Journal of Visual Languages and Computing 25 (2014) 545–571

Contents lists available at ScienceDirect

Journal of Visual Languages and Computing

journal homepage: www.elsevier.com/locate/jvlc

Palimpsest: A layered language for exploratory image processing ☆

Alan F. Blackwell

University of Cambridge Computer Laboratory, UK

ARTICLE INFO

Article history:
Received 28 May 2013
Received in revised form
24 July 2014
Accepted 25 July 2014
Available online 9 August 2014

Keywords:
Visual programming
Visual arts
Live coding
Constraint semantics

ABSTRACT

Palimpsest is a novel purely-visual language intended to support exploratory live programming. It demonstrates a new paradigm for the visual representation of constraint programming that may be appropriate to future generations of keyboardless and touchscreen devices. The current application domain is that of creative image manipulation, although the paradigm can support a wider range of computational expression. The combination of constraint semantics expressed via a novel image-layering metaphor provides a new approach to supporting a gradual slope of abstraction from direct manipulation to behaviour specification. Exploratory evaluations with a range of users give an indication of likely audiences, and opportunities for future development and application.

1. Introduction

How are visual *programming languages* related to the software used to manipulate visual *images*? Visual artists often use applications such as Photoshop to create or modify images. This can involve a long sequence of manipulations and transformations. From a programming language perspective, we might think of that sequence as a program: a composition of individual operations whose output is a new artwork. However this kind of visual interaction sequence is more often described as direct manipulation, rather than programming, despite the fact that Shneiderman originally defined direct manipulation in relation to programming [28].

The advantage of direct manipulation is that the user can easily anticipate and evaluate the effect of each action as it is taken, rather than waiting until program execution time (or command execution time, in the more familiar comparison between command line and GUI). The disadvantage of direct manipulation is that reduced abstraction and expressive power makes it laborious to repeat the same operations, or variants on them. It is possible, in powerful tools like Photoshop, to construct a simple kind of program in the form of a macro recording – a sequence of operations that can be replayed. However, it is difficult to parameterise a macro recording, or change its behaviour in response to new data or external events.

The goal of this research is to develop an interaction paradigm that shares properties of both image editors and programming languages, offering increased abstraction power to people who work with images. Although the research is theoretically motivated (by the relationship between direct manipulation and visual abstraction), there are a number of possible applications in the visual arts. One of these is the creation of animations, by replaying sequences of image transformations. Another is a kind of version or configuration control, in which different combinations of image transformations might be derived from each other or compared. A third is the performance practice of live coding, in which digital artworks are

☆ This paper has been recommended for acceptance by S.K. Chang.

E-mail address: alan.blackwell@cl.cam.ac.uk

http://dx.doi.org/10.1016/j.jvlc.2014.07.001
1045-926X/

created in front of an audience, using tools that minimise the separation between run-time and edit-time. Each of these offers increased connection between the program editor and execution environment, in the manner defined by Tanimoto as "liveness" [33].

This paper describes the design, implementation and evaluation of Palimpsest, a novel image manipulation environment and live visual programming language that demonstrates these properties. The focus on image manipulation revives the long-standing challenge of the "purely visual" language, of which Smith's *Pygmalion* [29], Furnas's *BitPict* [16], and Citrin's *VIPR* [11] have been creative earlier examples. Although often mooted in this journal and related venues, purely-visual languages have seldom seemed compelling on machines that do, after all, have keyboards. In the age of tablets and touch interaction, keyboards have suddenly become a real inconvenience and hindrance in routine interaction, so this seems a better time than any to explore text-free notations.

The name Palimpsest is inspired by the fact that on touch devices, abstract notations are often superimposed over directly manipulated content. In media and cultural studies, the word palimpsest has been extended from its original sense of a text written over an erased original, to refer to the layered meanings that result when different cultural or historical readings are superimposed[1] [15]. The palimpsest thus offers a metaphor for the integration of computational capabilities into the visual domain.

1.1. Application example

As a simple example of how Palimpsest might be applied, consider a typical image-editing operation in which a person's face has been separated from the background of an image, and is placed over a coloured frame. The frame is initially red, but the artist decides that the colour should be related to the skin tone in the face. In Photoshop, this would involve using an eyedropper to select the new default colour, and then filling the frame with that colour. However in Palimpsest, the colour can be treated as a visual variable. The variable initially had a constant value of red, but is now bound to a sampled value. The artist then decides that, rather than skin tone, the frame should match the collar of the shirt the person is wearing. In Photoshop, the default colour would be changed, and another fill carried out. However in Palimpsest, the change can be defined simply by modifying the location of the sampled value. Now imagine that the artist likes both of the moods provided by these two colours, and decides to create an animation alternating between them. In Photoshop this would not be possible. In Palimpsest, the colour-sample location can be bound to a function that changes over time, turning the static picture into a dynamic one contrasting the effect of the two frames.

As an introductory tutorial, illustrating typical operation of Palimpsest, Appendix A shows a sequence of screenshots corresponding to the scenario just described. A complete video sequence of the interaction is included in the supplementary materials published with this paper.

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.jvlc.2014.07.001.

1.2. Outline of paper

The remainder of this paper is structured as follows. The programming paradigm itself is first described, including an overview of the novel execution model, data types, edit/debug facilities, and support for alternative flow of control and encapsulation. Throughout the discussion, the user-interface design considerations of Palimpsest are discussed in terms of the Cognitive Dimensions of Notations ([17] The current prototype implementation is summarised, followed by preliminary evaluation of this prototype with three different kinds of user population. A discussion of related work addresses previous systems that have combined elements of direct manipulation and abstract behaviour specification, as well as related approaches to user functionality and interaction techniques. The final discussion proposes a model for future work of this kind, and sets out an agenda for future research that will develop the Palimpsest concept.

2. The Palimpsest programming paradigm

This section provides a high-level overview of Palimpsest operation. An extended step-by-step tutorial introduction is included in Appendix B. The supplementary material published with this paper also includes a video demonstrating the operation of these basic aspects.

Palimpsest allows users to combine source material (photographs, simple shapes or ink) and treatments of that material (e.g. filters or geometric transformations). Source material and treatments are partially transparent, and are layered on top of one another such that a final image is built up from elements in many layers, just as in professional image manipulation tools such as Photoshop. The behaviour of individual layers can be modified, as usual in such tools, by adjusting parameter values.

However, unlike conventional systems such as Photoshop, parameter values are also represented as image layers, making them first class values in this novel visual language. Interaction with the system involves creating new layers, superimposing them on layers already created, and adjusting values. The resulting stack of source material, values and treatments provides both a visual palimpsest (in the media studies sense), and a layered historical record of the process by which it was achieved. The resulting image can also be viewed in a simplified exhibition mode, with control information hidden and the layers composited together.

The primary Palimpsest display (Fig. 1) is an image composed from multiple layers overlaid on top of each other. This stack of overlapping layers is also rendered as thumbnails at the side of the main display, but spread out so that the order of the individual layers is visible. The current viewpoint can be moved up and down the

[1] For example in the work *Cambridge Palimpsest* by artist Issam Kourbaj, which was created to commemorate the 800th anniversary of the University, and was one of the inspirations for the title.