Novel Multi-Modal Tools to Enhance Disabled and Distance Learners' Experience of Mathematics

Dilaksha Attanayake, Gordon Hunter, James Denholm-Price, Eckhard Pfluegel

***Abstract*— This paper discusses ICT-related needs, with particular reference to studying mathematically-based disciplines, of several types of users – notably people with physical disabilities, online (distance) learners and people working or studying "on the move", relying on mobile devices. We note the inadequacy of existing interfaces for mathematical input and output for these groups and investigate several options for addressing the problems. We present two tools that employ novel human computer interaction methods which allow users to create and edit mathematical content in electronic documents and address some of the above issues. We apply one of these in a user study in a classroom environment, aiming to investigate whether this system can assist students to learn mathematical concepts via creating mathematical e-content. We show that there is some evidence that, by using this tool, some students can improve their general understanding of mathematical concepts, comparing "deep" natural language descriptions with representations based on "shallow" application-specific interactions.**

***Index Terms* — HCI, Learning Mathematics, Spoken Mathematics, Web-based Mathematical Interfaces, Mathematical Text Editing, Assistive Technology, Educational Technology.**

I. Introduction

Over recent years, standards of literacy and education have been greatly improving in many emerging countries, contributing significantly to their economic development. However, many such countries have suffered from conflicts, leading to many young people losing limbs or the use of limbs. Some such disabilities are due to direct combat, but many are caused by exploding land mines, cluster bombs and similar munitions in conflict zones from Afghanistan and Angola to the former Zaïre. According to The Guardian [38], between 1999 and 2008 Landmine Monitor documented 73576 mine and unexploded ordinance (UXO) casualties Worldwide. Of these, around 55000 were maimed rather than killed, the majority being civilians and nearly a third of them children. In 2009 alone, over 2800 people were injured, in addition to over 1000 deaths, due to such UXO [39]. Furthermore, despite widespread immunisation and treatment campaigns around the globe, diseases such as polio continue to cause disabilities, cases due to such causes being more common in developing regions than in developed countries. (Polio, or formally poliomyelitis, in some central Asian and sub-Saharan African countries results in age-standardised Disability-Adjusted Life Year (DALY) rates in excess of 5 - in the case of one country, as high as 19 – whereas no developed country has a DALY rate due to polio greater than 1 [32]. DALY is a measure of the average number of effective person-years lost due to disability caused by a particular disease, including contributions from "years of life lost" and "years lived with disability".)

Disabilities due to the above, as well as other causes such as accidents, can seriously impair the educational and career opportunities of people who suffer from them. This may occur in an environment where a good education, leading to a professional career, can provide a route out of poverty. As discussed previously in [10], such opportunities in many disciplines – most scientific and engineering fields, along with many in the commercial domain - are underpinned by achievement in mathematically-based subjects (at least at an elementary level). However, the teaching and learning of mathematics is considered very difficult by many people, especially those with a variety of special needs. Although assistive educational technology has provided help to various groups with such needs (see [47] for a review), the development and use of such technology in mathematical disciplines has been rather limited to date. This is in part due to the rather complicated two-dimensional layout of conventional mathematical text, plus the specialised terminology, notation and symbols used in mathematics.

In addition, creating & editing electronic mathematical content still remain difficult for both able and disabled students, and particularly for people studying "at a distance" (e.g. via Open University) and those relying on mobile computing devices such as smartphones and tablet PCs [21]. The sophistication of Human Computer Interaction (HCI) methods in the domain of mathematics is still lagging behind those for other disciplines, meaning that the cognitive load required of people using conventional mathematical editors may have a considerable impact on the individual's learning of mathematical concepts.

Manuscript updated on January 31, 2013.

Dilaksha Attanayake & Eckhard Pfluegel are in the School of Computing & Information Systems, Kingston University, UK (e-mail: k0962524@kingston.ac.uk, E.Pfluegel@kingston.ac.uk)

Gordon Hunter & James Denholm-Price are in the School of Mathematics, Kingston University, UK (e-mail: G.Hunter@kingston.ac.uk, J.Denholm-Price@kingston.ac.uk)

This project aims to study how users create and edit mathematical formulae which are typed into our *TalkMaths* system (described in Section III) using natural language commands, rather than using conventional equation editing software (such as the *Microsoft Word equation editor*). In particular, our goal is to evaluate whether such an approach can aid participants' understanding of particular mathematical concepts, such as the "numerator" and "denominator" of fractions, to reinforce their understanding of these and related ideas. This is carried out through questionnaires, both to test participants' knowledge and understanding and to quantify their own perception of these, with respect to appropriate mathematical concepts. The current paper describes a small pilot study of this type within an introductory non-specialist mathematics course and measures the students' performance in both pre- and post- task tests. We intend to use the results of such studies in the development of new teaching and support materials aimed at improving students' understanding of key ideas. These activities will be rolled-into more courses in the future, which should give useful insights and data for additional studies.

The remainder of this paper is structured as follows. Firstly, we discuss the need for novel interfaces for creating and editing mathematical content in electronic documents and discuss several possible options for these. We then review the development of, and facilities provided by, our own natural language interfaces for mathematics, *TalkMaths* and *SWIMS*. We subsequently present details and results of an initial user trial to evaluate *TalkMaths* as a tool to help improve students' understanding of mathematical concepts. Finally, we put forward our conclusions and propose possible future developments.

II. Novel Interfaces for the Target User Groups

Conventional human-computer interfaces based on input from a keyboard and mouse and output via a visual display are clearly far from ideal for the three target user groups identified above. This is particularly true for the task of creating and editing mathematical content in electronic documents − the standard notation for representing mathematics and commonly-used mark-up languages (such as LaTeX and MathML) are not particularly accessible for these types of users. However, as noted by Österholm [36], the use of specialised symbols and notation gives mathematics great strengths, saving time and space relative to attempts to represent mathematical expressions in ordinary narrative language, which are likely to result in lengthy and potentially ambiguous descriptions. Österholm's study [36] concluded that reading and understanding mathematical text, in standard notation, required rather different skills, which took much time and effort to acquire, from the corresponding task on text written in ordinary language. Although conventional mathematical interfaces are constantly evolving and improving, the problem remains that such systems are not very user-friendly or helpful to naïve users. The learning curve is somewhat steep and lengthy for these systems and issues regarding ease and efficiency of their use are highly significant for our target groups of disabled, mobile device-using and on-line (distance) learners [21]. The following subsections present some previous approaches to addressing these issues.

A. Solutions using Tactile Output

Braille has been a successful tactile medium for nearly 200 years, enabling the blind to read and write. Conventional Braille codes use a two-dimensional representation for each alphanumeric character, but these are arranged in a "linear" format, much like the normal orthography of English and many other languages, to represent words, phrases and sentences. This is not particularly well-suited to represent mathematics. However, novel extensions and modifications to Braille have allowed blind and other visually-impaired students a much wider access to mathematical resources (e.g. [37]). Nevertheless, all of these coding schemes need to be learned by both the students and teachers, which is a problem for most teachers since they (except specialist teachers of the blind) will probably only teach a rather small number of blind students over many years. Some previous authors have also investigated the use of other forms of tactile (haptic) interfaces (e.g. [33]) to replace visual output, or have implemented a hybrid interface combining both Braille and synthetic speech (see below), such as the LAMBDA (Linear Access to Mathematics for Braille Devices and Audio synthesis) projects [34, 35].

B. Solutions using Optical Character Recognition (OCR)

For some groups, typing, but not writing, mathematics is a problem. These would include people with certain types of repetitive strain injuries, people using small mobile devices and, in some cases, on-line distance learners [21]. It has been noted that a growing proportion of studying and academic exercises are being carried out using mobile devices such as smartphones and notebook or tablet computers, often "on the move" or in potentially noisy public places such as cafés [21]. In such cases, input using a keyboard becomes difficult and use of spoken input using Automatic Speech Recognition (ASR – see sub section *C* below) is impractical. Thus, some users might prefer to create and edit mathematical expressions using a handwriting-based system.

For such people, an appropriate option might be to write the necessary mathematical expressions using a smart pen or stylus. The characters used could then be identified using an optical character recognition (OCR) system, and converted into (correctly) typeset mathematical text in electronic form. Some previous authors [26, 27, 28, 29] have developed systems following this approach and indeed OCR has been included into the latest Samsung Galaxy Note II smartphone, with the aid of an innovative stylus called the S Pen [30]. However, two remaining issues are; how to deal with the possibilities of misidentified symbols (potentially a big problem, since many people have poor "on-screen" handwriting) and mistakes by the user. Previous researchers have used syntactic [28] or statistical [29] approaches in attempts to resolve these issues. The latter approach is to some extent similar to the methodology (SLMs) we use in this paper (see section III E below).

C. Solutions using Spoken Input and/or Output

In recent times, speech has become a realistic alternative method for communicating with computers, and speech technology - especially synthetic speech output [48] and automatic speech recognition (ASR) [23, 49] - is becoming quite sophisticated and reliable. Furthermore, humans' learning and teaching of mathematics [19] has not been simply a matter of silently reading and/or writing of the mathematical text, but also tightly associated with speech, gestures and other "side effects" for many years [18]. There have been a variety of systems - primarily designed to assist blind and other visually impaired people - attempting to provide synthetic speech descriptions of mathematical text, including *AsTeR* (*Audio Systems for Technical Readings* [40], *MathGenie* [41], *REMathEx* [42], the commercial system *MathPlayer*™ [43], and *AudioMath* [44]. The latter system is open-source but, unfortunately, only functions in Portuguese. The LAMBDA system [34, 35], mentioned in section II A above, also gives the option of the automated reading-out of mathematical text by synthetic speech. In contrast, systems employing ASR (and further processing) to allow spoken dictation of mathematical expressions can benefit not only the visually impaired but also people with disabilities affecting their hands or arms, and people working or studying "at a distance" (e.g. on-line) or "on the move" using mobile devices [21]. Previous approaches to allowing spoken input of mathematics include the research prototype systems of Bernareggi & Brigatti [45] (which only works in Italian) and Hanakovič and Nagy [46] (which is restricted to use with the Opera web browser), plus the commercial systems *MathTalk*™ [3] (which is only compatible with certain commercial editors) and *Math Speak & Write* [6] (which has a rather limited mathematical vocabulary). All of these systems have serious limitations, prompting us to develop our own system, *TalkMaths*. ASR is a key component of any mathematical interface which allows input via speech, and it can be carried out either on client side or on server side, or even partly on each side in a "distributed" approach. A more detailed discussion of issues relating to these options for speech recognition within the domain of mathematical text editors can be found in [1].

III. TalkMaths & SWIMS

TalkMaths [4, 7, 8] is a web-based editing system for mathematical e-content that can be controlled using a variety of modalities (keyboard, mouse and speech).

Our other prototype system, *SWIMS*, is a proof of concept for a web-based mathematical document editor developed to assist the user by intelligently predicting and/or correcting mathematical input by the use of statistical language models created from several sources of data [10, 11]. The remainder of this section will provide a technical overview of *TalkMaths* and *SWIMS*. We refer the user to our other papers [4, 7, 8, 10, 11] for further details of how they work. A demonstration video, showing *TalkMaths* in use, can be viewed at the webpage http://www.youtube.com/talkmaths.

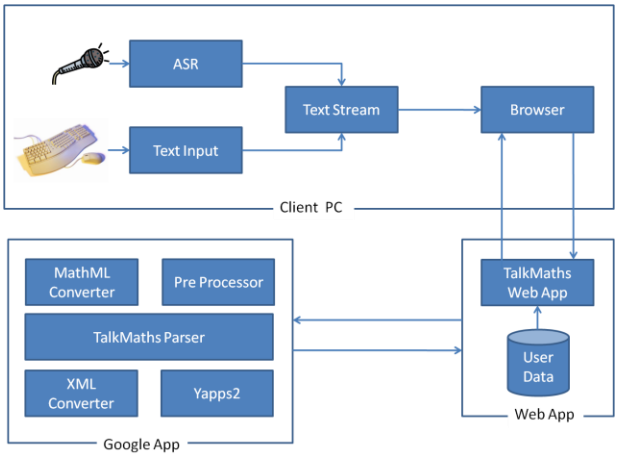


Fig. 1. *TalkMaths* system architecture

Figure 1 illustrates the system architecture of the *TalkMaths* prototype that has been used for this experiment. Two input modalities are available, namely spoken input (using ASR) and textual input from a keyboard & mouse. Each of these result in a text stream which describes the mathematical expression of current interest in relatively natural language (see section III B below). The text stream is then passed through the browser interface, encapsulated in a HTTP request, to the web server. The web application on that server then handles the so-called application logic (covering general security, session management and storage tasks) [1] and calls the parser, running on the Google Apps server [2], which ultimately processes the input text stream, transforming it into the desired format, such as MathML or XML. Again, web app requests and parser responses use HTTP. Upon receiving the parsed output from the Google Apps server, the web app sends the output (in the appropriate marked-up form) to the browser on the client for rendering in conventional mathematical notation. We have used a recursive-descent context-free grammar parser generator, called Yapps2 [5], to develop our *TalkMaths* parser. Due to the bi-modal nature of our system, and the consequential (small) differences between the correct spoken and typed descriptions of any given mathematical expression (see section III B below), a pre-processor has been used to convert the input from either spoken or typed form (as appropriate) into a canonical form that is suitable for analysis by our parser.

A. Multimodality

While *SWIMS* is only currently used with keyboard and mouse, *TalkMaths* accepts input from keyboard, mouse and via speech, which allows this new version of the application to reach a wider user base than its previous counterpart [8].

B. Natural Language Commands

Both *SWIMS* and *TalkMaths* employ an input language that is much closer to how people actually speak mathematics in a classroom environment, compared with specialised mark-up or formatting languages for mathematics, such as *LaTeX* or *MathML*. The rationale behind this is to make learning to command the system to be as easy and intuitive as possible for the user, keeping the "naturalness" of the task to a maximum.

For example, Figure 2 shows encodings of the same mathematical expression, namely

$$\frac{n}{k(n-1)},$$

in *LaTeX*, *MathML*, *TalkMaths* spoken input form and *TalkMaths & SWIMS* keyboard input command language.

LaTeX →

```
$\frac{n}{k(n-1)}$
```

MathML →

```
<mfrac>
  <mi>n</mi>
  <mrow>
    <mi>k</mi>
    <mo>&InvisibleTimes;</mo>
    <mfenced close=")" open="(">
      <mrow>
        <mi>n</mi>
        <mo>-</mo>
        <mn>1</mn>
      </mrow>
    </mfenced>
  </mrow>
</mfrac>
```

TalkMaths speech input →

november over begin kilo open bracket november minus one close bracket end

TalkMaths & SWIMS keyboard input →

n over begin k (n - 1) end

Fig. 2. Encoding of same mathematical expression in LaTeX, MathML, *TalkMaths* and *SWIMS* command languages respectively.

Note that both forms of the *TalkMaths* command language are easy to read and easy for a person to speak/type. They should be much more accessible and easy to learn for novice users than either LaTeX or MathML. Also, note that the *TalkMaths* speech input language [8] requires use of the *NATO* alphabet [9] for the dictation of single characters, due to issues of potential confusion between conventional letter names by ASR systems (e.g. "bee" (b), "cee" (c), "dee" (d)). An example of a simple mathematical expression is the equation for velocity under uniform acceleration $v = u + at$ which in our spoken mathematical language would be read as: "*victor equals uniform plus alpha tango*". A more complex example is the formula for the solutions of a general quadratic equation:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \qquad (1)$$

which would be spoken as "*minus bravo plus or minus square root of bravo squared minus four alpha charlie all over begin two alpha end*". Greek characters, such as α , β , etc. can be inserted using the prefix "greek" before the name of the character. For example, the trigonometric identity :

$$sin(\alpha + \beta) = sin\,\alpha\ cos\,\beta + cos\,\alpha\ sin\,\beta$$

would be read as "*sine begin greek alpha plus greek beta end equals sine greek alpha cos greek beta plus cos greek alpha sine greek beta*". An example of a more complicated expression which can be interpreted by *TalkMaths* is the van der Waals equation from thermal physics, rendered as shown in Figure 3 below. This would be dictated as "*open bracket capital papa plus begin november to the power of two alpha end over begin capital victor to the power of two end close bracket open bracket capital victor minus november bravo close bracket equals november capital romeo capital tango*".

$$\left(P + \frac{n^2 a}{V^2}\right)\left(V - nb\right) = nRT$$

Fig. 3. The van der Waals equation, read as stated above, rendered by *TalkMaths*.

C. Editing paradigms

TalkMaths allows users to edit mathematical expressions that they have input, as rendered on the computer screen, by issuing relatively intuitive commands which are close to how a human-to-human interaction would deal with the same tasks. For example, editing the numerator of a fraction can be invoked using the "edit numerator" command. Here, the application "understands" that the user requires the editing of only a part (in this case, the top part) of the fraction. Another example is where the "edit functions" command will invoke editing of all available functions within the current mathematical expression. We refer to these types of edit commands as "*semantic editing*" commands, as they refer to the meaning of the mathematical structure/components which the user sees on the screen. Other types of commands include "*selective editing*" and "*exhaustive editing*". The former is used to select specific sub-expression(s) within the full expression shown on the screen. For example, if a user wishes to change the sub-expression *2a* in the quadratic formula (1), he/she can use the command *"edit two alpha*", which will make the denominator (*2a*) of the above quadratic formula editable by placing indexed bounding boxes around all possible selections of the sub-expression *2a* - in this case, just the denominator. In contrast to this, exhaustive editing makes use of a displayed set of nested indexed boxes superimposed over the expression to allow the user to select the whole, or any part, of the expression for editing. Three such methods, highlighting all sub-expressions, all individual symbols and all operators, respectively, are illustrated in Figure 4. A more detailed explanation of these editing paradigms can be found in [4, 20].

Fig. 4. Different editing paradigms for editing mathematics by speech, each applied to one of equations of uniformly accelerated motion.

D. Natural Language Search-Driven Help Facility

TalkMaths has a *Help* facility that includes a natural language search option. A user can search through the help information using this tool, by entering search terms in natural language form. If the search phrase contains a word that is not in the vocabulary (*V*) of the *Help* content, we use the *Damerau – Levenshtein* algorithm [12, 13] to calculate the Levenshtein distance between the entered word and each word in *V*. This distance is based on the minimum number of insertions, deletions, substitutions and transpositions of characters

required to transform one string into the other. The word within *V* with the shortest distance from the entered word is selected as the "best guess" for what the user intended to enter.

The *Damerau – Levenshtein* method was originally introduced to compare the similarity of text strings, and we found that this strategy works well when the user makes minor misspellings when typing using the keyboard and mouse. For each *Help* term, we assign a score based on its length, the length of the entered search term and how similar the terms are (based on the Levenshtein distance). Our mechanism assists the user to find appropriate commands using their existing mathematical knowledge. For example, typing, "*How to create a fraction*" command will present all the *TalkMaths* commands associated with fractions and rank them according to how relevant they are (see Figure 5).

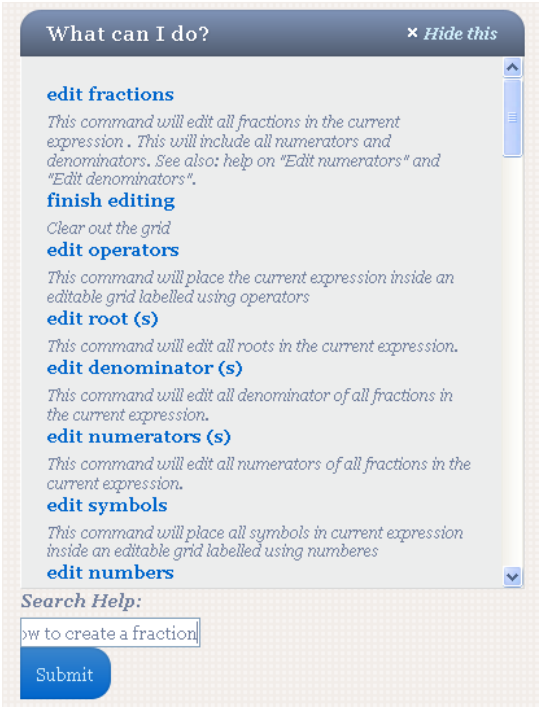


Fig. 5. The *TalkMaths* Help facility.

E. Intelligent Prediction & Correction

A user is bound to make some mistakes during any typing or dictating process. This is likely to be an even bigger problem when entering mathematics than for ordinary text. In contrast to what is required for a conventional text editor, a mathematical text editor has to encode the complex spatial layout and specialised symbols of mathematical expressions. However, the user may need to type or dictate the content in a *linear* manner. For example, the linear statement/command "*fraction a over b end fraction*", or just "*a over b*", should be displayed on the screen in the conventional non-linear mathematical layout:

$$\frac{a}{b}$$

Such differences between the spatial layout and format of the input and output make the editing and correction much harder for mathematical text than for conventional "ordinary" text.

From our earlier work, [7, 8], we found that mathematical text can be reasonably predictable within a sufficiently large dataset. We have used this observation to develop a prediction feature into *SWIMS* [10, 11] so that a user can get intelligent assistance from the system in order to help them type mathematical expressions quickly and efficiently. This prediction mechanism, which is in some ways analogous to predictive text in ordinary word processing systems and on SMS text editors on mobile 'phones, is based on N-gram Statistical Language Models (SLMs) [16, 17, 23], which have been key components of most successful ASR systems for many years. N-gram models are based on statistics of sequences of N consecutive words in previously encountered example ("training") text of the appropriate nature – in our case, descriptions of mathematical expressions using the near-natural command language described in section III B above. We have also included a semi-automatic correction system, based on the *Damerau – Levenshtein* method [12, 13], which prompts the user with suggested corrections when an out of vocabulary word is observed in the input. These features are analogues of predictive text and auto correction facilities in ordinary text editors. In *SWIMS*, these facilities offer likely alternatives, in transcribed "spoken mathematics" form and/or displayed in conventional mathematical notation, to what the user actually entered (see Figures 6 & 7 for examples).

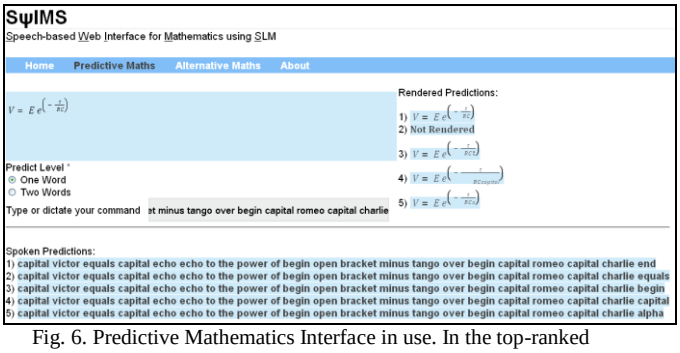


Fig. 6. Predictive Mathematics Interface in use. In the top-ranked suggestion, the SLM predicts that "charlie" will be followed by "end".

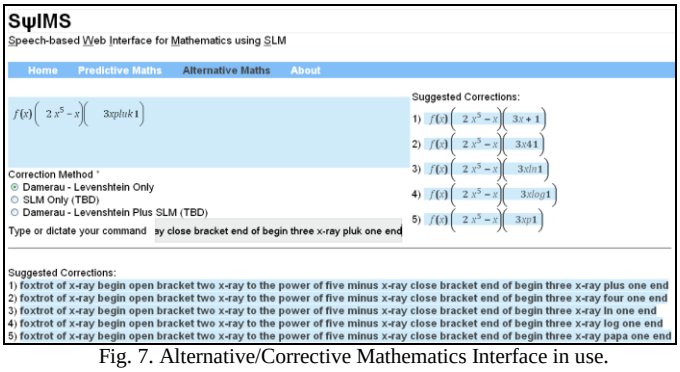


Fig. 7. Alternative/Corrective Mathematics Interface in use.
In the top-ranked suggestion, the OOV word "pluk" is replaced with "plus".

F. Research Challenges

The general case of natural language parsing is believed to be an unsolved problem. However, whilst our systems,

TalkMaths and *SWIMS*, require the parsing of "close-to-natural" language, namely *spoken mathematics* and typed mathematical language, in both these situations we are dealing with a much more controlled case than for general natural language. In these cases under study here, we have a restricted vocabulary (of the order of 100 words) and a fairly prescriptive syntax which, with the exception of a few "pathological" examples, can be well-described by a context-free grammar [4]. Nevertheless, incomplete or ill-formed mathematical input can be highly ambiguous and is much harder to parse and then render compared to well-formed, unambiguous mathematical expressions. We have used a top-to-bottom LL(1) parser algorithm [24] and pre- and post-parser error recovery strategies in our prototype version of *TalkMaths* for the work discussed in this paper. Currently, we are looking into novel ways of parsing mathematics by using only the precedence of operators [22] with non-deterministic GLR (generalized-LR) parsing techniques [14, 15]. However, in this paper we do not discuss the technical advances made by using these methods.

IV. Experiment

In this section, we present the design, implementation and results of an experiment on using *TalkMaths* with real students in a real classroom environment. This was to assess the *TalkMaths* application in terms of its usability and impact on learning of mathematical concepts compared with use of a conventional mathematical editor. A user evaluation had previously been carried-out on the original, desktop-based version of *TalkMaths* [4]. However, this focused on the system's ease of use and how fast and accurate users were in performing various mathematical editing tasks. It was found that the majority of participants, who did not have any disability, took longer and produced more errors using *TalkMaths* than when using a conventional keyboard & mouse based editor. Nevertheless, the only participant who did have a major disability (Duchenne Muscular Dystrophy) performed better, both in term of speed and accuracy, when using *TalkMaths*, and out-performed many of the non-disabled group when using this modality [4]. This illustrated the potential benefits of *TalkMaths* to one of the user groups for which it was primarily designed. The present study investigates how the new version of *TalkMaths* influences the users' understanding of mathematical concepts related to the prescribed editing tasks, instead of speed and accuracy in performing the editing. The results of a preliminary evaluation of the new version have already been published elsewhere [31], but scrutiny of these revealed some weaknesses in the design of the original experiment – notably ambiguities in some parts of the questionnaires. These limitations were addressed and the revised materials were tested using a new group of students. The details and results of this refined, follow-up study are presented below.

A. Design of Learning Activities

We developed a set of classroom mathematical materials and learning tasks for undergraduate Life-Science students who were taking a basic mathematics module at Kingston University. The tasks to be carried out by the participants, and the questions on their mathematical knowledge, were designed to be appropriate to their typical level of mathematical expertise. The volunteer student participants were allocated to two groups randomly. Both groups carried out the same tasks, but using two different tools. The first group (A) used a conventional editor (*Microsoft Word Equation Editor*) while the other group (B) used our research prototype system, *TalkMaths*. Note that all subjects had previously used *MS Equation Editor* but none had used the *TalkMaths* system before. Each participant was asked to complete three questionnaires. The first questionnaire was about the participant's own perception of their mathematical competence at the start of the experiment. The second was a diagnostic test related to the tasks they were about to carry out and the final questionnaire, given at the end of the exercise, was similar to the second, in order to assess improvements to the participants' understanding, but also included questions concerning their experience of using whichever system they were allocated. From our pilot experiments [31], it was observed that some participants omitting to answer some of the questions – particularly in the post-task questionnaire - led to results of dubious reliability. Hence, in the follow-up study we instructed the subjects that all questions on both pre- and post-task questionnaires were mandatory. Furthermore, after detailed scrutiny of the original versions, the revised questions were re-designed and re-worded to be as unambiguous as possible. Details of the tasks and the questionnaires are given in the appendices.

B. Undertaking of Learning Activities

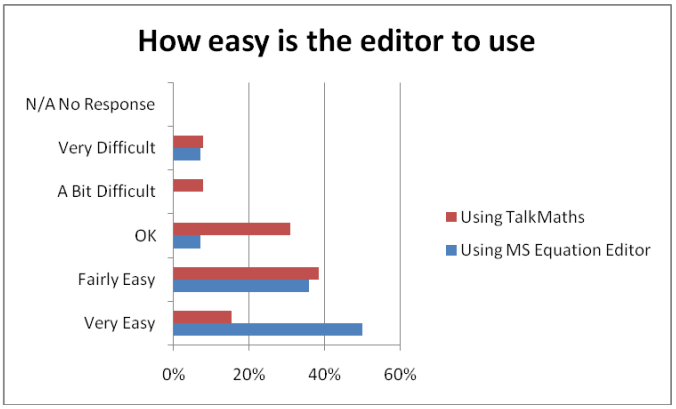
The regular teaching staff and demonstrators for the module supervised the participants carrying out the tasks of the experiment, without actually instructing them. The students were required to learn by themselves, with the only resources available being the instructions given in the worksheet and the *Help* facility of whichever tool they were assigned to. After completing the first two questionnaires, participants were required to undertake three tasks (Tasks 1, 2 & 3 – see the Appendices) creating and editing mathematical expressions involving fractions, functions and square roots respectively. Each task required them to create a specified equation and then carry out a minor modification to this using the editor they were assigned. As noted previously, all participants had used *Microsoft Equation Editor* in earlier practical sessions but no prior training on *TalkMaths* was given. (Thus, we expected "better" performance and possibly higher levels of satisfaction amongst the group using *MS Equation Editor*.) All participants were encouraged to use the *Help* facility of the editor to resolve any questions they might have while completing the tasks.

C. Evaluation of Learning Activities

Table I presents the results on the post task feedback from the participants on the ease of use of the tool they were using to complete the tasks. The group using the *MS Equation Editor* seemed to find it easy to use, more-so than the group using *TalkMaths*. When these qualitative evaluations were each put onto a 5 point Likert scale, the differences in perceived ease of use of the system by the two groups proved

just to be statistically significant ($p \approx 0.047$ in a one-tailed t-test). However, this was to be expected due to the participants' previous experience with *MS Equation Editor*, in contrast to none of them having used *TalkMaths* before.

TABLE I
HOW EASY IS THE EDITOR TO USE?



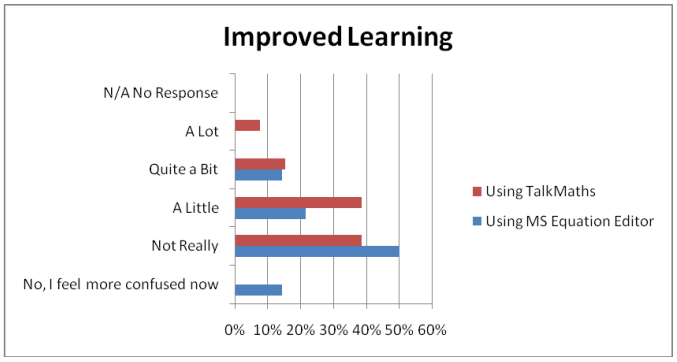
Editor Used	Participants	Very Easy	Fairly Easy	OK	A Bit Difficult	Very Difficult	No Response
MS Equation	14	50%	36%	7%	0%	7%	0%
TalkMaths	13	15%	38%	31%	8%	8%	0%

Table II presents the results on the post task feedback from the participants regarding any improvements they perceived to their understanding of the mathematical concepts involved in the task after using the appropriate tool they were allocated.

Most users of *TalkMaths* (61%) seemed to believe it had improved their understanding, while only 35% of users of *MS Equation Editor* reported any improvement. In fact, 14% of users of *MS Equation Editor* thought it had impeded their understanding, whereas no *TalkMaths* users held this opinion. However, when these qualitative evaluations were converted to a 5 point Likert scale, the difference between the two groups' perceptions proved not quite to be statistically significant ($p \approx 0.065$ for a one-tailed t-test).

We also evaluated the students' performance on knowledge of relevant mathematical terminology, both before and after the tasks, to investigate whether the exercise, possibly including use of the *Help* facility of whichever tool they were allocated, had improved this knowledge. We gave a score of *+1* whenever the participant correctly gave the prescribed answer to one of these "mathematical knowledge" questions, or a score of *+0.5* for an answer we considered to be a relevant "near miss". If the answer given was incorrect or the student did not respond, a score of *0* was given for that question. We observed the change to each student's score between the pre- and post-task questionnaires.

TABLE II
IMPROVED UNDERSTANDING?



Editor Used	Participants	No, I feel More confused now	Not Really	A Little	Quite a Bit	A Lot	No Response
MS Equation	14	14%	50%	21%	14%	0%	0%
TalkMaths	13	0%	38%	38%	15%	8%	0%

Although, in both groups, most individuals gave the same response to any given question in both pre- and post-task questionnaires, indicating no change to their knowledge, one participant in group *A* (*MS Word Equation Editor*) actually did worse the second time, indicating a decline in understanding ! In the previous, pilot study [31], such observations could possibly have been due to laziness, where the subject did not complete all the questions in the post-task questionnaire. However, this was not the case in the current study. Although the overall average score for the group did increase slightly after performing the tasks, this improvement was very small and was not statistically significant ($p \approx 0.18$ in a one-tailed t-test). In contrast, no participants using *TalkMaths* did worse on the second questionnaire, five students improved their scores and overall the group average score increased by a noteworthy amount, which proved statistically significant ($p \approx 0.011$ in a two-tailed t-test, or $p \approx 0.006$ in a one-tailed test). The difference between the two groups in mean improvement after performing the tasks also proved to be weakly significant ($p \approx 0.048$ in a one-tailed t-test), where the *TalkMaths* group showed greater improvement.

D. Discussion

We found that use of the *TalkMaths* system did make a modest, but statistically significant, improvement to the average of the students' scores on their knowledge and understanding of the mathematical concepts relevant to the given tasks. They also perceived this to be the case in their own qualitative self-evaluations of how they had performed. Although both the mean knowledge score and students' self-perception of their understanding improved by a small amount for the *MS Equation Editor* group, neither of these were significant statistically. This suggests that *TalkMaths* has better potential as a tool to aid people's mathematical skills than does *MS Equation Editor*. At first sight, the results of this

study on the ease of use of the two systems might appear disappointing, with students on the whole finding the *MS Equation Editor* easier to use than the *TalkMaths* editor. However, it must be born in mind that all the students had used *MS Equation Editor* before, whereas none of them had any previous experience or training in the use of *TalkMaths*. In the future, we would hope to perform a more controlled experiment, either using participants who had not previously used either editor, or providing some initial training in the use of *TalkMaths*, in order to reduce any bias in favour of one system or the other. Furthermore, previous researchers have investigated students' preferred methods and styles of learning. Some such studies have suggested considerable variations in the ways different individuals learn. For example, Fleming [25] has identified four rather different learning styles adopted by students, categorising these as Visual, Auditory, Reading/Writing and Kinesthetic within his so-called *VARK* model. It is also possible that differences in individuals' preferred learning styles, and other similar factors, might affect whether an editor which is conventionally text-based, GUI-based, or which employs a more novel modality (such as *TalkMaths*) would be best suited to a particular user. As we have already noted, different modalities may also be appropriate for users with various disabilities [4,8], people relying on mobile devices, and online (distance) learners [21]. Some users may prefer to create and edit mathematical expressions using a handwriting-based system with optical character recognition (OCR). Some researchers have already investigated this [26, 27, 28, 29], and indeed OCR has been included into some recent mobile devices, including the latest Samsung Galaxy Note II smartphone [30].

V. Conclusions

This paper has discussed the ICT-needs, with reference to studying mathematically-based disciplines, of certain groups – people with disabilities, online (distance) learners and people relying on mobile devices – which represent a notable section of the population in the emerging regions of the World. We have discussed various ways in which some of these needs can be addressed and presented two innovative tools which make some progress towards achieving these goals. These have been employed in a user evaluation study, with real students as participants, comparing our own mathematical text editor, *TalkMaths* with a conventional, commonly-used commercial system (*MS Word Equation Editor*) for ease of use and participant opinion on whether it had improved their mathematical knowledge and understanding. We also performed a quantitative assessment to investigate whether there was any actual evidence of such an improvement. The results from this initial evaluation suggest that there is still significant room for improvement for *TalkMaths* in these respects. However, although fewer users found our system, *TalkMaths*, easy to use than the more conventional *MS Equation Editor*, *TalkMaths* did appear to have a greater beneficial effect on the participants' knowledge and understanding of mathematical concepts. This was the case both from the users' own perception of their mathematical understanding, and from the results of pre- and post- task testing of their mathematical knowledge. It should also be noted that all the users in the study had used the conventional editing system (*MS Equation Editor*) previously but none had used *TalkMaths* before. This fact could contribute to biasing the users' opinions in favour of the conventional system. Furthermore, none of the participants in this study were really from the target groups – namely the disabled, distance learners and people relying on mobile devices – for which *TalkMaths* was designed to help. We would expect more positive results if participants were drawn from such groups of users. In the light of these observations, the results of this study are quite encouraging, bearing in mind that *TalkMaths* was a system of which the users had no previous experience.

VI. Future work

In addition to the work described in this paper, we are investigating novel parsing strategies for spoken mathematical expressions in the hope of obtaining more powerful, reliable and robust analysis of natural language descriptions of mathematics. Once these expressions have been rendered into conventional mathematical notation, further innovative methodologies for editing them, both in terms of the paradigms the user can employ to carry out the necessary modifications, and assistive tools such as predictive or corrective text, will be implemented and evaluated.

In the future, we are also aiming to carry out a large scale usability study on our *TalkMaths* system, with participants from the target user groups: we would like to present our editing paradigms to experienced speech-interface users, disabled users, distance learners and mobile device users and evaluate how user-friendly and useful the system is for them. Eventually, we would like to integrate *TalkMaths* features with other systems, including editors, mathematical and educational tools – such as computer algebra systems - to enhance the experience of creating, using and modifying mathematical content easily and intuitively, whilst also assisting people's learning and understanding of mathematical concepts.

Acknowledgments

This work was partly supported by the Research Informed Teaching initiative at Kingston University. One of us (Dilaksha Attanayake) is partly funded by a SEC Faculty research bursary. We would like to thank Dr Nigel Atkins, Dr Mastaneh Davis, Dr Vincent Lau and Maryam Hajiesmaeili for allowing us to run the tests in their classes and assisting with supervising the tests, and all the student volunteers for their participation in the study.

References

[1] D. Attanayake, E. Pfluegel, J. Denholm-Price, G. Hunter, Architectures for Speech-Based Web Applications, 4th Int. Conf. on Semantic E-business & Enterprise Computing (SEEC2011), U.K., July 2011

[2] Severance, C., Using Google App Engine. O'Reilly Media, Incorporated, 2009

[3] Metroplex Voice Computing, Inc. mathtalk.com. http://www.mathtalk.com/Accessed: 25/01/2013

[4] A. Wigmore, Speech-Based Creation and Editing of Mathematical Content. Ph.D. Thesis, Kingston University, U.K., 2011

[5] A. Patel, Parsing with yapps2, http://theory.stanford.edu/~amitp/yapps, Accessed: 25/01/2013

[6] C. Guy, M. Jurka, S. Stanek, and R. Fateman. Math speak & write, a computer program to read and hear mathematical input. Technical report, University of California, Berkeley, Electrical Engineering and Computer Sciences Department, 2004

[7] D.R. Attanayake, G. J. A. Hunter, J. C. W. Denholm-Price, E. Pfluegel, Interactive error correction using statistical language models in a client-server interface for editing mathematical text, Designing Inclusive Systems – Designing Inclusion for Real-World Applications, Ed. P Langdon et al. Springer Verlag, London, Chapter 13, pp 125-132, 2012

[8] A. Wigmore, G. Hunter, E. Pfluegel, J. Denholm-Price, M. Colbert TalkMaths Better ! Evaluating and improving an intelligent interface for creating and editing mathematical text. Proc. of 6th International Conf. on Intelligent Environments, Kuala Lumpur, Malaysia, July 2010

[9] G. Law, Phonetic alphabets (alpha bravo charlie delta), 2007 http://www.faqs.org/faqs/radio/phonetic-alph/full/

[10] D.R. Attanayake, G. J. A. Hunter, J. C. W. Denholm-Price, E. Pfluegel, Intelligent Assistive Interfaces for Editing Mathematics, 1st Workshop on Future Intelligent Educational Environments (WOFIEE'12), Guanajuato, Mexico, 2012

[11] D.R. Attanayake, G. J. A. Hunter, J. C. W. Denholm-Price, E. Pfluegel, SWIMS (Speech-based Web Interface for Mathematics using Statistical language models): An intelligent editing assistant for mathematical text, 8th Int. Conference on Intelligent Environments, IE2012, Guanajuato, Mexico, 2012

[12] F. Damerau, A technique for computer detection and correction of spelling errors. Comm. ACM, 1964, Vol. 7 (3), pp 659-664

[13] V. Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics-Doklady, 1966, Vol. 10 (8), pp 707-710

[14] A. Johnstone, E. Scott, and G. Economopoulos. Generalized parsing: Some costs. In Proceedings of International Conference on Compiler Construction, volume 2985 of LNCS, pages 89–103. Springer-Verlag, 2004

[15] T. A. Wagner and S. L. Graham. Incremental analysis of real programming languages. In PLDI '97: Proc. of the ACM SIGPLAN 1997 conference on Programming language design and implementation, pages 31–43, New York, NY, USA, 1997. ACM Press

[16] R. Rosenfeld, Two decades of statistical language modeling: where do we go from here?, *Proceedings of the IEEE*, 2000, Vol. 88 (8) (August 2000), pp. 1270 – 1278

[17] S. Young, Large Vocabulary Speech Recognition : A review, *IEEE Signal Processing Magazine*, Vol. 13, Issue. 5, pp. 1-4, 1996

[18] L. Leventhall, Bridging the Gap Between Face to Face and Online Math Tutoring, ICME-10, Copenhagen, 2004

[19] R. Skemp, The psychology of learning mathematics, Lawrence Erlbaum, 1987

[20] A. Wigmore, E. Pfluegel, D. Attanayake, Speech-Based Editing Paradigms for Mathematical Content, Technical Report KU-CISM-2011-1, 2011

[21] J. Cuartero-Olivera et al, Reading and writing mathematical notation in e-learning environments, eLC Research Paper Series, Universitat Oberta de Catalunya, Spain, http://elcrps.uoc.edu/ojs/index.php/elcrps/about, 2012

[22] A. Aho, R. Sethi, and J. Ullman, Compilers: Principles, Techniques, and Tools, world student series edn, Addison-Wesley Publishing Company, USA, 1986

[23] S. Young, Talking to machines (statistically speaking). In Proceedings of International Conference on Spoken Language Processing (ICSLP), Denver, Colorado, USA, 2002

[24] A. Patel, Parsing with yapps, http://theory.stanford.edu/~amitp/yapps/, 2009

[25] N.D. Fleming, "Teaching and learning styles: VARK strategies." Christchurch, 2001, also http://www.vark-learn.com/english/index.asp

[26] M. Suzuki, F. Tamari, R. Fukuda, S. Uchida, & T. Kanahori, Infty - an integrated OCR system for mathematical documents. In C. Vanoirbeek, C. Roisin, E. Munson (eds.) *Proceedings of ACM Symposium on Document Engineering*, 2003, pp. 95–104, http://www.inftyproject.org/en/index.html

[27] S. M. Watt and X. Xie, Recognition for Large Sets of Handwritten Mathematical Symbols , *Proc. IEEE Int. Conf. on Document Analysis and Recognition, (ICDAR 2005)*, Korea, IEEE Press, pp. 740-744

[28] A. Fujiyoshi, M. Suzuki, S. Uchida, Syntactic Detection and Correction of Misrecognitions in Mathematical OCR, *Proceedings of The 10th International Conference on Document Analysis and Recognition, ICDAR 2009*, Barcelona, Spain, pp.1360-1364

[29] E. Smirnova & S.M. Watt, Context-Sensitive Mathematical Character Recognition, *Proc. IAPR Int. Conf. on Frontiers in Handwriting Recognition, (ICFHR 2008)*, August 19-21 2008, Montreal, Canada

[30] Samsung, 2012, Samsung Galaxy Note II http://www.samsung.com/global/microsite/galaxynote/note/story.html?type=find

[31] D.R. Attanayake, G. J. A. Hunter, J. C. W. Denholm-Price, E. Pfluegel, A Novel Web-Based Tool to Enhance Learning of Mathematical Concepts, *Proceedings of International Conference on Advances in ICT for Emerging Regions (ICTER)*, Dec 13-14, Colombo, Sri Lanka. ISBN 978-1-4673-5527-8, 2012

[32] World Health Organisation (WHO), Death and DALY estimates for 2004 by cause for WHO member states (2009-11-12), http://www.who.int/healthinfo/statistics/bodgbddeathdalyestimates.xls, 2009

[33] K. Moustakas et al, Haptic rendering of visual data for the visually impaired, IEEE Multimedia, 14(1), 62-72

[34] Edwards, A. D. N., McCartney, H. and Fogarolo, F., Lambda: A multimodal approach to making mathematics accessible to blind students, *Proceedings of ASSETS 2006*, Portland, Oregon, ACM, 48-54.

[35] Schweikhardt, W., Bernareggi, C., Jessel, N., Encelle, B., Gut, M., LAMBDA: A European System to Access Mathematics with Braille and Audio Synthesis, Proceedings of ICCHP 2006 (10th International Conference on Computers Helping People with Special Needs), 1223-1230, 2006

[36] M. Österholm, Characterizing Reading Comprehension of Mathematical Texts, Educational Studies in Mathematics, Vol. 63 (3), pp. 325-346, 2006

[37] Math in Braille Project, 2011, http://www.mathinbraille.at/en

[38] The Guardian (U.K.), From Laos to Libya, landmines still take their toll on civilians, http://www.guardian.co.uk/global-development/poverty-matters/2012/jul/06/landmines-toll-civilians-laos-bombs, 2012

[39] The Monitor, FAQ on Casualties, http://www.the-monitor.org/index.php/LM/The-Issues/FAQs#23913 , 2009

[40] T. V. Raman, Audio system for technical readings, Springer Verlag, Berlin, 1998

[41] N. Jacobs, "MathGenie : User's Guide and Teacher's Manual", 2006, http://logicalsoft.net/MathGenie.pdf

[42] P. Gaura, REMathEx - Reader and Editor of the Mathematical Expressions for Blind Students, *Proceedings of 8th ICCHP 2002*, Springer-Verlag, London, U.K, pp. 486-493.

[43] N. Soiffer, MathPlayer: web-based math accessibility. In Proc. of the 7th int. ACM SIGACCESS conf. on Computers and Accessibility (Assets '05), ACM, New York, NY, USA, pp. 204-205, 2005

[44] H. Ferreira, D. Freitas, Enhancing the Accessibility of Mathematics for Blind People: The AudioMath Project, 2004, K. Miesenberger et al (eds.) *Proc. ICCHP 2004*, Springer LNCS, vol. 3118, pp. 678–685.

[45] C. Bernareggi & V. Brigatti, Writing mathematics by speech: A case study for visually impaired, *Proceedings of 11th ICCHP 2008*, pp. 879–882

[46] T. Hanakovic & M. Nagy, Speech recognition helps visually impaired people writing mathematical formulas, Proceedings of 10th ICCHP 2006, pp. 1231–1234

[47] A. I. Karshmer, Access to mathematics and science, Proc. 11th International Conference on Computers Helping People with Special Needs (ICCHP), 2008, Vol. 5105, pp. 873–874.

[48] Carlson, R. and Granström, B., Speech Synthesis, in The Handbook of Phonetic Sciences, Second Edition (eds W. J. Hardcastle, J. Laver and F. E. Gibbon), Blackwell Publishing Ltd., Oxford, 2010

[49] Juang, B. H., & Rabiner, L. R., Automatic speech recognition–A brief history of the technology development. *Encyclopedia of Language and Linguistics, Elsevier*, 2005

APPENDIX A

Pre session questionnaire for both groups of participants (the correct answers for questions 2 to 6 are given in Appendix E):

1. How competent would you regard yourself in terms of your basic maths skills? (Doing simple algebraic manipulations, involving fractions, functions and square roots)

   Excellent ☐ Good ☐ Fair ☐ Poor ☐
   Don't Know ☐

2. Given the following fraction,

$$\frac{x^2 - 1}{x + 1}$$

   do you know what the expressions on the top and the bottom of the expression are normally called? If so, write them in the blanks below:
   The top is called the _____and the bottom is called the _____of the fraction.

3. Consider the expression,

   $(a + b)^n$

   In general, the number *n* in this example is called the _____ of *(a + b).*

4. The mathematical symbols +, -, ×, ÷ represent the processes of addition, subtraction, multiplication and division respectively. We call them binary _____.
   (Add a single word to complete the blank space above)

5. Given the following expression,

$$z + f(x^2 + y^2)$$

   do you know how the expression inside the bracket associated with the function *f* is normally referred to? If so, write it in the space:
   The expression inside the bracket is called the _____of the function of *f.*

6. Given the following square root,

$$y + \sqrt{x(1+x)^n}$$

   do you know how the expression inside the square root symbol is normally referred to? If so, write it in the blank space below:
   The expression inside the square root is called the _____of the root.

APPENDIX B

Mathematical tasks given for participants who were allocated with *Microsoft Equation Editor*:

Question on fractions
For this exercise, you are supposed to create the following mathematical expression, using *Microsoft Equation Editor*:

$$a + b\frac{2+c}{x-y} \quad -(1)$$

Now, replace the expression $x - y$ in (1) by $w - z$.

Question on functions
Still using the *Microsoft Equation Editor*, create the following expression:

$$a + f(2x - 5) \quad -(2)$$

Now, replace the expression *2x* in (2) by *y*.

Question on square roots
Still using the *Microsoft Equation Editor*, create the following expression:

$$a + \sqrt{x + 2y} \quad -(3)$$

Now, replace the expression *2y* in (3) by *3z.*

APPENDIX C

Mathematical tasks given for participants who were allocated with *TalkMaths*:

Question on fractions
Using the *TalkMaths* editor, create the following expression:

$$a + b\frac{2+c}{x-y} \quad -(1)$$

In order to do this, in the input field, type "*a + b* begin *2 + c* end over begin *x - y*" and press enter.

Now, replace the expression $x - y$ in (1) by $w - z$.
Hint: consult the help facility on fractions and selections. This will help you to edit the expression, select appropriate parts of it and replace the desired symbols.

Question on functions
Still using the *TalkMaths* editor, clear the previous expression and create the following expression:

$$a + f(2x - 5) \quad -(2)$$

In order to do this, in the input field, type "*a + f* of begin *2 x - 5* end" and press enter.

Now, replace the expression *2x* in (2) by *y*.
Hint: consult the help facility on functions and selections. As in the previous example, this will help you to edit the expression, select appropriate parts of it and replace the desired symbols.

Question on square roots
Still using the *TalkMaths* editor, create the following expression:

$$a + \sqrt{x + 2y} \quad -(3)$$

In order to do this, in the input field, type "a + square root of begin *x + 2 y* end" and press enter.

Now, replace the expression *2y* in (3) by *3z.*

Hint: consult the help facility on roots and selections. As in the previous example, this will help you to edit the expression, select appropriate parts of it and replace the desired symbols.

APPENDIX D

Post session questionnaire for both groups of participants (the correct answers for questions 3 to 7 are given in Appendix E):

1. How competent would you regard yourself in terms of your basic maths skills? (Doing simple algebraic manipulations, involving fractions, functions and square roots)

Excellent ☐ Good ☐ Fair ☐ Poor ☐
Don't Know ☐

2. Which of the computer-based tasks(s) from this session did you manage to complete? Please circle.

Task 1(i) Task 1(ii) Task 2(i) Task 2(ii)
Task 3(i) Task 3(ii)

3. Given the following fraction,

$$\frac{x^2 - 1}{x + 1}$$

do you know what the expressions on the top and the bottom of the expression are normally called? If so, write them in the blanks below:

The top is called the _____ and the bottom is called the _____ of the fraction.

4. Consider the expression,

$(a + b)^n$

In general, the number n in this example is called the _____ of $(a + b)$.

5. The mathematical symbols +, -, ×, ÷ represent the processes of addition, subtraction, multiplication and division respectively. We call them binary _____.
(Add a single word to complete the blank space above)

6. Given the following expression,

$z + f(x^2 + y^2)$

do you know how the expression inside the bracket associated with the function f is normally referred to? If so, write it in the space below:

The expression inside the bracket is called the _____of the function of f.

7. Given the following square root,

$$y + \sqrt{x(1 + x)^n}$$

do you know how the expression inside the square root symbol is normally referred to? If so, write it in the blank space below:

The expression inside the square root is called the _____of the root.

8. How easy did you find the Equation Editor system to use? Please tick the appropriate answer.

Very Easy ☐ Fairly Easy ☐ O.K. ☐
A Bit Difficult ☐ Very Difficult ☐

9. Did you feel that using the *TalkMaths/ Microsoft Equation Editor* for doing this exercise improved your understanding of the relevant mathematical concepts? Please tick the appropriate answer.

No, I feel more confused now ☐ Not Really ☐
A Little ☐ Quite a Bit ☐ A Lot ☐

10. Have you any other comments? If so, please write them below.

APPENDIX E

The prescribed correct answers for questions 2 to 6 and 3 to 7 in pre and post questionnaires respectively (the numbers not in parentheses refer to the pre-questionnaire and those in parentheses to the post- questionnaire) are as follows.

2(3). The top is called the **numerator** and the bottom is called the **denominator** of the fraction.

3(4). In general, the number n in this example is called the **power** of (a + b).

4(5). We call them binary **operators**.

5(6). The expression inside the bracket is called the **argument** of the function of f.

6(7). The expression inside the square root is called the **radicand** of the root.