Mental Imagery in Program Design and Visual Programming

Marian Petre and Alan F. Blackwell

International Journal of Human-Computer Studies (1999) **51**, 7-30

ABSTRACT

There is widespread anecdotal evidence that expert programmers make use of visual mental images when they are designing programs. This evidence is used to justify the use of diagrams and visual programming languages during software design. This paper reports the results of two studies. In the first, expert programmers were directly questioned regarding the nature of their mental representations while they were engaged in a design task. This investigative technique was used with the explicit intention of eliciting introspective reports of mental imagery. In the second, users of a visual programming language responded to a questionnaire in which they were asked about cognitive processes. The resulting transcripts displayed a considerable number of common elements. These suggest that software design shares many characteristics of more concrete design disciplines. The reports from participants in the two studies, together with previous research into imagery use, indicate potential techniques for further investigation of software development support tools and design strategies.

1. INTRODUCTION

Much research into visual programming languages is motivated by the assumption that visual languages bear a closer relationship than do textual languages to the mental representation of the programmer (Blackwell 1996a). This claim, as with statements about mental imagery in other fields, is difficult to evaluate empirically; previous studies of textual programming languages have found unexpected results when some participants "re-code" program text into an imagistic form (Gilmore & Green 1984). There is consequently a strong tradition in research into mental imagery of taking a phenomenological approach (using questionnaires and introspective reports) to supplement experimental investigations. This paper reports the results of two studies which try to refine the widespread anecdotal evidence that expert programmers use mental images in order to specify, predict, and simulate program behavior. Anecdotes of this type appear throughout Susan Lammers's interviews of well-known programmers (1986), e.g.:

> *"The first step in programming is imagining … I like to imagine the structures that are being maintained, the structures that represent the reality I want to code … The code for the most part writes itself, but it's the data structures I maintain that are the key. They come first and I keep them in my mind throughout the entire process." (Charles Simonyi, page 15)*

> *"You have to simulate in your mind how the program's going to work, and you have to have a complete grasp of how the various pieces of the program work together." (Bill Gates, page 73)*

And similar anecdotes emerged in Petre's previous empirical study of expert programmers (1989):

> *"One of the earliest things is to visualize this structure in my head, a dynamic structure, so I can think about how things fit together and how they work … and once I have the structure fairly strong and clear in my mind, I move it around and move around inside it, examining it and tweaking it … "*

> *"I think of these systems of relationships as alive, interacting beings … I make this dirty dynamic mental representation, a sort of organic thing … "*

Each of these anecdotes includes two key elements: a structure of information, and how it works. These mental structures, part of initial solution planning, apparently embody information inter-relationships and take into account manipulations and functions to be performed in processing information. They do not necessarily map onto data structures coded in programming languages — nor onto the 'objects' of recent popularity — yet they highlight the importance of structuring information, of making that structure apparent and accessible, and of making it amenable to required manipulations.

So, what is the nature of these mental structures? What might they reveal about how experts think about solutions and hence how might they inform our design of programming tools and representations? This paper presents two studies which attempted to elicit statements about mental imagery from expert programmers. The first of these involved detailed interviews with ten expert programmers. The second was based on a questionnaire study of over 200 users of a commercial visual programming language. The evidence from these studies is set in the context of the psychology literature on mental imagery.

2. THEORETICAL CONTEXT

2.1. Definitions of mental imagery

The term "mental imagery" has been used in many different ways. It is therefore necessary to define the range of phenomena that we are addressing in this paper. We have tried to avoid the problems of the imagery debate (summarized in Kosslyn, 1994), particularly the question of whether there is a uniform, underlying (propositional) encoding of mental representations (Pylyshyn 1973). We recognise that there are multiple neural subsystems, for example supporting spatial and visual memory (Farah et al. 1988). These subsystems allow single pieces of information to be encoded in multiple forms. These multiple encodings within mental representations are, in fact, a productive resource for problem-solving (Mani & Johnson Laird, 1982; Chi, et al., 1989; Payne, 1993).

Despite the fact that we can make some general statements of this kind, the use of mental imagery during problem solving is not universal. Individuals differ in terms of their memory, transfer ability, cognitive preferences, repertoire of notational conventions, and so on. It also appears that - just as perceptual skill in notation use can be trained (Petre, 1995) - imaging ability can also be developed (for example in expert abacus users - Hishitani, 1990). In order to capture this range of variability, the current study uses as broad a definition of imagery as possible. Wherever the participants in this study describe inspectable mental representations, we take their statements to be a relevant description of imagery, regardless of the sensory modality of the image (for example, some participants describe verbal images rather than pictorial images).

We therefore address a phenomenon which is both complex and non-uniform. What makes it a suitable topic for investigation? There are two reasons for the current investigation. The first is that new programming tools are often justified in terms of the way that they support mental representations (Blackwell 1996a). The second is that both theoretical arguments (e.g Bartlett, 1927; Lindsay, 1988; Goel, 1995) and controlled experimental studies (e.g. Kaufmann (1979) Finke, Pinker and Farah (1989) Anderson & Helstrup (1993) do provide substantial support for the importance of imagery in creative problem solving.

2.2. Multiple external representations

The significance of this study obviously lies in the way in which it can inform the development and application of software tools, and of the other external representations used in design. There is a substantial literature addressing the relationships between external representation and problem-

solving (e.g. Cox & Brna 1995, Scaife & Rogers 1996), some of which specifically addresses programming tasks (e.g. Green & Petre 1996, Davies 1996). It appears reasonable that there should be some relationship between the external representation used, and the ability to generate solutions from mental representations. The cognitive role of external representations in architectural design, for example, have been addressed by Goldschmidt (1991) and Goel (1995). Navarro-Prieto (1998) has explored the use of visual representations to support images for programming, and Fish and Scrivener (1990) have made a coherent argument for the processes through which sketching can facilitate creative design.

These questions are not the main emphasis of the current paper. We have no doubt that external representations are important, and have addressed the question of interaction between mental representations and external representations elsewhere. The current paper, by contrast to these other studies, attempts to characterise the internal representations used by programmers.

2.3. Previous investigations

The first study reported here followed from earlier research (Petre and Winder, 1988) in which expert programmers' descriptions of their problem solving were elicited as part of an investigation of how they matched programming languages to solutions. Many of those observed reported that they "start with the data structure", and so Petre (1989) investigated how experts derived data structures from problems. It became clear that experts' use of the term 'data structures' in the context of constructing abstract solution strategies doesn't necessarily correspond closely to 'data structures in code' — it encompasses much more information, such as what relationships exist, how the information will be manipulated, how the problem is being partitioned and structured, and so on.

In experts' usage in this context, 'data structures' (as they conceived them) included considerable algorithmic information; the structure derived, not from the problem (although some data relationships are), but from the solution (in which the programmer interprets 'inherent' data relationships, imposes others, and organizes these to serve the strategy devised by the programmer). So, what are these internal 'data structures', which are the currency of the experts' problem solving? Are there elements in common among experts, or other patterns of usage? How closely do these internal structures correspond to external representations, whether program code or visualisation tools? These are the questions which led to the first study reported here, an attempt to elicit mental imagery used by expert programmers in designing solutions.

The second study was motivated by a project investigating the cognitive factors affecting visual programming language design. Visual programming language researchers often claim that their languages will provide cognitive benefits (Blackwell 1996a). These benefits are often explained in terms of an intuitive relationship between visual depictions and mental imagery. Are these intuitions common to all programmers, or simply to visual language researchers? A survey of commercial programmers who had no experience of visual languages found little support for the intuitive appeal of visual languages (Blackwell 1996b). A third study was therefore conducted, amongst a population who were experienced users of one particular visual programming language (National Instruments' LabVIEW), but who had no particular commitment to, or familiarity with, the research literature in visual programming. The current paper discusses the findings of that study with regard to mental imagery; a more detailed report comparing the three surveys has been published by Whitley and Blackwell (1997).

3. METHOD OF INVESTIGATION

The work reported here involved structured observational studies, interviews and questionnaires attempting to elicit mental imagery, *not* controlled laboratory experiments. The dangers of these techniques are well-known, especially as reported in the review by Ericsson and Simon (1985). The main problems with data from verbal reports are the likelihood that cognitive processes of interest are not accessible to introspection, and the possibility that the experimenter might bias the response by

asking only for certain types of report. In this paper we therefore consider two separate studies, the first of which uses data gathered in interviews, and the second data collected from questionnaires with comparatively non-directive questions.

The content of the reports we collected must not be taken at face value either. Information in non-verbal modalities will be verbalised in different ways by different subjects — this is a particular problem in studies of imagery. Finally, many people when asked to describe their thoughts do not base their reports on memories of specific cognitive events, but theorise about cognitive processes in general. This last type of report obviously requires a special analytic stance when compared to the results of psychological research, but even this metacognitive data is relevant to the execution of cognitive tasks, in that people's metacognitive beliefs influence their selection of problem-solving strategies. The great advantage of the approach we have taken in these studies is that they allow us to survey variability, rather than focusing on specific phenomena as in experimental work. The intention was to provoke a rich response from experts, in order to elicit an impression of their mental imagery and to begin to look at whether there are common elements among the imageries of individuals.

The first study used a combination of observation and interview techniques which had, in previous studies, proven effective at eliciting rich qualitative data of this ilk. It adopted a loose protocol of questions which appeared in pilot interviews to elicit descriptions that both surprised and satisfied those questioned. The ability to engender surprise and satisfaction have, in previous studies, been effective criteria for elicitation questions. When a subject is surprised during an interview, it suggests a divergence from routine rehearsed response, whether in the form of a new expression, a new perspective, or a re-consideration. Surprise and satisfaction together can be an indication of insight: “Is that what I did? — that explains why...” or “Did I say that? — yes, that is the way I think about it!” The study relied on subjects whose reports of activity in earlier studies corresponded well to other evidence of their activity, such as notes and observed actions, i.e., it relied on subjects who appeared to be ‘good self-reporters’.

Experts are well-known for rationalizing their practice ‘on-the-fly’. As reported by Schooler, Ohlsson & Brooks (1993), there is evidence that solving insight problems relies on essentially non-reportable processes, even that verbalisation interferes with some important thought processes. On the other hand, although subjective tests may be suspect, they have in some cases been shown to be reliably consistent, and to produce results just as good as those from more objective tests (Katz, 1983) There is some evidence that self-ratings do correlate with demonstrated ability (Ernest, 1977) and are stable in cases where they do.

The second study asked more general questions, and the questionnaire format did not prompt respondents to elaborate their descriptions. It therefore provides an alternative view of the respondents' theories about mental representations.

3.1. Verbal protocol study

The first study was based on the reports of ten experts, from both industry and academia, and from several countries in Europe and North America. They share the same general background: all have ten or more years of programming experience; all have experience with large-scale, real-world, real-time, data- and computation-intensive problems; all are acknowledged by their peers as expert. Half are proficient with one or more declarative languages. All had participated in previous studies undertaken by the first author on various aspects of expert programmer behaviour, mainly design and generation.

The coding language used was not of particular interest in this investigation, but, for the record, a variety of styles was exercised in the examples, using languages including APL, C, C++, Hypercard, macro-assembler, common LISP, Miranda, Prolog, and SQL. Those with experience of declarative languages did not design significantly different solutions from those without.

Tasks

Programmers were asked to design solutions to one of the following problems (also used in the previous data structures study), or to a problem of their choice. The experts were asked to imagine themselves free of coding restrictions, and they were not asked to implement the solutions as code.

Noughts and crosses player

The program acts as an interactive player of noughts and crosses.

Academic timetable maker

The program coordinates available resources (e.g., lecturers, equipment, rooms) with available times so as to maximize access to facilities, to avoid conflicts among related facilities, and to consider independence or interdependence of resources (e.g., course pre-requisites, mandatory courses).

Lexicon for sub-anagram solver

The lexicon is organized so that the sub-anagram solver can find quickly all words in a given language which can be constructed from some or all of the letters in a given word or phrase.

Pinball path predictor

The path of a pinball is determined by the pinball machine architecture (including bumpers, flippers, etc.), the initial push given the ball, and the current state of the pinball machine. (Bonus question: Consider changes in data structures required if two balls could be in motion at the same time.)

The prescribed task is not the most important element of these activities. The problems were chosen to evoke rich discussions by addressing classic issues in data representation and by admitting both standard and innovative treatments (the effective task is therefore the generation of potential solutions within this framework). 'Noughts and crosses player' is a small but non-trivial problem with a familiar superficial structure. 'Academic timetable maker' is a classic data manipulation problem whose solution relies on resolving competing priorities applied to complex relationships among potentially substantial data. The 'sub-anagram solver' is another classic problem in which the task is limited and the data extensive, and which suggests a strong inter-dependence between data structure and algorithm. 'Pinball path predictor' requires a re-interpretation of a physical structure.

Questions

Programmers were questioned during and after the programming tasks, depending on the time available. Transcripts taken throughout the task provided a record of programmer remarks and the contexts in which they were made; i.e., the transcripts provided a record of which remarks were spontaneous and which were prompted by questions. The analysis gave priority to spontaneous, concurrent verbalisation. Some questions interrupted their activity (allowing immediate description of images), some were introduced in the subject's own pauses and some followed the task, allowing reflection. Prompting questions were keyed to the moments when (or the moments just after) the experts showed signs of internally-focussed thinking, such as:

- pauses in writing activity
- closing eyes
- staring 'into space'
- staring at blank paper (fixedly or with eye movement)
- gesturing in the air

Sometimes the questions interrupted the reverie, sometimes they followed, allowing completion of the thought episode. The prompts were general, e.g.,:

"What do you see?"

"What colour is it?"

"Does it move?"

"Give me instructions to see what you're seeing"

"What strikes you about what's there?"

"What's there that you can't see?"

"What can you hear?"

"Do you hear any words?"

"What's going on?"

"Where are you now?"

Prompts were chosen in a modality (a sensory mode or perceptual form) other than that of recent answers. The immediacy of the response was noted.

The initial answers were followed up with other probes, e.g.,:

"Is that what it's *like*, or is that what you see?"

"How much have you left out?"

"Are there bits you can't describe?"

"What was this [gesture]?"

After the tasks were completed, the experts were questioned about their previous responses and about their imagery in general. All notes and other products were collected, and the sessions were recorded.

The analysis was data-driven. Notes and transcripts were examined for the imagery accounts they contained, and imagery descriptions were grouped in terms of common elements. Attention was given to the context in which the account was given. Particular attention was given to the spontaneity of the accounts (as reflected by the fluency and immediacy of description and by the amount of prompting required), to the programmer's satisfaction with the account, to any discrepancies, and to any indications of surprise.

3.2. Questionnaire study

The participants in the first study were personally recruited on the basis that they were expert programmers. Although they used visual representations in designing software, as most programmers do, there was no reason to believe that these experts would have a particular commitment to visual representations of software. The participants in the second study, however, were selected *because* of their commitment to a particular visual representation.

The second study also employed a less directed approach to eliciting descriptions of mental imagery. It used a questionnaire which was advertised on an e-mail discussion list dedicated to users of the National Instruments LabVIEW product, with responses collected via an HTML form on the World-Wide Web (Whitley and Blackwell 1997). One of the most notable features of LabVIEW is the fact that it includes a general purpose visual programming language, based on the data flow paradigm, called "G". Note that although many of the respondents to this survey were professional programmers, they had a wide range of experience of programming – unlike participants in the first study, who were individually selected on the basis of their programming expertise.

The questionnaire dealt with a wide range of aspects of the LabVIEW environment, including aspects related to G, and other features of the product. Those findings are described in detail in (Whitley and Blackwell 1997) and (Whitley and Blackwell 1998). The final question in the questionnaire,

however, was designed to elicit, in a comparatively non-directive manner, statements of the kind that have been described above in the first study: "How and why does the graphical nature of G affect the 'brain-work' involved in programming?" The same question had been used to encourage introspective statements about cognition in an earlier study (Blackwell 1996b). This question was an open format response, but the software controlling the HTML form encouraged all respondents to make some entry – if the form was submitted with this field blank, a message would be returned stating that partially complete responses were less valuable, and inviting him or her to complete the missing field.

As a result of this precaution, 209 of the 227 survey respondents gave some kind of response to the question about 'brain-work' in programming. These responses were coded into one of 28 different themes, as described by Whitley and Blackwell (1997). Check-coding found an 85% inter-coder reliability in assignment to themes. For the present study, however, only one of those themes is of interest – statements about mental representations. Some respondents also made statements of this nature when answering other open format questions in the questionnaire – those statements are also considered here.

Of the 227 LabVIEW users responding to this survey, only 63 (28%) made statements about mental representations being relevant to visual programming. This figure supports the surprised reactions of participants in the first study when considering their mental representations during programming tasks - it seems that many programmers give little explicit thought to their own mental processes, even when using a visual programming language of the type that is claimed to encourage mental imagery.

4. RESULTS

4.1. Imagery reports during programming tasks

The experts in the first study demonstrated a readiness to describe the form and contents of their thinking, although sometimes they found it difficult to articulate, and sometimes they were more adept at describing what their imagery was *not* … What follows is a digest of the sorts of imagery described. The descriptions in italics are in the experts' own words; the quotations given here are all immediate responses from the elicitation during problem solving.

Dancing symbols

> *" … it moves in my head … like dancing symbols … I can see the strings [of symbols] assemble and transform, like luminous characters suspended behind my eyelids … "*
>
> *"I see lines of code … text and variables … changes of variables against time … "*

One of the experts described textual imagery as "text with animation".

Mental description/discussion

> *"I'm just talking to myself … "*
>
> *"In order to design something, you need to be able to describe it to yourself … "*

Most of the experts described a sort of verbal imagery, in which parts of the problem were described or 'discussed' mentally. Sometimes this verbal imagery accompanied other imagery; sometimes it occurred in isolation. Sometimes it took visual form (see 'dancing symbols' above), but more usually it did not.

Auditory imagery

> *"It buzzes … there are things I know by the sounds, by the textures of sound or the loudness … it's like I hear the glitches, or I hear the bits that aren't worked out yet … "*

All of the experts described sound as an element in their imagery, although most did not describe it as a typical element. Some described mental verbalizations, but not all of the auditory imagery was verbal; the experts also described auditory presentations of solution characteristics, with auditory qualities like loudness or tone reflecting some aspect of the solution, like level of activity or type of data.

Visual imagery

> *"values as graphs in the head … flip into a different domain … transform into a combined graph … (value against time; amplitude against frequency; amplitude against time) … "*

All of the experts described some visual imagery, and a variety of mental visualisations were described. Some were mental simulations, as described below. Others were described as "spaces of possibility", e.g., a look-up table for techniques, used in pre-solution reasoning when the expert was choosing a general strategy. Many of the visual images described bore some resemblance to standard external representations, although often these would be dynamic in the mind, changing with different dimensions, or augmented by other views or additional information.

Machines in their minds

i. Abstract machines

> *"slow-speed time … movement of coefficients and button pushes … a machine grovelling through it (munch munch)"*

All of the experts described a sort of dynamic mental simulation, a 'machine in the mind'. Some of these were vivid, colourful, 'physical' structures — described by one expert as a "great, bristling, multi-coloured scaffolding of pipework and gadgets floating in space" — sometimes described as being in more than four dimensions. Others were logical structures, e.g., "The machines are like data structures — but as operated on … " Often, data was visible as it flowed through the simulation. All of the experts described moving inside these active simulations, as though they (their perspective) moved around, inside, and through the image.

One expert described setting such a mental machine going to run 'independently':

> *"[imagery] is a way of harnessing the mental machine for tandem problem solving … if there's a problem I can't solve … I get the mental machine to solve a subset of the problem, and I need to see what it's doing in order to understand how that activity relates to the larger problem … "*

ii. Pictures of implementations

> *"object oriented … nets of stuff … poking inputs and see what filters through … sucking: I'm ready, send me another one … "*

> *"It's a dynamic map … this process talks to that process: this processor talks to that processor … "*

Some of the mental simulations were described as augmented pictures of a 'physical machine' or a picture representing implementation structure (in a form such as a block diagram, a structure of

objects, or a schematic) overlaid with data flow or process flow. This appears consistent with claims by Finke, Pinker and Farah (1989) that mental images can be manipulated in order to simulate transformations of physical objects, thus allowing successive states in a design to be inspected.

Many of the difficult programming problems — and those most difficult to reason about — involve the concatenation of tools or programs, e.g., outputting probe results into files (or pipes) to another tool, rather than just displaying results on screen. Hence, experts spend time modelling the behaviour of partially known entities, e.g., this computer must talk to that remote peripheral, which we don't know much about. It is for this sort of problem that the experts seem to use pictures of implementation structures. Despite the use of the term "object-oriented" by one participant, we do not feel that this evidence can be used to justify the occasional claims that object oriented programming languages directly exploit physical world experience or intuitive cognitive processes (as argued in the past by Blackwell, 1993)

iii. Mechanical analogy

These mental simulations were distinguished by the experts from mechanical analogies, which were discussed as a different (inferior) form of imagery:

> *"some people, in order to make things thinkable about, generate mechanical analogies — e.g., two-scalar values represented as sliders"*

The implication was that mechanical analogy was either constraining or potentially misleading.

Surfaces

Many of the experts described a strongly spatial, mathematically-oriented imagery of 'solution surfaces' used in "prospecting around in the equation space" either to identify solutions or to identify simplifications that will lead to solutions. This imagery typically arises in connection with finding the correct solutions to a set of multi-dimensional, complex equations; it is described as a reduction of the multi-dimensional equation space into a two-dimensional surface where height or depth indicates 'goodness'.

> *"It's like describing all the dimensions of a problem in 2D, and in the third dimension you're putting closeness to a solution."*

Seeking a solution in the space is imagined as traversing this surface, either to identify appropriate numeric values or to identify algebraic simplifications.

> *"It's like driving across a desert looking for a well. What you actually have is good solutions distributed across this desert like small deep wells and your optimizer trundling along looking for them..."*

The surfaces were described variously in terms of wells (hence well-finding), attractors ("like heavy objects dropped in a rubber sheet"), or hills (hence hill-climbing), but the experts considered them equivalent. The surfaces in the imagery appear to vary among problems and classes of problems, so that experts described classes in terms of surface characteristics, e.g.: flat plains with evenly distributed wells, clustered wells of varying depths, wells with 'lips', prickly valleys with many local minima appearing as 'noise' obscuring a large depression, wells in clusters of known number, and so on.

Landscapes

All of the experts described a strongly spatial imagery, a surface or landscape of solution components over which they could 'fly', with different parts of the solution residing in different regions, and with

a sense of different parts of the solution being visible even when distant in the landscape. Experts talked about knowing where things were in the landscape:

> *" … oh, that happens over there … it's on the horizon, so I can keep an eye on it,but I don't really need to know … "*

This landscape imagery was typically discussed in connection with large, multi-component solutions. There seems to be more variation in this type of imagery than was elicited in this study; the elicited descriptions had insufficient detail to enable us to elaborate this category.

Presences

> *" … no place holders, no pictures, no representation … just the notion, the symbol entities, semantic entities and the linguistic token … atomic notions. They just 'are'"*

All of the experts described a sort of imagery that was not verbal, visual, or physical. This was an imagery of presence (or knowledge) and relationship (sometimes location) which included qualities of the 'entities' present. These qualities conveyed the nature or character of the entity and were 'perceived' as a sort of notional (not tactile) pressure or texture, or as a tension or relative density between non-visual regions. Several experts described the qualities in terms of sensation, but then clarified that this did not correspond to external sensory experience. These images were subject to movement, change, or transformation. One expert described this imagery as "knowing in the dark". Another described it as "perturbations in non-visual space … not always in 3D space or in sensual space".

4.2. Common Elements in Concurrent Reports

Despite considerable individual variation, a number of common elements emerged in the experts' descriptions. Some elements were not used to the same extent; for example, most experts incorporate colour in their imagery some of the time, and a few usually do so, but some of them never do. The same was true of non-verbal sound. Other elements or characteristics were common, not just among the experts, but among their different sorts of imagery. Those are described here.

Stoppably dynamic

All of the images were described as dynamic, but subject to control, so that the rate could be varied, or the image could be frozen. In some cases, the activity could be reversed or the image restored to a previous state.

Focal attention

The distribution of information in the imagery was not uniform; the experts chose where to put their attention at any given moment, and different regions of the imagery were described as coming in and out of focus. Information outside the focus might be undefined, or unsolved, or soluble, or solved; mainly, it was deemed not important at the moment.

> *"I don't need to think about that; it's enough to know it's over there … "*

This seems similar to Arnheim's (1969) discussion of "complete things perceived incompletely" in mental imagery as a positive quality, "the product of a selectively discerning mind", p. 105.

Adjustable granularity of abstraction

The experts described reasoning about aspects of solutions in more or less detail, depending on what interested them:

> *" … don't worry about what the detail is … don't plow through each of the internal objects … "*

They chose what was general, and what particular. Some also described holding a simplified image in tandem with a more elaborate one, in order to consider a simpler analog to a knotty problem.

Provisional, variable

All of the imagery could accommodate incompleteness and provisionality, which were usually signalled in the imagery in some way, e.g., absence, fuzziness, partial shading, distance in a landscape, change of tone. This confirms Fishbein's (1987) argument that internal images have two characteristics which contribute to creative discovery: that they are not fully developed, and that they are associated with intrinsic feelings about their correctness; also Miller's claim (1993) that the vagueness of an image is critical to its utility.

Many dimensions

All of the experts reported using more than four dimensions. The extra dimensions were usually associated with additional information (such as overlaid data flows or links to external representations), different views, or strategic alternatives. These strategic alternatives seem to allow single physical dimensions to be represented with more complexity in the design, thereby allowing different sets of problem constraints to be represented along different axes, even where they refer to the same quantity:

> *" … small number of dimensional structures, 4 + 1 say, (e.g., two sizes of time that are different enough to be different dimensions) … "*

Multiplicity

All of the experts described simultaneous, multiple imagery. Some alternatives existed as different regions (so that one alternated imagery by 'turning your head'). Some existed as overlaid or superimposed images, which became more or less transparent depending on the programmer's focus. Some existed as ready transformations or "morphing", e.g., *"changes of scrutiny become textual changes"*, or as *"projection onto translations"*. Some existed on different mental planes: "levels … alternative representations, not connected together … "

'The name business'

Although some of the imagery is largely non-verbal, the experts all talked about the ready ability to label entities in the imagery. They clarified that not all imagery carried explicit labels (although some did encompass a 'name awareness'), but that the labels were available and accessible. This supports our contention that the mental representations we are investigating are multi-modal, apparently extending to linguistic information as well as the traditional "mental pictures" of visual imagery research.

> *"The name business is absolutely necessary to understand things."*

> *"Of course everything is labelled."*

4.3. Imagery reports in questionnaire responses

In the second of these studies, 32 respondents (14%) made unsolicited statements that could be compared to those collected in the first study, even though these descriptions of mental imagery were not as elaborate. (If we had been using an interview technique, we could of course have asked for elaboration). This level of report provides an estimate of general awareness of these issues among professional programmers. Although all the participants in the first study were willing to discuss mental representations in an interview situation, we note that it may be relatively rare for programmers to have carried out any degree of introspection on this topic.

The perceived relationship between LabVIEW use and visual representations led several respondents to describe LabVIEW itself as a kind of mental representation. This reflects earlier findings from the research literature that the discourse surrounding visual programming languages does make free reference to idealised mental representations:

> *Working through of problems … is facilitated by using [LabVIEW] symbols "in your head"; LabVIEW structures become second nature; design with the language itself; LabVIEW is very conceptual; LabVIEW - like representations of data manipulation; [LabVIEW] lets me program the way I think*

The LabVIEW literature emphasizes the value of the "data flow" programming paradigm, and respondents often alluded to this. It was rare for respondents in the study to describe data flow representations as being internal, but several did so:

> *The mind wants to "see" what is going on as a graphical flow; I'm used to thinking in [data flow diagram] terms; I see pictures of data flow; I think in a data flow manner; I think in terms of flow charting and data flow; Data flow closely parallels the problem-solving logic I use; [LabVIEW] allows a concept flow.*

Respondents also described using specific mental representations derived from other design notations:

> *[I] envision program blocks in my head; [LabVIEW] is already a block diagram*

Several respondents described mental representations by reference to verbal alternatives, especially pseudo-code, which some seem to consider as a design aid even when the programming language is visual. We interpret this pseudo-code as corresponding to the labelled representations reported in the first study.

> *Write out a "pseudo-code" simple text outline of the steps and hierarchy I intend to accomplish; the "word" version of what I want to do; transition from pseudo-code to actual program.*

Although others manage to avoid pseudo-code:

> *don't have to clutter the brain with lines of code ; no need for pseudo-code*

Most introspection describes the general properties of mental representation rather than representational details, however:

> *the procedural description of the program I have in my head; an overview type of perspective before I focus in on details; [I can] visualize a programming concept much easier using nodes; I find I can visualize algorithms; [I] visualize a program as a collection of functionality; [I] think problems more structurally.*

The value of these general properties of representations lies in the way that they structure the problem solving task. This is considered in more detail in the next section.

> *Your focus is how and where the data should be directed; think linearly rather than procedurally; think through in a stepwise fashion; a 3-dimensional environment; changes from 1-dimensional (linear, oral) to 2-dimensional (planar, visual).*

It is likely, however, that most of the respondents in this study would have felt uncomfortable making any more detailed claims about the properties of mental representations. The majority did not provide

any information about mental representations, and some were not able to express any information beyond the vaguest impressions:

> *my mental visualization of my program's code is different, in a hard to describe way*

Design representations

When describing the mental representations used in visual programming, respondents were often speaking in the context of particular cognitive activities. It is clear that their perceptions regarding useful features of a representation were based on the degree to which the representation supported these activities. Visualization, for example, was often mentioned:

> *Visualization of the code is primary for me; visualization allows for more global, higher-level thinking; [I] "visualize" graphically my programs which lets me "see" them, create them, and modify them; [the] program becomes easier to visualize; find a solution to a problem visually.*

Mental representations are contrasted with external representations. The process of programming is often described as "translation" of a complete internal representation. Petre (1991) observes that programmers do not design their code directly in the "target" programming language. They create an abstract solution using some kind of mixed representation that cannot be equated with any particular programming language. These responses provide introspective data that support that hypothesis:

> *It takes a while to translate what I want to do to the correct form; I translate into conventional code; take your task conceptually and "translate" it; [requiring] the brain to translate language representations; translating [the] functional outline.*

External representations may be more or less successful in assisting with the maintenance of conceptual structures

> *understand the project conceptually; conceptualize a programming problem; conceive project's overall appearance; generate a solution which follows the nature of the problem; assimilate and control the thought process; my brain "switches modes" … as if it's some instinct … it just feels right; allows you to follow your "programming hunches".*

In fact, some respondents described strategies that were already in place. LabVIEW was seen as providing a representation that corresponded to previous devices. Petre (1996) has proposed that paradigms evolve by reflecting this internalised practice:

> *I usually used sketches before I started writing; I have dreamt in most of the programming languages I have used.*

4.4. Metacognitive statements in questionnaire responses

Some respondents in the questionnaire described their own mental imagery, providing data that can be compared to that collected during programming tasks. However the respondents also made statements about mental imagery in others, comparing it to their own experience. These responses have a different status in the terms of our investigation. They are statements of metacognitive theories - beliefs that people have about their cognitive processes. The significance of metacognitive beliefs is not that researchers should take them at face value as descriptions of cognition, but that they provide the basis for the strategic choices that people make when undertaking a cognitive task such as programming or design. Choice of strategy is a very significant factor in complex cognitive tasks, and it is therefore important to consider them alongside more spontaneous introspective reports.

Because we could not ask for elaboration of these comments as in the first study, many of the comments are less specific than those described above. The advantage of this methodology, however, is that respondents were free to discuss any area of concern to them. This has highlighted information that was not apparent in the first study. Where numbers of respondents are given, this is as a guide to the proportion of respondents concerned with that issue, even though only representative quotes have been included from transcripts.

Individual differences

18 of the questionnaire respondents noted that the kind of mental representations involved in using a visual programming language were not universal amongst programmers. Most of them claimed that LabVIEW was well suited to their own needs, but implied that other programmers might work differently.

> *Some of us are more "visual"; I am a visually oriented type person; I am more of a picture person; I'm a visual person; I think in pictures; I think visually.*

Some more vehement statements about the nature of mental representations made more general claims, although they still allowed for exceptions to the representations they described:

> *My brain works much more efficient in the visual world; That is how my mind works; humans (especially engineers) tend to think graphically.*

Visual-ness in a representation often seems to be defined in opposition to textual representations.

> *Personally, I am not a "text" type of person; I'm not particularly good with languages*

But some respondents complained of difficulty in using LabVIEW because their natural mental representations were in fact textual.

> *It is difficult to take the "word" version of what I want to do and change it; A different mindset is required; For some it may make more sense to program in another manner … all people may not be alike; Something that is unique to each person.*

And one respondent considered their mental representation to be neither visual nor textual.

> *I am not inherently good at [visual] … kinesthetic is my way of learning.*

Creativity

10 respondents made an explicit link between visual mental representations and creative problem solving, along the lines that have been suggested by researchers such as Finke and Slayton (1988) and Dreistadt (1968).

> *I think creatively; [LabVIEW] allows me to problem-solve and be creative; [LabVIEW] sparks creativity; [LabVIEW] results in more creative solutions; using my imagination*

Introspective claims for creative use of imagery are of course subject to the observations made by Katz (1983) that a creative self-image is associated with reports of mental imagery.

> *[LabVIEW] allows me to express a great deal of personal style and panache; [LabVIEW] allows a large amount of personal style; [LabVIEW is] appropriate for a more intuitive programmer*

Some respondents did draw specific distinctions between images and "textual" thinking.

> *In text based languages one must think quite linearly to achieve results; [LabVIEW] fosters a different way of looking at things … different views lead to different solutions.*

5. DISCUSSION

5.1. Common elements of imagery in design

Section 4.2 reported the common elements of imagery reported by experts in the first study. Despite considerable individual variation, multiple participants concurred in describing imagery that was stoppably dynamic, allowed local attention, had adjustable granularity of abstraction, was provisional and variable, had many dimensions and included information in multiple sensory modes - multiple images in addition to accessible labels.

Several aspects warrant further investigation, but especially the relationship between imagery and explanations or labels applied to the image. The respondents to the questionnaire, although they were not addressing the content of their images explicitly, also spent some time considering the verbal content of their mental representation. References to pseudo-code appear to imply verbal representations that can be applied to the graphical conventions of LabVIEW.

5.2. Validity of the studies

The cohort interviewed in the first study, although small, is arguably representative of expert programmers – the findings of the separate population surveyed in the second study give broad support to the importance of visual images as representations, although not addressing specific details of image properties. But there is no assumption here that expert programmers or LabVIEW users are representative of any other population. The distillation is not claimed to cover all possible sorts of imagery, nor is the distribution of images taken to be representative. We do believe that, within the obvious limitations of the studies, the distillation does give a valid glimpse of some of the mental imagery actually used by such people.

The criteria of surprise and satisfaction provide some basis for suggesting that the descriptions given in the first study are arguably close representations of the experts' mental imagery. The experts' surprise at their own accounts indicates at least that they were not just repeating practiced descriptions, that they were articulating things not usually articulated, and that they were impressed by the differences from their own expectations — their own rationalized versions— of what they imagine. Their satisfaction indicates at least that they accepted their accounts as plausible. This is not true to the same degree of the second study, where several themes in the responses were obviously derived from the LabVIEW literature (in discussions of data flow, for example). The fact that so few questionnaire respondents referred to mental imagery, however, partially explains the grounds for surprise when participants in the interview study discovered this aspect of their mental representations.

Note that the imagery elicited here relates to solution *construction*. Many of the experts made comments suggesting that a different imagery may be employed during *debugging*:

> *" … the possibilities of debugging at bottom level from here are zero"*

> *"In debugging, you only do it mentally for the difficult ones: intermittent, incomplete capture of the stimulus … "*

Specific descriptions of debugging from the second study mostly dealt with specific features of LabVIEW itself, and have not been considered here.

5.3. Previous investigations of imagery as a design technique

Designers working in other fields than software development have traditionally described the importance of mental imagery as a design technique. We are not aware of other studies taking the approach we have done with relatively large numbers of designers, but there have been a number of smaller scale studies. These tend to be studies of exceptional performance - even of genius - in which the reports of famous designers or other creative individuals are taken at close to their face value. Ferguson (1992) presents historical arguments for the primacy of imagery in engineering design. Goel (1995) describes a cognitive theory of architectural design that relies on image-like representations. Miller (1984) follows Dreistadt (1968) in describing the importance of imagery in scientific discovery.

The problems with these historical studies and personal accounts is that it is unclear whether they merely express the common teachings of an established discipline, and whether they are even generalisable within the discipline. The current study has the advantage that there are no widely established beliefs of image use in software design, so we are better able to capture personal insights. At the same time, the range of participants in our studies have given us a slightly broader view of the software profession than is managed by Miller or Dreistadt in their more detailed studies of exceptional individuals.

Goel (1995) argues from the basis of cognitive theory that creative design must involve the use of images, which are less deterministic than propositional representations. It is clear from our data that many of the mental representations used by designers are vague or indeterminate. Nevertheless, the use of mental images is only one of the possible strategies that can be used to record undecided design alternatives. Studies of sketching, for example, suggest that designers making preliminary sketches purposely maintain ambiguities in their sketches, or even introduce them through scribbles and messy erasures (Fish & Scrivener 1990, Hewson 1994). In a practical design situation, it seems that designers might employ both external and internal representations, with each type needing to support ambiguity during the design process. Denis (1991) and Cox (1996) argue that the main challenge in problem solving is to identify the most appropriate visualisation for a given problem — and that some problems become trivial given a suitable representation. And yet the external representations — the programming notations — on which many programmers believe that they focus their problem solving are at some remove from the internal images adopted by experts in constructing solutions (Petre 1991). Goldschmidt (1994), and Suwa and Tversky (1997) describe protocol based studies of architects sketching, while Waisel et. al. (1997) have made a similar study of mathematician's sketches made during problem solving. It is likely that detailed protocols of programmers at work may be able to extend those studies.

5.4. Techniques for future investigations of mental imagery

Would it be possible to investigate the use of mental images during software design in an experimental context, or using other empirical techniques? There are several possibilities. One of the most long-standing empirical techniques has been the use of correlational studies in which psychometric measures of verbal and non-verbal IQ are compared to abilities in a particular field of achievement. A strong correlation between achievement and non-verbal IQ is taken to be evidence of the importance of mental imagery in a field, especially where there is no associated correlation with verbal IQ. For example, Winner and Casey (1992) argue on the basis of psychometric measures that fine art students display a relative superiority in image generation. It is our opinion that these studies are overly simplistic for the description of an activity as complex as programming. The base assumptions in many such studies (though not especially in Winner & Casey) show rather simplistic allocation of cognitive activities to propositional and image-like (sometimes even left/right hemisphere) representations. Our data suggest that design representations are complex and multimodal, as well as being reliant on individual strategies (e.g. Chi, et al. 1989; Payne 1993), so correlational techniques do not seem ideal for the detailed study of programming, despite the fact that there are broad correlations between programming ability and most psychometric tests.

Experimental techniques for studying imagery use have often relied on tasks contrived to require an element of creativity, but constrained to be purely image based. These include Finke, Pinker and Farah (1989) and Finke and Slayton's (1988) tasks in which some collection of image components must be rearranged to form a novel composite. However if any element of verbalisation is introduced into simplified tasks of this kind, it can have the effect of radically changing both strategy and performance. Hitch, Brandimonte & Walker (1995) concluded that "verbalization overshadows insight", that is, that requiring people to talk about the task they are carrying out can over-ride insights that might otherwise be achieved through imagery. In complex design tasks, it is easier to study external representations used by the designer rather than attempt to isolate internal representations. However, it is the interaction between external and internal representations that is critical.

Another commonly used experimental technique is the dual task paradigm, in which experimental subjects carry out a main task (in our case, some software design activity) while also performing a secondary task that may or may not rely on the same cognitive resources. Brooks (1968) used dual-task experiments to show that visual-spatial and auditory-linguistic systems represent different processing modes. Brooks (1967, 1968) and Byrne (1974) detected interaction between perception and imagery tasks, suggesting that an image may be an internally constructed perceptual description, using the same resources with which external perceptions are coded. Logie et al. (1991) showed that image tasks are disrupted by intentional eye or arm movement tasks, suggesting that they are spatial rather than visual. Dual task experiments are difficult to design and control, and the main task must often be so simple that its relevance to complex cognitive tasks is questionable. The increasing availability of functional imaging techniques should be able to replace dual-task studies in many cases - we hope that they will provide a major source of data regarding the application of localised cognitive subsystems in specific design activities.

Most of the techniques used to investigate mental imagery have concentrated on distinguishing *modality* in imagery. Modality is only one of the aspects that arises from the data presented here. Several of the other properties that we have identified are amenable to experimental investigation. A detailed methodological review of previous work is beyond the scope of this paper, but in table 1 we summarise possible experimental approaches that could be used to investigate the image properties reported in our studies (including references to some other experimental studies that have investigated those aspects of mental imagery in the past).

Image Property	Experimental Paradigms
Modality (labels, multiple modes)	Dual task experiments (e.g. Byrne 1974; Logie et al. 1991)
Extension (dimensions, surfaces)	Temporal measures of image-scanning and search (e.g. Kosslyn et. al. 1978), image axes (e.g. Franklin & Tversky 1996)
Elaboration (provisionality, detail)	Effects of complexity on image memory (e.g. Chechile et. al. 1996), discovery and ambiguity (e.g. Chambers & Reisberg 1985)
Dynamics (machines, motion)	Mental animation (e.g. Sims & Hegarty 1997, Schwartz & Black 1996)
Attention (locality, lumpiness)	Hemispatial neglect in images (e.g. Bisiach & Luzatti 1978), preference in image formation (e.g. Shuren et. al. 1996)

Table 1 - experimental techniques for investigation of imagery

5.5. Implications for tool design

One of the places to look next is at the sorts of tools experts build for themselves, which presumably complement and supplement their thinking, even if they don't mimic their internal representations. We have noted that an essential component of designing is the process of externalising concepts through sketching or visualisation. It is likely that when software experts draw visual representations of their designs, they gain similar benefits from the externalisation of concepts (Schenk 1991). A number of the experts in our study did develop their own visual representations, and even software tools to support those representations.

This is very different from the function of visual programming languages such as LabVIEW. There has been a great deal of confusion in the visual languages community about the relationship between mental representations of a design and the programming notation. As noted by Green, Petre & Bellamy (1991), the superlativist position (claiming that visual languages always improve performance in programming tasks) is not justified by experimental evidence. Blackwell (1996a) has analysed the variety of beliefs among visual language researchers about the cognitive effects of using different notations. These tend to range far beyond the practical benefits of the notation or the programming paradigm, but their statements about mental representations bear little resemblance to our findings in the present studies.

The visual conventions developed for a particular paradigm, such as those provided in LabVIEW and other visual programming languages, are unlikely to reveal as much about designers' mental representations as the tools that the designers build for themselves. The things that one can visualise 'clearly' within one domain tend not to be what experts build their own tools for. Most errors that consume expert time are not within-paradigm, but looking below the paradigm at internal operation in order to debug abnormal behaviour of entities that are correct within the paradigm, i.e., unknown side-effects, rather than syntactic errors. So visualisations within-paradigm are uninformative. As a result, experts have a tendency to create a visualisation for a particular problem (e.g., specific data structure) even if it will never be useful for another problem. This sort of custom visualisation may well reveal something about how experts identify the most appropriate visualisation for a given problem.

5.6. Conclusion

Any investigation of what lies inside people's minds is problematic. Experimental psychology has led us to value only what can be demonstrated, but that discipline has not provided sufficient insight into the realm of complex, abstract, imagined structures. For our early ventures into that realm, careful elicitation can provide valid insight. Strong commonalities emerged among the images elicited from ten expert programmers during design sessions, and these were supported by references to imagery from a further 63 questionnaire respondents. These results suggest that there are accessible lessons to be elicited about how programmers construct solutions. The results presented here provide a glimpse of what experts do when they 'sit and stare.' Although we have concentrated on diversity in mental representations, and believe that is an important goal for studies of this type, we have identified potential avenues for specific investigation in the future using experimental techniques as well as further observational studies.

6. ACKNOWLEDGEMENTS

The authors are grateful to the expert programmers interviewed in the first study, without whom the paper would not be possible. Kirsten Whitley collected the data described in the second study, and has generously agreed to its use in this comparison. Thomas Green and Peter Eastty each provided essential commentary at crucial moments. Four anonymous reviewers provided very valuable feedback on an earlier draft, and suggested several themes that have been expanded in the paper.

Alan Blackwell's research was funded by a collaborative studentship from the Medical Research Council and Hitachi Europe Ltd. He is grateful to the Advanced Software Centre of Hitachi Europe for their support.

7. REFERENCES

Anderson, R.E. & Helstrup, T. (1993). Visual discovery in mind and on paper. Memory and Cognition, 21(3), 283-293.

Arnheim, R. (1969). ***Visual Thinking***. University of California Press.

Bartlett, F.C. (1927). The relevance of visual imagery to thinking. ***British Journal of Psychology***, **18** (1), 23-29.

Bisiach, E. & Luzatti, C. (1978). Unilateral neglect of representational space. ***Cortex***, **14**(1), 129-133.

Blackwell, A.F. (1993). Bottom-up design and this thing called an 'object'. ***EXE Magazine***, **8**(7), 28-32.

Blackwell, A.F. (1996a). Metacognitive Theories of Visual Programming: What do we think we are doing? In ***Proc. IEEE Workshop on Visual Languages, VL'96***.

Blackwell, A.F. (1996b). Do Programmers Agree with Computer Scientists on the Value of Visual Programming? In A. Blandford & H. Thimbleby (Eds.), ***Adjunct Proceedings of the 11th British Computer Society Annual Conference on Human Computer Interaction, HCI'96***, pp. 44-47.

Brooks, L.R. (1967). The suppression of visualization in reading. ***Quarterly Journal of Experimental Psychology***, **19**, 289-299.

Brooks, L.R. (1968). Spatial and verbal components of the act of recall. ***Canadian Journal of Psychology***, **22**, 349-368.

Byrne, B. (1974). Item concreteness vs. spatial organization as predictors of visual imagery. ***Memory and Cognition***, **2** (1A), 53-59.

Chambers, D. & Reisberg, D. (1985). Can mental images be ambiguous? ***Journal of Experimental Psychology: Human Perception and Performance***, **11**, 318-328.

Chechile, R.A., Anderson, J.E., Krafczek, S.A. & Coley, S.L. (1996). A syntactic complexity effect with visual patterns: Evidence for the syntactic nature of the memory representation. ***Journal of Experimental Psychology: Learning, Memory & Cognition***, **22**(3), 654-669.

Chi, M.T.H., Bassok, M., Lewis, M.W., Reimann, P., and Glaser, R. (1989). Self explanations: how students study and use examples in learning to solve problems. ***Cognitive Science***, **13**, 145-182.

Cox, R. (1996). ***Analytical reasoning with multiple external representations***. Unpublished doctoral dissertation. University of Edinburgh.

Cox, R. & Brna, P. (1995). Supporting the use of external representations in problem solving: the need for flexible learning environments. ***Journal of Artificial Intelligence in Education***, **6**(2), 239-302.

Davies, S.P. (1996). Display-based problem solving strategies in computer programming. In W.D. Gray & D.A. Boehm-Davis (Eds.), ***Empirical Studies of Programmers****: Sixth Workshop*. Norwood, NJ: Ablex, pp. 59-76.

Denis, M. (1991). Imagery and thinking. In C. Cornoldi & M.A. McDaniel (eds.), ***Imagery and Cognition***. New York: Springer-Verlag.

Dreistadt, R. (1968). An analysis of the use of analogies and metaphors in science. ***The Journal of Psychology***, **68**, 97-116

Ericsson, K.A. & Simon, H.A. (1985). ***Protocol analysis: Verbal reports as data***. Cambridge, MA: MIT Press.

Ernest, C.H. (1977). Imagery ability and cognition: a critical review. ***Journal of Mental Imagery***, **1** (2), 181-216.

Farah, M.J., Hammond, K.M., Levine, D.N., & Calvanio, R. (1988). Visual and spatial mental imagery: dissociable systems of representation. ***Cognitive Psychology***, **20**, 439-462.

Ferguson, E.S. (1992). ***Engineering in the Mind's Eye***. Cambridge, MA: MIT Press.

Finke, R.A., Pinker, S., & Farah, M.J. (1989). Reinterpreting visual patterns in mental imagery. ***Cognitive Science***, **13** (1), 51-78.

Finke, R.A. & Slayton, K. (1988). Explorations of creative visual synthesis in mental imagery. ***Memory and Cognition***, **16**, 252-257.

Fish, J. & Scrivener, S. (1990). Amplifying the mind's eye: sketching and visual cognition. ***Leonardo***, **23** (1), 117-126.

Franklin, N. & Tversky, B. (1990). Searching imagined environments. ***Journal of Experimental Psychology: General***, **119** (1), 63-76.

Gilmore, D.J. & Green, T.R.G. (1984). Comprehension and recall of miniature programs. ***International Journal of Man-Machine Studies,*** **21** (1), 31-48.

Goel, V. (1995). ***Sketches of Thought***. Cambridge, MA: MIT Press.

Goldschmidt, G. (1991). The dialectics of sketching. ***Creativity Research Journal***, **4**(2), 123-143.

Goldschmidt, G. (1994). On visual design thinking: The vis kids of architecture. ***Design Studies***, **15**(2), 158-174.

Green, T.R.G. & Petre, M. (1996). Usability analysis of visual programming environments: a 'cognitive dimensions' approach. ***Journal of Visual Languages and Computing***, **7**,131-174.

Green, T.R.G., Petre, M. & Bellamy, R.K.E. (1991). Comprehensibility of visual and textual programs: A test of superlativism against the 'match-mismatch' conjecture. In J. Koenemann-Belliveau, T.G. Moher & S.P. Robertson (Eds.): ***Empirical Studies of Programmers: Fourth Workshop*** Norwood, NJ: Ablex, pp. 121-146.

Hewson, R. (1994). ***Marking and making: A characterisation of sketching for typographic design***. Unpublished PhD thesis, Open University, UK.

Hishitani, S. (1990). Imagery experts: How do expert abacus operators process imagery? ***Applied Cognitive Psychology***, **4**, 33-46.

Hitch, G.J., Brandimonte, M.A., and Walker, P. (1995). Two types of representation in visual memory: evidence from the effects of stimulus contrast on image combination. ***Memory and Cognition***, **23**, 147-154.

Katz, A.N. (1983). What does it mean to be a high imager? In J.C. Yuille (ed.), ***Imagery, Memory and Cognition: Essays in Honor of Allan Paivio***. Hillsdale, NJ: Erlbaum.

Kaufmann, G. (1979). Visual imagery and its relation to problem solving. Oslo, Norway: Universitetsforlaget.

Kosslyn, S.M. (1994). ***Image and the Brain: The Resolution of the Imagery Debate***. Cambridge, MA: MIT Press.

Kosslyn, S.M., Ball, T.M. & Reiser, B.J. (1978). Visual images preserve metric spatial information: Evidence from studies of image scanning. ***Journal of Experimental Psychology: Human Perception and Performance***, **4**, 47-60.

Lammers, S. (1986). ***Programmers at Work***. Redmond, Washington: Microsoft Press.

Lindsay, R.K. (1988). Images and inference. ***Cognition***, **29** (3), 229-250.

Logie, R.H., and Marchetti, C. (1991). Visuo-spatial working memory: visual, spatial or central executive? In R.H. Logie & M. Denis (eds.), ***Mental Images in Human Cognition***. Amsterdam: Elsevier. 105-115.

Mani, K., & Johnson-Laird, P.N. (1982). The mental representations of spatial descriptions. ***Memory and Cognition***, **10** (2), 181-187.

Miller, A.I. (1984). ***Imagery in Scientific Thought: Creating 20th-Century Physics***. Cambridge MA: MIT Press.

Miller, G.A. (1993). Images and models, similes and metaphors. In A.Ortony (ed.), ***Metaphor and Thought***, 2nd edition. Cambridge: Cambridge University Press, 357-400.

Navarro-Prieto, R. (1998). ***The role of imagery in program comprehension: Visual programming languages***. Doctoral Thesis, University of Granada. ISBN 84-8497-957-1.

Payne, S.J. (1993). Memory for mental models of spatial descriptions: an episodic-construction-trace hypothesis. ***Memory and Cognition***, **21** (5), 591-603.

Petre, M. (1989). Finding a Basis for Matching Programming Languages to Programming Tasks. Unpublished doctoral dissertation. University College London.

Petre, M. (1991). What experts want from programming languages. ***Ergonomics*** (Special Issue on Cognitive Ergonomics), 34 (8), 1113-1127.

Petre, M. (1995). Why looking isn't always seeing: readership skills and graphical programming. ***Communications of the ACM***, **38** (6), 33-44.

Petre, M. (1996). Programming paradigms and culture: Implications of expert practice . In M. Woodman (Ed.), ***Programming Language Choice: Practice and Experience***. Chapman and Hall. 29-62.

Petre, M., and Winder, R.L. (1988). Issues governing the suitability of programming languages for programming tasks. In ***People and Computers IV: Proceedings of HCI'88***. Cambridge University Press.

Pylyshyn, Z. (1973). What the mind's eye tells the mind's brain. ***Psychological Bulletin***, **80**, 1-24.

Scaife, M. & Rogers, Y. (1996). External cognition: how do graphical representations work? ***International Journal of Human Computer Studies***, **45**, 185-214.

Schenk, P. (1991). The role of drawing in the graphic design process. ***Design Studies***, **12** (3), 168-181.

Schooler, J.W., Ohlsson, S., and Brooks, K. (1993). Thoughts behond words: when language overshadows insight. ***Journal of Experimental Psychology: General***, **122** (2), 166-183.

Schwartz, D.L. & Black, J.B. (1996). Analog imagery in mental model reasoning. ***Cognitive Psychology***, **30**, 154-219.

Shuren, J.E., Greer, D. & Heilman, K.M. (1996). The use of hemi-imagery for studying brain asymmetries in image generation. ***Neuropsychologia***, **34**(6), 491-492.

Sims, V.K. & Hegarty, M. (1997). Mental animation in the visuospatial sketchpad: Evidence from dual-task studies. ***Memory & Cognition***, **25**(3), 321-332.

Suwa, M. & Tversky, B. (1997). What do architects and students perceive in their design sketches? A protocol analysis. ***Design Studies***, **18**, 385-403.

Waisel, L., Wallace, W.A. & Willemain, T.R. (1997). Using diagrammatic representations in mathematical modeling: The sketches of expert modelers. In M. Anderson (Ed.), ***Reasoning with Diagrammatic Representations II: Papers from the AAAI 1997 Fall Symposium***. Technical Report FS-97-02. Menlo Park, CA: AAAI Press, pp. 125-135.

Whitley, K.N. and Blackwell, A.F. (1997). Visual programming: the outlook from academia and industry. In S. Wiedenbeck & J. Scholtz (Eds.), ***Proceedings of the 7th Workshop on Empirical Studies of Programmers***, pp. 180-208.

Whitley, K.N. and Blackwell, A.F. (1998). ***Visual programming in the wild: A survey of LabVIEW programmers***. Technical Report CS-98-03, Computer Science Department, Vanderbilt University. Nashville, TN.

Winner, E. & Casey, M.B. (1992). Cognitive profiles of artists. In G.C. Cupchik & J. László (Eds), ***Emerging visions of the aesthetic process: Psychology, semiology and philosophy***. Cambridge: Cambridge University Press.