Hoyles, C. and Noss, R. (2003) 'What can digital technologies take from and bring to research in mathematics education?' In A.J. Bishop, M.A. Clements, C. Keitel, J. Kilpatrick and F.K.S. Leung (eds), *Second International Handbook of Mathematics Education*. Dordrecht: Kluwer Academic Publishers.

What can digital technologies take from and bring to research in mathematics education?

Celia Hoyles & Richard Noss
Mathematical Sciences Group
Institute of Education, University of London

Preface

There are many reasons to write a chapter of a research handbook; among these are a recognition of recent developments, the need to summarise a field of research, and the importance of providing a ready source of bibliographic references. It is also a chance for the authors to give their particular view of a given area, ride their particularly hobby-horse, or simply focus on what seems to them to be important, even if it runs the risk of offending those whose work is not adequately referenced. Research in the use of digital technologies in mathematics has proliferated, using a wide range of theories and methodologies. Whereas it might have been possible a decade or so ago to write a comprehensive review in just one chapter of research on the effects of digital technologies on the whole of mathematics teaching and learning, the vast corpus of study that now exists makes this no longer feasible (indeed, as we complete this review, there is an entire edited book being prepared on this one subject: Heid & Blume, in preparation). As the title of this chapter makes clear, we have decided to take as our focus neither the technology itself nor the ways pedagogies have changed under its influence. Rather, our focus is on how the incorporation of technologies has afforded insights for mathematics education; and reciprocally, how research with digital technologies is beginning to be informed by the development of new theoretical frameworks.

Our aim therefore, is to bring the field of research with and on computationally-based technologies in mathematical learning closer to the broader field of mathematics education research. We take it as axiomatic that each has much to learn from the other; but we are fully aware of just how insulated the work with digital technology has been. Given the knowledge explosion in all modern research fields, it is no doubt tempting for workers in the field of mathematics education to feel that there is at least one source of literature that they can safely ignore; reciprocally, we are aware that the broader research effort that focuses on digital technologies is often oblivious to the continuing growth in the depth and diversity of mathematics education research. Both positions miss an essential point. Our claim is that there are major research issues for mathematics education that are shaping and being shaped by the issues confronting 'technologists'. In what follows, therefore, we attempt to address a range of themes that are thrown up by placing the two fields into closer proximity, and to map out interesting avenues for investigation that arise from examining the interrelationship between them.

Hoyles, C. and Noss, R. (2003) 'What can digital technologies take from and bring to research in mathematics education?' In A.J. Bishop, M.A. Clements, C. Keitel, J. Kilpatrick and F.K.S. Leung (eds), *Second International Handbook of Mathematics Education*. Dordrecht: Kluwer Academic Publishers.

Introduction

It seems, at least in the domain of research and development, that something significant is beginning to happen in the application of digital technologies to the learning and teaching of mathematics. Perhaps the hitherto inescapable forces which have wedded mathematical learning and teaching to a methodology of the precomputational era are at last beginning to be relaxed.

In part, the sheer ubiquity of personal computers has brought about a cultural shift in how people think with and about computers. The possibilities are evident, and above all, the kinds of technological potential that are now emerging contain the seeds of radical change. Added to this is the ease of widespread dissemination of ideas and practice, in ways which were unthinkable except by a very few when the last handbook of research in mathematics education was published[1].

Changes in the computational domain open up only the potential for change, not actual change in the didactical field. Only those suffering from acute technological determinism could fail to acknowledge how much research remains to be done in order to effect radical and fundamental change in mathematics education even when this is supported by accessible technology and based on research. Kaput (1992) laments the lack of technology-related research of any kind, attributing the continuing marginalisation of technology in mathematics education to the complex issues that surround its use. Among the causes, he cites:

- Technology requires one to continually rethink pedagogical and curricular motives and contexts.
- Classroom-based research is difficult, since exploiting the real power of the technology requires such innovative approaches that comparison to a traditional class is inappropriate.
- The practical complications of student access to computers, cost of software, and development of curricular materials often prohibit research.
- Due to the rapid changes in technology, research is often outdated by the time it is complete.

This list is clearly not exhaustive, and there are other issues that could now be appended to it (for an earlier review, see Heid, 1997). A key element is the existence of two inevitable and interrelated tensions in software use. The first is that learners need to be able to cope with the syntax and semantics of the software: they have to find out how it works, what it affords, and how it might be employed. It is often considered, therefore, that the employment of technology adds an overhead to learning. Nevertheless, as some have pointed out (for example diSessa, 2000; Wilensky, 2001), this is not necessarily the case any more, as learning about software increasingly becomes an integral part of learning mathematics.

[1] When the first volume was published, the World Wide Web was a novel tool just beginning to be used by academic scientists.

The second tension is that students will tend to use the power of the technology to avoid the cognitive load of 'mathematical thinking' – not a surprising observation when we consider that this is precisely the role that software mostly plays in the larger culture. In this respect, it is important to distinguish the needs of mathematical *learners* from the needs of mathematical *users* – learners need to search for and appreciate generality and structure, while users want simply to get a particular job done or a problem solved. This distinction, while it applies generally to tools, is particularly crucial with regard to digital technologies; the 'point' of a spreadsheet (from the program designer's point of view) is precisely that the actions of pointing and clicking replace the need to think algebraically (for discussion of this tension in relation to Logo work, see Noss & Hoyles, 1992; and in relation to graphing software, Goldenberg 1991).

Our first task is to delineate explicit boundaries for our review. One piece of the boundary arises from the restriction of our interest to software with at least some claim to transformative potential for mathematics learning. Of course, this begs some difficult questions; how do we judge such software? On what criteria? In fact, our response to such questions is somewhat self-referential: we have made our choice in order to maximise the extent to which a piece of technology assists in gleaning insights into students' conceptions and practices, an idea that we elaborate in our book (Noss & Hoyles, 1996). The key point is that expressive computational engagement on the part of students offers observers a *window* onto mathematical meaning under construction; or put another way, while students use and construct tools to build models to explore and solve problems, their thoughts become simultaneously externalised and progressively shaped by their interactions with the tools.

This restriction of interest has led us to exclude serious consideration of some significant genres of digital-technology use that are beginning to impact mathematics teaching and learning. First, "Intelligent tutor" programs are making some headway in becoming more sensitive to students' partially-formed mathematical responses and to the role of the teacher; one of the most cited references here is Koedinger, Anderson, Hadley, & Mark (1997). Second, we make no reference to the 'puzzle-style' software, which in the UK is possibly the most ubiquitous application of technology in mathematics classrooms but whose impact remains largely unresearched. Third, we make little reference to handheld technologies in the form of personal data assistants (PDA's) and calculators, despite their growing popularity[2]. Fourth, we have chosen not to review the role of digital technology in collaborative mathematical learning, where studies have touched on issues of epistemology and design alongside social-psychology[3], and where networking technologies (such as the Internet and digital video) are opening up new possibilities for research. Fifth, we mention that much technology that is highly rated for learning in general, is rather

[2]Of course some of the software to which we will refer also runs on handheld devices, and the general distinctions between handheld, laptop and desktop computers are becoming ever more blurred.

[3] In relation to design, if the computer is to act as a genuine mediator of social interaction through which shared expression can be constructed, careful thought must be given to the kinds of software to be used (see Confrey et al 1991; Roschelle, 1992). In relation to social-psychological questions, Healy, Pozzi & Hoyles (1995), point to the importance of a balanced co-construction at the computer coupled with the co-ordination of others' perspectives orchestrated by didactical intervention.

little researched in the context of mathematical learning, such as the use the Internet as an on-line resource.

Representational forms and cultures

Some 20 years ago, Taylor (1980) suggested a framework for categorising software into the roles of "tutor, tool and tutee', and maintained the fundamental difference between these kinds of use. A little later, Pea (1987) offered a description of a 'cognitive technology' as a medium that helped to "transcend the limitations of the mind" (p. 910), and distinguished the 'amplifying' and 'reorganising' roles of technologies. More recently, Salomon (1992) traced three historical approaches to educational computing: computer-aided instruction/intelligent tutoring systems (CAI/ITS), programming, and tools.

Although these delineations have proved helpful in classifying software, some are now beginning to outlive their usefulness, partly because of developments in technology: the tutoring role for example is not as distinct a category as it once was, with programming languages and software 'tools' now offering 'tutorial' elements, and a kind of interactivity not possible in the past (see Lesh and Kelly, 1996). However, it is not only technological development that has pointed to the limitations inherent in these distinctions, but also a shift in research paradigm towards a recognition of the need, not simply to classify software, but to understand how software enters the activities of communities of learners. One way to approach this – essentially cultural – challenge is to consider digital technologies in terms of their representational contingency for mathematical learning i.e. the ways in which dynamic, manipulable and interactive representational forms mediate and are mediated by mathematical thinking and expression.

The notation systems that we use to present or re-present our thoughts to ourselves and to others, to create and communicate records across space and time, and to support reasoning and computation, constitute a central part of our cultural infrastructure. In this respect, the evolution of mathematical notation systems provides a paradigmatic example of the ways in which the development of notational forms has shaped intellectual development. diSessa (2000) discusses the huge intellectual gains produced by the development of an algebraic representation for mechanics and argues that Leibniz's notation for calculus was at least as important as the physical insights that it encoded. Olson (1994) provides a more general analysis, including an interesting discussion of a range of evolutionary changes in the semiotics of representational systems in the case of art and mathematics. Kaput & Shaffer (in press), by adopting a historical perspective on mathematical notations, also shows how representational infrastructures developed in response to the needs of specific social groups. For example, they demonstrate how algebraic symbolism gradually freed itself from the functional ambiguities and general expressiveness of natural language so that, by the 17^{th} century, it had succeeded in embodying general mathematical relations and functions (see also Kaput, Noss and Hoyles, in press).

Is the computer a qualitatively different entity, or does it merely present one more development in representational power? Kaput (1999) argues for the former position, and suggests that the computer heralded a new kind of culture – a virtual culture – which differs crucially from preceding cultural forms. Not only is there a new representational infrastructure but also the externalisation (from the human mind) of general algorithmic processing. For the first time, neither the functions of recording nor of processing require human intellectual activity (see Kaput and Schaffer, in press, for elaboration and examples).

Precomputational infrastructures made it necessary for individuals to pay attention to calculation: generations of ‘successful’ students can testify to the fact that calculational ability was (and still is) sufficient for passing examinations, without the necessity to understand how the symbols worked. While the need to think creatively about representational forms arose less obviously in settings where things were mechanical and much more visible (i.e. objects had gears, levers, pulleys etc.), the devolution of processing power to the computer has generated the need for individuals *to represent for themselves* models of how things work, what makes systems fail, and what would be needed to correct them (see Noss, 1997 for an elaboration of this point; see also Hoyles, Morgan and Woodhouse, 1999).

The representational perspective affords both a means to classify technologies and to revisit some of the distinctions with which we began. For example, the “computer as tutee” (programming) and “computer as tool” both provide the means to model mathematical relationships but have a different relationship with the representational framework. On the one hand, *programming* or building programmable tools, presents novel ways of modelling and representing mathematics, while, on the other hand, what we shall term *expressive tools* aim to provide ready access to the results of procedures and algorithms without the necessity for learners to attend to their production, to open up the tools or to evaluate alternative representations. The outcome of using the tools rather than the tool structures is the focus of the users’ thinking – to obtain an answer or some information, to calculate a result, to construct a graph. There are, as we shall see, commonalities between these two genres; but since they have different histories and developmental trajectories, we will use the distinction to structure our review.

Programming and Microworlds

We begin this section with a brief summary of programming languages, a software genre that has the longest and perhaps the most controversial history in its relation with mathematics education. We then describe research with *microworlds*, environments based on a (visible) programming language that allow students to construct and reconstruct elements of the environment.

The history of research on programming as part of mathematical learning and teaching is more than 30 years old. The 1980's saw the first widespread implementations of computers and programming software into schools, and this was accompanied by numerous studies that aimed to evaluate the ‘effect’ of programming. Even today the majority of studies are centred around the use of Logo, a programming language first developed in the late nineteen-sixties (Feurzeig and Papert, 1969; Papert 1980). We do not propose to review research in this area over such a long period: for an overview, see Noss and Hoyles (1996, Chapter 3); and for recent and comprehensive surveys see Yelland (1995); Clements *et al*., (2001).

Our focus will not centre on 'effects' of programming. Rather we will draw attention to a more nuanced discussion in the literature of what programming is, and what it might bring to mathematics education research[4]. 'Mathematical' programming languages today feature a range of representational enhancements that have brought about qualitative changes in what it is possible to do and what kinds of mathematics can be expressed (see for example, Wilensky, 1995; Sfard & Leron, 1996; Noss & Hoyles, 1996; Clements, 1999; diSessa, 2000).

What does it mean, then, to engage in programming for the purposes of mathematical education in the twenty first century? The general case for programming is nicely put by diSessa (2000) who argues that it "turns analysis into experience and allows a connection between analytic forms and their experiential implications that algebra and even calculus can't touch" (*ibid.*, p. 34). Earlier, Sendov & Sendova (1995) had stressed the affordance of a programming language for expressing, elaborating and communicating ideas. These issues of expressiveness and collaboration are central to the theoretical framework of *constructionism* (Harel & Papert 1991), which developed out of research with Logo but continue in other studies (for example, with *Boxer,* diSessa, 2000). Some studies that place programming at the heart of their endeavour are taking a further step away from recasting mathematical activity in new representational forms, and towards recasting the nature of school mathematical activity itself. Wilensky (1997) describes how, as 'therapy' for 'epistemological anxiety' about statistical distributions, he offered students the opportunity to build models of problem situations, using an object-based parallel modelling language called StarLogo[5]. This use of parallel modelling systems is a paradigmatic case of a wider set of investigations of how students can come to understand the surprising but pervasive fact that large-scale patterns in the world are more often than not the result of the interactions of a large number of small, independently-acting components (see, for example, Resnick, 1994; Wilensky & Stroup 2000). The crucial point is that such phenomena can *only* be explored and understood in an environment that affords learner-construction, such as a programming language.

Despite the potential of programming environments such as Boxer and StarLogo, there are clearly limitations to text-based interaction, not least since textual literacy is a prerequisite. How can programming environments be used to develop new infrastructures for expressing relationships? Hoyles & Noss have been involved in building and evaluating systems – "Playgrounds"– in which young, pre- and early-literate children (aged 6-8) can explore through programming, the mathematics of video-game construction (see, for example, Hoyles, Noss & Adamson, in press)[6].

---

[4] The following paragraphs owe much to a workshop on programming at the AERA conference 2001 in Seattle, with participants Andy diSessa, Chris Hancock, Bruce Sherin, Uri Wilensky, and ourselves. For an earlier, in-depth discussion see diSessa, Hoyles & Noss (1995).

[5] Like its ancestor, StarLogo allows the programmer to control the state (position, heading etc) of a turtle on the screen. Unlike its ancestor, StarLogo allows the programmer to control many thousands of turtles in parallel.

[6] The approach of Playground in which children design and build their own games, as well as play them, can be contrasted with the use of computer games for children to play in order to learn mathematics, which has found to be effective in children's learning and enjoyment of mathematics (see for example the E-GEMS project based at University of British Columbia)

One Playground consists of higher-level tools built on ToonTalk, a programming language with a radically new representational infrastructure in which the medium of expression is not text or icons, but consists of the direct manipulation of animated cartoon-like characters (Kahn, 1999).

The approach of the Playground project highlights a characteristic element of many studies mentioned in this review, namely that the development of the system and the evaluation of its efficacy proceed iteratively and in tandem. The Playground project's findings demonstrate how the specificities of the design of the playground 'layer' mediate the formalisation of the rules which underpin the children's constructed video games. In particular, results from studies of the different ways in which children articulate a simple rule they had programmed themselves, indicate that rule-expression is shaped by the type of prompting given to help children express their thinking ('predict', 'describe' or 'explain'), the medium of expression (computational, spoken or written), the narrative of the game, and its collaborative context (face-to-face or remote): Hoyles, Noss, Adamson & Lowe (2001).

In general, we can identify a shift away from studying the possibilities of expressing mathematical relationships within a language towards a stronger focus on what might be written and read with it. Accordingly, we now turn to studies that fall within this new direction; that is investigations of what can be done *with* programs, and how sets of programmable tools or *microworlds,* designed to explore mathematics might be interconnected, manipulated and modified in pursuit of mathematical learning goals.

Edwards (1998) reviews the different ways in which the term "microworld" came to be used within the mathematics and science education research communities, and makes a useful distinction between a *structural* definition, which focuses on epistemological facets of microworld design, and a *functional* definition that points to the ways in which students may learn within them. She concludes:

> It is perhaps in this sense that we can speak of a microworld as "embodying" a subdomain of mathematics or science: not because of some reifying link between the representation and the mathematical or scientific entity, but because of the opportunity that such environments provide for learners to kinaesthetically and intellectually interact with the designers' construction of these entities, as mediated through the symbol system of a computer program. (Edwards, 1998, p. 74)

Similarly, Hoyles (1993) traces the origins of the microworld idea, noting that the early vision was of tools 'embodying' mathematics. This vision developed over time to include playful and informal interactions in which "software tools and knowledge would grow together interactively in the pursuit of epistemologically rich goals" (*ibid.,* p. 3). Much of the current microworld-based research is concerned with studying learning trajectories within carefully designed microworlds, and this provides perhaps the most compelling vision of computational systems as 'windows', or tools for better understanding of what learners can do and think. Edwards' description of the "opportunity [to] interact with the designers' construction" is nicely put: it is an affordance that all microworlds designers seek to maximise. One might even say that, in the standard sense, it is this affordance that distinguishes a microworld from other kinds of tools.

But even this distinction is problematic. It fails to take account of the potential differences in learning arising from interaction with 'open' microworlds – based on programming languages – and those that are not. In the former case (and only then) the learner is able, not only to interact with the designer's intentions but negotiate with them.

One major way in which, historically, learners have effected this negotiation, is by the manipulation of symbolic code to explore graphical outputs, although as we have seen, there are various steps away from exclusive reliance on this single form. There has been a wide array of research in this field, spanning a range of mathematical topics and age and mathematical experience of student. For example, Kynigos (1991) reports that after experience with a Logo microworld, students were able to construct bridges between intrinsic and cartesian geometry. Duncan Jones (1998) devised a set of Logo programming tasks for young children to explore ideas of function and proof. Working with much older students, Sacristán (2001) found that the tools of a Logo microworld allowed students to discriminate and coordinate subtle process-oriented features of infinite processes. In a study with older students, Stevenson (2000) built a Logo-based microworld that afforded exploration of non-euclidean geometries, and notes that students were able to develop "a feel" for hyperbolic space by interacting with the tools he had iteratively designed.

Many early studies took the symbolic notation afforded by the programming environment as a focus of investigation. For example, Sutherland (1989) shows that Logo experience enhanced students' understanding of variable, but notes that the links made between variable in Logo and variable in algebra are sometimes problematic; Ursini (1994) demonstrates that by engaging with Logo programming, different characterisations of variable could be developed prior to formal algebra teaching. More recently, some programmable microworlds have attempted to bring the symbolic-graphical relationship more into balance, by exploiting new interfaces in the design of the tools. This was done, for example, by Noss, Healy and Hoyles (1997) in their design of a microworld, *Mathsticks*. Their studies showed how the interactions within the microworld helped students appreciate mathematical structure by forging links between the rhythms of their actions on the computer and the corresponding visual and symbolic representations developed on the screen. Using a similar interface design in a microworld for reflective symmetry, *Turtle Mirrors*, Hoyles & Healy (1997) conclude that learning evolved in tandem with tool development, that is from "thinking about reflection with the tools". Similarly, Pratt (1998, 2000) tracks the co-evolution of young children's meanings of randomness and the tools he designed in a *Boxer* microworld designed to make visible some of the mechanisms of random behaviour.

It has been well documented that without careful consideration of interfaces, students can tend to lose the psychological connection between symbolic (programming) code and graphical output, especially when they are encouraged to use procedures rather than work in immediate mode (see Hoyles & Sutherland, 1989; Clements, 1999). In the development of TurtleMath (Clements & Samara, 1995), the relationship between these two representational forms was redesigned, not at the interface but at the system level. In a series of classroom studies, Clements and his colleagues showed that TurtleMath supported students' mathematical development, and that students showed positive learning gains in their conceptions of length (Clements *et al*. 1997) and angle (Clements *et al*., 1996).

As befits research on novel and carefully designed environments, all the researchers mentioned above note that learning evolved in ways that were contingent on design, both of the software and the activities presented to the students. Thus a further qualitative change in programming-based research can be discerned. Alongside a growing recognition that outcomes for learning are not only contingent on tasks, activity structures and pedagogical context, it has increasingly become recognised that student learning is deeply sensitive to changes in software design. This insight, in turn, invites reconsideration of epistemological issues. Perhaps the boldest conclusion in this regard has been proposed by Sherin (2001), who suggests that programming (in Boxer) could shift the ontological foundations of school physics and mathematics, and that "the nature of the understanding associated with programming-physics might be fundamentally different than the understanding associated with algebra-physics". (p. 1). For a similarly radical point of view, see Papert's (1996) exploration in "the space of [possible] mathematics educations".

Expressive tools for learning mathematics

In this section we turn to review research with expressive tools, tools which their authors might consider as essentially 'black box', but which might nonetheless be customised to some extent by a user through, for example, the use of 'macros' or 'scripting'[7]. As such, these tools offer the learner an expressive sub-system within which mathematical knowledge may be explored. Expressive tools tend to fall into two categories: pedagogic tools finely-tuned for the exploration of a mathematical domain, and calculational instruments, often adapted to, rather than designed for, educational purposes.

We first present a survey of a small selection of expressive tools, before providing a more extensive review of developments in the research culture around one particular class of tool in each category; *dynamic geometry systems* as an example of a pedagogical tool, and *computer algebra systems* as an example of a calculational instrument.

A selective survey

We begin with a set of studies with physical devices linked to a computer that attempt to exploit body syntonicity – relating learning to one's sense and knowledge about one's own body (Papert 1980) and aiming to make exploration of mathematics more tactile, experiential, and intimate. Nemirovsky *et al.,* (1998); Noble *et al*., (2001) have exploited this approach in a number of ways. For example, *Contour Analyser* allows the construction of computer-generated graphs corresponding to contours on a physical object, aiming to make exploration of mathematics more

[7] We think it likely that the firm delineation between programmable and non-programmable systems may, in the not-too-distant future, become further blurred, although it is clear that the future will be determined by social, economic and educational-political judgements as much, if not more than, technical ones.

tactile, experiential, and intimate. The researchers suggest that the traditional dichotomous relationship between internal and external representations unnecessarily restricts the possibilities of researching the ways mathematical meanings are constructed. This work has continued, employing a range of other devices, see for example, Noble *et al*. (2001).

The issue of syntonicity provides a new vantage point on representational forms, inviting us to consider not only the forms of new inscriptions, but the means by which they are inscribed. Just as writing this paragraph with a pen would represent a very different engagement with the ideas within it (and undoubtedly result in a *different* paragraph), the expression of mathematical knowledge – especially incomplete 'knowledge-under-construction' – will be different as keyboards give way to more kinaesthetic devices, to eye movement sensors and beyond. It is unlikely that learnable mathematical knowledge will remain invariant under such radical changes in representational medium.

In a further development, Kaput and his colleagues on the *SimCalc* project (Kaput & Roschelle, 1999; Roschelle *et al*., 2000) identify a variety of new representational forms, none of which require algebraic infrastructure for their use and comprehension. For example, Kaput & Roschelle (1999) report how students are able to investigate fundamental ideas such as the mean value theorem, and the theoretical concept of continuity over an interval, without the necessity to have already mastered the algebraic representational forms within which these ideas are usually expressed.

Recently, the SimCalc team have begun to study the effects of wireless connectivity afforded by cheap and powerful handheld devices, a direction which, they argue, might realise the kind of systemic change in mathematics education which has eluded technology-based curriculum innovators until now. Research on handheld technologies is also being undertaken by a variety of other workers: see, for example, Wilensky & Stroup (2000) and Resnick, Berg & Eisenberg (2000).

The examples of kinaesthetic devices and manipulable graphical representations mentioned in this section share the aim of providing tools that are designed at an appropriate level, or 'grain size', to provide descriptive and manipulative power for mathematical ideas. They suggest a challenge to the curricular givens of pre-computational mathematics, and particularly to rethink assumptions of what mathematical knowledge is appropriate to teach to whom, and when (see Confrey, 1993, for a discussion of challenge in the context of functions and algebra). There are, as well, significant research efforts to work more closely within the confines of existing curricula by focusing on particular mathematical topics. For example, Chazan (1999) concludes that technological tools written from a pedagogical perspective and developed around central mathematical objects and processes, are especially effective in supporting teachers' understandings of mathematics. Similarly, Dreyfus (1993) and Yerushalmy (1999) offer design principles for the creation of tools that are intended explicitly for learners to explore a part of the mathematical curriculum. With respect to the elementary curriculum, Olive (2000) has designed a computer tool to provide children with a medium for constructing fractions. Similarly, Carraher & Schliemann, 1998 investigated children's developing ideas on interacting with their software and pointed to the importance of the pedagogical intervention. A characteristic of these and similar efforts, is the constraining of the mathematical domain for the purpose of learning, and the careful tuning of the functionalities of the system in ways that would not be necessary (or desirable) for more open, programmable systems.

Changing goals in research with dynamic geometry systems

During the last decade, dynamic geometry systems (DGS) have become increasingly common classroom tools to support the teaching and learning of plane geometry, providing a setting in which students can construct and experiment with geometrical objects and relationships. Key to every DGS is an interface that affords direct manipulation of geometrical figures, particularly by *dragging* parts of them with the mouse. Despite differences in detail in the DGS systems available[8], all set out to model Euclidean geometry, and to support constructions by user-defined macros. At a first level, therefore, it appears as though the visual artefact produced via direct manipulation resembles in essence the traditional representations of paper-and-pencil geometry and ruler and compass constructions (see Laborde and Laborde, 1995). In this sense, DGS are designed to be transparent, allowing the user to form the impression that they are actually interacting with the Euclidean figure.

However, in contrast to the paper-and-pencil representation, the visual output of DGS does not represent an instance of a geometry figure but a class of drawings; it can be dragged around the screen with its constructed properties or underlying geometric relationships preserved. Thus the system provides a kind of feedback that is not readily evident in paper-and-pencil constructions[9], that distinguishes between a result, a *drawing*, created without concern for the underlying geometrical relationships, and one, a *figure*, that has been constructed through the use of geometrical primitives and relationships (Laborde and Laborde, 1995).

In its initial phase, research with DGS was largely focussed on its potential as a conjecturing tool and as a way to investigate students' processes of construction in geometrical contexts (Goldenberg & Cuoco, 1998; Laborde & Capponi, 1994; Arcavi & Haddas, 2000). Over time, however, studies of student interactions with DGS began to raise issues of constraints and student difficulties of interpretation and construction. We also note, in relation to the drag mode, a gradual recognition of its multiple functionality, and the problematic nature of the tool's visibility — the other side of transparency.

Dragging is a crucial instrument of mediation between figure and drawing in DGS. Balacheff and Kaput (1996) point out that dynamic geometry environments make the distinction between drawings and figures "a visible part of the geometric activity of the learner" (p. 476). But this is not necessarily the case without appropriate teacher intervention. Healy, Hölzl, Hoyles & Noss (1994) demonstrate the value of designing activities where the distinction between drawings and figures are made evident, by focussing students' attention on the difference between

[8] *Cabri Geometry*: www-cabri.imag.fr/index-e.html; *The Geometer's Sketchpad*: www.keypress.com/catalog/products/software/Prod_GSP.html; *Geometry Inventor*: www.riverdeep.net/products/math/tm_geo_invent.jhtml; *Cinderella:* www.cinderella.de/en/index.html.

[9] Traditionally, of course, teachers insisted that construction lines be left visible: but even these are not necessarily a uniquely viable proof of correct construction.

drawings that can be 'messed up' by dragging (since the relationships between lines and points are only spatial coincidences) and figures whose geometrical properties are retained under dragging. In fact, when this distinction is left to chance, students are likely to see the goal of the task as producing a printout (Foletta, 1994) or derive meanings connected with drawing instead of constructing (Pratt and Ainley, 1996).

More recently, Hölzl (2001) describes how for many students, the drag mode is viewed as a graphics tool to modify the appearance of a drawing, or to check a construction, although he notes that with more experience students also learn to use the drag mode (even without explicit help) to check their own conjectures. However Hölzl sounds a note of caution by reporting: "students primarily used dragging to explore the movement properties of a drawing rather than the relational properties of a figure" (p. 84). Findings are though (once again) necessarily sensitive to the choice of task. For example, Arcavi and Hadas (2000) exploit the drag mode to design a task in which students made sense of functional relationships and graphs without the necessity of an algebraic representation and conclude that the "initial absence of the algebraic representation, does not seem to impede genuine and deep mathematical reasoning" (p. 41).

Thus many researchers suggest that students switch between figures and concepts, between empirical and theoretical considerations while dragging. Arzarello *et al*. (1998) identify a further function of dragging and report how students who produce 'good' conjectures tend to use dragging in a particular way, which they call *lieu muet* or dummy locus. A dummy locus enables the learner to develop a sense of the 'trace' of a point moving according to a given relationship with one or more other points. Arzarello *et al.* show how students who formulate a conjecture on the basis of a dummy locus not only make constructions to validate a hypothesis, but also go on to use dragging for a new function i.e. to test the hypothesis.

Researchers with DGS often draw attention to the important differences in the ontology of the objects and relationships in dynamic geometry and in traditional geometrical knowledge, due to what Balacheff (1993) terms the *computational transposition*. Examples of this transposition are the construction of geometrical objects that do not exist in theory (Strässer, 2001); and the distinction, that is not geometrical, between the points that form the vertices of a constructed triangle in DGS and the behaviour of 'points on object' (Hölzl 1996; Goldenberg, 1995; Goldenberg and Cuoco, 1998). As Goldenberg (1995) puts it "dynamic geometry should not be treated as if it is merely a new interface to Euclidean construction. Line segments that stretch and points that move relative to each other are not trivially the same objects that one treats in the familiar synthetic geometry, and this suggest new styles of reasoning" (*ibid.,* p. 220). Dynamic geometry also generates new heuristics: for example, Goldenberg (2001), among others, draws attention to the strategy of 'relaxing' a constraint, so that a learner can construct not a single solution but a set of solutions to a more open problem, in which the solution to the original problem appears as a particular case.

In reviewing research into the use of DGS it is notable how attention is turning away from the investigation of the process of construction and conjecturing with DGS, and towards consideration of how the new tools mediate the nature of explanation, verification and even proof. It is commonplace to note that when students interact with interactive digital technologies, some (though not all of course) spontaneously articulate justifications of their actions along with explanations of why their actions produce the expected feedback (or not). It is therefore reasonable to surmise that interactions with DGS might provide an excellent opportunity for students to consider the "why..?" in addition to the "what if..?" and the "what if not...?", although it is not obvious that the facility to drag and conjecture will necessarily encourage an engagement with proof (in fact, many opponents of technology in mathematical learning have argued quite the reverse, especially if DGS is simply used to produce data).

Fortunately, research is stepping in to this debate (see, for example, deVilliers, 1997; Hoyles, 1998; Chazan and Yerushalmy, 1998; Hoyles, 1998). These studies are attempting, by careful design of tasks and activities to provide a basis for proving, together with a rationale for its necessity. Hadas, Hershkowitz & Schwartz (2001) set out to strengthen students' recognition of the need for deductive proof through a series of novel activities aimed at generating surprise and uncertainty leading to contradictions between conjectures and findings. Jones (2001), by studying 12-year-old students' interpretations of geometrical objects and relationships while using DGS, concludes that the DGS experience can mediate between 'everyday' and mathematical explanations that transcend the tool itself.

Thus the question of mediation and tasks are once again reported as critical. So too is the role of the teacher. Healy and Hoyles (2001) demonstrate how interactions with sets of DG-based construction tasks can assist 14-15 year-old students to connect their informal explanations of geometrical phenomena with logical, deductive argument when these tasks are undertaken with teacher support and alongside a teacher-introduction to writing proofs. Similarly, Mariotti (2001), after analysing extracts of classroom discussion that form part of a teaching experiment to introduce students to the theoretical world of geometry, draws attention to the role of the teacher in emphasizing relationships between geometrical theory and the tools and figures of the DGS, and in maintaining a delicate balance between constructive activity at the computer and reflections upon this activity. The teacher is the explicit focus of attention in the context of DGS in the research of Laborde (2001), who presents an analysis of teaching sequences involving DGS developed by teachers over a period of three years. She demonstrates how DGS moves with familiarity from being a visual amplifier or provider of data towards becoming an essential constituent of the meaning of tasks. In this latter stage, the technology begins to shape the conceptions of the mathematical objects that the students construct, a finding, Laborde argues, that explains why the integration of computer technology in mathematics classrooms is a long and difficult process.

Calculational instruments

There are several candidates that could serve as exemplars for considering the role of calculational instruments in mathematics education: among these are spreadsheets, graphing calculators, databases and computer algebra systems. All of these software continue to be the subject of research. There is, for example, a considerable body of work involving spreadsheets for studying algebraic relationships (Rojano, 1996; Sutherland & Rojano, 1993; Dettori et al 1998, 2001) or for modelling (for example, Lingefjärd and Kilpatrick, 1998). Similarly, databases have provided a backdrop against which research has been undertaken into students' learning of classification and querying (see, for example, Hancock, Kaput and Goldsmith, 1992; Falbel and Hancock, 1993; Hoyles, Healy and Pozzi, 1994).

In addition to these widely-used standard applications, there are studies of the learning outcomes of the use of pedagogically-oriented graphing software (see Balacheff & Kaput, 1996, for an overview; Ainley, Nardi and Pratt, 2000; Ainley, 2000; Goldenberg, 1991; Gomes Ferreira, 1997). There are fascinating issues involved, even with what appears at first sight to be an unproblematic extension of traditional representational systems. For example, Tall (1996) discusses how a simple graph-plotter translation generates difficulties in interpretation (*ibid.,* p. 301-2) a difficulty which is, more generally, symptomatic of subtleties that underpin the mathematical knowledge domain (Smith & Confrey, 1994). Significantly, these subtleties surface precisely because of the technology, and these findings signify a recognition of the complexity of the ways in which technology enters and reshapes the cultures of mathematical learning.

This complexity suggests that rather than surveying all the different calculational instruments we mentioned above, we could profitably focus on one, and survey the literature in a little more depth as well as trace its development. We have chosen Computer Algebra Systems (CAS) as our focus since (as with DGS), there exists a thriving community of researchers and practitioners who simultaneously gain some intellectual cohesion by focussing attention on a particular piece of software, yet derive insight from borrowing (and contributing to) research in the broader field[10].

Tracing the research trajectory of Computer Algebra Systems

Computer algebra systems (CAS)[11] enable students to define, manipulate, transform, compare and visualise algebraic expressions in any of their traditional representational forms (Balacheff and Kaput, 1996). In the 1980's, CAS was quickly embraced by the mathematical community as affording an escape for students from having to learn manipulation skills, so that they could focus their learning on more conceptual issues. At university level, for example, research by Heid (1988) and Palmiter (1991) served as a springboard for later work, by showing that students who had used CAS and had experienced rather less emphasis on algorithmic procedures than in traditional skills-oriented courses, achieved greater understanding and higher test scores in conceptual knowledge with no parallel reduction in performance in associated computational skills.

These kinds of studies, which sought to describe the effectiveness of CAS, proved useful in laying the groundwork for thinking about what CAS might bring to mathematical learning. Although the overall trends have been generally positive,

[10] This was, incidentally, true of the Logo programming community (see, for example, Hoyles & Noss, 1992).

[11] CAS include *Maple* (www.maplesoft.com/); *Mathematica* (www.wolfram.com/products/mathematica/); *MatLab* (www.mathworks.com/products/matlab/); *Derive* (www.chartwellyorke.com/derive.html).

results are beginning to show that using CAS does not necessarily lead to improvements in performance, and to raise questions concerning students' interpretations of their use of the systems (some of these are referenced in Tall, 1996), as well as identifying obstacles and misconceptions of mathematical ideas as a result of CAS use (Hillel, Lee, Laborde and Linchevski, 1992; Hillel, 1993; Drijvers, 2000; Monaghan *et al*., 1994).

In a useful summary of research with CAS undertaken up to 1995, Mayes (1997) concludes that using a CAS as a cultural reorganizer (in the sense of Pea, 1987) leads to beneficial results in general, but if the CAS is used primarily to increase efficiency and speed in implementing standard approaches to solving problems, the outcomes are less positive.

An important question faced by CAS research is to gauge the extent to which the use of CAS opens a window onto conceptual problems students already have (for example, in distinguishing variables from parameters) or whether existing problems are actually exacerbated by tool use. In order to decide between these alternatives (if indeed they can legitimately be seen as such) it became clear that there was a pressing need for more detailed qualitative studies of student/tool interaction. At the very least, the inevitable 'overhead' of using these tools was invariably highlighted (an issue that we will return to later) along with the necessity to take seriously the goals and activity structures of tool use and practice.

After analysing case studies of students' interactions with CAS, Pozzi (1994) suggested that CAS could be used to support students in making sense of their algebraic generalizations at a semantic level provided the software is used to explore and verify generalizations and not simply as a symbolic calculator. He also notes that using a CAS may necessitate a close conceptual understanding of syntactic manipulations, since when students do not fully comprehend a CAS output – that is, when the tool is not transparent – they frequently develop informal and possibly erroneous models of what the computer is doing to explain the output. On the question of defining roles and activity structures, Dreyfus and Hillel (1998) point to a further complexity, by distinguishing among different roles for the CAS: as a graphic calculator, as an investigative resource that prompted more precision in language, and as a "silent moderator".

A further aspect of the broadening of the CAS research paradigm has been to turn the spotlight on to the teacher. Kendal & Stacey (1999) present results suggesting that teachers in CAS studies influence students' responses, not only through their intervention, but also through their attitudes to mathematics and how these are played out in terms of what they stress and what they ignore in the software.

It must also be noted that changes in hardware (that is, for example, the availability of 'computer-like' calculators such as DERIVE on the TI-92) has stimulated CAS research to revisit some of its earlier research questions. These have widened access to CAS systems and made it worthwhile to revisit the interrelationship of a symbolic capability with students' conceptualisations (Lagrange, 1999). Lagrange adopts a cognitivist analysis that takes as a premise that the calculator is not transparent – that the user does *not* trivially distinguish the interface from the internal logic of the system. This implies that the computational transposition that has occurred is completely hidden. Drawing on the notion of 'technique' used by Chevallard (1992), Lagrange identifies a set of *schemes* related to CAS use, such as linking algebraic and analytic interpretation, transformation and expression of a function, and argues that new techniques must be identified, taught and discussed to help to develop these schemes – they are not simply obvious from computer use. Examples of techniques are the ways an algebraic expression can be transformed with a CAS into an equivalent one, or zooming to coordinate algebraic, graphical and numerical representations. What he calls *instrumental genesis* – the development of schemes which evolve into techniques – has its own constraints deriving from the specificity of the calculator and the mathematical topic (see Verillon & Rabardel, 1995, for a seminal analysis of the idea of *instrumentation*).

The process of instrumentation presents a further theoretical viewpoint on the question of transparency of computational systems, the extent to which the learner is aware of the system, and is able to look *through* it as well as look *at* it (Artigue, 2001). In particular, it suggests that attention be given to the ways in which the mathematical needs of techniques change as computational technology enters the institutional setting (Balacheff, 1994), and to issues of designing for instrumental genesis (Guin and Trouche, 1999).

An agenda for research

In this final section, we aim to draw together the threads we have left hanging in previous sections, and make good, at least partially, our promise to address the ways in which research with and on digital technologies in the mathematical domain may assist in clarifying some outstanding issues in the study of mathematical learning and teaching in general.

We began this chapter by considering the representational infrastructure for learning mathematics as an organising framework for classifying different uses of technologies. We distinguished two categories of software, programmable microworlds and expressive tools, that have been widely researched. We chose this classification in order to highlight the ways in which digital technology is shaped and shaped by its incorporation into mathematical learning and teaching environments. From the review it emerges that tools do shape learning: but they do so often in unpredicted ways. Furthermore, apart from its unsurprising dependence on tasks and activity structures, research with programmable microworlds suggests that learning is highly sensitive to small changes in technologies, and that the design of tools and learning have tended to co-evolve. We have also identified a common research trajectory for the study of digital technologies in mathematical learning: that is, one that starts by documenting potentials and obstacles in software use and then gradually shifts to discussions of tool mediation, tasks and activities, and the role of the teacher. Perhaps this is the process that has to be followed in order to develop an intellectual discipline for the study of teaching and learning in these new settings (see Ruthven, 19 91 for an early discussion of the potential role of technology in the rationalisation of teaching)?

We now conclude the chapter by elaborating two themes that have emerged from the review: these are (1) the openness of tools, and (2) the reconceptualisation of mathematical learning.

The openness of tools

The research reviewed regarding programming and microworlds points to the potential for student learning in situations where students are able to reconstruct and mould software tools to the task as a means to appropriate and coordinate new representational infrastructures for mathematics. There is, as we have seen, a blurring of the distinction between the two genres of tool-use and programming, particularly in situations where tools are partially open for reconstruction or, reciprocally, in cases where programming systems are presented not only as systems with which to model, but as cases of models themselves.

The tool-use genre includes, for example, DGS, and the issue of their programmability is ripe for further research. Currently, programmability is disguised as 'macros', although this allows only limited openness to the system (see Healy and Hoyles, 2001). The programming genre involves systems like the Boxer, StarLogo and Playground work reviewed earlier, in which whole usable programs are presented to learners for reconstruction where appropriate.

The dichotomy dividing these two genres is not so clear-cut as it was, and from a technical point of view research is proceeding along three axes. First, in the direction of component architectures, in which more-or-less opaque tools can be combined and reused to generate higher-level functionalities (Kynigos *et al*., 1997); second, "open toolsets" (diSessa, 2000) a genre of software that involves a greater number of transparent smaller units than conventional educational 'applications'; and third, "programmable applications" which would allow learners to tune (large) applications for themselves (Eisenberg, 1995), breaking down altogether the idea of a monolithic application that is closed to the "user".

From a cultural rather than technical perspective, the issue of programmability or openness raises a number of interesting questions touched upon in this review. In the first place, it raises the issue of the transparency of computational models, conceived of as a relationship between learner and artefact, and how far tools are used to explore a mathematical domain without awareness of their workings (see Hancock, 1995,). When tools are used in mathematics classrooms they are often (naïvely) assumed to be transparent, in that activities are shaped only by structures of classroom discourse. Yet as the review illustrates, tools are sometimes used in unpredicted ways, and often as a result of such use or in the face of unexpected feedback, become visible as the focus of attention. At this point, the learner becomes aware of the constraints upon her, the mediation of her mathematical ideas, or the imperatives set by the tool itself. In systems in which programmability plays a non-trivial role, this awareness is necessarily explicit. The review indicates the complexities revealed at this point are not a matter of "overhead" to be bypassed or ignored, but a matter for study. The question remains as to what extent, therefore, the notion of instrumentation should be expanded to take account of this reflexive aspect of tool use? How might the activity of construction mediate the ways in which learners come to develop techniques and how might this constructive dimension influence the relationship between technical and conceptual fluency?

A second issue points towards mathematical learning research in general. In tool design, a question that looms large is the extent to which it can be said to "embody" a piece of mathematics, or at least to facilitate its exploration[12], whether or not we may speak of a tool's intrinsic structures or relationships. Tools matter: they stand between the user and the phenomenon to be modelled, and shape activity structures. Recognising that tool-mediation is subject to its user's participation in a practice does *not* mean that we can ignore what the tool was designed for: the structural facets of the tool are at least as important as what is done with them.

The tools of an environment encapsulate mathematical relationships in some sense: but these relationships lie dormant until they are mobilised, and it is in their mobilisation that meanings are created. The individual steps onto an already-built structure: and what is seen – and taken – from that structure is mediated by the activity structures, intentions and pedagogical goals of the setting. This phenomenon is both more explicit and more visible when learners are (re-)constructing tools for themselves (for further discussion of this point, in relation to the idea of "webbing", see Noss & Hoyles, 1996).

From a sociocultural point of view therefore, we believe that taking account of the design and intention of technological artefacts brings epistemology to centre stage. Tools do not, by themselves, make explicit how they work: yet this is a clear priority for the design of educational tools, and educational artefacts in general, and it is this imperative that points to the importance of tools which are open, malleable and programmable. How should we understand epistemological structures and how are they mediated by learning communities? Reciprocally, how are learning communities shaped by the tools, artefacts and technologies embedded within them, and to what extent are there epistemological imperatives delineated by these tools?

Reconceptualising mathematical learning

Throughout the review, research has pointed to the ways that tools shape evolving conceptions of learners through, for example, the representational infrastructures that frame them, the connections between different knowledge elements of the system that they afford, the balance between technique and concept, and the extent to which feedback encourages exploration and engagement with specific mathematical knowledge.

If computationally-based research has taught the broader field anything, it is that Logo-maths, or DG-maths is not the same as maths *per se,* and that – by implication – neither is the knowledge that learners develop. The wider community, therefore, needs to seek ways to describe the knowledge structures which are characteristic of interaction in particular learning environments, how they develop in relation to particular tools and discourses, and how relationships form between different (and differently-formed) knowledge structures. Our personal solution has involved elaborating, over the last decade or so, the idea of *situated abstraction,* a notion that seeks to describe how a conceptualization of mathematical knowledge can be both tuned to its constructive genesis within a practice, yet simultaneously can

[12] See Wenger, 1987, on the question of "epistemic fidelity", and Meira, 1998 for a discussion of the sense of instructional devices in mathematical activity.

retain mathematical invariants abstracted within that practice (see, for example, Noss, Hoyles & Pozzi, in press). Other workers are developing related lines of attack on the fundamental dilemma of situated cognition: how mathematical knowledge gains generality within a situated perspective that sees every act of cognition as bounded within the setting of its genesis.

It seems that much, though by no means all, of the work in this area (particularly the strand of work studying abstraction) has emerged from a concern with technological environments. This is not an accident: we have seen how the constraints and boundedness of static media are often invisible due to their ubiquity. Here, then, is a challenge for the broader community, to generalise this work beyond technological settings, and to reconceptualise individual mathematical knowledge (necessarily a cognitive preoccupation) in terms that take adequate account of the tools and discourses with which it was constructed.

More generally, work with computational tools and the development of learning communities that have been established around their use, has pointed the way to a new and more robust paradigm for thinking about tool use that has moved beyond simple student/tool interaction or a merely cognitivist paradigm studying the individual's acquisition of knowledge, towards a consideration of the complex process of instrumental genesis, the role of the teacher, and the connection of tool use and traditional techniques. This points the way to reconciling cognitive and sociocultural approaches – a task which is, as Cobb and Bowers (1999) argue, a pressing one for research in the field. Research with digital technologies shows promise in assisting this endeavour.

Acknowledgement

We acknowledge the valuable help and advice of John Monaghan in preparing the section on CAS. Our thanks to Lulu Healy for her comments on an earlier draft, and to Phillip Kent for his intellectual contribution as well as his assistance in preparing the references.

References

Ainley, J. (2000). Transparency in graphs and graphing tasks: An iterative design process. *Journal of Mathematical Behavior* 19, 365-384.

Ainley, J., Nardi, E. and Pratt, D. (2000). The construction of meanings for trend in active graphing. *International Journal of Computers for Mathematical Learning* 5, 2, 85 – 114.

Arcavi A. & Hadas N. (2000). Computer mediated learning: and example of an approach. *International Journal of Computers for Mathematical Learning*, 5 (1) 25-45.

Artigue, M. (2001). Learning Mathematics in a CAS environment: The genesis of a reflection about instrumentation and the dialectics between technical and conceptual work. Presentation to the Computer Algebra in Mathematics Education Symposium, Utrecht University.

Arzarello, F., Domingo, P., Gallino, G., Micheletti, C. and Robutti, O. (1998). Dragging in Cabri and modalities of transition from conjectures to proofs in geometry. *Proceeding of the Twenty Second Annual Conference of the International Group for the Psychology of Mathematics Education* (Stellenbosch), vol. 2, 32 – 39.

Balacheff, N. & Kaput, J. (1996), Computer-Based Learning Environments in Mathematics. In A. Bishop, K. Clements, C. Keitel, J. Kilpatrick, & C. Laborde (eds.), *International Handbook of Mathematics Education*, pp. 469 – 504. Dordrecht: Kluwer Academic Publishers.

Balacheff, N. (1993). Artificial intelligence and real teaching. In C. Keitel & K. Ruthven (Eds.), *Learning from computers: Mathematics education and technology* pp. 131-158. Berlin: Springer-Verlag.

Balacheff, N. (1994), La transposition informatique. Note sur un nouveau problème pour la didactique, in, M. Artigue & al. (Eds), *Vingt ans de Didactique des Mathématiques en France,* pp. 364-370. Grenoble, La Pensée Sauvage.

Carraher, D. and Schliemann, A.(1998) *Giving Prominence to Pedagogical Issues* Paper to NCTM Technology meeting

Chazan, D. (1999). On teachers' mathematical knowledge and student exploration: A personal story about teaching a technologically supported approach to school algebra. *International Journal of Computers for Mathematical Learning*, 4(2-3), 121 – 149.

Chazan, D. and Yerushalmy, M. (1998). Charting a course for Secondary Geometry. In Lehrer, R. and Chazan, D. (eds) *Designing Learning Environments for Developing Understanding of Geometry and Space*. Hillsdale, NJ: Lawrence Erlbaum. Pages 67 – 90.

Chevallard, Y. (1992). Concepts fondamentaux de la didactique : perspectives apportées par une approche anthropologique. *Recherches en Didactique des Mathématiques*, 12, 1, 77-111.

Clements, D. H. (1999). The future of educational computing research: The case of computer programming. *Information Technology in Childhood Education Annual*. Norfolk, VA: Association for the Advancement of Computing in Education. Pages 147 – 179.

Clements, D. H. and Sarama, J. (1995). Design of a Logo environment for elementary geometry. *Journal of Mathematical Behaviour*, 14, 381 – 398.

Clements, D. H., Battista, M. T. and Sarama, J. (2001). *Logo and Geometry* (Journal for Research in Mathematics Education, Monograph Number 10). Reston, VA: National Council for Teachers of Mathematics.

Clements, D. H., Battista, M. T., Sarama, J., & Swaminathan, S. (1996). Development of turn and turn measurement concepts in a computer-based instructional unit. *Educational Studies in Mathematics,* 30, 313 – 337.

Clements, D. H., Battista, M. T., Sarama, J., Swaminathan, S., & McMillen, S. (1997). Students' development of length measurement concepts in a Logo-based unit on geometric paths. *Journal for Research in Mathematics Education,* 28(1), 70 – 95.

Cobb, P., and Bowers, J. (1999). Cognitive and situated learning perspectives in theory and practice, *Educational Researcher*, 28 (2), pp. 4-15.

Confrey, Jere (1993) The role of technology in reconceptualizing functions and algebra. In B. Pence (ed.) *Proceedings of the XVIIth Annual Meeting of the North American Chapter of the International Group for the Psychology of Mathematics Education* . Asilomar, CA.

Dettori, G. Greco, S. and Lemut, E. (1998). Complementing different software environments to mediate the introduction of variable, parameter and unknown. In Tinsley, J. and Johnson, D. (eds) *Information and Communications Technologies in School Mathematics*. London: Chapman and Hall.

Dettori, G., Garuti, R. and Lemut, E. (2001). From arithmetic to algebraic thinking by using a spreadsheet. In Sutherland, R. Rojano, T., Bell, A. and Lins, B. (eds.) *Perspectives on School Algebra*. Dordrecht: Kluwer Academic. Pages ???

diSessa, A. A. (2000). *Changing Minds: Computers, learning and literacy*. MIT Press.

diSessa, A. A., Hoyles, C. and Noss, R. (Eds.) (1995). *Computers and Exploratory Learning*. Berlin: Springer-Verlag. (NATO ASI Series F, Volume 146.)

de Villiers, M. (1997). The role of proof in investigative, computer-based geometry: Some personal reflection. In King, J. & Schattschneider, D. (Eds.) *Geometry Turned On: Dyamic software in Learning, Teaching, and Research* (*MAA Notes Series 41)* (pp.15-28). Washington. D.C.: Mathematical Association of America.

Dreyfus, T. (1993), Didactic Design of Computer-based Learning Environments, in C. Keitel and K. Ruthven (eds), *Learning from Computers: Mathematics Education and Technology,* (NATO ASI Series F, vol 121) pp. 101-130, Springer-Verlag.

Dreyfus, T. & Hillel, J. (1998). Reconstruction of Meanings for Function Approximation *International Journal of Computers for Mathematical Learning* **3** (2):93-112, 1998.

Drijvers, P. (2000). Students encountering obstacles using CAS. *International Journal of Computers for Mathematical Learning*, 5, 3, 189 – 209.

Edwards, L. D. (1998). Embodying mathematics and science: Microworlds as representations. *Journal of Mathematical Behaviour* 17, 1, 53 – 78.

Eisenberg, M. (1995). Creating software applications for children: Some thoughts about design. In diSessa, A. A., Hoyles, C. and Noss, R. (Eds.), *Computers and Exploratory Learning*. Berlin: Springer-Verlag. Pages 175 – 196.

Falbel, A. and Hancock, C. (1993). Coordinating sets and properties when representing data: The group placement problem. *Proceedings of the 17th International Conference on the Psychology of Mathematics Education* vol. 2, 17 – 24.

Feurzeig, W. and Papert, S. (1969). Programming Languages as a Conceptual Framework for Teaching Mathematics. Report No.1889. Cambridge, MA: Bolt, Beranek and Newman.

Foletta, G. (1994), Technology and guided inquiry: Understanding of students' thinking while using a cognitive computer tool, the Geometer's Sketchpad, in a geometry class. *Dissertation International,* (The University of Iowa Microfilms)

Goldenberg, E. P. (1995). Ruminations about dynamic imagery (and a Strong Plea for Research). In R. Sutherland and J. Mason (Eds.), *Exploiting Mental Imagery with Computers in Mathematics Education*, pp. 202-224. Berlin: Springer.

Goldenberg, P. (1991). The difference between graphing software and educational graphing software. In W. Zimmerman & S. Cunningham (Eds.), *Visualisation in Teaching and Learning Mathematics* (MAA Notes no. 19). Washington DC: Mathematical Association of America. Pages 77 – 86.

Goldenberg, P. (2001). Getting Euler's line to relax. *International Journal of Computers for Mathematical Learning*, 6, 2 .

Goldenberg, P. and Cuoco, A. (1998), What is Dynamic Geometry? In Lehrer, R. and Chazan, D. (eds) *Designing Learning Environments for Developing Understanding of Geometry and Space*. Hillsdale, NJ: Lawrence Erlbaum. Pages 351 – 367.

Gomes Ferreira, V. (1997) Exploring Mathematical Functions through dynamic microworlds. *Unpublished Doctoral Thesis*, Institute of Education, University of London.

Guin, D. and Trouche, L. (1999). The Complex Process of Converting Tools into Mathematical Instruments: The Case of Calculators, *International Journal of Computers for Mathematical Learning,* 3 (3), 195-227.

Hadas, N., Hershkowitz, R. & Schwartz, B. (2001) The role of contradiction and uncertainty in promoting the need to prove in dynamic geometry environments. *Educational Studies in Mathematics.* Special issue on Proof in Dynamic Geometry Environments. Vol. 44, 1-2. 127-150.

Hancock, C. (1995). The Medium and the Curriculum: Reflections on Transparent Tools and Tacit Mathematics, in A. diSessa, C. Hoyles, R. Noss, (eds),*Computers and Exploratory Learning*. Berlin: Springer. Pages 221 – 240.

Hancock, C., Kaput J., and Goldsmith (1992) Authentic inquiry with data: Critical barriers to classroom implementation. *Educational Psychologist*, 27(3), 337-364.

Harel, I. and Papert, S. (Eds.) (1991). *Constructionism*. Norwood, NJ: Ablex Publishing.

Healy, L. and Hoyles, C. (2001). Software tools for geometrical problem solving: Potentials and pitfalls. *International Journal of Computers for Mathematical Learning*, 6, 3.

Healy, L. Hoelzl, R., Hoyles, C. & Noss, R. (1994), Messing Up: Reflections on Introducing Cabri Géomètre. *Micromath*, 10, 1, 14-16.

Healy, L., Pozzi, S. and Hoyles, C. (1995). Making sense of groups, computers and mathematics. *Cognition and Instruction* 13, 4, 505 – 523.

Heid, K. (1997) The Technological Revolution and the reform of School Mathematics. *American Journal of Education* 106(1), pp. 5-61.

Heid, K. & Blume G. (Eds.) (in preparation). Research on Technology in the Learning and Teaching of Mathematics: Syntheses and Perspectives.

Hillel, J. (1993). Computer algebra systems as cognitive technologies: Implications for the practice of mathematics education. In Keitel & K. Ruthven (eds.), *Learning from computers: Mathematics education and technology*, Berlin: Springer-Verlag.

Hillel, J. ,Lee, L., Laborde, C. and Linchevski, L. (1992). Basic functions through the lens of computer algebra systems. *Journal of Mathematical Behavior,* 11, 119-158.

Holzl, R. (1996). How does 'dragging' affect the learning of geometry? *International Journal of Computers for Mathematical Learning* , 1, 169-187.

Holzl, R. (2001). Using dynamic geometry software to add contrast to geometric situations — a case study. *International Journal of Computers for Mathematical Learning* 6(1), 63 – 86.

Hoyles C, Healy, L. and Pozzi, S. (1994). Homing in on data handling: A case study. *Computers in New Zealand Schools,* 6 (3), 5-13.

Hoyles, C. & Noss, R. (Eds.) (1992), *Learning Mathematics and Logo*, Cambridge Mass., MIT Press.

Hoyles, C. (1993). Microworlds/schoolworlds: The transformation of an innovation. In C. Keitel & K. Ruthven (eds.), *Learning from Computers: Mathematics Education and Technology* (NATO ASI Series F, vol. 121). Berlin: Springer-Verlag. Pages 1 – 17.

Hoyles, C. and Healy, L. (1997). Unfolding meanings for reflective symmetry. *International Journal of Computers for Mathematical Learning*, 2, 27 – 59.

Hoyles, C. (1998).A culture of proving in school mathematics. In Tinsley, D. & Johnson, D. C.(eds) *Information and Communications Technologies in School Mathematics* London: Chapman Hall . Pages 169 – 182.

Hoyles, C. and Sutherland R. (1989) *Logo Mathematics in the Classroom.* London: Routledge.

Hoyles, C., Morgan, C. and Woodhouse, G. (Eds.) (1999). *Rethinking the Mathematics Curriculum*. London: Falmer Press.

Hoyles, C., Noss, R., Adamson, R. and Lowe, S. (2001). Programming rules: What do children understand? *Proceedings of the 25th Conference of the Psychology of Mathematics Education* (Utrecht), vol. 3, 169 – 176.

Hoyles, C., Noss, R., and Adamson, R. (in press). Rethinking the Microworld Idea. *Journal of Educational Computing Research*, Special issue on Microworlds in Mathematics Education.

Hunter, M. Monaghan, J. D. & Roper, T. (1993), The effect of computer algebra use on students' algebraic thinking. In R. Sutherland (ed) *Working Papers for ESRC Algebra Seminar,* London: Institute of Education.

Jones, D. (1998) Young children proving in computational settings. Unpublished Doctoral Dissertation. Institute of Education, University of London.

Jones, K. (1998). The Mediation of Learning within a Dynamic Geometry Environment. In the *Proceedings of the Twenty-second International Conference for the Psychology of Mathematics Education* Vol. 3. pp. 96-103, Stellenbosch, South Africa.

Jones, K. (2001). Providing a foundation for deductive reasoning: students' interpretations when using dynamic geometry software and their evolving mathematical explanations. *Educational Studies in Mathematics.* Special issue on Proof in Dynamic Geometry Environments. Vol. 44, 1-2. 55-85.

Kahn, K. (1999). Helping children learn hard things: Computer programming with familiar objects and activities. In A. Druin (Ed.), *The design of children's technology.* San Francisco: Morgan Kaufman. Pages 223 – 241.

Kaput, J. & Shaffer, D. (in press). On the development of human representational competence from an evolutionary point of view: From episodic to virtual culture. In K. Gravemeijer, R. Lehrer, B. van Oers, & L. Verschaffel (Eds.), *Symbolizing, modeling and tool use in mathematics education*. London: Kluwer.

Kaput, J. (1992). Technology and Mathematics Education. In: Grouws D. (Ed.) *Handbook of Research on Mathematics Teaching and Learning*. New York: Simon and Schuster Macmillan. pp. 515-556.

Kaput, J. (1999). Algebra and technology: New semiotic continuities and referential connectivity. In F. Hitt, T. Rojano, & M. Santos (Eds.), *Proceedings of the PME-NA XXI Annual Meeting*, Cuernavaca, Mexico.

Kaput, J. and Roschelle, J. (1999). The mathematics of change and variation from a millennial perspective: New content, new context. In C. Hoyles, C. Morgan, & G. Woodhouse (Eds.), *Rethinking the Mathematics Curriculum*,, (pp. 155-170). London: Falmer Press.

Kaput, J., Noss R., and Hoyles C. (in press). Developing New Notations for a Learnable Mathematics in the Computational Era. In English, L. (Ed) *Handbook of International Research in Mathematics Education*. Mahwah, NJ: Lawrence Erlbaum.

Kendal, M. & Stacey, K. (1999). Varieties of teacher privileging for teaching calculus with computer algebra systems, *The International Journal for Computer Algebra in Mathematics Education*, 6(4), 233-247.

Klinger, W. (1994) Using DERIVE in the Third and Fourth Form of Grammar Schools in Austria. In *DERIVE in Education: Opportunities and Strategies* (eds. Heugl, H. & Kutzler, B.), Chartwell-Bratt Ltd, Sweden.

Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, *8,* 30-43.

Kynigos, C. (1991). Can children use the turtle metaphor to extend their learning to include non-intrinsic geometry? *Proceedings of the 15th International Conference for the Psychology of Mathematics Education* (Assisi), vol. 2, 269 – 276.

Kynigos, C., Koutlis, S. and Hadzilacos, T. (1997). Mathematics with component oriented exploratory software. *International Journal of Computers for Mathematical Learning* 2(3), 229-250.

Laborde, C. & Capponi, B. (1994), Cabri-géomètre constituant d'un mileu pour l'apprentissage de la notion de figure géomètrique*, Recherches en didactiques des mathematiques*, vol. 14, no. 1.2, 165-210.

Laborde, C. & Laborde, J.-M. (1995). What about a learning environment where Euclidean concepts are manipulated with a mouse? In A. A. diSessa, C.

Hoyles, & R. Noss (Eds.), *Computers and exploratory learning,* Berlin: Springer-Verlag. Pages

Laborde, C. (2001). Integration of technology in the design of geometry tasks with cabri-geometry. *International Journal of Computers for Mathematical Learning*, 6, 3.

Lagrange, J-B. (1999). Complex calculators in the classroom: theoretical and practical reflections on teaching pre-calculus. *International Journal of Computers for Mathematical Learning*, 4 (1), 51-81.

Lesh, R. and Kelly, A. (1996). A constructivist model for redesigning AI tutors in mathematics. In J. M. Laborde (Ed.), *Intelligent Learning Environments: The case of geometry* (NATO ASI Series F, vol. 117) . Berlin: Springer. Pages 133 – 156.

Lingefjärd, T., & Kilpatrick, J. (1998). Authority and responsibility when learning mathematics in a technology-enhanced environment. In D. Johnson & D. Tinsley (Eds.), *Secondary school mathematics in the world of communication technologies: Learning, teaching and the curriculum* (pp. 233-236). London: Chapman & Hall.

Mariotti, M. A. (2001), Justifying and proving in the Cabri Environment. *International Journal of Computers for Mathematical Learning* 6, 3.

Mayes, R. (1997). Current state of research into CAS in mathematics education. In Berry, J., Monaghan, J., Kronfellner, M. and Kutzler, B.: *The state of computer algebra in mathematics education*. Bromley: Chartwell-Bratt. Pages 171 - 189.

Meira, L. (1998), Making sense of instructional devices: the emergence of transparency in mathematical activity. *Journal for Research in Mathematics Education,* 29 (2), pp. 121-142.

Monaghan, J., Sun, S. & Tall, D. (1994). Construction of the Limit Concept with a Computer Algebra System. *Proceedings of the 18th International Conference for the Psychology of Mathematics Education* (Lisbon), vol. 3, 279-286.

Nemirovsky, R., Tierney, C., and Wright, T. (1998). Body Motion and Graphing. *Cognition and Instruction*, 16(2), 119 – 172.

Noble, T. Nemirovsky, R. ,Wright T. and Tierney C. (2001) Experiencing change; the mathematics of change in multiple environments. *Journal for Research in Mathematics Education*, 32, 1, 85 – 108.

Noss, R. & Hoyles, C. (1992) Looking back and looking forward. In: Hoyles C. & Noss R. (Eds.) *Learning Mathematics and Logo.* pp. 431-468. Cambridge: MIT Press.

Noss, R. (1998). New numeracies for a technological culture. *For the Learning of Mathematics* 18, 2, 2 – 12.

Noss, R. and Hoyles, C. (1996). *Windows on Mathematical Meanings: Learning cultures and computers*. Dordrecht: Kluwer Academic.

Noss, R., Healy, L. and Hoyles, C. (1997). The Construction of Mathematical Meanings: Connecting the Visual with the Symbolic. *Educational Studies in Mathematics*, 33, 2, 203 – 233.

Noss, R., Hoyles, C. and Pozzi, S. (in press). Abstraction in expertise: A study of nurses' conceptions of concentration. To appear in *Journal for Research in Mathematics Education*.

Olive, J. (2000) Computer Tools for Interactive Mathematical Activity in the Elementary School. *International Journal of Computers for Mathematical Learning.* 5(3). 241-262.

Olson, D. (1994) *The World on Paper. The Conceptual and Cognitive Implications of Writing and Reading.* Cambridge: Cambridge University Press.

Palmiter, J. (1991). Effects of a Computer Algebra System on Concept and Skill Acquisition in Calculus. *Journal for Research in Mathematics Education*, 22, 151-156.

Papert, S. (1980), *Mindstorms: Children, Computers, and Powerful Ideas*, New York, Basic Books.

Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning* 1, 1, 95 – 123.

Pea, R. (1987), Socializing the knowledge transfer problem, *International Journal of Educational Research,* 11 (6), pp. 639-663.

Pozzi, S. (1994), Algebraic Reasoning and CAS: Freeing Students from Syntax? In *DERIVE in Education: Opportunities and Strategies* (eds. Heugl, H. & Kutzler, B.). Bromley, UK: Chartwell-Bratt.

Pratt, D. (1998). The Co-ordination of Meanings for Randomness. For the Learning of Mathematics, 18, 3, 2 – 11.

Pratt, D. (2000). Making Sense of the Total of Two Dice. *Journal for Research in Mathematics Education*, 31(5), 602 – 625.

Resnick, M. (1994). *Turtles, Termites and Traffic Jams: Explorations in massively-parallel microworlds.* Cambridge, MA: MIT Press.

Resnick, M., Berg, R. and Eisenberg, M. (2000). Beyond Black Boxes: Bringing Transparency and Aesthetics Back to Scientific Investigation. *Journal of the Learning Sciences*, vol. 9, no. 1, pp. 7-30.

Rojano, T. (1996). Developing algebraic aspects of problem solving within a spreadsheet environment. In N. Bednarz, C. Kieran, & L. Lee (Eds.), *Approaches to algebra: Perspectives for research and teaching*. Dordrecht: Kluwer Academic Publishers. Pages 137-145.

Roschelle (1992). Learning by collaboration: Convergent conceptual change. *The Journal of the Learning Sciences* 2(3), 235-276.

Roschelle, J., Kaput, J., & Stroup, W. (2000). SimCalc: Accelerating students' engagement with the mathematics of change. In M. Jacobson & R. Kozma (Eds.), *Educational technology and mathematics and science for the 21st century* (pp. 47-75). Mahwah, NJ: Lawrence Erlbaum.

Ruthven, K. (1991). Technology and the rationalisation of teaching, in C. Keitel and K. Ruthven (eds), *Learning from Computers: Mathematics Education and Technology,* (NATO ASI Series F, vol 121) pp. 187-202, Springer-Verlag.

Sacristán, A. I. (2001). Students' shifting conceptions of the infinite through computer explorations of fractals and other visual models. *Proceedings of the 25th International Conference for the Psychology of Mathematics Education*, vol. 4, 129 – 136.

Salomon, G. (1992). Computer's First Decade: Golem, Camelot, or the Promised Land? A Paper Presented at the Annual Meeting of the American Educational Research Association. San Francisco, April, 1992.

Sendov, B. & Sendova, E. (1995). East or West-GEOMLAND is best, or Does the answer depend on the question? In A. diSessa, C. Hoyles and R. Noss (Eds), *Computers and exploratory learning*. Berlin: Springer-Verlag. Pages 59 – 78.

Sherin, B. (2001). A Comparison of Programming Languages and Algebraic Notation as Expressive Languages for Physics. *International Journal of Computers for Mathematical Learning*, 6, 1, 1 – 61.

Sfard, A. & Leron, U. (1996). Just give me a computer and I will move the earth: programming as a catalyst of a cultural revolution in the mathematics classroom. *International Journal of Computers for Mathematical Learning.* 1(2), 189-195.

Smith, E and Confrey, J. (1994), Using a dynamic software tool to teach transformations of functions. In L. Lum (ed.), *Proceedings of the Fifth Annual Conference on Technology in Collegiate Mathematics*. Reading, MA: Addison-Wesley. Pages 225-242.

Stevenson, I. (2000). Modelling Hyperbolic Space: Designing a Computational Context for Learning Non-Euclidean Geometry. *International Journal of Computers for Mathematical Learning*, 5, 2, 143 – 167.

Strässer, R. (2001). Cabri-géomètre: Does Dynamic Geometry Software (DGS) Change Geometry and its Teaching and Learning? *International Journal of Computers for Mathematical Learning.* 3.

Sutherland, R. & Rojano, T. (1993), A Spreadsheet Approach to Solving Algebra Problems. *The Journal of Mathematical Behavior*, 12, 4, 353-383.

Sutherland, R. (1989) Providing a Computer Based Framework for Algebraic Thinking. *Educational Studies in Mathematics.* 20: 317-344.

Tall, D. (1996). Functions and calculus. In A. Bishop et al, *International Handbook of Mathematics Education*. Dordrecht: Kluwer Academic. Part 1, Pages 289 – 325.

Taylor, R. (Ed.) (1980). *The Computer in School: Tutor, tool, tutee*. New York: Teachers College Press. Pages 215 – 230.

Ursini, S. (1994). Pupil's Approaches to Different Characterisations of Variable in Logo. Unpublished PhD thesis, Institute of Education, University of London.

Verillon, P. & Rabardel, P. (1995). Cognition and Artefacts: a contribution to the study of thought in relation to instrumented activity*, European Journal of Psychology of Education*, 10 (1), 77-101.

Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems*. San Francisco: Morgan Kaufman.

Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems*. San Francisco: Morgan Kaufman.

Wilensky, U. (1995). Paradox, programming and learning probability: A case study in a connected mathematics framework. *Journal of Mathematical Behavior*, 14, 2, 231 – 280.

Wilensky, U. (1997). What is Normal Anyway? Therapy for Epistemological Anxiety. *Educational Studies in Mathematics*, 33, 2, 171 – 202.

Wilensky, U. (2001). Emergent Entities and Emergent Processes: Constructing Emergence through Multi-agent programming. Paper presented at the annual meeting of the American Educational Research Association. Seattle, Wa.

Wilensky, U. and Stroup, W. M. (2000). Networked gridlock: Students enacting complex dynamic phenomena with the HubNet architecture. In B. Fishman and S. O'Connor-Divelbiss (Eds.), *Proceedings of the Fourth International Conference of the Learning Sciences*. Mahwah, NJ: Lawrence Erlbaum. Pages 282 – 289.

Yelland, N. (1995). Mindstorms or a storm in a teacup? A review of research with Logo. *International Journal of Mathematics Education, Science, and Technology,* 26(6), 853 – 869.

Yerushalmy, M. (1999) Making Exploration Visible: On software design and school algebra curriculum. *International Journal of Computers for Mathematical Learning.* 4(2-3). pp. 169-189.