CrawLogo: empowering end-users to program the Web

Kevin McGee and Johan Nilsson
Department of Computer and Information Science
Linköping University
581 83 Linköping, Sweden
kevmc@ida.liu.se
x03johni@ida.liu.se

Abstract

In order to create applications that programmatically use the Web as an expressive medium, the current choice is largely between those conventional programming languages that are difficult to master and those that are less expressive. CrawLogo is a Logo-inspired programming environment in which Web-space is represented as part of a Crawler Geometry—and in which the coordinates in this geometry are programmable "Crawltures." In initial trials of the system, end-users were able to quickly create their own Web-enabled applications; however, further research is needed to improve different aspects of the geometry and the interface of the CrawLogo development environment.

1. Introduction

In recent years the Web has given rise to a number of innovative applications and activities, such as "collaborative browsing", collaborative collage-making, group-editing and publishing of Web logs ("blogs") and Wikis, file-sharing and "friend-sharing", chat, IM, and distributed, interactive story-telling. Although end-user programming systems have been developed for a variety of domains (see [2] for a summary and review), very few tools exist that empower non-programmers to implement their own versions of these applications—or even to invent new kinds of Web-enabled activities and applications.

To address this, we have been developing CrawLogo, a Logo-inspired programming environment in which the two-dimensional Turtle Geometry [4, 1] of the screen is extended to a Crawler Geometry that includes a geometric representation of the Web—and in which the coordinates in this geometry are programmable "Crawltures."

A key innovation in the design of Logo and Turtle Geometry was the transformation of the geometric *point* into an agent (the "turtle") with state-variables for *position, heading,* and the like. Thus, in Logo users can create drawings on the screen by "talking to the Turtle" using such well-known Turtle commands as `FORWARD` and `RIGHT`. This, in turn, transforms geometry into something *useful* for novice programmers.

The development of CrawLogo is motivated by a similar interest: to develop a geometry that is usable by non-programmers who want to make different kinds of Web-enabled applications. In the sections below, the initial version of Crawlture Geometry and the CrawLogo development environment is described—as are some sample applications and the initial response of end-user programmers who successfully used CrawLogo to build Web-enabled applications. (For a more extensive treatment, see [3].)

2. CrawLogo

The metaphor for developing CrawLogo applications involves the creation and control of Crawltures that exist within a Crawlture-space. The CrawLogo development environment consists of a user interface, the CrawLogo interpreter, and a number of programmable objects. CrawLogo applications are developed by creating and managing procedures and various objects such as Web browsers and filters, using Logo-like syntax and semantics. These applications can be local to one computer—or distributed across two or more.

Crawlture Geometry In CrawLogo, the agents are similar to Turtles in that they have a state-variable for their position; however, the geometric space of CrawLogo includes coordinates for Web-addresses (URLs). Thus, a Crawlture exists in a position comprised of the X- and Y-coordinates of the screen—as well as a URL-space that is three-dimensional in the following sense: the URL links explicitly referenced on the Crawlture's current URL location are mapped as a plane (X- and Y-coordinates in URL-space) of discrete, Crawlture-relative nodes—and the "absolute" directory/file-structure space of the current URL location is represented as a Z-axis in URL-space (with nodes potentially "above" and "below" the Crawlture).

CrawLogo Language Similar to other versions of Logo, CrawLogo contains a fairly standard set of primitives (mathematics, symbols, control, graphics, I/O, and the like). There are also custom primitives that include support for threading, networking, and creating and manipulating CrawLogo objects. The new CrawLogo primitives were designed to extend the CrawLogo metaphor to correspond to some of the task-specific features that programmers might want to include in their CrawLogo applications. For example, starting a server that can be accessed by other users does not require the programmer to know or use details about socket handling and sending packages; it is possible within the CrawLogo metaphor to quickly get a server up and running and start interacting with other users.

CrawLogo Objects As traditional Logo has Turtle objects, CrawLogo has Crawlture objects. There are a number of different categories of Crawltures, including Web browsers (which can retrieve and display Web pages); shapes (most similar to the traditional Logo Turtle, these are simple graphical objects—quadrangles and ellipses of different sizes and colors—that can be instructed to move on the screen and interact with other objects); and filters (which can graphically modify other Crawltures that they come across—blurring or embossing them). Crawltures can also potentially have different sensors and effectors, allowing them to respond to—and act upon—other Crawltures and I/O data (display, audio, and the like). Thus, for example, a Crawlture can be programmed to "click on" certain kinds of hyperlinks—and "have them spawn as new Crawltures."

CrawLogo: Programming Programming in CrawLogo is very similar to programming in other Logo environments. The actual syntax is similar and there are commands for making and controlling Crawltures; there is support for creating, applying, and saving complex procedures and subprocedures; and there are mechanisms for controlling different aspects of URL-space, network activity, and connectivity.

CrawLogo: Sample Applications A number of demonstration CrawLogo applications have been developed. *CrawLogo Pong* is a version of the classic Atari game in which players compete across the network, and in which Crawltures are the "ball" and "paddles"—and in which different state-conditions for both the balls and paddles have unexpected consequences for the players (such as changing the speed, size, or direction of the ball—or modifying the player's ability to control the paddles). *CrawLogo Co-Browse* is an application in which users can browse the Web together, show each other interesting Web pages and chat about what they see. *CrawLogo Slideshow-maker* is an application in which someone can programmatically specify Web-generated slide-shows. *"Guess Who?"* is a Web-enabled multiplayer guessing game in which Crawltures are programmed to find other Crawltures with, say, pictures of rock-stars; the pictures are then blurred (or otherwise disguised) and players send either guesses or Crawltures to "de-blur" the images slightly.

3. CrawLogo: Actual Use

To date, most of the CrawLogo research effort has gone into the initial design and implementation of Crawlture Geometry, the CrawLogo programming language, and the development environment. However, there have been some trials of the system with end-users. The focus of these preliminary studies was to see whether the CrawLogo environment inspired and empowered users to make up their own applications—and whether it supported them in implementing specific target-applications.

For the trials, participants were told about the Logo Turtle, the idea behind CrawLogo, and then given access to the CrawLogo environment and a short manual of CrawLogo commands. They were then asked to do two kinds of activities. The first was an open-ended activity in which they should develop "whatever they wanted", based on their own explorations and understanding of the language. Most participants quickly "made contact" with each other across the network. In several cases, participants also discovered that they themselves continued to have control over the Crawltures they sent to each other, and began to develop applications that combined collaborative browsing with competitive manipulations of each other's screen activity. Other participants spontaneously created their own versions of some of the applications described above.

The second activity presented participants with a specific challenge: create a Web-crawler application that would generate an interesting slide-show by finding a URL that matched some criteria, displaying it, waiting a certain amount of time, and then repeating this process. All participants succeeded quite easily in creating such an application.

The overall response of participants was that not only did the CrawLogo environment empower them to make interesting applications, but the actual process of making the applications was fun. The results are encouraging, but we also observed a number of problems with the current system. Some of these are related to the CrawLogo geometry and some are related to the lack of certain kinds of visual state- or process-feedback. We highlight a few of the issues below.

4. Discussion

Empowering users to create and control Crawltures that move along complex and unpredictable paths calls for a meaningful and intuitive representation of such movement—and the geometry of the space within which such movement takes place. This work is still in its early stages and there are already a number of obvious research challenges related to the CrawLogo language, the design of Crawlture Geometry, and the visualizations of different phenomena.

Syntonicity and Design of Primitives Much of the learnability of Logo's Turtle Geometry lies in its *syntonicity*—the possibility for a user to identify with the Turtle and mentally (or physically) "play Turtle." Currently, the CrawLogo primitives fall into three broad categories, depending on the degree to which they can be said to be consistent with the Crawlture metaphor: consistent with "playing Crawlture" (commands such as forward, up, and the like); consistent with "talking to the Crawlture" (commands such as setcolor, setpower and setURL); and those that are "outside the metaphor" (startserver and the like). This raises an ongoing design issue about whether (and how) to try and "force" certain programming activities into a consistent metaphor.

Turtle Goes Crawling Certain aspects of representing the Web geometrically raise problems that are not present in traditional 2-dimensional Turtle Geometry. To name only a few examples, consider that in Turtle Geometry, headings of 0 and 360 are equivalent; it is not clear to what extent it is meaningful to think of Web space as being "closed" in the same sense. Even more problematic, it is not clear where "the screen" is located along the Web z-axis: is it the "origin"—and if so, is such an origin best conceived in terms of a polar coordinate system? Similarly, the notion of "reversible operations" within Crawlture Geometry is not always straight-forward: whereas moving up is unambiguous (e.g. moving up from any particular URL will always lead to a parent URL) moving down from the same parent URL can potentially lead to a different URL.

Crawlture Visualization One major challenge is how to visualize certain aspects of a Crawlture's state; for example, *heading* and *position* in CrawLogo space. Some issues, such as the visual representation of a rectangular browser's orientation on the screen, will not be difficult to solve. Others, such as providing orientation or movement cues along other dimensions, will be more challenging. In particular, the current implementation does not try to visualize a unified, 5-dimensional space; much of the spatial movement and orientation of Crawltures is left to the imagination of the programmer. As a related visualization problem, it is not obvious what the equivalent of "pen down" should be for CrawLogo. Programmers in traditional Logo benefit from seeing the history of the pictures they try to make; we are currently exploring techniques to provide similar, concrete feedback about Crawlture histories.

5. Conclusion

A long-term goal of this research is to empower computational creativity of different kinds; in particular the ability of end-users to program the Web as an expressive medium. In addition to such success criteria as ease-of-use, the evaluation metrics include such things as the pleasure with which people use the tools, the pride they take in their creations, and the degree to which they are empowered in the having (and realizing) of their own innovations. The current version of CrawLogo still requires much work to fulfill this goal, but the results to date are encouraging.

References

[1] H. Abelson and A. diSessa. *Turtle Geometry: The Computer as a Medium for Exploring Mathematics*. MIT Press, Cambridge, MA, 1980.

[2] C. Kelleher and R. Pausch. Lowering the barriers to programming: a survey of programming environments and languages for novice programmers. Technical Report CMU-CS-03-137, Computer Science Department, Carnegie Mellon University, 2003.

[3] J. Nilsson. Crawlogo: An experiment in end-user programming for web-enabled applications. Master's thesis, Department of Computer and Information Science, Linköping University, Sweden, 2003.

[4] S. Papert. *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books, New York, 1980.