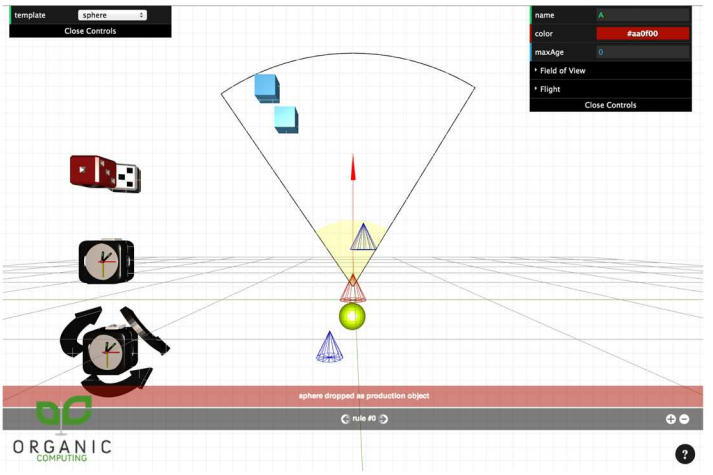
The evolutionary robotics projects described in the previous section constitute a major strand of current WebAL research. But science- and education-based WebAL is not limited to robotics; here we look at projects in other subject areas.

6.3.2.1 *Avida* and *Avida-ED* The *Avida* software[104] is a well-known research test bed that allows users to perform experiments on populations of digital organisms to investigate questions about evolutionary or ecological dynamics [74]. *Avida-ED*[105] is the educational version of this software, intended to allow students to get first-hand experience manipulating evolving populations, both to develop intuition about evolution and to generate a deeper understanding of the nature of science [85]. The *Avida* development team, based in the Digital Evolution Lab at Michigan State University, have recently created a fast JavaScript version of the code by compiling the original C++ version with Emscripten.[106] A new *Avida-ED* web interface[107] has been built for this web version of Avida and is in final testing before initial release. A full web reimplementation of the *Avida* research platform is also under development, and progressing rapidly.

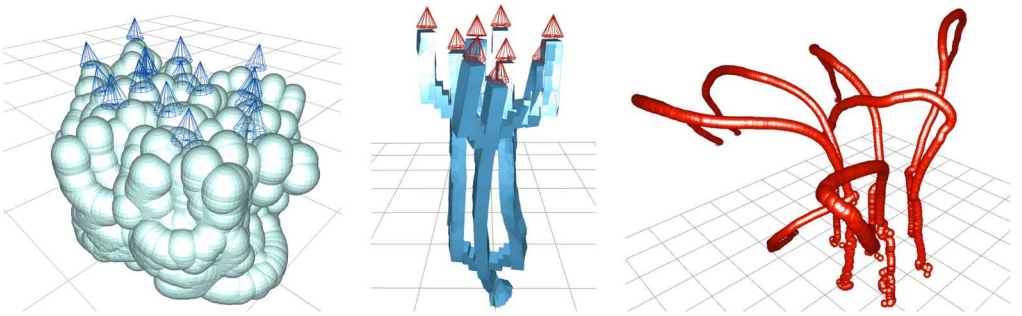
6.3.2.2 *The Ladybug Game* In 2012, Terence Soule and colleagues at the University of Idaho developed a web-based system called *The Ladybug Game*[108] as an interactive tool for teaching children about evolution.[109] The system was originally written in Processing[110] (from which a Java applet was generated), and later rewritten in JavaScript. *The Ladybug Game* presents the student with a simulation of a ladybug and a population of aphids on a colored background, where each aphid's color is genetically determined and the closeness of match between an aphid's color and the background affects its chance of being eaten by the ladybug. Over time the aphids' colors evolve in response to predation by the ladybug. The simulation was used as a tool in a series of five interactive lessons to demonstrate the roles of *selection*, *inheritance*, and *variation* in evolution. The user could interactively change the color of the leaf as the simulation proceeded, as well as select which of the three evolutionary features were active. *The Ladybug Game* was exhibited at the USA Science and Engineering Festival in 2012.[111]

6.3.2.3 *Swarm Grammars GD* Another recent example of an educational WebAL system is *Swarm Grammars GD*,[112] a WebGL-driven website by Sebastian von Mammen and Sarah Edenhofer at the University of Augsburg, Germany. The authors developed the tool as a means of introducing STEM research, and specifically ALife ideas, to high-school students. The system allows students to interactively configure and manipulate a *swarm grammar* system, which combines concepts from L-systems and flocking algorithms (see Figure 13). Initial studies were conducted as part of a girls-in-STEM program at the University of Augsburg, Germany. In their 2014 article, the authors discuss their experience of introducing the system to the students (aged between 12 and 15) and allowing them to explore its generative capabilities [130]. This preliminary study generated

104 http://avida.devosoft.org/
105 http://avida-ed.msu.edu/
106 http://kripken.github.io/emscripten-site/
107 https://github.com/devosoft/avida/tree/AvidaEd-Web
108 http://www2.cs.uidaho.edu/~tsoule/ladybug-lesson/
109 For further details, see also http://beacon-center.org/blog/2012/12/03/beacon-researchers-at-work-teaching-evolution-the-ladybug-game/.
110 https://processing.org/
111 http://www.usasciencefestival.org/component/content/article/66-2012-exhibits/213-2012-exhibits?&offset=12
112 http://www.vonmammen.org/SG-GD/. "GD" stands for generative design.



(a) Rule editor interface



(b) (c) (d)

Figure 13. The *Swarm Grammars GD* user interface (a), and examples of agent behavior with traces enabled (b–d).

mixed results, but the authors use the experience to identify several ways in which the system should be developed in future work.

6.3.2.4 *StringmolWeb* Tim Hutton's work on *Organic Builder*, a web-based platform for allowing users to learn about and experiment with artificial chemistries, was discussed in Section 5.1.1. A more recent system with similar goals is *StringmolWeb*,[113] developed in 2010 by Adam Nellis, Ed Clark, and Simon Hickinbotham at the University of York, UK. The system provides a web-based interface to the *Stringmol* artificial chemistry system [36]. The client-side application is written in native HTML and JavaScript, and communicates via CGI[114] scripts with a server running an instance of the *Stringmol* system (which is written in C++). It was used as an interactive educational tool at a tutorial workshop on artificial chemistries at the ECAL 2011 conference.[115]

113 http://stringmol.york.ac.uk/webapp/. Source code available at https://github.com/adamnellis/StringmolWeb, and for the underlying *Stringmol* simulator at https://github.com/franticspider/stringmol.
114 https://en.wikipedia.org/wiki/Common_Gateway_Interface
115 https://www.cs.york.ac.uk/nature/plazzmid/external/RUTSAC11/

6.3.2.5 *OpenWorm* A prominent example of an ALife-related open science project is *OpenWorm*, which aims to create a whole system simulation of a living organism. Specifically, the goal is to develop a detailed 3D dynamic simulation of the nematode *C. elegans* [82]. Although the simulation itself is not web-based, the core team are distributed across the world and have regular team meetings using web-based collaboration tools. The project website actively seeks to recruit new members to the team, including scientists, programmers, artists, and writers.[116] All code, data, and models produced by the project are open-source under the MIT license. The project also pursues a crowdfunding approach, seeking donations via the website and via a successful Kickstarter campaign that raised over US$120,000 in 2014.[117] The *OpenWorm* project therefore demonstrates multiple ways in which the Web can successfully facilitate large, open, collaborative projects of this kind.

6.3.2.6 The Broader Landscape The WebAL-related science and education projects described above are part of the broader landscape of developments in web-based education. These projects were highlighted because of the connection of their subject matter or their investigators (or both) to the artificial life community. A review of the broader field is beyond the scope of the current article, but we here provide some pointers to other work to give a flavor of more general developments.

In the area of web-based tools for primary and secondary education in biology, the *e-Bug* project[118] is a more elaborate example than *The Ladybug Game* described in Section 6.3.2.2. *e-Bug* is a large European Union-funded project to develop educational material to teach microbiology. A number of different web-based games have been developed in the project, aimed at different age groups, and the final versions have been translated into 11 different European languages [27].

A somewhat different example of the intersection of biological science education and web technology is provided by Kayhan Moharreri and colleagues' work on the *EvoGrader*[119] system [70]. While the projects described above have focused on web-based delivery of learning and educational *content*, the *EvoGrader* system provides a free web-based tool for the automated *assessment* of undergraduate biology students' understanding of concepts relating to evolution and natural selection.

Of course, the web also serves as the underlying delivery platform for many Massive Open Online Courses (MOOCs) beyond the more ALife-specific projects discussed in the preceding sections. The major MOOC providers[120] and other, more focused platforms[121] provide courses covering many topics in biology, evolution, complex systems, and many other topics relevant to ALife.[122]

6.4 Frameworks for WebAL

Recent years have seen the emergence of a number of more general frameworks and platforms for WebAL research. These systems provide users with the tools and infrastructure to conduct particular kinds of WebAL experiments without having to write everything from scratch; within a given domain, they provide general-purpose facilities that can be reused by multiple research projects. Four such frameworks are highlighted in this section, followed by a brief discussion of other relevant work.

116 http://www.openworm.org/

117 https://www.kickstarter.com/projects/openworm/openworm-a-digital-organism-in-your-browser

118 http://www.e-bug.eu/

119 http://www.evograder.org

120 For example, https://www.coursera.org/, https://www.edx.org and http://online.stanford.edu/.

121 For example, http://www.complexityexplorer.org/.

122 The implementation of scalable assessment mechanisms is a challenge for MOOCs. Current approaches typically involve either the use of simple forms of assessment (e.g., multiple choice questions or formulaic questions with well-defined answers), or peer grading [10]. Systems such as *EvoGrader*, which allow automated assessment of free-form written responses, could provide an important additional tool for realizing the full potential of MOOCs and other web-based educational tools.

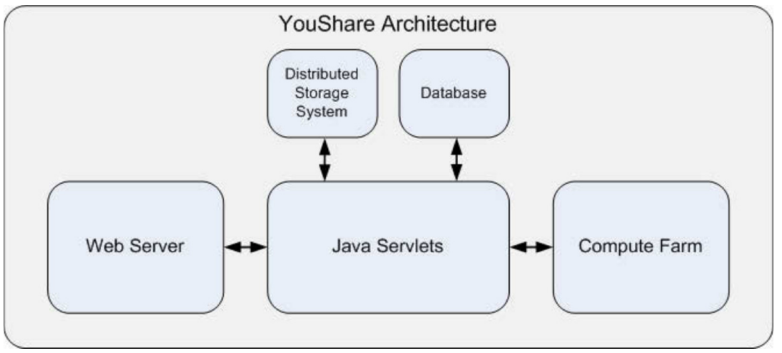


Figure 14. Overview of the YouShare architecture.

6.4.1 YouShare and the ALife Zoo

In 2013, Hickinbotham and colleagues reported WebAL-related work that made use of YouShare, a system that has the capability to run software designed for any platform in a web-based environment [37]. YouShare adopts a software-as-a-service model that exploits virtual machine (VM) technology and links it to a web browser. This design removes the need to port software to each target operating system (OS), since the software can be run inside a VM that runs the OS that the software was originally developed on. The only requirement is that the OS must also be able to run Java. The system architecture is illustrated in Figure 14. A web server, built using the Google Web Toolkit (GWT), interacts with a suite of Java servlets. These servlets, in turn, process the request for a service: A database maintains the service, its VM requirements, and the data to be analyzed by the service. The requisite components of the service are then fetched from a (distributed) storage system to a compute farm, which deploys the VM and the service inside it. The service is run, and the user is notified when the job has finished, again via the web portal. By this process, it is possible to hook up to a browser any ALife system that can run in a Java-enabled VM. It is also possible to chain services together with analysis tools to build *workflows* of analysis.

To explore the way that ALife systems can exploit this technology, Hickinbotham et al. wrapped three well-known ALife software technologies for deployment on YouShare, to create the ALife Zoo [37]. The systems were *Stringmol* [38], *Avida* [74], and *Tierra* [94]. The authors wrapped each software package as a service, and the test implementations that were provided with the source code were used as examples of how to use the system.

The YouShare framework has the potential to run or access a cloud- or grid-based high-performance computing (HPC) system, allowing large-scale ALife experiments to be run relatively easily. For example, it might form a suitable platform on which to implement a modern version of the *Network Tierra* project discussed in Box 1, with considerably less coding effort than was required for the original system.

Priorities identified by the authors for future development of YouShare and the ALife Zoo include facilities for visualization of a simulation's progress, and the provision of software hooks to more easily allow new systems to utilize the services provided in a modular fashion.

6.4.2 COEL

Another recent project aimed at providing a framework for ALife-related work is COEL, an open-source web-based chemistry simulation framework introduced in 2014 by Peter Banda and colleagues at Portland State University [11].[123] In contrast to the approach taken with YouShare of

123 http://coel-sim.org

providing infrastructure and virtual machine technology to run existing heterogeneous simulation software, COEL provides a single, purpose-built web-based simulation framework.

The COEL designers aimed not only to introduce a well-designed system that relieves researchers of having to reinvent the wheel by coding their own simulations, but also to create a simulation that: (1) runs on COEL's own grid rather than on the client's machine (as with YouShare); (2) allows geographically distributed teams to work together on a single platform; (3) provides remote database storage and backup facilities for experimental data; and (4) provides integrated visualization tools (a feature identified by the YouShare developers as an important next step).

COEL offers a unified, web-based environment for the definition, simulation, and analysis of chemical reaction networks. It can be run from any browser without installation—embedded visualization is implemented using Google's Chart API,[124] and the server side is implemented using a variety of modern technologies based on Java virtual machines (JVMs). Without the need for installation, the authors argue that COEL has a larger potential audience than existing desktop-based systems. They also suggest that the cloud-based data storage facilities promote collaboration, sharing of results, and building upon past work by others.

The authors' vision for COEL extends beyond chemical reaction network simulation: Their hope is for it to "become a common platform for diverse unconventional computing models" [11, p. 20].

6.4.3 Worldwide Infrastructure for Neuroevolution

Systems such as *Picbreeder* (Section 5.1.1) and related projects use the web as a platform for long-running, open, collaborative experiments in artificial evolution, but each required considerable effort to develop. Paul Szerlip and Ken Stanley are currently developing the Worldwide Infrastructure for Neuroevolution (WIN) project, which aims to provide a general-purpose solution that will greatly reduce the effort required to build such systems [124]. Also under development is an associated public web-based front end for ongoing experiments built with the WIN platform, called WIN Online.[125] WIN has been designed as a lightweight and expandable collection of event-driven Node.js[126] modules, allowing developers to use just those packages that they require.

The designers of WIN aim "to make it trivial to connect any individual or lab platform to the world, providing both a stream of online users, and archives of data and discoveries for later continuation" [124, p. 901]. Although currently still at prototype stage, the authors have demonstrated its use in rapidly reimplementing both *Picbreeder* and *IESoR* (the recent browser-based version of *Sodarace*, described in Section 4). The WIN-based reimplementation of the latter, *win-IESoR*, additionally featured interactive evolution, which was not present in the original *IESoR* implementation. These reimplementations show that WIN can be successfully employed to dramatically reduce coding effort by reusing standard libraries. Furthermore, the data-archiving facilities of the system provide the possibility of reuse of *results* for further evolution in future experiments.

6.4.4 Empirical

Several of the projects mentioned in previous sections have used the Emscripten[127] source-to-source compiler that converts C or C++ code into high-performance JavaScript.[128] The Empirical library,[129] currently being developed by Charles Ofria and others at Michigan State University, works with Emscripten to simplify the construction of ALife experimental test beds. Empirical provides tools to: (1) build data-driven web pages as part of Emscripten-compiled C/C++ software, (2) build dynamic configuration systems, (3) serialize results for easy analyses or continuation of runs, and

124 https://developers.google.com/chart
125 http://winark.org
126 http://nodejs.org
127 http://kripken.github.io/emscripten-site/
128 See http://devosoft.org/an-introduction-to-web-development-with-emscripten/ for a quickstart guide to working with Emscripten.
129 https://github.com/mercere99/Empirical

(4) hook into a variety of JavaScript packages, including D3.js,[130] for visualization. Many of the tools are built to work appropriately for either web distribution or native compilation, using any standard C++ compiler, and various high-performance core C++ tools are provided that are generally useful in scientific software. Empirical is currently being used to develop the full reimplementation of the *Avida* platform, described in Section 6.3.2.1.

6.4.5 Other Work

We conclude this subsection with a brief look at some other projects of relevance to WebAL frameworks.

The SimWorld Agent-based Grid Experimentation System (SWAGES), initially conceived over 15 years ago and more recently developed as a component of the Agent Development Environment (ADE) project,[131] is an extendable distributed experimentation platform for large-scale agent-based simulations [107, 106]. At a high level, the architecture is somewhat similar to that of COEL. Implemented in Java, the system allows for the automatic parallelization and distribution of simulations, as well as for providing analysis and visualization facilities, all controlled via a web-based interface. A full review of the field of distributed multi-agent systems is beyond the scope of this article, but such a review can be found in [103].[132]

Elsewhere, and also bearing some similarities to COEL (Section 6.4.2), Brucer Damer and colleagues' EvoGrid project sought to employ large-scale, distributed artificial chemistry simulations to study the origin of life.[133] The project was based upon a distributed architecture linking compute clusters via a web-based simulation management system. Some initial proof-of-concept results were reported in 2010–2012 [20, 21], but Damer and colleagues are currently concentrating on more traditional origins-of-life research (e.g., [18]).

In contrast to the kinds of projects mentioned above, which involve the development of large, comprehensive frameworks for distributed WebAL, an alternative approach is to use more general-purpose existing tools—separately or in combination—to create cheap (in terms of development effort), lightweight WebAL frameworks.

One example of such an approach is provided by Atanas Radenski, who showed how the MapReduce model[134] could be used to distribute large-scale lattice-based ALife simulations in the cloud [91]. Specifically, he demonstrated how discrete and continuous versions of Conway's *Game of Life* could be implemented on Amazon Elastic MapReduce.[135] Radenski investigated various optimization techniques for his approach, and discussed how his design could be used as a prototype for other such work.

Some interesting examples of lightweight approaches to web-based evolutionary algorithms have been published by Juan-Julián Merelo and colleagues in recent years. These include studies employing popular cloud-based storage services (specifically, Dropbox[136] and SugarSync[137]) [67], and others implementing JavaScript-based evolutionary algorithms [66, 65]. Other work has studied how the performance of different architectures of web-based evolutionary algorithms is affected when client nodes join and leave the cluster during an evolutionary run [51]. These studies present some interesting ideas that could be applied to WebAL projects. A full review of the general field of web-based distributed evolutionary algorithms is beyond the scope of this article, but recent reviews can be found in [30] and [31].

130 http://d3js.org/

131 http://ade.sourceforge.net/

132 We also give an honorable mention to breve.js (http://artificial.com/breve.js/), a slimmed-down, browser-based version of Jon Klein's popular Breve 3D simulation platform for multi-agent systems http://www.spiderland.org/. At the time of writing, breve.js is described as "work in progress."

133 http://www.evogrid.org/

134 https://en.wikipedia.org/wiki/MapReduce

135 https://aws.amazon.com/elasticmapreduce/

136 https://www.dropbox.com/

137 https://www.sugarsync.com/

6.5 New Directions for WebAL

The work discussed up to this point in Section 6 has fairly naturally fallen into a handful of distinct categories. In this final subsection, we discuss preliminary work in a couple of projects that look at somewhat different aspects of WebAL.

6.5.1 *EvoPopcorn:* Distributed Client-centric WebAL

Luis Zaman has recently described exploratory work with a distributed evolutionary system called *EvoPopcorn*, implemented with native web technology [137]. The main component of the system is a client-side application, written in JavaScript and HTML5, that presents the viewer with a simulation of a 2D arena in which simple organisms compete for food. Organisms die at random, and are replaced by mutated offspring of parents chosen according to their fitness (amount of food consumed). The resulting evolution of morphology and behavior of the organisms depends upon the distribution of food sources in the environment, which the user can manipulate.

Perhaps the most interesting aspect of this work in the current context is that Zaman extended the system so that evolved organisms could hop from one browser to another, using the WebSocket API.[138] Although the migration of organisms from one browser to another was physically routed through the system's server, this work stands in contrast to most of the projects discussed above in that computation is happening on the client side (in the browser) rather than on a central server. The server is only acting as a relay station to route organisms from one browser to another. The conceptual architecture of *EvoPopcorn* is therefore similar to that of the *Golem@Home* project described in Box 2.[139] The main novelty is that *EvoPopcorn* is implemented using native web technology that works by a user simply visiting the website, with no plug-ins or software installation required. This work hints at the growing possibility of a much more distributed kind of WebAL, in which ALife organisms live "in the wild" (on browsers and local storage on client machines around the world) rather than being penned in on a central server and only released to clients on demand.

6.5.2 The Internet as a Living System

All of the work discussed in preceding sections has used the web as a platform upon which to implement WebAL of one sort or another. Recent work by Mizuki Oka and colleagues takes a different perspective by asking the question: By measuring the characteristic dynamics and activity of the web, might we consider the web itself to be a living system?[140] In order to address this question, the dynamics of the web are examined in terms of four characteristics that are essentially associated with living systems: excitability, autonomy, homeostasis, and capacity to evolve.

A series of studies argues that aspects of the Internet can be regarded as: an *excitable medium*—specifically with reference to the dynamics of activity on social media sites [76]; an *autonomous system*—by applying the concepts of reactive and default modes of activity from brain sciences to social media and web search behavior [78]; a *homeostatic system*—by studying the adaptation and robustness of packet switching networks under varying data input loads [75]; and an *evolving system*—in relation to vocabularies used in tags on a social media system [77].

These studies regard the web as a complex chemical soup analogous to the primitive state of the Earth. The work is guided by the possibility that the web's massive scale, complex dynamism, open richness, and social character could potentially be developing living systems spontaneously. Thus,

138 http://www.w3.org/TR/websockets/

139 In a similar vein to *Golem@Home*, the long-running *DarwinBots* desktop ALife simulation (http://www.darwinbots.com) added an Internet mode in 2011 (http://wiki.darwinbots.com/w/Internet_Mode). If enabled, this mode allowed bots evolved on one user's machine to be teleported to other users' machines via an FTP server.

140 While the analogies between networked computer systems and living systems have been highlighted before (e.g., [28]), Oka et al.'s work takes the idea of the web as a living system more literally. Bedau et al. had previously suggested that social communities on the web might display some aspects of genuinely living systems (in their terminology, they are borderline between "primary" and "secondary" living technology) [12], but Oka et al. develop this idea much further.

the authors argue that it may be more profitable to study it using tools and concepts appropriate for understanding nervous systems, organisms, ecosystems, and society, rather than using more traditional analytical tools employed in engineering and technological systems.

Elsewhere, Yuki Takeichi and colleagues have recently reported a study of activity on Twitter during major sporting events [126]. On the basis of their analysis, the authors are led to regard Twitter as an emergent "social sensor"—a distributed bionic extension to human sensory capability that exists and evolves in cyberspace but senses events in the physical world.

7 Emerging Themes and Future Directions

In the previous sections we surveyed the current landscape of WebAL and have seen how it has developed and expanded over its first two decades. In this final section we highlight some emerging themes that are apparent in current WebAL, and consider how these topics might influence future work.

7.1 Human or Hybrid Computation and Crowd Creativity

A great deal of the work surveyed here, especially the evolutionary art and design systems covered in Section 6.1 and some of the earlier systems discussed in Sections 4 and 5, involves the idea of *human* or *hybrid computation*, where some part of the computation is performed by human users of the system. Many of these systems use humans for the selection stage, but some allow users to exert more direct control over the design of the evolving artefact—for example, *TechnoSphere* and *Nerve Garden* (Section 4), *DrawCompileEvolve* (Section 6.1.2), and *BrainCrafter* (Section 6.2.1). The *Organic Builder* project (Section 5.1.1) is another example of a human computation system, this time based upon artificial chemistries rather than evolutionary systems. Elsewhere, the web-based survey of the ALife community reported by Rasmussen and colleagues (discussed at the start of Section 5) provides a somewhat different example of using the web to harness the collective intelligence of a distributed group of users.

There is an increasingly large and active research community investigating methods for combining human and computational intelligence in appropriate ways so as to leverage and complement the strengths of both. Kosorukoff presented an interesting early exploration of different ways in which humans and computers can be deployed to create hybrid evolutionary algorithms [48]. There is a large literature on the more general areas of *human computation* and *crowd creativity*: For good reviews, see [60], [59], [90], and [136].[141] An ALife-oriented review of *crowdsourcing* and discussion of important factors determining the success of crowdsourcing platforms, together with a report of simulation studies investigating such parameters, can be found in [13].

As web-based distributed computation systems become more powerful (see Section 7.2) and infrastructure frameworks such a WIN (Section 6.4.3) emerge to simplify the development of such systems, human-computer hybrid architectures will undoubtedly remain a prominent feature of much WebAL research in future years.

7.2 Distributed Computation

The kind of web-based human computation described above is a special case of the more general concept of *distributed computation*. As mentioned in Section 3, theoretical and practical work on distributed artificial evolution goes back many decades. It is now becoming increasingly possible to use the web as a distributed computation platform: HTML5 and related APIs such as WebSocket,[142] Web Workers,[143] and Web Storage[144] allow these kinds of distributed computation systems to be implemented using native technology.

141 An interesting recent study that conceptualizes human decision making and creativity as evolutionary computation is described by Sayama and Dionne [105]. Elsewhere, Woolley and Stanley have also recently explored novel architectures for human-computer collaboration within an ALife context [135].

142 http://www.w3.org/TR/websockets/

143 http://www.w3.org/TR/workers/

144 http://www.w3.org/TR/webstorage/

At the same time, a number of technologies are currently being developed to allow fast client-side processing at speeds approaching those of local compiled code: Mozilla's asm.js[145] and Google's Native Client[146] are the most prominent of these. The Emscripten compiler,[147] used in several of the projects discussed in Section 6, converts C and C++ code to highly efficient JavaScript (using asm.js). Furthermore, the W3C has recently launched a WebAssembly Community Group[148] that aims to develop an efficient low-level programming language for fast client-side applications.[149] An alternative form of web-based distributed computation is provided by cloud computing platforms such as the established Amazon Web Services[150] and the more recent Google Cloud Platform.[151]

Elsewhere, David and Elena Ackley have advocated a more radical approach to designing distributed, scalable computational architectures that embrace stochastic events rather than attempt to prevent them at all costs [4]. This hardware-based approach has a strong ALife flavor that could be relevant to future WebAL work and ideally bring WebAL techniques into more mainstream applications.

All of these developments contribute to easier implementation and faster execution of client-side processing and web-based distributed computation systems. We can therefore expect to see more WebAL projects along these lines in the near future.

7.3 Persistent Systems

Traditional ALife experiments typically run for a few hours, days, or maybe weeks on a local machine or compute cluster: Data are collected, results are written up, and no further experimentation is done. A feature of many of the web-based ALife systems reviewed here is that they are designed to run indefinitely, for as long as there are users who are interested in interacting with them. This requirement represents a profound change in the way that experiments are designed, showing some parallels with long-running evolution studies of real biological systems such as Richard Lenski's *E. coli* long-term evolution experiment [29]. Such a shift in methodology also introduces challenges in developing appropriate methods for data capture and analysis—we expect improvements in these areas to be a feature of future work in the area. Furthermore, using the HTML5 APIs mentioned above for client-side processing and data storage, or cloud-based processing and data-storage services, these systems can potentially be massively distributed and extended across space as well as time. Systems such as *Pfeiffer* (Section 5.1.4) and *Picbreeder* (Section 5.1.1) give some indication of the potential benefits of web-based experiments, and many other types of long-term experiment can be imagined for future projects.

7.4 Cumulative Progress: Building upon Past Results

An important aspect of some of the projects reviewed, in addition to employing human computation and distributed, persistent systems, is that they allow users to build upon the results of previous users' work. Early examples include the International Interactive Genetic Art series and the *Electric Sheep* project in the 1990s (see Section 4). The enthusiast forums that developed around the *Creatures* game, allowing users to exchange their *Norns* with other users, can also be seen in this light (Section 4). This approach was introduced more explicitly in *Picbreeder*, with its *branching* method that allowed users to select and continue evolving images previously evolved by other users (Section 5.1.1). The branching process is also central to the more recent *EndlessForms* project (Section 6.1.1). From a more general perspective, open, web-based science frameworks such as COEL (Section 6.4.2) and WIN (Section 6.4.3), which provide data-archiving facilities for previous experiments, facilitate the process of using and building upon past results by others. Furthermore, platforms such as YouShare (Section 6.4.1)

145 http://en.wikipedia.org/wiki/Asm.js
146 http://en.wikipedia.org/wiki/Google_Native_Client
147 https://github.com/kripken/emscripten
148 https://www.w3.org/community/webassembly/
149 https://en.wikipedia.org/wiki/WebAssembly
150 http://aws.amazon.com/ (as used, for example, in Radenski's work [91] discussed in Section 6.4.5).
151 https://cloud.google.com/

facilitate archiving and reuse of whole software systems from previous research projects. Thus, these kinds of web-based platforms can potentially provide tremendous advantages for pursuing collaborative WebAL research (and similarly for many other branches of science as well).

7.5 Open Science, Open Education, and Public Outreach

Many of the systems reviewed here, particularly in Section 6.3 (Science and Education) and Section 6.4 (Frameworks for WebAL), are designed to be platforms for open science or open education. Others are focused on public outreach and communication of ALife concepts. These systems are generally available for free, and, being web-based, require no installation—most of this work now uses native technology rather than relying on browser plugins, which might have been the case in the past. This work hints at the huge potential offered by web-based systems to open up science and education in ALife (and, of course, other topics too) to a much wider audience. Over the coming years we will surely continue to witness great innovations in the way that research, education, and outreach are conducted on the web, in ALife as in many other academic disciplines.[152]

7.6 The Web as an Arena for Multi-user Competitions

In addition to providing mechanisms for collaboration, web platforms can also provide arenas in which competitions can be held between agents submitted by multiple users. Examples of work employing this kind of approach include *Sodarace* (Section 4), *NERO* (Box 2), and *EvoCommander* (Section 6.2.1). Taking a somewhat different approach, *Galactic Arms Race* (*GAR*) (Box 2) investigated how in-game content can be evolved based upon the behavior of multiple concurrent online players. The user challenges set in *Organic Builder* (Section 5.1.1) can also be seen as a form of multi-user competition, although the discussion of results on the online forum also gave the project a more cooperative flavor. It is likely that many additional forms of multi-user games and competitions will be explored in future WebAL projects.

7.7 Client-centric ALife: WebAL "in the Wild"

Some of the early publications on WebAL, such as Ray's article on *Network Tierra* [95] (Box 1) and Langdon's article on *Pfeiffer* [49] (Section 5.1.4), discuss the possibility of ALife agents roaming the Internet and evolving in the complex environment that it provides. In the architectures employed by most of the work described here, the system's server plays a central role in performing computations, farming data to client browsers, and receiving the results. However, one can imagine much more client-centric architectures whereby artificial life forms exist primarily on users' machines (made persistent between browser sessions by client-side storage mechanisms such as cookies or the new Web Storage API), and the server is used primarily as a routing system to allow organisms to migrate from one client to another (which can now be implemented natively using the WebSocket API).[153] There are glimmers of this approach in the *Pfeiffer* system (Section 5.1.4), and it is more strongly emphasized in the recent exploratory work on *EvoPopcorn* (Section 6.5.1) and in the forthcoming augmented-reality mobile games *Polly Peck's Journey* and *TechnoSphere 2.0* (Section 6.2.2). It seems likely that this kind of ability for free-roaming agency—for WebAL "in the wild"—could be explored and exploited much more thoroughly, and we expect many more developments in this area in future work. However, for this kind of work to truly flourish, important concerns about safety, security, and preventing free-roaming ALife agents from evolving out of control—concerns raised by, among others, Chris Langton in the early days of ALife (see Section 3)—must be fully addressed.

152 See [72] for a good review and outlook on the topic of *citizen science*, which intersects open science, education, and outreach and is therefore relevant to the future direction of the WebAL projects discussed here.

153 In other words, effectively a persistent, native, worldwide, peer-to-peer (P2P) architecture.

7.8 Cloud APIs

The work by Auerbach [9] (Section 6.1.2) illustrates an approach to utilizing cloud interfaces and APIs (in his case, Google Search by Image) as a component of a computational intelligence system. Of the work reviewed here, this is the only example of a WebAL system that directly utilizes a cloud API to provide an intelligent component to its architecture. Nguyen and colleagues' work [73], also described in Section 6.1.2, makes indirect use of Amazon's Mechanical Turk web-based crowdsourcing platform. It is not hard to imagine ways in which future WebAL systems could make more direct use of Mechanical Turk. More broadly, we can imagine many other ways in which cloud APIs could be employed to provide enhanced capabilities to WebAL systems. As the number of cloud services and open data sources continue to expand and Web API ecosystem architectures mature, we can expect to see many more WebAL systems using this kind of approach in future.

7.9 Crowdfunding

While not related to WebAL technology as such, another important way in which the web can enhance ALife is through *crowdfunding* of research and applications.

In 2011, Steve Grand (author of the *Creatures* game discussed in Section 4) successfully secured Kickstarter funding of nearly US$57,000 to develop a new ALife-powered game called *Grandroids*, currently still under development.[154] More recently, the *OpenWorm* project (see Section 6.3.2.5) has raised substantial funds through crowdfunding efforts, including over US$120,000 through a Kickstarter campaign in 2014. Another example is the company *Wiggle Planet*, discussed in Section 6.2.2, which raised over US$15,000 in 2014 through a Kickstarter campaign to develop its augmented-reality ALife game.

Among them, these three projects have raised nearly US$200,000 of funding through Kickstarter. These examples demonstrate that it is possible (although still far from easy) to obtain substantial funding for ALife projects via crowdfunding.

8 Conclusion

We have explored many different ways in which web technologies and ALife techniques have intersected in past and current work. The projects we have reviewed demonstrate the diverse domains in which WebAL has been applied, including: collaborative design; human computation; education; outreach; persistent and long-running experiments; the archiving, sharing, reproduction, and reuse of scientific experiments and platforms; collaborative open science; art; computer games; crowdfunding; and more besides.

As web technology continues to develop, and particularly with the move towards native APIs in place of proprietary plugins, the potential for developing complex web-based ALife research and applications grows greater each year.

Whether or not a WebAL project is primarily focused on education or public outreach, the very nature of the web means that WebAL research is inherently open and can reach a wide audience (unless steps are taken to actively prevent this accessibility). As funding councils around the world place increasing emphasis on the public understanding of science, WebAL is well placed to play an important role in the communication of ALife research to a wide and diverse audience. Furthermore, WebAL not only enables wide dissemination of results, but it also promotes public engagement with and participation in ALife research.

Looking back over the research reviewed here, it is clear that great strides have been made over the last 21 years. However, as web technology and APIs develop, it is likely that current work represents only the tip of the iceberg of what could be possible. The work surveyed here represents a great showcase of some of the possibilities of WebAL, and yet we suspect there are many other

154 The initial target platforms for the game are PCs and Macs. For more details, see https://www.kickstarter.com/projects/1508284443/grandroids-real-artificial-life-on-your-pc.

possibilities, some as yet unimagined. Advances will doubtless be made in all of the areas outlined in our discussion of emerging themes and future directions (Section 7), and likely in completely different areas as well.

It is a truly exciting time to be involved in WebAL research. We expect the current rapid pace of development to continue, and indeed to accelerate, over the next few years. We look forward to witnessing the advances and achievements, both expected and unexpected, that will emerge from these efforts.

Acknowledgments

We would like to thank two anonymous reviewers for their insightful comments and constructive criticism on an earlier draft of this article.

Thanks to Alan Dorin for providing references to relevant early work on ALife art. Thanks also to Bruce Damer, Steve Grand, Tim Hutton, Jane Prophet, Craig Reynolds, Hiroki Sayama, Terence Soule, and Sebastian von Mammen for verifying details relating to their work.

We gratefully acknowledge use of Overleaf, the free, online collaborative LaTEX authoring tool (https://www.overleaf.com/), in preparing the draft of this article.

Tim Taylor and Simon Hickinbotham acknowledge funding from the EU FP7 project EvoEvo, grant number 610427. Joshua Auerbach acknowledges funding from the EU FP7 project Insight, grant agreement number 308943. Josh Bongard acknowledges funding from the National Science Foundation awards PECASE-0953837 and INSPIRE-1344227. Jeff Clune was supported by an NSF CAREER award, grant number 1453549. Charles Ofria acknowledges funding for the BEACON Center for the Study of Evolution in Action from the US National Science Foundation, under Cooperative Agreement DBI-0939454. Jason Yosinski was supported by a NASA Space Technology Research Fellowship.

Image credits: Figure 2 courtesy of Jane Prophet, used with permission; Figure 3 courtesy of http://www.biota.org/nervegarden/publish.html, used according to the licensing information on that page; Figure 6 is a screenshot taken by one of the authors (T.T.), used with permission of Bill Langdon; Figure 4 courtesy of Tim Hutton, used with permission; Figure 13 courtesy of Sebastian von Mammem, used with permission. All other figures were provided by the authors.

References

1. Ackley, D. H. (1996). ccr: A network of worlds for research. In C. G. Langton & K. Shimohara (Eds.), *Artificial life V: Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems* (pp. 116–123). Cambridge, MA: MIT Press.
2. Ackley, D. H. (2000). Real artificial life: Where we may be. In M. A. Bedau, J. S. McCaskill, N. H. Packard, & S. Rasmussen (Eds.), *Artificial life VII: Proceedings of the Seventh International Conference on Artificial Life* (pp. 487–496). Cambridge, MA: MIT Press.
3. Ackley, D. H. (2013). Bespoke physics for living technology. *Artificial Life*, *19*(3–4), 347–364.
4. Ackley, D. H., & Ackley, E. S. (2015). Artificial life programming in the robust-first attractor. In P. Andrews, L. Caves, R. Doursat, S. Hickinbotham, F. Polack, S. Stepney, T. Taylor, & J. Timmis (Eds.), *Proceedings of the European Conference on Artificial Life 2015* (pp. 554–561). Cambridge, MA: MIT Press.
5. Anderson, D. P., Cobb, J., Korpela, E., Lebofsky, M., & Werthimer, D. (2002). Seti@Home: An experiment in public-resource computing. *Communications of the ACM*, *45*(11), 56–61.
6. Astor, J. C., & Adami, C. (1998). Development and evolution of neural networks in an artificial chemistry. In *Proceedings of the Third German Workshop on Artificial Life*. Bochum: Verlag Harri Deutsch. arXiv preprint: http://arxiv.org/abs/adap-org/9807003.
7. Astor, J. C., & Adami, C. (2000). A developmental model for the evolution of artificial neural networks. *Artificial Life*, *6*(3), 189–218.
8. Auerbach, J., Aydin, D., Maesani, A., Kornatowski, P., Cieslewski, T., Heitz, G., Fernando, P., Loshchilov, I., Daler, L., & Floreano, D. (2014). RoboGen: Robot generation through artificial evolution.

In H. Sayama, J. Rieffel, S. Risi, R. Doursat, & H. Lipson (Eds.), *Artificial life 14: Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems* (pp. 136–137). Cambridge, MA: MIT Press. http://www.robogen.org/.

9. Auerbach, J. E. (2012). Automated evolution of interesting images. In C. Adami, D. M. Bryson, C. Ofria, & R. T. Pennock (Eds.), *Artificial life 13: Proceedings of the Thirteenth International Conference on the Simulation and Synthesis of Living Systems* (pp. 629–630). Cambridge, MA: MIT Press.
10. Balfour, S. P. (2013). Assessing writing in MOOCs: Automated essay scoring and calibrated peer review. *Research and Practice in Assessment*, *8*, 40–48.
11. Banda, P., Blount, D., & Teuscher, C. (2014). COEL: A Web-based chemistry simulation framework. http://arxiv.org/abs/1407.4027.
12. Bedau, M. A., McCaskill, J. S., Packard, N. H., & Rasmussen, S. (2010). Living technology: Exploiting life's principles in technology. *Artificial Life*, *16*(1), 89–97.
13. Buecheler, T., Lonigro, R., Füchslin, R. M., & Pfeifer, R. (2011). Modeling and simulating crowdsourcing as a complex biological system: Human crowds manifesting collective intelligence on the Internet. In T. Lenaerts, M. Giacobini, H. Bersini, P. Bourgine, M. Dorigo, & R. Doursat (Eds.), *Advances in artificial life, ECAL 2011* (pp. 109–116). Cambridge, MA: MIT Press.
14. Cantú-Paz, E. (1998). A survey of parallel genetic algorithms. *Calculateurs parallèles, réseaux et systèmes répartis*, *10*(2), 141–171.
15. Carlson, S. (2000). Boids of a feather flock together. *Scientific American*, *283*(5), 112–114.
16. Cliff, D., & Grand, S. (1999). The *Creatures* global digital ecosystem. *Artificial Life*, *5*(1), 77–94.
17. Clune, J., & Lipson, H. (2011). Evolving three-dimensional objects with a generative encoding inspired by developmental biology. In T. Lenaerts, M. Giacobini, H. Bersini, P. Bourgine, M. Dorigo, & R. Doursat (Eds.), *Advances in artificial life, ECAL 2011* (pp. 141–148). Cambridge, MA: MIT Press.
18. Damer, B., & Deamer, D. (2015). Coupled phases and combinatorial selection in fluctuating hydrothermal pools: A scenario to guide experimental approaches to the origin of cellular life. *Life*, *5*(1), 872–887.
19. Damer, B., Marcelo, K., Revi, F., Furmanski, T., & Laurel, C. (2005). Nerve Garden: Germinating biological metaphors in net-based virtual worlds. In A. Adamatzky & M. Komosinski (Eds.), *Artificial life models in software*, chap. 3 (pp. 67–80). Berlin: Springer.
20. Damer, B., Newman, P., Gordon, R., Barbalet, T., Deamer, D., & Norkus, R. (2010). The EvoGrid: A framework for distributed artificial chemistry cameo simulations supporting computational origins of life endeavors. In H. Fellermann, M. Dörr, M. M. Hanczy, L. L. Laursen, S. Maurer, D. Merkle, P.-A. Monnard, K. Støy, & S. Rasmussen (Eds.), *Artificial life XII: Proceedings of the Twelfth International Conference on the Synthesis and Simulation of Living Systems* (pp. 73–79). Cambridge, MA: MIT Press.
21. Damer, B., Newman, P., Norkus, R., Graham, J., Gordon, R., & Barbalet, T. (2012). Cyberbiogenesis and the EvoGrid: A twenty-first century grand challenge. In J. Seckbach (Ed.), *Genesis—in the beginning* (pp. 267–288). Berlin: Springer.
22. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition, 2009* (pp. 248–255). Piscataway, NJ: IEEE.
23. Denning, P. J. (1989). The science of computing: The Internet worm. *American Scientist*, *77*(2), 126–128. http://www.jstor.org/stable/27855650.
24. Dick, P. K. (1968). *Do androids dream of electric sheep?* New York: Doubleday.
25. Draves, S. (2008). Evolution and collective intelligence of the electric sheep. In J. Romero & P. Machado (Eds.), *The art of artificial evolution* (pp. 63–78). Berlin: Springer. http://dx.doi.org/10.1007/978-3-540-72877-1_3.
26. Durland, S. (1987). Defining the image as place: A conversation with Kit Galloway, Sherrie Rabinowitz and Gene Youngblood. *High Performance*, *37*, 52–59. Text available at http://www.ecafe.com/museum/hp_gy_1987/hp_gy_1987.html.
27. Farrell, D., Kostkova, P., Lazareck, L., Weerasinghe, D., Weinberg, J., Lecky, D. M., Adriaenssens, N., Herotová, T. K., Holt, J., Touboul, P., et al. (2011). Developing e-Bug Web games to teach microbiology. *Journal of Antimicrobial Chemotherapy*, *66*(suppl 5), v33–v38.

28. Forrest, S., Balthrop, J., Glickman, M., & Ackley, D. (2005). Computation in the wild. In E. Jen (Ed.), *Robust design: A repertoire of biological, ecological, and engineering case studies* (pp. 207–230). Oxford, UK: Oxford University Press.

29. Fox, J. W., & Lenski, R. E. (2015). From here to eternity—the theory and practice of a really long experiment. *PLOS Biology*, *13*(6), e1002185.

30. García-Sánchez, P., González, J., Castillo, P. A., Arenas, M. G., & Merelo-Guervós, J. (2013). Service oriented evolutionary algorithms. *Soft Computing*, *17*(6), 1059–1075.

31. García-Valdez, M., Trujillo, L., Fernández de Vega, F., Merelo Guervós, J. J., & Olague, G. (2013). EvoSpace: A distributed evolutionary platform based on the tuple space model. In A. I. Esparcia-Alcázar (Ed.), *Applications of evolutionary computation* (pp. 499–508). Berlin: Springer. http://dx.doi.org/10.1007/978-3-642-37192-9_50.

32. Goldberg, K., Mascha, M., Gentner, S., Rothenberg, N., Sutter, C., & Wiegley, J. (1995). Desktop teleoperation via the World Wide Web. In B. Edmonds, N. Gilbert, S. Gustafson, D. Hales, & N. Krasnogor (Eds.), *Proceedings of the 1995 IEEE International Conference on Robotics and Automation* (pp. 654–659). Piscataway, NJ: IEEE.

33. Goldsby, H., Dornhaus, A., Kerr, B., & Ofria, C. (2012). Task switching costs promote the evolution of division of labor and shifts in individuality. *Proceedings of the National Academy of Sciences of the USA*, *34*(109), 13686–13691.

34. Hastings, E. J., Guha, R. K., & Stanley, K. O. (2009). Evolving content in the Galactic Arms Race video game. In *Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG09)* (pp. 241–248). Piscataway, NJ: IEEE.

35. Hastings, E. J., Guha, R. K., & Stanley, K. O. (2009). Interactive evolution of particle systems for computer graphics and animation. *IEEE Transactions on Evolutionary Computation*, *13*(2), 418–432.

36. Hickinbotham, S., Clark, E., Stepney, S., Clarke, T., Nellis, A., Pay, M., & Young, P. (2012). *Specification of the Stringmol chemical programming language version 0.2* (Tech. Rep. YCS-2010-458). University of York.

37. Hickinbotham, S., Weeks, M., & Austin, J. (2013). The ALife Zoo: Cross-browser, platform-agnostic hosting of artificial life simulations. In P. Liò, O. Miglino, G. Nicosia, S. Nolfi, & M. Pavone (Eds.), *Advances in Artificial Life, ECAL 2013* (pp. 71–78). Cambridge, MA: MIT Press.

38. Hickinbotham, S. J., Clark, E., Stepney, S., Clarke, T., Nellis, A., Pay, M., & Young, P. (2010). Diversity from a monoculture—effects of mutation-on-copy in a string-based artificial chemistry. In H. Fellermann, M. Dörr, M. M. Hanczy, L. L. Laursen, S. Maurer, D. Merkle, P.-A. Monnard, K. Støy, & S. Rasmussen (Eds.), *Artificial life XII: Proceedings of the Twelfth International Conference on the Synthesis and Simulation of Living Systems* (pp. 24–31). Cambridge, MA: MIT Press.

39. Hutton, T. (2009). The organic builder: A public experiment in artificial chemistries and self-replication. *Artificial Life*, *15*(1), 21–28.

40. Jallov, D., Risi, S., & Togelius, J. (2016). EvoCommander: A novel game based on evolving and switching between artificial brains. *IEEE Transactions on Computational Intelligence and AI in Games*, *PP*(99), 1–1.

41. Jepsen, D. (1999). They're dead, Jim: Natural selection goes awry in this depressing update. *Computer Gaming World*, *174*, 364–366.

42. Johnson, N., Rasmussen, S., Joslyn, C., Rocha, L., Smith, S., & Kantor, M. (1998). Symbiotic intelligence: Self-organizing knowledge on distributed networks driven by human interaction. In C. Adami, R. Belew, H. Kitano, & C. Taylor (Eds.), *Artificial life VI: Proceedings of the Sixth International Conference on Artificial Life* (pp. 403–407). Cambridge, MA: MIT Press.

43. Jónsson, B., Hoover, A., & Risi, S. (2015). Interactively evolving compositional sound synthesis networks. In *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*. New York: ACM.

44. Kelly, K. (1994). *Out of control: The new biology of machines, social systems and the economic world*. Boston: Addison-Wesley.

45. Kephart, J. O., & Chess, D. M. (2003). The vision of autonomic computing. *Computer*, *36*(1), 41–50.

46. Khatib, F., DiMaio, F., Cooper, S., Kazmierczyk, M., Gilski, M., Krzywda, S., Zabranska, H., Pichova, I., Thompson, J., Popović, Z., et al. (2011). Crystal structure of a monomeric retroviral protease solved by protein folding game players. *Nature Structural & Molecular Biology*, *18*(10), 1175–1177.

47. Klusch, M. (Ed.) (1999). *Intelligent information agents: Agent-based information discovery and management on the Internet*. Berlin: Springer.

48. Kosorukoff, A. (2001). Human based genetic algorithm. In *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 5 (pp. 3464–3469). Piscataway, NJ: IEEE.

49. Langdon, W. B. (2005). Pfeiffer: A distributed open-ended evolutionary system. In *AISB'05: Proceedings of the Joint Symposium on Socially Inspired Computing (METAS 2005)* (pp. 7–13). London: SSAISB.

50. Langton, C. G. (1991). Introduction. In C. G. Langton, C. Taylor, J. D. Farmer, & S. Rasmussen (Eds.), *Artificial life II* (pp. 3–24). Boston: Addison-Wesley.

51. Laredo, J. L. J., Bouvry, P., González, D. L., de Vega, F. F., Arenas, M. G., Merelo, J., & Fernandes, C. M. (2014). Designing robust volunteer-based evolutionary algorithms. *Genetic Programming and Evolvable Machines*, *15*(3), 221–244.

52. Lewis, M. (2008). Evolutionary visual art and design. In J. Romero & P. Machado (Eds.), *The art of artificial evolution: A handbook on evolutionary art and music* (pp. 3–37). Berlin: Springer.

53. Lim, S. L., & Bentley, P. J. (2012). App epidemics: Modelling the effects of publicity in a mobile app ecosystem. In C. Adami, D. M. Bryson, C. Ofria, & R. T. Pennock (Eds.), *Artificial life 13: Proceedings of the Thirteenth International Conference on the Simulation and Synthesis of Living Systems* (pp. 202–209). Cambridge, MA: MIT Press.

54. Lim, S. L., & Bentley, P. J. (2012). How to be a successful app developer: Lessons from the simulation of an app ecosystem. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, GECCO '12* (pp. 129–136). New York: ACM. http://doi.acm.org/10.1145/2330163.2330182.

55. Lim, S. L., & Bentley, P. J. (2013). Investigating app store ranking algorithms using a simulation of mobile app ecosystems. In *2013 IEEE Congress on Evolutionary Computation (CEC)* (pp. 2672–2679). Piscataway, NJ: IEEE.

56. Lintott, C. J., Schawinski, K., Slosar, A., Land, K., Bamford, S., Thomas, D., Raddick, M. J., Nichol, R. C., Szalay, A., Andreescu, D., et al. (2008). Galaxy Zoo: Morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. *Monthly Notices of the Royal Astronomical Society*, *389*(3), 1179–1189.

57. Lipson, H., & Pollack, J. B. (2000). Automatic design and manufacture of robotic lifeforms. *Nature*, *406*(6799), 974–978.

58. Lowood, H. (2014). Stanford University libraries acquire MUD1 source code. http://web.stanford.edu/group/htgg/cgi-bin/drupal/?q=node/1191.

59. Maher, M. L. (2010). Design creativity research: From the individual to the crowd. In T. Taura & Y. Nagai (Eds.), *Design creativity 2010 (Proceedings of the International Conference on Design Creativity)* (pp. 41–47). Berlin: Springer.

60. Malone, T. W., Laubacher, R., & Dellarocas, C. (2009). *Harnessing crowds: Mapping the genome of collective intelligence* (Tech. Rep. 2009-001). MIT Center for Collective Intelligence.

61. Mataric, M. J. (2004). Robotics education for all ages. In *Proceedings, AAAI Spring Symposium on Accessible, Hands-on AI and Robotics Education*. Menlo Park, CA: AAAI Press.

62. McLaughlin, M. L., Osborne, K. K., & Ellison, N. B. (1995). Virtual community in a telepresence environment. In S. Jones (Ed.), *Virtual culture: Identity and communication in cybersociety* (pp. 146–168). Thousand Oaks, CA: Sage Publications.

63. McOwan, P. W., & Burton, E. J. (2005). Sodarace: Adventures in artificial life. In A. Adamatzky & M. Komosinski (Eds.), *Artificial life models in software*, chap. 5 (pp. 97–111). Berlin: Springer.

64. Menczer, F., Belew, R. K., & Willuhn, W. (1995). Artificial life applied to adaptive information agents. In C. Knoblock & A. Levy (Eds.), *Information gathering from heterogeneous, distributed environments: Papers from the AAAI Spring Symposium* (pp. 128–132). (Technical Report SS-95-08.) Menlo Park, CA: AAAI Press.

65. Merelo, J.-J., Castillo, P., Mora, A., Esparcia-Alcázar, A., & Rivas-Santos, V. (2014). NodEO, a multi-paradigm distributed evolutionary algorithm platform in JavaScript. In *GECCO Comp '14: Companion publication of the Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation* (pp. 1155–1162). New York: ACM.

66. Merelo-Guervós, J. J., & García-Sánchez, P. (2015). Modeling browser-based distributed evolutionary computation systems. arXiv preprint arXiv:1503.06424.

67. Meri, K., Arenas, M. G., Mora, A. M., Merelo, J., Castillo, P. A., García-Sánchez, P., & Laredo, J. L. J. (2013). Cloud-based evolutionary algorithms: An algorithmic study. *Natural Computing*, *12*(2), 135–147.

68. Métivier, M., Lattaud, C., & Heudin, J.-C. (2002). A stress-based speciation model in LifeDrop. In R. Standish, M. A. Bedau, & H. A. Abbass (Eds.), *Artificial life VIII: Proceedings of the Eighth International Conference on Artificial Life* (pp. 121–126). Cambridge, MA: MIT Press.

69. Mignonneau, L., & Sommerer, C. (2001). Creating artificial life for interactive art and entertainment. *Leonardo*, *34*(4), 303–307.

70. Moharreri, K., Ha, M., & Nehm, R. H. (2014). EvoGrader: An online formative assessment tool for automatically evaluating written evolutionary explanations. *Evolution: Education and Outreach*, *7*(15).

71. Moore, J. M., Clark, A. J., & McKinley, P. K. (2014). Evolutionary robotics on the Web with WebGL and Javascript. In *Proceedings of WebAL-1: Workshop on Artificial Life and the Web 2014, held at the 14th International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14)*. http://arxiv.org/abs/1406.3337.

72. Newman, G., Wiggins, A., Crall, A., Graham, E., Newman, S., & Crowston, K. (2012). The future of citizen science: Emerging technologies and shifting paradigms. *Frontiers in Ecology and the Environment*, *10*(6), 298–304.

73. Nguyen, A., Yosinski, J., & Clune, J. (2015). Innovation engines: Automated creativity and improved stochastic optimization via deep learning. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*. New York: ACM.

74. Ofria, C., Bryson, D. M., & Wilke, C. O. (2009). Avida. In M. Komosinski & A. Adamatsky (Eds.), *Artificial life models in software* (2nd ed.) (pp. 3–35). Berlin: Springer.

75. Oka, M., Abe, H., & Ikegami, T. (2015). Dynamic homeostasis in packet switching networks. *Adaptive Behavior*, *23*(1), 50–63.

76. Oka, M., Hashimoto, Y., & Ikegami, T. (2014). Self-organization on social media: Endo-exo bursts and baseline fluctuations. *PLOS ONE*, *9*(10), e109293.

77. Oka, M., Hashimoto, Y., & Ikegami, T. (2015). Open-ended evolution in a Web system. In *Late breaking papers at the European Conference on Artificial Life (ECAL 2015)*. https://www.cs.york.ac.uk/nature/ecal2015/late-breaking/159.pdf.

78. Oka, M., & Ikegami, T. (2013). Exploring default mode and information flow on the Web. *PLOS ONE*, *8*(4), e60398.

79. Ølsted, P. T., Ma, B., & Risi, S. (2015). Interactive evolution of levels for a competitive multiplayer FPS. In *Proceedings of the 2015 IEEE Congress on Evolutionary Computation*. Piscataway, NJ: IEEE.

80. Pagliarini, L., Dolan, A., Menczer, F., & Lund, H. H. (1998). ALife meets Web: Lessons learned. In J.-C. Heudin (Ed.), *Virtual worlds: First International Conference, VW'98, Paris, France, July 1–3, 1998, Proceedings* (pp. 156–167). Berlin: Springer. http://dx.doi.org/10.1007/3-540-68686-X_15.

81. Pagliarini, L., Lund, H. H., Miglino, O., & Parisi, D. (1996). Artificial life: A new way to build educational and therapeutic games. In C. G. Langton & K. Shimohara (Eds.), *Artificial life V: Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems* (pp. 152–156). Cambridge, MA: MIT Press.

82. Palyanov, A., Khayrulin, S., Larson, S. D., & Dibert, A. (2012). Towards a virtual *C. elegans*: A framework for simulation and visualization of the neuromuscular system in a 3D physical environment. *In Silico Biology*, *11*(3), 137–147.

83. Parkin, S. (2014). The man who made a game to change the world. Eurogamer (website). http://www.eurogamer.net/articles/2014-10-30-the-utopia-that-never-died.

84. Paulsen, K. (2013). Image as place: The phenomenal screen in Kit Galloway and Sherrie Rabinowitz's Satellite Arts 1977. *Leonardo Electronical Almanac*, *19*(2), 98–111.

85. Pennock, R. T. (2007). Learning evolution and the nature of science using evolutionary computing and artificial life. *McGill Journal of Education*, *42*, 211–224.

86. Piskur, J., Greve, P., Togelius, J., & Risi, S. (2015). Braincrafter: An investigation into human-based neural network engineering. In *Proceedings of the IEEE Congress on Evolutionary Computation (IEEE CEC 2015)*. Piscataway, NJ: IEEE.

87. Prophet, J. (1996). Sublime ecologies and artistic endeavors: Artificial life and interactivity in the online project TechnoSphere. *Leonardo*, *29*(5), 339–344.

88. Prophet, J. (2001). *TechnoSphere*: "Real" time, "artificial" life. *Leonardo*, *34*(4), 309–312.

89. Prophet, J., & Pritchard, H. (2015). SE Asian Ubicomp and ALife: Roaming and homing with TechnoSphere 2.0 computational companions. In *CHI 2015 extended abstracts (Workshop on Between the Lines: Reevaluating the online/offline binary)*. New York: ACM.

90. Quinn, A. J., & Bederson, B. B. (2011). Human computation: A survey and taxonomy of a growing field. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1403–1412). New York: ACM.

91. Radenski, A. (2013). Using MapReduce streaming for distributed life simulation on the cloud. In P. Liò, O. Miglino, G. Nicosia, S. Nolfi, & M. Pavone (Eds.), *Advances in artificial life, ECAL 2013* (pp. 284–291). Cambridge, MA: MIT Press.

92. Rasmussen, S., Knudsen, C., Feldberg, R., & Hindsholm, M. (1990). The Coreworld: Emergence and evolution of cooperative structures in a computational chemistry. *Physica D: Nonlinear Phenomena*, *42*(1), 111–134.

93. Rasmussen, S., Raven, M. J., Keating, G. N., & Bedau, M. A. (2003). Collective intelligence of the artificial life community on its own successes, failures, and future. *Artificial Life*, *9*(2), 207–235.

94. Ray, T. S. (1991). An approach to the synthesis of life. In C. G. Langton, C. Taylor, J. D. Farmer, & S. Rasmussen (Eds.), *Artificial life II* (pp. 371–408). Boston: Addison-Wesley.

95. Ray, T. S. (1995). A proposal to create a network-wide biodiversity reserve for digital organisms (Tech. rep. TR-H-133). ATR, Japan.

96. Ray, T. S. (1998). Selecting naturally for differentiation: Preliminary evolutionary results. *Complexity*, *3*(5), 25–33.

97. Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *Computer Graphics (SIGGRAPH '87 Conference Proceedings)*, *21*(4), 25–34.

98. Risi, S. (2013). A compiler for CPPNs: Transforming phenotypic descriptions into genotypic representations. In *Proceedings of the 2013 AAAI Fall Symposium on How Should Intelligence be Abstracted in AI Research*. Menlo Park, CA: AAAI Press.

99. Risi, S., Lehman, J., D'Ambrosio, D. B., Hall, R., & Stanley, K. O. (2012). Combining search-based procedural content generation and social gaming in the Petalz video game. In *Proceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE 2012)*. Menlo Park, CA: AAAI Press.

100. Risi, S., Lehman, J., D'Ambrosio, D. B., Hall, R., & Stanley, K. O. (2015). Petalz: Search-based procedural content generation for the casual gamer. In *IEEE Transactions on Computational Intelligence and AI in Games*, *PP*(99), 1–1.

101. Risi, S., Lehman, J., D'Ambrosio, D. B., & Stanley, K. O. (2014). Automatically categorizing procedurally generated content for collecting games. In *Proceedings of the Workshop on Procedural Content Generation in Games (PCG) at the 9th International Conference on the Foundations of Digital Games (FDG-2014)*. New York: ACM.

102. Risi, S., Zhang, J., Taarnby, R., Greve, P., Piskur, J., Liapis, A., & Togelius, J. (2014). The case for a mixed-initiative collaborative neuroevolution approach. In *Proceedings of WebAL-1: Workshop on Artificial Life and the Web 2014, held at the 14th International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14)*. http://adsabs.harvard.edu/abs/2014arXiv1408.0998R.

103. Rousset, A., Herrmann, B., Lang, C., & Philippe, L. (2014). A survey on parallel and distributed multi-agent systems. In *Euro-Par 2014: Parallel Processing Workshops* (pp. 371–382). Berlin: Springer.

104. Sayama, H. (2009). Swarm chemistry. *Artificial Life*, *15*(1), 105–114.

105. Sayama, H., & Dionne, S. D. (2014). Studying collective human decision making and creativity with evolutionary computation. http://arxiv.org/abs/1406.6291.

106. Scheutz, M., & Harris, J. (2012). An overview of the SimWorld agent-based grid experimentation system. In W. Dubitzky, K. Kurowski, & B. Schott (Eds.), *Large-scale computing techniques for complex system simulations* (pp. 59–80). Hoboken, NJ: Wiley.

107. Scheutz, M., Schermerhorn, P., Connaughton, R., & Dingler, A. (2006). SWAGES—an extendable distributed experimentation system for large-scale agent-based ALife simulations. In L. M. Rocha, L. S. Yaeger, M. A. Bedau, D. Floreano, R. L. Goldstone, & A. Vespignani (Eds.), *Artificial life X: Proceedings of the Tenth International Conference on the Simulation and Synthesis of Living Systems* (pp. 412–419). Cambridge, MA: MIT Press (Bradford Books).

108. Secretan, J., Beato, N., D'Ambrosio, D. B., Rodriguez, A., Campbell, A., Folsom-Kovarik, J. T., & Stanley, K. O. (2011). Picbreeder: A case study in collaborative evolutionary exploration of design space. *Evolutionary Computation*, *19*(3), 345–371. http://www.mitpressjournals.org/doi/pdf/10.1162/EVCO_a_00030.

109. Secretan, J., Beato, N., D'Ambrosio, D. B., Rodriguez, A., Campbell, A., & Stanley, K. O. (2008). Picbreeder: Evolving pictures collaboratively online. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1759–1768). New York: ACM.

110. Sims, K. (1994). Evolving 3D morphology and behavior by competition. In R. A. Brooks & P. Maes (Eds.), *Artificial life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems* (pp. 353–372). Cambridge, MA: MIT Press.

111. Sklar, E. (2007). Software review: NetLogo, a multi-agent simulation environment. *Artificial Life*, *13*(3), 303–311.

112. Sommerer, C., & Mignonneau, L. (1999). VERBARIUM and LIFE SPACIES: Creating a visual language by transcoding text into form on the Internet. In *IEEE Symposium on Visual Languages (VL'99) conference proceedings* (pp. 90–95). Piscataway, NJ: IEEE.

113. Sommerer, C., & Mignonneau, L. (2000). Modeling emergence of complexity: The application of complex system and origin of life theory to interactive art on the Internet. In M. A. Bedau, J. S. McCaskill, N. H. Packard, & S. Rasmussen (Eds.), *Artificial life VII: Proceedings of the Seventh International Conference on Artificial Life* (pp. 547–554). Cambridge, MA: MIT Press.

114. Spafford, E. H. (1994). Computer viruses as artificial life. *Artificial Life*, *1*(3), 249–265.

115. Spence, K. (1995). Genetic exhibitionism. *Wired*, *3.01*, 148.

116. Stanley, K. O. (2007). Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines*, *8*(2), 131–162.

117. Stanley, K. O., Bryant, B. D., & Miikkulainen, R. (2005). Evolving neural network agents in the NERO video game. In *Proceedings of the IEEE 2005 Symposium on Computational Intelligence and Games (CIG-05)* (pp. 182–189). Piscataway, NJ: IEEE.

118. Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation*, *10*, 99–127.

119. Stanley, K. O., & Miikkulainen, R. (2004). Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research (JAIR)*, *21*, 63–100.

120. Steels, L., & Kaplan, F. (1999). Collective learning and semiotic dynamics. In D. Floreano, J.-D. Nicoud, & F. Mondada (Eds.), *Advances in artificial life: 5th European Conference, ECAL'99, Lausanne, Switzerland, September 13–17, 1999 proceedings* (pp. 679–688). Berlin: Springer.

121. Steels, L., Kaplan, F., McIntyre, A., & Van Looveren, J. (2002). Crucial factors in the origins of word-meaning. In A. Wray (Ed.), *The transition to language* (pp. 252–271). Oxford, UK: Oxford University Press.

122. Stockhausen, K. (1996). Helikopter-streichquartett. *Grand Street*, *56*, 213–225.

123. Szerlip, P., & Stanley, K. O. (2013). Indirectly encoded Sodarace for artificial life. In P. Liò, O. Miglino, G. Nicosia, S. Nolfi, & M. Pavone (Eds.), *Advances in artificial life, ECAL 2013* (pp. 218–225). Cambridge, MA: MIT Press.

124. Szerlip, P., & Stanley, K. O. (2014). Steps toward a modular library for turning any evolutionary domain into an online interactive platform. In H. Sayama, J. Rieffel, S. Risi, R. Doursat, & H. Lipson (Eds.), *Artificial life 14: Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems* (pp. 900–907). Cambridge, MA: MIT Press.

125. Szor, P. (2005). *The art of computer virus research and defense*. Boston: Addison-Wesley.

126. Takeichi, Y., Sasahara, K., Suzuki, R., & Arita, T. (2014). Twitter as social sensor: Dynamics and structure in major sporting events. In H. Sayama, J. Rieffel, S. Risi, R. Doursat, & H. Lipson (Eds.), *Artificial life 14: Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems* (pp. 778–784). Cambridge, MA: MIT Press.

127. Taylor, T. (2014). Artificial life and the Web: WebAL comes of age. In *Proceedings of WebAL-1: Workshop on Artificial Life and the Web 2014, held at the 14th International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14)*. http://arxiv.org/abs/1407.5719.

128. Taylor, T., Auerbach, J., Bongard, J., Clune, J., Hickinbotham, S., & Hornby, G. (Eds.) (2014). *Proceedings of WebAL-1: Workshop on Artificial Life and the Web 2014*. http://arxiv.org/abs/1406.2507.

129. Thimbleby, H. W., Whitten, I. H., & Pullinger, D. J. (1995). Concepts of cooperation in artificial life. *IEEE Transactions on Systems, Man, and Cybernetics*, *25*(7), 1166–1171.

130. von Mammen, S., & Edenhofer, S. (2014). Swarm grammars GD: Interactive exploration of swarm dynamics and structural development. In H. Sayama, J. Rieffel, S. Risi, R. Doursat, & H. Lipson (Eds.), *Artificial life 14: Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems* (pp. 312–319). Cambridge, MA: MIT Press.

131. Wagy, M., & Bongard, J. (2015). Crowdseeding: A novel approach for designing bioinspired machines. In S. P. Wilson, P. F. Verschure, A. Mura, & T. J. Prescott (Eds.), *Biomimetic and biohybrid systems* (pp. 293–303). Berlin: Springer. http://dx.doi.org/10.1007/978-3-319-22979-9_29.

132. Wait, A. (2004). The quantum coreworld: Competition and cooperation in an artificial ecology. In J. Pollack, M. A. Bedau, P. Husbands, R. A. Watson, & T. Ikegami (Eds.), *Artificial life IX: Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems* (pp. 280–285). Cambridge, MA: MIT Press.

133. Whitelaw, M. (2004). *Metacreation: Art and artificial life*. Cambridge, MA: MIT Press.

134. Witbrock, M., & Neil-Reilly, S. (1999). Evolving genetic art. In P. Bentley (Ed.), *Evolutionary design by computers* (chap. 10, pp. 251–259). Burlington, MA: Morgan Kaufmann.

135. Woolley, B. G., & Stanley, K. O. (2014). A novel human-computer collaboration: Combining novelty search with interactive evolution. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation* (pp. 233–240). New York: ACM.

136. Yu, L. L., Nickerson, J. V., & Sakamoto, Y. (2012). Collective creativity: Where we are and where we might go. In T. W. Malone & L. von Ahn (Eds.), *Proceedings of Collective Intelligence 2012*. http://arxiv.org/abs/1204.2991.

137. Zaman, L. (2014). EvoPopcorn: A Web-native distributed artificial life simulation. In *Proceedings of WebAL-1: Workshop on Artificial Life and the Web 2014, held at the 14th International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14)*. http://arxiv.org/abs/1406.2507.

138. Zhang, J., Taarnby, R., Liapis, A., & Risi, S. (2015). DrawCompileEvolve: Sparking interactive evolutionary art with human creations. In C. Johnson, A. Carballel, & J. Carreia (Eds.), *Evolutionary and biologically inspired music, sound, art and design 4th International Conference, EvoMUSART 2015, Copenhagen, Denmark, April 8–10, 2015, Proceedings.* (pp. 261–273). Berlin: Springer.

Appendix: Bibliography of Other NetAL Work

This appendix provides details of articles discovered in the systematic search of the *Artificial Life* journal and conference proceedings that have not been covered elsewhere in this review article. For the purposes of this review, they were deemed to be either insufficiently focused on the web (i.e., NetAL rather than WebAL), or insufficiently focused on ALife. They are listed here for completeness. For details of the search methodology adopted, see Section 2.

A1. Aguilera, M., Morer, I., Barandiaran, X. E., & Bedia, M. G. (2013). Quantifying political self-organization in social media. Fractal patterns in the Spanish 15M movement on Twitter. In P. Liò, O. Miglino, G. Nicosia, S. Nolfi, & M. Pavone (Eds.), *Advances in artificial life, ECAL 2013* (pp. 395–402). Cambridge, MA: MIT Press.

A2. Best, M. L. (1997). An ecology of text: Using text retrieval to study ALife on the net. *Artificial Life*, *3*(4), 261–287.

A3. Best, M. L. (1997). Models for interacting populations of memes: Competition and niche behavior. In P. Husbands & I. Harvey (Eds.), *Fourth European Conference on Artificial Life* (pp. 154–163). Cambridge, MA: MIT Press.

A4. Bollen, J., Gonçalves, B., Ruan, G., & Mao, H. (2011). Happiness is assortative in online social networks. *Artificial Life*, *17*(3), 237–251.

A5. Humphrys, M. (2001). Distributing a mind on the Internet: The world-wide-mind. In J. Kelemen & P. Sosík (Eds.), *Advances in artificial life: 6th European Conference, ECAL 2001, Prague, Czech Republic, September 10–14, 2001, Proceedings* (pp. 669–680). Berlin: Springer.

A6. Kephart, J. O., Hanson, J. E., & Sairamesh, J. (1998). Price and niche wars in a free-market economy of software agents. *Artificial Life*, *4*(1), 1–23.

A7. Kim, K.-J., & Cho, S.-B. (2006). A comprehensive overview of the applications of artificial life. *Artificial Life*, *12*(1), 153–182.

A8. Noser, H., Pandzic, I., Capin, T., Thalmann, N., & Thalmann, D. (1996). Playing games through the virtual life network. In C. G. Langton & K. Shimohara (Eds.), *Artificial life V: Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems* (pp. 135–142). Cambridge, MA: MIT Press.

A9. O'Leary, C., & Humphrys, M. (2003). Building a hybrid society of mind using components from ten different authors. In *Advances in artificial life: 7th European Conference, ECAL 2003, Dortmund, Germany, September 14–17, 2003, proceedings* (pp. 839–846). Berlin: Springer.

A10. Saffre, F., & Shackleton, M. (2008). "Embryo": An autonomic co-operative service management framework. In S. Bullock, J. Noble, R. A. Watson, & M. A. Bedau (Eds.), *Artificial life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems* (pp. 513–520). Cambridge, MA: MIT Press.

A11. Stow, D., & Roadknight, C. (2001). Antigens, antibodies, and the World Wide Web. In J. Kelemen & P. Sosík (Eds.), *Advances in artificial life: 6th European Conference, ECAL 2001, Prague, Czech Republic, September 10–14, 2001. Proceedings* (pp. 161–165). Berlin: Springer.