Supportive Methodology and Technology for Creating Interactive Art

by Greg Turner

M.Comp. (Hons) Loughborough University

A thesis submitted to:
Faculty of Information Technology
University of Technology, Sydney
Australia

For the degree of:
Doctor of Philosophy in Computing Science

Supervised by:
Professor Ernest Edmonds, UTS
Dr. Tim Mansfield, Industry

Submitted for Examination: December 20th, 2006
Final revision: May 18th, 2007

Statement of Sources

I, Greg Turner, declare that the work presented in this thesis is, to the best of my knowledge and belief, original, except as acknowledged in the text, and that the material has not been submitted, either in whole or in part, for a degree at this or any other university.

Signed:

Acknowledgements

This is the one part of the thesis I can be sure the examiners won't read, so I think it is safe to relax the rules of scholarship for two pages, because the justification for thanking these very special people need not be very rigorous. It's been a big journey, and my thanks are literally the size of the world: I shall begin with the far hemisphere and perform several circumnavigations.

Thanks heaps, as Aussies say, to mum and dad, for the selfless kindness and support, keenly felt from the other side of the planet, that only loving parents can give. I love you too! To Alankar, for reminding me that life is best when it's lived, and to Nick, for reminding me well that life isn't to be lived in books alone.

Formal, yet heartfelt, thanks to the institutions that have supported my study with their kind and forward-thinking scholarships, for what is unusual and difficult research. These institutions are the Faculty of IT, and the ICAN group at UTS.

Thank you to all the fine minds at CCS, but most of all to the desk quintet: Lizzie, who taught me the true meaning of true meaning and tidied up afterwards; Alastair, for setting the pace—in love as well as labour (both sorts!); Yun for blazing a trail with her strength and honesty; Deb for her ability to single-handedly arrange the universe such that everyone enjoys themselves (with Canada at the middle).

Like thanx or whatever to the juvey ledges that I like live with you know Claire, Amber and like Sebastian for looking after me loads and making life outside the matrix so like bodacious and like totally an in-joke haven. Woo, Collins St. rocks, you guys!

Gerhard, the most skilled, in-demand, non-programmer I know, gets a special nod for his support and his strange genius, and on behalf of the examiners, I thank him for making this document so easy on the eye.

Back on the other side of the world, here's to all of my real-life heroes who I haven't seen in too long: Ruth, who knows me better than anyone, and still thinks that's a good thing (likewise!). Robert and Lindsay, for sharing life's joys through great hospitality and the Socratic method; Russ and Jill for sharing life's joys through great hospitality and an excellent Bloody Mary. Jim, whose inspired and inspiring flex of phrase delights even more through its comparative rarity; Kate and Tim who are so big, and so much fun it was almost rude to get married on top of it all; and David, who more than anyone, embodies all the things I miss about Britain.

But for this thesis, of course, no-one deserves more thanks than Ernest Edmonds and Tim Mansfield, my supervisor and co-supervisor. Ernest, I will always admire you for your hard work, your taste, and for your willingness to make amazing things happen for people like me. Tim, you are friendlier, funnier, wiser and more quotable than any mentor I could have hoped for, and I love that the pace of learning hasn't slowed for you after your PhD.

And thank you finally to "the artists", particularly George, Dave, Norie, Maria, Petra, Keir and Daniel, without whom "all this" would be decidedly pointless, and much less fun.

Simple things should be simple.
Complex things should be possible

Alan Kay's law of simplicity.

…and there should be a smooth curve
between them.

Greg Turner's corollary.

Abstract

Computation, as a medium for programming, supports scientists, mathematicians and "algorithmically-creative" (Amabile, 1996) workers very well. 'Deep' programming environments, with few, or flexible constraints, are designed for these kinds of computation. However, most artists, designers and other "heuristically-creative" (Amabile, 1996) workers must make do with more 'gentle' programming environments, such as Max/MSP or Processing, which support particular conceptual spaces well. Yet once the constraints of those spaces are come up against, they are found to be rigid.

The new media world is, by now, used to seeing interdisciplinary work that involves artists and technologists in collaboration, sometimes in response to this difficulty. These collaborations combine the power of artistic modalities of thinking with the full capabilities of computational media, but still the computing medium must be mediated for the artist by the technologist. Such mediation is at risk of reinforcing boundaries between artists and technologists, and denies artists 'hands-on' creativity in the medium, which is not only frustrating but also can destroy artistic meaning (Candy & Hori, 2003).

How can we make computational media better support creative workers, in and out of collaborations? My answer stems from the roles of constraints which surround conceptual spaces, but which can support creativity only as far as they can be changed in response to a change in conceptual spaces (Boden, 2004). Computation is an attractive medium because potentially supports highly changeable constraints. However, this potential is not realised—there are plenty of constraints within computing today which are neither inherent nor useful for creativity, but imposed as a result of industrial practices which are decreasingly relevant in today's techno-society. An example is the constraint around every compiled program preventing any modification of that program. Since these constraints cannot be changed in response to changing conceptual spaces, creativity is limited.

To remedy this technological disjunction between conceptual spaces and supportive media, I have made recommendations for future computing systems in which imposed constraints are not rigid. For example, if someone wishes to explore or change a particular constraint in such a computing system, they can 'lift the hood' and discover what's happening and change it, recursing if necessary to the level of computing fundamentals, but using a similar interface paradigm to that which they have already been using. Such a computing system allows people to change a computing medium to fit with their changing conceptual spaces.

To illuminate the accompanying social issues of supporting interdisciplinary collaboration, I carried out a grounded theory inquiry into the roles of collaborating experts—predominantly artist and programmer—working in interactive art collaborations. By studying first-hand reports and conducting interviews, I was able to build a rich theory of technology's role in the collaborative process. Most importantly, I found that non-programming artists prefer to use shared language and boundary objects (Fischer & Ostwald, 2003) that are also meaningful in computing terms. An example is when a programmer constructs 'computational toys', which sit between conceptual spaces and thus can be manipulated to create technical, aesthetic and computational meaning simultaneously.

To evaluate these findings, I synthesised the computing recommendations and the toy-making methodology, and examined prototypical examples of them in the light of a real-world art collaboration called *Cardiomorphologies v. 2.* The collaboration involved the development of several computational toys in the Max/MSP computing system, and also a technology for quickly creating toys.

Contents