

TAYOUKI: A SKETCH-BASED TUTORING SYSTEM FOR YOUNG KIDS

A Thesis

by

FRANCISCO ALFONSO VIDES CERÓN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2012

Major Subject: Computer Science

TAYouKi: A Sketch-Based Tutoring System for Young Kids

Copyright 2012 Francisco Alfonso Vides Cerón

TAYOUKI: A SKETCH-BASED TUTORING SYSTEM FOR YOUNG KIDS

A Thesis

by

FRANCISCO ALFONSO VIDES CERÓN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Tracy Hammond
Committee Members,	Erin McTigue
	Frank Shipman
Head of Department,	Duncan M. Walker

August 2012

Major Subject: Computer Science

ABSTRACT

TAYouKi: A Sketch-Based Tutoring System for Young Kids. (August 2012)

Francisco Alfonso Vides Cerón, B.S., Universidad de los Andes

Chair of Advisory Committee: Dr. Tracy Hammond

Intelligent tutoring systems (ITS) have proven to be effective tools for aiding in the instruction of new skills for young kids; however, interaction methods that employ traditional input devices such as the keyboard and mouse may present barriers to children who have yet learned how to write. Existing applications which utilize pen-input devices better mimic the physical act of writing, but few provide useful feedback to the users. This thesis presents a system specifically designed to serve as a useful tool in teaching children how to draw basic shapes, and helping them develop basic drawing and writing skills.

The system uses a combination of sketch recognition techniques to interpret the handwritten strokes from sketches of the children, and then provides intelligent feedback based on what they draw. Our approach provides a virtual coach to assist teachers teaching the critical skills of drawing and handwriting. We do so by guiding children through a set of exercises of increasing complexity according to their progress, and at the same time keeping track of students' performance and engagement, giving them differentiated instruction and feedback. Our system would be like a virtual Teaching Assistant for Young Kids, hence we call it TAYouKi.

We collected over five hundred hand-drawn shapes from grownups that had a clear understanding of what a particular geometric shape should look like. We used this data to test the recognition of our system. Following, we conducted a series of case studies with children in age group three to six to test the interactivity efficacy of the system. The studies served to gain important insights regarding the research challenges in different domains. Results suggest that our approach is appealing and engaging to children and can help in more effectively teach them how to draw and write.

DEDICATION

To my guide.

To my support.

To my inspiration.

To my mother.

ACKNOWLEDGEMENTS

Formality recommends using this section to give thanks to the committee chair and members, as well as peers in the same research area. This time is slightly different: this has nothing to do with formality. I really and truly want to thank Dr Tracy Hammond. She was my advisor. Not only my academic advisor, in general *my advisor*. I came to the United States and to Texas A&M University with high expectations about my academic experience and the professor that I would be working with. During these two years she went far beyond these expectations. She was a constant guide and an amazing source of knowledge and ideas. With her astonishing balance between hard work, research and personal life, she also served as a role model for me. I learned a lot from her in this journey, and I will be in permanent debt with her because of this. For the moment I want to say thank you! And hope that in the future our paths can cross again.

Similarly I want to thank Dr Frank Shipman; he was not only a committee member but my teacher for three consecutive terms. It was in his classes that this project was conceived and raised. The project that we show you today, has the signature of all these classes in it.

I also want to thank Dr Erin McTigue. She gave final direction to this project when she introduced me into the educational world in particular literacy teaching. She gave me very relevant advice and added to this work a completely new dimension that makes it a grounded solution for a real problem.

I could not have gone so far without the help of my peers; particularly the Sketch Recognition Lab. Someone from another major once visited the lab and commented this lab did not look like a lab: it didn't have giant machines, beakers, or supercomputers. She had not met the people yet, that is what empowers our lab and allows us to create science out of a couple of home desktops. And it is to note that in particular the SRL lab has amazing people not only as researchers but moreover as persons. This was like a family that helped me through all the process with ideas, code-snippets, references, practice-talks and lots of fun to keep me encouraged. In particular I feel I should thank personally Hong-Hoe Kim, Paul Taele and Jimmy Ho who helped in the development of an early version of the system for the IUI class.

The internal contributors are just a part of it. I also want to thank all the people that provided external support, my family, and my friends, and the departmental staff in CS at TAMU. They were always there, giving support to me when I needed it the most. In particular I want to mention Carmen Osorio, Stella Cerón and Rita Ferguson who provided tremendous support in this finishing track of my thesis.

NOMENCLATURE

ITS	Intelligent Tutoring System
HCI	Human Computer Interaction
TAYOUKI	Teaching Assistant for Young Kids
SME	Subject Matter Expert
OCR	Optical Character Recognition
CAD	Computer Aided

TABLE OF CONTENTS

	Page
ABSTRACT	iii
DEDICATION	v
ACKNOWLEDGEMENTS	vi
NOMENCLATURE	viii
TABLE OF CONTENTS	ix
LIST OF FIGURES	xii
LIST OF TABLES	xiv
INTRODUCTION.....	1
A Vision	5
TAYouKi: A Teaching Assistant for Young Kids	7
RELATED WORK	11
Drawing and Writing Instruction	12
Educational Software	13
Educational Software Evaluation Methods	14
Computer Usage in Education.....	16
Sketch Recognition	17
Ink Digitizers	18
Approaches to Recognition	21
Template Matching	23
Gesture-Based Recognition.....	24
Geometric Recognition	26
Applications.....	28
IMPLEMENTATION	31
The Interface's Visual Structure	31
Recognition	33
Geometric Recognition.....	34
Symbol Recognizers.....	36

Triangle	37
Square.....	37
Letter A	37
Number 1.....	37
House.....	38
Person	38
Car	38
Template Matching.....	38
Agent Intelligence	43
Agent Personality	44
Question Navigation.....	46
Measuring Engagement.....	48
Feedback Generation.....	49
INSTRUCTIONAL DESIGN	56
Instructional Goal.....	57
Instructional Analysis.....	58
Learner Context and Analysis.....	59
Learner Analysis.....	59
Performance Context Analysis	60
Learning Context Analysis.....	61
Performance Objectives	61
Terminal Objective.....	62
Performance Objectives	62
Assessment Instruments	63
EVALUATION.....	64
Evaluation Overview.....	65
Sketch Recognition	66
Participants	67
SOUSA.....	67
Sketch Analyzer	68
Results	71
Computer Human Interaction (CHI)	73
Participants	74
Location Conditions	74
Pre-Instructional Activities.....	75
Step 1: Breaking the Ice	76
Step 2: Showing Them How Cool It Is to Draw	76
Step 3: The Importance of Geometric Shapes.....	76
Wizard of Oz Experiment.....	77
Results	78

Education.....	79
Participants	80
Pre-Testing	80
Activity Log	81
Post-Testing.....	82
Results	83
DISCUSSION	84
Sketch Recognition	84
What Is the Accuracy of Each Recognizer?	84
Which Recognition Approach Works Better for Our Case?	84
What Can Be Improved?	85
Is the Accuracy Similar for Kids?	85
What Features Are Most Relevant?.....	86
What Are the Major Challenges?	86
Computer Human Interaction.....	87
Was the Feedback Method Appropriate and Useful? (Sounds, Pedagogical Agent..).....	87
Do the Kids Find the Software Engaging? For How Long Do They Remain Engaged?	87
What Is The Children’s/Teachers’ Attitude Toward the Use of Digital Pens?	88
Was the Timing Appropriate?	88
Education.....	88
Are Kids Reaching One of the Defined Instructional Goals by Using Our System?	88
Does the Digital Pen Interaction Resemble Enough to Regular Pen/Paper ? (Delivery Method and Transfer Learning)	89
Were the Feedback Messages Appropriate and Useful?	89
Where the Questions (Shapes) Relevant and Easy to Understand?	90
General Discussion.....	90
CONCLUSIONS	91
REFERENCES	93
APPENDIX A: USER LOGS	98
APPENDIX B: ASSESMENT INSTRUMENTS	105
APPENDIX C: FEEDBACK MESSAGES	110
VITA	114

LIST OF FIGURES

FIGURE	Page
Figure 1 TAYouKi users. The system acts as a mediator between instructor and learner.	8
Figure 2 Interface of TAYouKi showing a correct answer	9
Figure 3. Different representations of the sampling process.....	19
Figure 4 Different examples of ink digitizing hardware	20
Figure 5 Representations of a drawing.....	22
Figure 6 Examples of two completely different gestures that look very similar	25
Figure 7. Open arrow composed of a shaft and 2 head lines	27
Figure 8 Different forms of drawing the same shape.....	28
Figure 9 TAYouKi Interface	32
Figure 10 Initial shape set	36
Figure 11. Re-sampled stroke.....	40
Figure 12 Activity diagram of the interaction flow of the agent.....	43
Figure 13. Different expressions of the avatar	45
Figure 14. The state diagram that rules the change of emotional state of the agent	46
Figure 15. Box diagram showing the system's inputs, processes and outputs	50
Figure 16. Answer Checker.....	51
Figure 17. Box diagram showing the sub modules of the Message Manager.....	55
Figure 18 Dick & Carey Model for instructional design	57
Figure 19. Goal decomposition	58

Figure 20 Different domains from which point of view we evaluated our system	64
Figure 21. SOUSA is a web-based tool for sketch data collection	68
Figure 22. Screenshot of the Sketch Analyzer	69
Figure 23 Recognition rates of each recognizer for the sample shape set	73
Figure 24 Wizard of Oz experiment.....	78
Figure 25 Instructor interface for the activity log	82

LIST OF TABLES

TABLE	Page
Table 1. Shape definition for an open arrow in the LADDER language.	27
Table 2. Variables used for engagement measurement.....	48
Table 3. Example messages for different message types	52
Table 4. Learner analysis summary.....	59
Table 5. Performance context summary.....	60
Table 6. Learning context summary.....	61
Table 7. Performance objectives	62
Table 8 Confusion matrix for the geometric recognizer	71
Table 9 Confusion matrix for the visual recognizer.....	71
Table 10 Accuracy summary.....	72

INTRODUCTION

Drawing is a form of visual expression that has shaped the history of mankind. From the rough sketches found on prehistoric caves to the contemporary pieces of art we find in modern art museums we find the innate desire of mankind to express and communicate visually. Drawing by itself is an important part of modern society, as it is the case with visual arts. But moreover, it is also a very important form of communicating ideas visually in many other domains. In the design stage of a varied number of projects in different domains, ideas are expressed and documented using hand-drawn sketches in a blackboard or notebook. Storyboards of a movie, floor-plans of a house, or layout of a webpage, are examples of designs that are usually conceived first in a piece of paper. Along with drawing comes the written word, were with a finite set of symbols drawn in a particular order we can communicate virtually any idea. Handwriting is still today a very common form of the written word, despite the technological advances such as the printing press, the typewriter and the computer word processor. The exercise of writing with pen and paper results very natural and gives less inhibitions for creativity than a regular word processor and a keyboard.

This thesis follows the style of the Association for Computing Machinery (ACM).

The significant importance of drawing is a reason why so much emphasis has been made to make individuals learn this basic skill at a young age. Although a writer and an artist can spend a lifetime perfecting these skills, the fundamentals are usually learned by people at the age 3 to 5. At this age, children not only learn to draw but draw to learn [2]. Drawing is a form of knowledge representation at this age as valid as technical papers and documents like this one are at a later stage of life. Drawing skills blend with those of early writing, where learning how to write can be seen as a process of gradually acquiring the ability to give separate meanings to the two forms of graphic symbolism, drawing and writing [9]. This creates a number of expectations about the drawing and writing skills a kid should have once finished preschool.

Unfortunately, these expectations are not always met. We can claim a number of reasons that cause the delay in learning these basic skills, but perhaps an important one is not using the right tools and delivery methods in the instructional material offered to these young learners. The rapid pace of technological advances has not only produced tools that aid in the critical task of teaching, but has also led to the declining cost of computers. As a result, more schools are widely adopting computers within classroom settings, and more children are achieving computer literacy. While technology is still no replacement for traditional human instruction, the advantages of highly-accessible highly-capable computers have created an interest from educators for tools and applications which complement and expand traditional instructional methods. From simple tutorials found on the Internet to full-fledged educational systems, a plethora of software exists with aims of augmenting the knowledge of students. Educators can

benefit from a wide variety of systems, from physics tutoring systems [45] to robotics training for astronauts [14].

One of the challenges we face in order to use computer software to instruct children at this early age, is that we cannot longer employ many of the traditional user interfaces which rely heavily on text. With the rise of multimedia capabilities in computers and mobile devices there is visible progress in this area and we can now use sound and animations to overcome this challenge. Many existing computer applications now use these multimedia capabilities to assist in reading instruction, but only a few projects incorporate an explicit focus on writing. We believe that this is mainly because traditional input methods such as keyboard and mouse are not appropriate for early writers. This is concerning as research in reading development has demonstrated the need for synchrony of reading and writing development for effective instruction, particularly for young learners [3]. If we are teaching children how to write we need them to actually hand-write instead of type. There is research that shows a clear link between drawing and practicing the formation of letter shapes and the solidification of the students' alphabetic knowledge [54].

Besides the challenge of being able to establish appropriate communication between the kid and the computer, we face the challenge of grabbing the attention of the kids and engaging them in continuous interaction. Children naturally possess only limited knowledge, but are quick learners in the initial stage of their lives. One of the major challenges that instructors face when dealing with children is precisely, keeping their attention. Therefore, it is important for instructors to engage children so that they

stay motivated to learn at this critical stage of development [35]. If we can come up with some interaction method that they find more appealing and involves them in the teaching process, as opposed to having them as passive observers, there is a higher chance of grabbing their interest to explore and learn. The early adults of today have grown up with keyboard and mouse, as previous generations grew up with a typewriter and showed stiffness going to a PC. Children on the other hand are open to new creative ways of interaction. And they may find it appealing to interact with computers in a different, more authentic manner.

Another important challenge comes when teaching multiple kids at the same time. Every child tends to be different across many important dimensions. This diversity in background and skills presents a big challenge to the instructor. Even if we have a limited scope for our users in terms of age and background the idea of having a completely homogeneous classroom is almost a utopia. Our challenge is to work closer to the model of differentiated instruction [26]. In this model each kid is provided with the content that goes with their current knowledge skill and ability. This is known to be very important, it is discussed in the literature that effective learning is achieved when a kid works in his particular *zone of proximal development* [47]. Although the use of this term is debated [4], we refer to it here as the set of skills that the children is not yet able to solve completely independently, but shows promise to do so when he successfully performs when guided by an instructor or a more capable peer. Ideally each kid should be provided with the particular content that works best for his learning stage. But the

logistics required to do this by a single instructor with more than one student are very difficult to meet with traditional methods.

As presented, we confront several challenges in this domain, including communicating with the kids, engaging them in learning, and providing them with differentiated instruction. In the next subsection we share with the reader a long term vision we have for this system, we follow by a brief description of the concrete system we developed and explain how we plan to tackle the before mentioned challenges.

A Vision

Tom is 5 years old. Many of his classmates and friends are already learning how to read and write, but Tom is having difficulties with it. It doesn't appear to be a learning disability, Tom is a very active boy and his interactions in class and with his friends denote that he is a very smart boy. However, he struggles with remaining focused during long periods of time and it is often that Mrs. Ferguson finds him doing something else besides paying attention. For Mrs. Ferguson, his Reading teacher, it is very difficult to monitor him all the time as she has many other students. And even when she tries to be innovative at the blackboard, by painting quick drawings and using different colors, it is exhausting for her to really sustain the attention of the young learners.

One day the principal announces to her that new equipment and software has arrived to the computer lab so she can use it with the kids. When she arrives to the lab she notices that the same computers she used often now have a pen input peripheral and they have installed new educational software that seems really engaging for the kids. After only 30 minutes of playing with the software, in both instructor- and student-mode,

she gets familiar with it and comfortable enough to decide to try it out with her class. She then proceeds to create a particular profile for each of the kids, based on their preferences and learning levels. She also decides to fill in the picture option in the profile with the yearbook pictures she took last month. So when the kids come into class they are happy to see their face in the screen. They start by playing a mini-game that the teacher instructs them to do, which helps them familiarize with the new input. They grasp the new program quickly and help each other out. Then they also have some time to play any game of their preference. They enjoy and laugh at the animations in the screen, while at the same time they are learning new skills.

Tom picks the “Draw the ABC” game. In this game, the avatar TAYouKi guides Tom to follow his strokes of each alphabet character on the screen. At first Tom cannot follow easily, but TAYouKi gives him positive and constructive feedback and advice on audio so he can repeat his strokes until he gets it right. Contrary to other classroom activities Tom becomes engaged in this one, as for each of his actions he receives immediate feedback from an interesting avatar on screen. Yet he is using the same motor skills that will allow him to write on regular pen and paper. After learning new concepts, while having fun, the kids leave the lab with a smile on their faces, and so does Mrs. Ferguson after seeing so much progress in their writing skills. After the class she stays to look at the activities done by the kids, happy to see Tom has now reached closer to mastery on writing both vowels and most of the consonants. However she notices he still has problems with the letter b and the letter p. So she configures the system such that next time Tom plays he will receive more emphasis on these letters. After successfully

working with the software in school, at the next parents-night, Mrs. Ferguson encourages parents to continue using the software with their kids at home, telling them that it can be easily installed and that it can be used with virtually any pen-input device or in case they don't have one even a regular mouse can be used to practice. After some time with the new system Tom has learned how to write all the letters in the alphabet and he even has a very beautiful handwriting. Mastering the shape of letters is one critical step in achieving the “alphabetic principal” which is a precursor to all formal reading tasks. Ms Ferguson takes a look at Tom's profile and follows all the progress he made feeling a deep sense of satisfaction.

TAYouKi: A Teaching Assistant for Young Kids

We base on the premise that an instructor or parent that can accompany the kid at this learning stage is fundamental. An exclusively computer-assisted instruction at this age is not only very difficult to achieve, but is also not encouraged. This, however, does not imply that we should discard the use of computers. We think they provide a perfect tool to assist the teacher or parent in this stage, where the kids can actively interact with the system while the instructor supervises giving a more complete and didactical instruction that can enhance the learning process.

Thinking of this premise we built a system that can be viewed as a *Teaching Assistant* for these *Young Kids*. Hence we call this system *TAYouKi*. We have two target user groups for our system. The first are young learners around ages 3-5 that are learning how to draw basic geometric shapes, letters and numbers. The other group is that of the instructors (teachers or parents) that are guiding the kids in this global process of

learning. Figure 1 shows the stakeholders or target users of the system and the relationship between them.

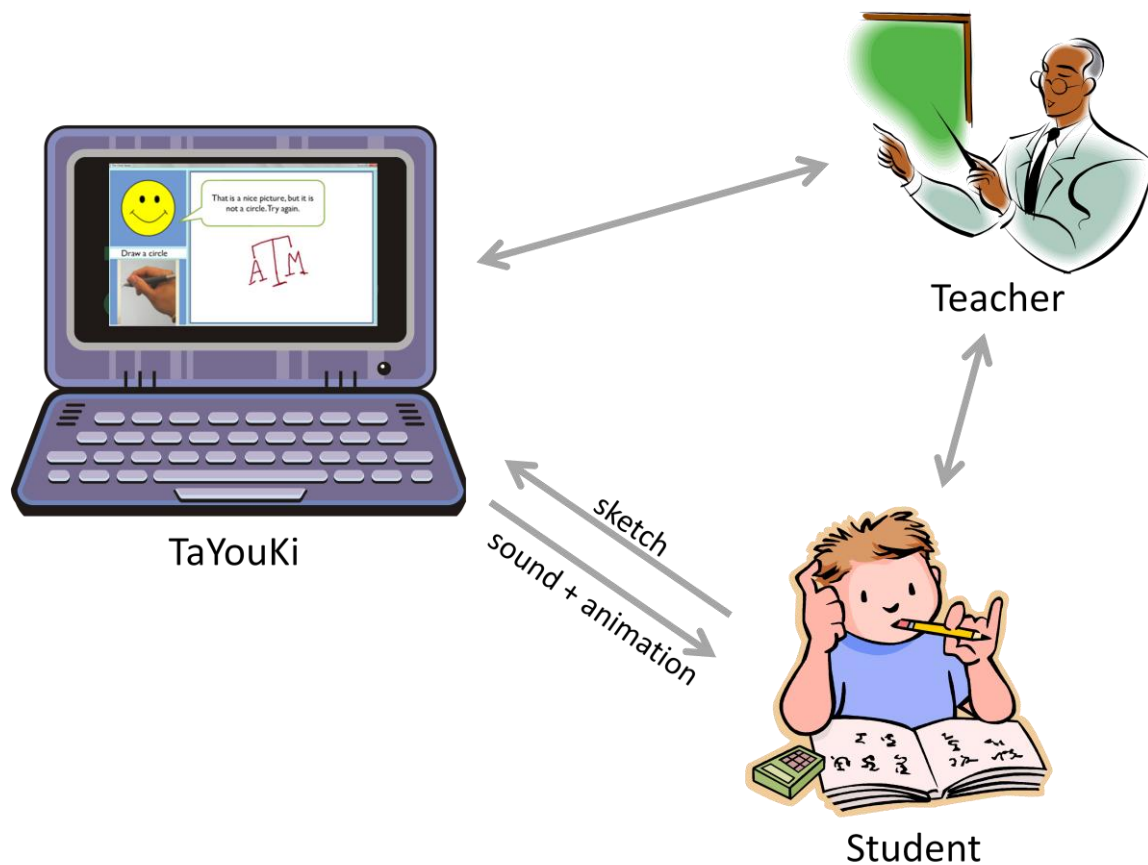


Figure 1 TAYouKi users. The system acts as a mediator between instructor and learner

Following the discussion in the previous section, we see the need of having a user interface where the kids could input information on the system where they yet do not master the alphabet well enough to use a keyboard, and that also provided them with the possibility of enhancing psychomotor skills that they will need in the future. The role

of physical objects in the development of young children has been studied extensively in the past. In particular, it has been shown that a careful choice of materials can enhance children's learning. Fortunately, we can rely on pen-input devices and touch screens to capture what the kids are drawing. In our system, we use advanced sketch recognition techniques to analyze the strokes made by the novice artists and writers, and after processing the information, the system gives them feedback based on what they have done. The feedback message displays itself as text so the instructor can follow, but more importantly is said out loud in the form of audio taunts so it can be understood by students. Figure 1 also shows this communication schema.

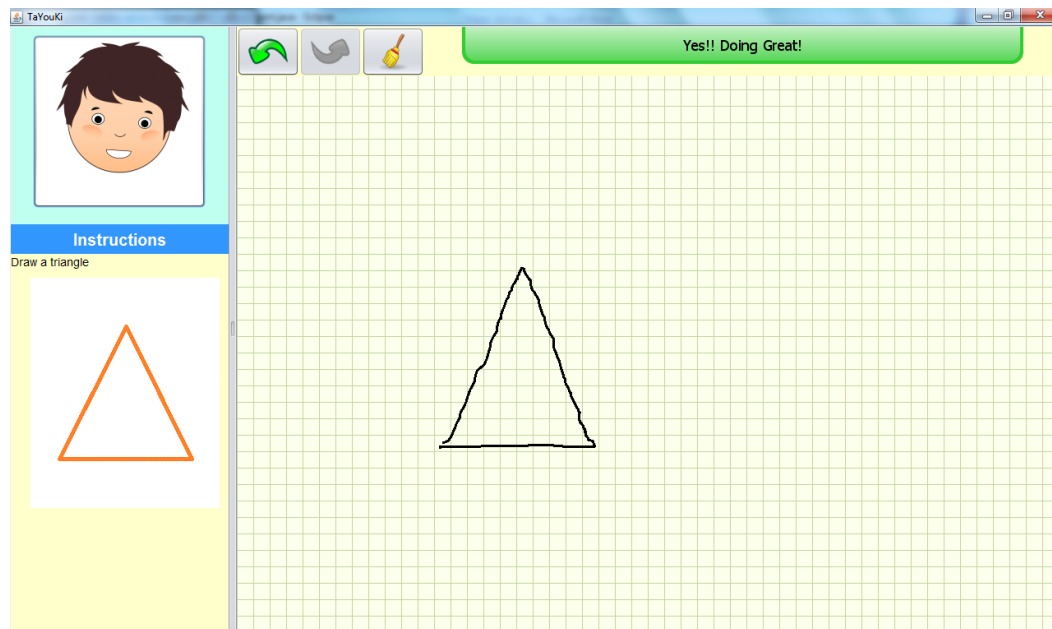


Figure 2 Interface of TAYouKi showing a correct answer

Another challenge we discussed was grabbing and retaining the attention of the kids, we do so by having a pedagogical agent at all times on screen that is providing constant feedback on what the student is drawing. The use of colorful animations and response to the interaction keeps the children interested in the activity. Figure 2 shows the main interface and an example of a correctly drawn shape. Finally, we designed this software to be an effective instructional tool. The system follows the principle of scaffolding by providing decreasing levels of help and increasingly complex questions. The system assesses the current performance of the students based on their interaction history and will then decide the difficulty level of the next question as well as the amount of help given in the instructions. The system gives the opportunity to instructors to tweak some parameters of the system so it better adjusts to the needs of the kids.

The remainder of this document goes deeper in explaining and analyzing this system. We first explore the related work that is relevant for the explanation and discussion of this system. Then we go into the implementation details of the current prototype. Following, we share with you a vision of a bigger system and explain how the current prototype fits into such system, and after this we explain the instructional design process that led to our development. When all the content has been said, we can explain how we put our system to the test, the evaluation procedures and the obtained results. Finally, we conclude with result analysis and some closing remarks.

RELATED WORK

Although the advantages of employing computers in the curriculum have been embraced by the academic community, is still challenging to reach and engage young children in educational software and have them enhance psychomotor skills such as drawing writing. This research problem has been explored by different communities and provides an interesting challenge where the use of multi-disciplinary knowledge is needed in order to arrive to an effective solution. For one side there is the academic community of psychologists and educators that have studied the cognitive issues presented in the instruction of this skill. On other side we have the instructional designers and educational software developers who now have well founded models that can be used to develop instructional material for kids this age. Very closely, we have HCI researchers who have studied in general the design of user interfaces that result appropriate for effective and intuitive use of computers. And given our approach to the proposed solution, we are also very interested in the research area that studies the algorithms and techniques required to interpret the drawings of the user, namely, *sketch recognition*. This multi-disciplinary problem requires some background on each of the mentioned domains before we get to a particular solution. This section, does not intend to provide an exhaustive survey in all of these fields, but provides some relevant examples in each of them and links such examples to our current work.

Drawing and Writing Instruction

As grownups, we read textual information, write notes by hand, or sketch shapes and annotations on a daily basis without hesitation. These skills are so well mastered for us that we do not pause to think about the cognitive process that happens in our brain to achieve these tasks. They happen automatically and it results hard to remember know how we learned them. At what point did we go from not being able to recognize any letter of the alphabet to being able to read words and sentences until we gave meaning to entire paragraphs and stories? Most of us do not remember the details of how we became fluent readers because, in fact, we rarely drew conscious attention to our learning [3]. Many of these skills are accomplished tacitly, without conscious reflection. But these does not mean conscious instruction is meaningless. Giving the appropriate instruction at the appropriate time with the appropriate materials is a relevant challenge that drives the field of instructional design [6]. Teaching how to draw and write fits in the general model of designing instruction but has its own particularities and cognitive issues.

Yang and Noel [54] made a very interesting study of drawings made by children in the age 4 and 5. In this work they sampled the drawings of kids at two points in time over the course of a year to kids in this age range. They found that horizontal and vertical lines were very common; at both ages and that circles or spirals were less common. Multiple lines in the form of scribbles were increasingly popular in the age range. Over tracing is fairly common in their random drawings. Significant progress was made in the writing skills at this age, having most of the kids from not being able to write anything to write their own names at the second sample. The results of this work

are very significant for this project, as they can give us an insight of the shapes we should emphasize and the risks we need to mitigate in recognition.

Educational Software

Since the popularization of computers, educational software has grown to be a significant market that takes advantage of the computers as a powerful tool for teaching any kind of skills. Developers of educational software can now utilize a wide assortment of multimedia in order to provide entertainment value as a way of capturing the interest of students. And the use of virtual tutors and pedagogical agents has become fairly common.

There is a notable genre of educational software applications blending educational elements with an entertainment factor are often labeled as being edutainment or learn to play [34]. Positive contributions of edutainment software can be seen including research studies which identified that the core attributes found in interactive software programs that make them ideal for educational purposes [19]. Among those positive contributions are the capabilities of these to attract and grasp the attention of children and to retain their flow of concentration and promote to the development of good learning by stimulating creativity.

An important appeal of edutainment software is its applicability to students of all ages, and generally the key concepts of lessons hold for any age of the user. For example there is a system called U table, a novel interface and software application which allows mature-aged users to interface through games for a more natural experience while they are learning critical skills [20]. This work in particular demonstrates the importance of

having sufficient and adequate execution of human-computer interaction in order to appeal to the correct demographic and attract them into learning while playing.

Children generally remain the primary focal point in the development of edutainment software, and examples of such software include Music Journey, a novel program for teaching children how to play music. This software application was successfully employed in public schools within the United States, and important results from this application have shown that it effectively teaches music skills while maintaining low cost overhead and lacking the requirement for the expertise of a physically present human instructor in the subject [22].

The creation of virtual tutors that control the flow of interaction between the learner and the instructional material appeared parallel with the edutainment phenomena. With around four decades of research in this field, the area of knowledge has adapted many names and scopes, such as *CAI* (Computer-Aided Instruction), *CBT* (Computer-Based Training), or *CAL* (Computer-Assisted Learning). But latest research commonly converges to the term and concept of *ITS* (Intelligent Tutoring Systems) [53]. In this model, the system tries to emulate a real tutoring environment, grabbing all the main elements that make teaching effective. Some of the general concepts found in the literature of ITS were adopted in this system. More details will be explained in the implementation section.

Educational Software Evaluation Methods

Like other types of software, educational software must adhere to evaluation methods in order to achieve strong design and effective human-computer interaction. Unlike other

types of software though, educational software in general possesses its own particularities. Researchers such as Shiratuddin et al. expose evaluation methods which are lacking in the selection of appropriate activities for users of educational software, specifically the kind that focus on instructing children [39]. Observations from this discovery include the fact that the usefulness of the software from an interface, while necessary, is not sufficient, but that it fundamentally must actually possess pedagogical aspects for allowing children to learn. Moreover, a set of activities that a system should include was determined in order to stimulate knowledge in at least one of the identified learning styles. For the case of evaluating educational software for second language instruction, data from a set of interviews was collected and analyzed; from the analyzed data, categories were discovered that were determined to require taking into account when designing educational software [5]. Alternative usability evaluation methods were developed which employed more scenario-based design. A particular instance of those alternative methods was successfully used in a program called Children's Heaven to detect many of the bugs and problems in the early stages of developing that software [49].

As we see, there is a wide variety of educational computer games in the market. Some of them are very good and prove to be valuable in learning; however, they are usually generic games that do not take into account the current set of acquired skills and knowledge of the kid. Another common limitation of these games is that the interaction methods used are limited to the traditional input methods of personal computers, namely mouse and keyboard. Today there is a wide range of accessible peripherals that may be

more appropriate for children to interact with this type of system. Sketch and touch input is beginning to be explored by researchers but there are few systems that interpret user drawings to provide a more meaningful interaction as our proposed system does. We have found in the recent literature programs that take a similar approach as TAYouKi to teach literacy skills. We will discuss these examples in detail after we have provided the ground for discussion in the Sketch Recognition section.

Computer Usage in Education

Because participating students will have much opportunity to work with cutting edge learning technology, we feel it is important to highlight the use of computer science in education more generally. Educational software is becoming of common usage around the United States and the whole world. By 2005 more than 12 million computers were being use in public schools for educational purposes [25]. And this number has been growing and will likely keep growing over time. Both at educational institutions or at home people of all ages can benefit from educational software. Although it may not replace entirely the experience of being in a classroom it surely reinforces positively the learning process. There are studies that confirm that the use of technology has the potential to augment substantially the outcome of students [24]. However it is not enough to count with the right hardware tools in the classrooms around the world. It is also fundamental to have specially designed software that adapts to the needs of teaching. The adequate design of educational software is essential to the success of using technology in a teaching environment. Our use of computer software is unique because it

allows students to learn in an interactive environment, with a novel interaction technique and keeping the instructor in the loop to achieve truly differentiated instruction.

Sketch Recognition

The last four decades have seen amazing advances in computer software. During the first half of the past century people sent each other hand-written letters, documented knowledge using type-writers, and used rulers and protractors to draw technical drawings. The appearance of word processors, CAD systems, web forms, e-mail and other technologies heavily changed the way we communicate and express to each other. Yet, even in a world with all these advances in technology we experience numerous situations where people prefer to use hand-writing or hand-drawing as opposed to a keyboard or a mouse. This is particularly true in early stages of a cognitive process where the ideas that brainstorm into our minds seem to interact more naturally with a pen and a blank piece of paper than they do with a new word document or computer design tool. It is often the case that we make rough drafts in paper before using the computer to create more formal versions.

One of the reasons for this to happen is the constraints computer impose in their input methods. Using a mouse and a keyboard, and following layout conventions results very inconvenient for creative stages of design. Fortunately hardware and software advances have opened a new world of possibilities for new input methods that might result more natural. Particularly the ability of drawing ink strokes in a computer screen is made possible by different technologies that we will discuss in a later section. The

automated recognition of these strokes into meaningful symbols such as letters, numbers or shapes is a field that has been denoted as *Sketch Recognition* [13,32,46].

Ink Digitizers

In order for the computer to understand the drawings made by humans the first step is to convert these drawings into some sort of binary representation. In the case of a scanner or a digital camera, the drawings are interpreted as plain images and can be stored into a bitmap and further compressed into an image format such as jpeg, gif or tiff. But the form of digitizers that we are really interested here are those that enable to capture strokes in real time, emulating the use of pen and paper. These give us the possibility of creating a system such as this one where children can receive real-time feedback while having a look and feel very similar to that of pen and paper.

There is debate in the use of the term digitizer, but for the sake of clarity let us define an *ink digitizer* as a hardware device that has at least the following properties:

- Users can draw ink strokes on a screen using it.
- It has clearly defined pen-down and pen-up events such that it only draws whenever the pen is down.
- If the pen goes down at time t_0 and goes up at time t_f , the device will *sample* the position of the pen at different times t_i , where $t_0 < t_i < t_f$. A *sample* is a recording of the x-y position of the device relative to the screen at a given time. A *sample* will be taken every Δt between t_0 and t_f . A graphical description of this process is depicted in Figure 3.

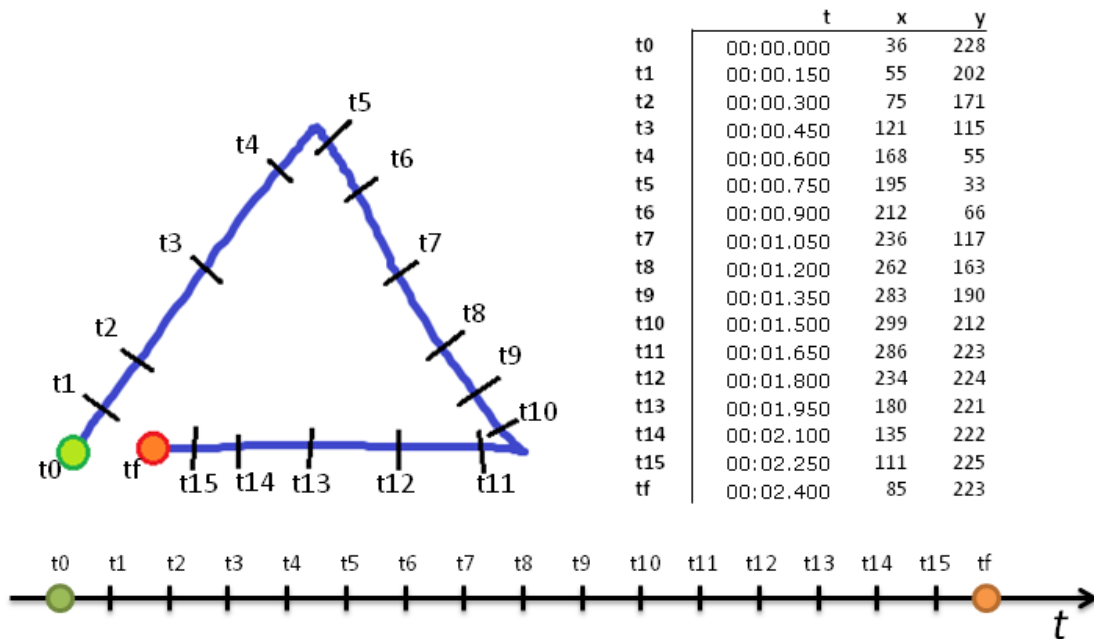


Figure 3. Different representations of the sampling process

Note that although these are the minimum requirements many ink digitizers have more information such as pressure or tilt. In this section we will see several examples of ink digitizers that comply with this definition and thus serve as input forms for our system.

We cannot really say this technology is new, pen input devices have been around for quite a while; they were actually conceived before the mouse. Systems such as the *SketchPad* [42] or the *Dynabook* [33] used a pen as their main source of input. However, different reasons such as costs, or response time of alternative forms of input made of the pen a less attractive option than a mouse and its use became shadowed for many years. However the advantages of the natural feel of pen and paper kept in the background of many researchers and hardware companies giving rise to what has been

denoted as *Pen Computing* [23,41]. In the 90's we saw a growing interest in pen computing with popular pen-based-hand-held devices such as the Palm Pilot [12] and operative systems such as the *PenPointOS* or the *Newton* [11] that were designed with pen-input in mind. Although these technologies became very popular they were not generally embraced by the whole community as the keyboard and mouse, however they laid ground for new developments. The last decade saw the boom of Smartphones and tablet PCs with embedded pen or touch input capabilities which urged for the inclusion of native support for these new forms of input in the mainstream operative systems.



Figure 4 Different examples of ink digitizing hardware

Currently there are several hardware mechanisms that allow strokes to be drawn into a computer screen. Some use special pens with electromagnetic resonance such as the Wacom screens [43]. Others use multi-touch capacitive technology or vision-based mechanisms. And sizes go from giant screens [40] to mobile phones [37]. Figure 4 shows some examples of ink digitizers.

We want to close this section by emphasizing that despite the input mechanism most of the concepts explained in the remainder of this document apply. And that our system is independent of the hardware employed. However, given the application of teaching how to write, a pen-input device is recommended as it will bring the student closer to the performance context.

Approaches to Recognition

There are different approaches and algorithms for Sketch Recognition that have been exposed in the literature, so far there is not an all-purpose-best-performing Sketch recognizer that can be applied in any domain maintaining superiority on its performance characteristics. Different approaches have their own advantages and disadvantages. In this section we will sweep through some of the most common techniques. We want to note that because of the importance of the written word, *handwriting recognition* is a field of sketch recognition that sometimes is considered on its own as a separate research domain. However, both sketch and handwriting recognition share most of the same principles and their preprocessing steps and general approaches are often the same. For the remainder of this section we will talk indistinctively about these two problems unless a relevant particularity arises.

One common categorization of sketch recognition approaches bases on the way the image is acquired, or more exactly on the available digital representation of the sketch at the moment of recognition. Based on this criterion we usually talk about *Offline Recognition* and *Online Recognition*. The former is concerned with recognizing characters and shapes that have been scanned and converted into a digital image in binary form. The later is concerned in interpreting the strokes drawn used specialized hardware as described in the previous section. Note that one crucial difference between these two is the availability of time information. In online recognition the location of the pen is sampled every certain time so we have as an output a set of points in x, y and t. Visual examples can be seen in Figure 5.

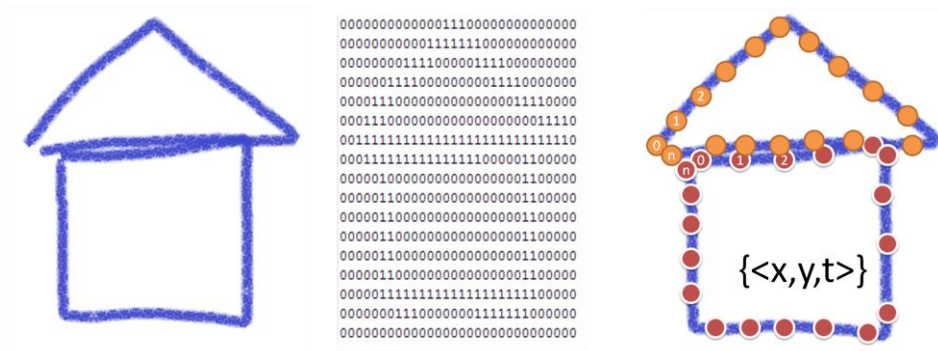


Figure 5 Representations of a drawing. a) Original drawing b) binary form c) sampled points

In offline recognition we depart from an image that is in its final form before attempting recognition. As opposed to other computer vision problems, for shape and character recognition we do not care much about color or grayscale information and

instead we want a black and white binary image that is as clear as possible. This implies a series of preprocessing steps for noise removal and thresholding when text is scanned or gathered from another analog form [31]. When text is already in digital form as when captured by one of the devices mentioned in the previous section the preprocessing may be omitted. Note that in this case we can still use offline recognition methods, where the only difference is if we decide not to take into account some features captured by these devices such as time and pressure information.

For our scope we will be based on the assumption we are sampling ink with one of the online capturing devices. Therefore we can use both online and offline approaches to recognize our sketches and we are not concerned with problems such as thresholding or background noise removal. Within this scope we can further categorize the recognition approaches into three main categories: *gesture-based recognition*, *template-matching* and *geometric recognition*.

Template Matching

In template matching, the recognition will be based mostly on the appearance of the drawn sketch. The input of the user will be compared to a set of examples and using some sort of comparison the recognizer will determine which one it resembles the most. Difficulties appear in template matching if we attempt to compare the sketch of the user directly with the templates. Even if they look alike they are not likely to have the same location, scale or rotation. Different methods have been proposed to normalize the input image before comparing it to the templates. A recent one that has been very well accepted because of its simplicity is the *\$I recognizer* [51]. This method is an online

method as it uses temporal information to resample and to establish the start-point of the shape. It then rescales and rotates the shape to match it against the best template. The clear advantage of this algorithm is its simplicity in implementation and training; however it has some disadvantages such as its running time that grows with the training set and the fact that it only supports single-stroke gestures. Other template matcher example is that of Kara and Stahovich [17]. This recognizer has the advantage that can be also used for offline recognition and that it supports multi-stroke shapes it is based on a distance metric of each point of the input sketch to other points in the matching template. A modified version of Kara and Stahovich recognizer was implemented by Valentine [44]. Both of these works will be explained in greater detail in the implementation section.

Gesture-Based Recognition

Another popular approach for the recognition of sketches is commonly known as gesture-based recognition. This type of recognition relies on feature extraction out of the position and timing information of the shapes to classify them accordingly using some sort of classifier. It is called gesture because of its time dependence, where the direction we draw is relevant for recognition as when we are making a gesture with our hands. Figure 6 shows an example of two gestures in the form of a triangle. Notice that despite the gestures look very similar they are actually to very different gestures because of the order we used to draw.

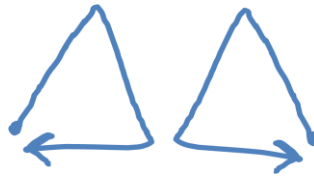


Figure 6 Examples of two completely different gestures that look very similar

One of the most cited works in this area is that of Rubine [36]. At the core of rubine's recognizer lies feature extraction. A feature is a single numeric value that can be extracted statistically out of the raw data provided by the *ink digitizer*. An example of a feature can be the average speed of a stroke or the width of the bounding box of the sketch. Ideally a feature should be cheap to calculate (constant time per input point) and meaningful to the recognition of the shapes. Rubine defined 13 meaningful features that can be extracted by any single-stroked gesture. He then proposed a way to determine from a stroke to which class (shape) it belongs. A linear classifier is used to determine the classified shape based on a vector of weights for defined for each target shape. The mentioned weights moreover do not have to be calculated by the programmer but instead they can be trained by a set of examples for each class.

After the work of Rubine, many other gesture-based recognizers appeared. Because of their simplicity and their performance these types of recognizers were very accepted as an input method for pen applications. However, in most drawing domains the stroke order is unimportant, which means that in order to use this approach a sample class or shape has to be defined for each possible stroke order, which is undesirable and can quickly become unmanageable in multi-stroked scenarios.

Geometric Recognition

The last main form of recognition we are mentioning here is geometric recognition. In this type of recognition shapes are recognized in a hierarchic manner based on their geometric properties. A first level of recognition is made to identify simple shapes such as lines, curves or circles. These atomic shapes that cannot be further decomposed are called primitives; all the other shapes can be described as a particular composition of primitives. The main advantage of geometric recognition is that it can describe shapes based purely on the primitives that compose them and the relationship between them. This allows for a great flexibility in the way the user draws, since depending on the constraints the shape can be drawn in different scales, rotations or variations.

Geometric recognition presents its own challenges such as corner finding or segmentation [52] and primitive disambiguation [38]. It is also challenging to describe the relationship between the primitives and the constraints that affect them in a natural manner. To overcome this problem Hammond introduced LADDER [13] which is a language that uses a natural vocabulary to describe, display and edit new shapes in a particular domain. This work shows that it is feasible to create compact and natural representations of new shapes using geometric recognition. Table 1 shows an example of how to describe an open-head arrow. And Figure 7 shows the corresponding image.

Table 1. Shape definition for an open arrow in the LADDER language*Shape definition for an open arrow*

<pre> components Line shaft Line head1 Line head2 constraints coincident shaft.p1 head1.p1 coincident shaft.p1 head2.p1 coincident head1.p1 head2.p1 equalLength head1 head2 acuteMeet head1 shaft acuteMeet shaft head2 </pre>

**Figure 7. Open arrow composed of a shaft and 2 head lines**

Before we move on to the next section we want to conclude by summarizing and emphasizing the advantages and disadvantages of each approach to recognition with concrete examples. Figure 8 shows our example of a house shape drawn on different forms; all of them are perceptually acceptable for a human but the recognizers encounter particular troubles in them. Figure 8 c) for example will not be recognized by a multi-stroked gesture based recognizer if the template is defined as in Figure 8 a). Note that although the shapes are similar in appearance their stroke order is different, likely

causing the recognizer to fail. Similarly Figure 8 c) might not be recognized by an appearance based template matcher, note that even if we rescale the shape, the roof (triangle) of the house does not maintain the proportions as the house described in the template of in Figure 8 a). In this case both gesture and geometric recognition have better chances of correctly identifying the shape. Figure 8 b) shows a case where geometric recognition fails if we only consider the shape hierarchy defined in Figure 8 a) (Triangle + square). The shape is visually correct and technically has the right components, but the triangle and square share an edge, causing the geometric recognizer to capture the wrong set of primitives (a square and a polyline of size 2).

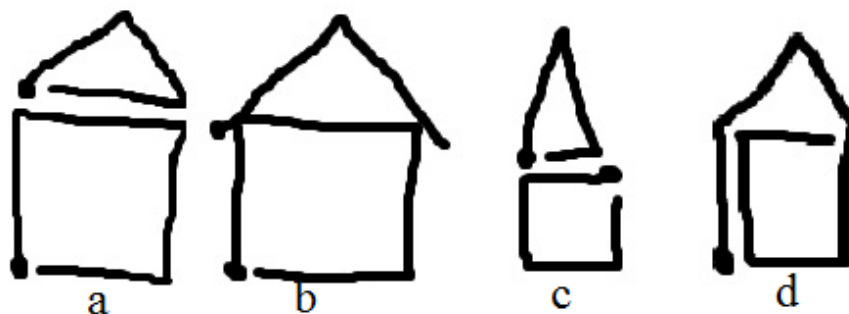


Figure 8 Different forms of drawing the same shape. Each one presents different challenges to each approach of recognition

Applications

Games that incorporate a sketching component have been explored to a certain extent in systems such as the Telephone children game-inspired system found in Picturephone [16]; the object of the game is for one play to describe a picture while the

other player draws their own interpretations of those descriptions. Other sketch-based systems have incorporated more of a subject-specific approach for their games as educational software, such as a system developed by Wang, et al. which presents a pen-enabled teaching system for young children [50]. The authors for that system were motivated in creating a more immersive system for the instruction of a foreign language from traditional tools and resources, and their system specifically provides novel and entertaining sketching interactive activities for young children to experience.

In addition to teaching specific classroom subjects, sketch-based educational software has also been developed to teach specific technical skills. One such system which uses sketching interaction called iCanDraw provides tailored feedback based on a user's sketched input for teaching how to draw realistic human faces [8]. The feedback in iCanDraw is controlled from interactions conducted in a step-by-step manner, and additional feedback is provided when the system detects that the user is completed with the sketch; if a user is stuck in a particular step of the sketching, the system also provides helpful feedback for accomplishing that step before moving on. There is also research oriented to draw mathematical expressions using a sketch-based education tool. Joseph J. LaViola Jr. introduced MathPad² [21]. Expressing a mathematical expression in a computer requires more effort than regular plain text. To alleviate this problem, the author used sketch recognition to recognize mathematical expression, and diagrams.

Applications that are closer to this work have also been developed recently. Pereira and her team designed and implemented IWA in 2009 [30]. IWA This work resembles the one presented here in many aspects. Its main purpose is to enhance the

drawing and handwriting skills of young children like is the case of TAYouKi, and as TAYouKi, it also uses sketch recognition to provide some kind of feedback. They also have an instructor side of the program that allows teachers to enter new letter templates and create new assignments. Following the work of Pereira, Alvin [1] created a simpler program to study the use of an OCR as an evaluator for their work. They used an opensource software called NeuronDotNet [7] which employs a neural network engine to train and classify different types of recognizers, in this case an optical character recognizer. As opposed to these works which employ an offline-style template matcher for recognition, TAYouKi provides an additional geometric recognizer which gives some advantages as explained in the Sketch Recognition section. Paulson [27] also used geometric recognition to teach children how to draw shapes. He used an earlier version of the same low-level recognizer we are using here. In this work Paulson only supported single stroked shapes and the feedback provided was mostly in terms of right or wrong.

With the use of different sketch recognition techniques combined with our help level evaluator, which is also novel in our work, the children can follow scaffolding principles and go from closely guided instruction to a more free form of drawing as they progress. In this way we reinforce learning and are able to give them tailored and relevant feedback at various stages of drawing and writing development.

IMPLEMENTATION

Now that we have established enough background for this work we can move into the implementation details behind our current prototype. In the TAYouKi system, children move through a set of questions that have varying complexity ranging from the drawing of basic primitive shapes to more shapes consisting of composite primitives. The tutoring agent tracks system events, such as a pen-up, pen-down, or timeout, and keeps track of the current state of the interaction to contextualize the feedback it will give. After a pen-up event, the agent attempts to recognize the shapes drawn by the child, and compares them with the expected input. The agent generates tailored and intelligent feedback, appropriate for the child's current state of progress. The following subsections give more detail on each of the parts that compose the TAYouKi system.

The Interface's Visual Structure

Our primary focus for the interface is simplicity. We use few buttons and little to no text to avoid clutter that would otherwise serve as a distraction. On the other hand, we make use of color and audio cues combined with limited animations that may be important to keep children engaged. There is a fixed space for the avatar so the children will become familiar with the avatar. The visual appearance of the interface is divided into different panels that serve various functions, as shown in Figure 9.

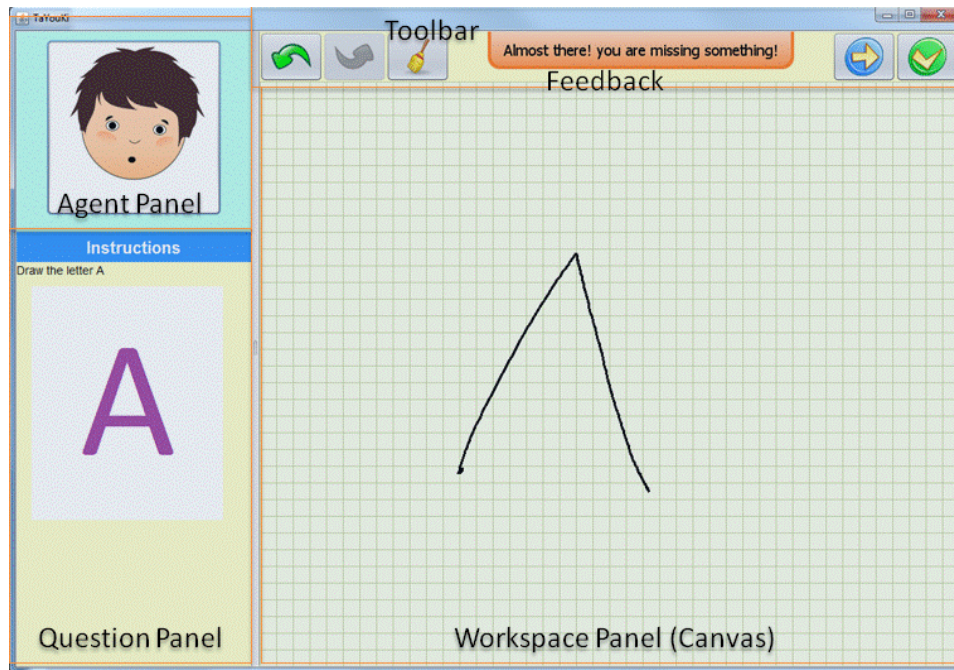


Figure 9 TAYouKi Interface

The **Question Panel** displays the instructions for the current question. These instructions are shown both in textual and image form. We provide vivid visual instructions composed of both static images and animations that, together with the audio prompts at the beginning of each question, result in more captivating and accessible interaction for our target users. The images and text that are shown for each question change depending on the current state of the interaction.

Users interact with the system through the **Workspace Panel** or Canvas. The panel consists of a drawing canvas for sketching an answer to the agent's instructions. Right on top of it there is a simple **Toolbar** for editing the sketch with options to undo, redo and clear. At the right hand side of this toolbar there are buttons to control the

interaction, by skipping a question or giving the agent the go to attempt recognition right away. This can prevent waiting for feedback after a timeout in multi-stroke shapes.

The **Feedback** box shows the result of children's drawings and the agent's feedback responses. This panel slides into view upon the completion of a sketch. The back of the panel is colored in green or red to make the feedback that immediately comprehensible with low cognitive overhead.

The **Agent Panel** displays a visual and emotional representation, or avatar, of the agent. The avatar is always present, acting as a companion, guiding him in the process. At anytime children can click or tap on the agent which will be equivalent to clicking on the checkmark button, causing the agent to give feedback right away. The agent's feedback responses and emotional state can vary depending on the current state of the child's drawing and the child's recent performance.

Recognition

At the core of our system lie the sketch recognition techniques we use to interpret children's drawings. Ideally TAYouKi users will use a stylus as primary input; however, as we discussed in the related work section, any ink digitizer will serve the purpose. Because of the advantages and disadvantages we discussed in our related work for each type of recognizer, we utilized two separate recognizers with distinct approaches, geometric and template-based. How we merge the output of these two subsystems will be explained in the agent's intelligence section.

Geometric Recognition

After we obtain the raw data from the device we begin our preprocessing steps. The system first pre-processes the stroke data to reduce noise and avoid false recognition. This is, getting rid of most likely unintended parts of each stroke. Second, the system converts each stroke into one or more primitive shapes. For this preprocessing steps we are using a low level primitive recognizer called PaleoSketch [28]. Paleo nicely integrates several techniques of preprocessing and primitive recognition such as corner finding and geometric perception to perform a series of possible matches over the supported shapes. Paleo then uses a novel ranking algorithm to determine which of these shapes has a better fit. The current version of Paleo supports more than 10 basic shapes including polylines, curves, arcs and circles with a reported accuracy of more than 98%.

A key feature of our system is that we provide sound feedback upon low-level recognition. Notice that this happen on pen-up events as opposed that on clock events like the high-level geometric recognition or visual recognition. Each primitive maps to a unique sound that is played to the user as soon as the primitive gets recognized by paleo. For instance, a line will make a *“pshh”* sound, a poly-line of size two will sound like *“psh-psh”* a circle will sound *“bauum”*, a spiral *“wiii”* and a larger poly-line (more than 5 composing lines) will sound *“pshohhh”*. This provides the kids with attractive and engaging feedback while allowing them to early identify problems with their drawings before attempting high level recognition. For example if they are drawing the wheel of a car and they hear *“bblok”* instead of *“bauum”* they could tell immediately that the wheel

of the car looks more like a square than like a circle, and they can correct this right away.

We make use of these primitive shapes as an input to domain specific high-level recognition algorithm, where we use context and the relationship between primitives to infer high-level recognition of more complex shapes. For each complex shape we have a particular recognizer that first verifies that it has the necessary primitives and then proceeds to analyze the relationship between them in terms of positioning, size and relative orientation. We use the context provided by the agent to optimize the recognizer for the current question. Particularly we take the meta-data of the current question to determine the minimum and maximum expected number of strokes for each shape. Note that a composite shape such as the number one shown in Figure 10f can be drawn in different ways using one to three strokes. We can use this meta-data to limit the recognition attempts up to a certain number of strokes (three in this case). Similarly for complex shapes like the *person* shape in Figure 10i there is a minimum number of expected strokes to draw it on a natural fashion without over-tracing.

Since we have multi-stroke recognition we need to take into account the complete set of strokes in the sketch. Moreover children often repeat their drawings over and over on the same page, as opposed to clearing the canvas or erasing the previous shape. We attempt to group the strokes in the sketch drawn by the children into different symbols. A symbol is a shape that is semantically meaningful and can be labeled such as a circle, a house, or a car. Note that some symbols are hierarchically built upon simpler symbols, for example the symbol car contains two circles. We built at least one symbol

recognizer for each of the supported shapes. For each symbol recognizer s we take the n strokes from the submitted sketch and try to group those in all groupings of size r_i , which is the number of components required by the symbol recognizer i . We repeat until we cannot find any new recombination. This process is bounded by the equation:

$$n \sum_{i=1}^s \binom{n}{r_i}$$

Symbol Recognizers

Figure 10 shows the currently supported shapes of our prototype system that serve as a proof of concept and future shapes can be included if needed. As shown in the work of Hammond [13] new shapes can be described in almost natural language to expand our question set with too much effort. The line, circle and spiral are primitives already supported by Paleosketch [28]. We now give a brief description of the particular recognizers implemented for the remaining shapes.

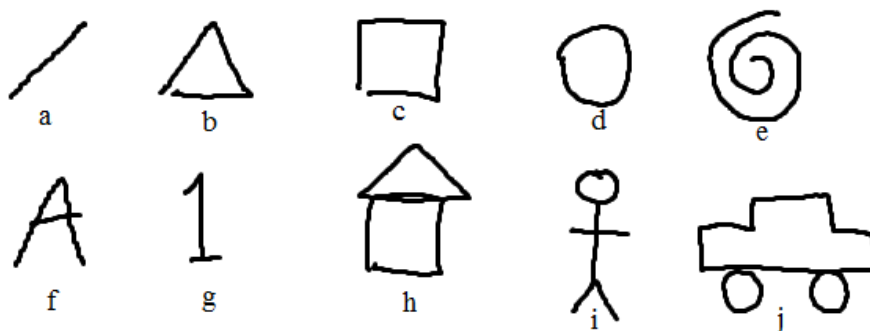


Figure 10 Initial shape set. a) Line b) Triangle c) Square d) Circle e) Spiral f) Letter A g) Number 1 h) House i) Person j) Car

Triangle

We expect up to three strokes that can be decomposed into three simple lines (e.g. polyline (2) + line). We check for each line that its endpoints meet with exactly one endpoint of the remaining lines.

Square

Paleo natively supports a square recognizer, however it has to be single stroked. We add a multistroke square recognizer using the same principle as in the triangle but requiring four lines.

Letter A

As in the triangle we check for the lines. We sort these by their angle with respect to the horizontal line and check that they have respectively a positive, negative and a near 0 slope. We check that the two lines with high magnitude slopes meet at their topmost endpoint. We check that the low magnitude slope line is about at half height the bounding box of the other two and intersects them.

Number 1

The approach is similar to that in A, but we start by checking that one of the lines is relatively much longer than the other two and that it is relatively vertical. The other two must intersect this long one on bottom and top and be relatively perpendicular to it. The top one should be at the left of the main line and the bottom one centered. We take into account the case of short over-tracing for single strokes omitting the extra line at the bottom.

House

We attempt recognition of a square and a triangle independently as the figure is drawn. We group the strokes in case one of these shapes is recognized. We then check that the triangle is on top of the square and that it “points up” meaning that the lowest slope line of it is at the bottom.

Person

For this person or stick-figure we also check for primitives, seeking a circle that is on top of the other shapes, a relatively long vertical line, and lines making a caret ^ and a horizontal one such as the ones we seek for in the letter A, just that now the caret should be below the horizontal with a distance of at least 0.25 times the length of the vertical line. All of these must be horizontally centered and the vertical line should start approximately below the circle and end above the caret.

Car

In the case of the car we need two circles vertically centered at the bottom which bounding boxes do not intersect. Above this we need multiple line segments making up a closed shape. We check this with a similar approach as in the triangle and square checking the meeting of the endpoints. Note that this criterion do not constrain the figure to be exactly as the one in Figure 10 as long as we have a closed shape made out of 7 or 8 simple lines we account it as valid.

Template Matching

The second type of recognizer we use is a template matcher. As explained in the related work section, a template matcher bases on the visual appearance of the input

shape to perform recognition. We think this is a desirable property due to the fact that children sometimes perform unpredictable ways of drawing a shape, that do not follow a process of conventional geometric standards, but that their final product shows a shape that fairly resembles what was expected. For our template matcher we used a slight variation of the recognizer developed by Valentine [44], which in turn is based in that of Kara and Stahovich [17]. This recognizer has some interesting properties such as the scale invariance and support for multiple strokes. We briefly explain the functioning of this recognizer next.

For each of the shapes in our system (Figure 10) we have around 10 templates. A *template* is a sample drawing of the image that is stored as a file using the same structure as any other drawing in our system: a series of two-dimensional points in time (See Figure 3). We want to compare the submitted drawing to all the templates in each shape and try to find the best match for classification. Note that in this case we are interested in the template that *looks* more similar to the submitted drawing, independently of how it was drawn (speed, stroke order, scale...). For this purpose we first go through a process called *re-sampling*. Remember from the ink digitizer section that we explained how the strokes drawn by the user were sampled in time to create a list of points, each having a time value, a x value and a y value. Remember as well, that these points are sampled at a fixed frequency meaning that every Δt there will be a point and the number of points in a line segment will not depend purely on its pixel length but also in the drawing speed. For instance, a line segment of constant pixel length can have fewer points if it is drawn fast, than if it is drawn slowly. Also segments where the user accelerates or decelerate the pen

are likely to have higher point density. In the re-sampling process we are interested in making the points approximately equally evenly distributed by the stroke length rather than the time information, meaning that two lines of the same pixel length should have the same number of points and these should be spread evenly along the drawing path. Also we want to resample to a fixed number of points to be able to compare templates independently of the drawing speed. Figure 11 shows a visual example of a stroke that originally has 10 points and is re-sampled to 8 points. Note that the figure on the right has the sample points more evenly distributed visually. Also it is relevant to note that when we choose the number of points to re-sample we need to be careful: a number that is too low might result in loss of information, one that is too high might increase time complexity as we will see later on.

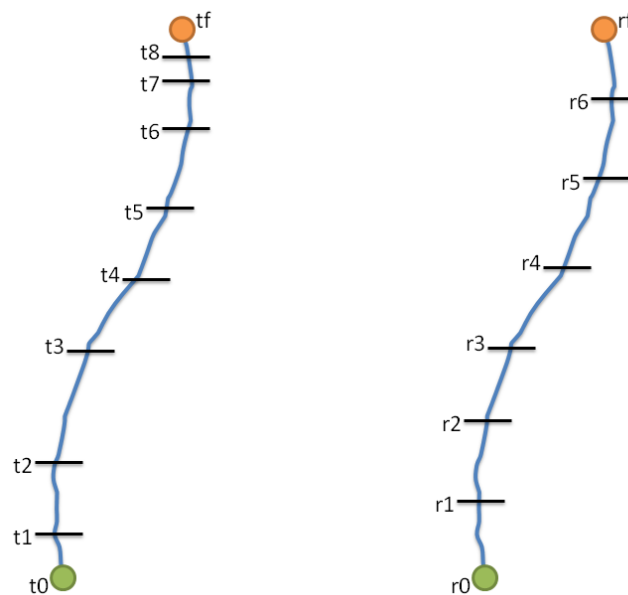


Figure 11. Re-sampled stroke

We re-sample both the template and the submitted image to be able to compare them; next we also scale them to a bounding box of the same size and translate them so they will have the same frame of reference. At this point we have two vector images with the same number of points and of the same size overlapping each other. In order to determine how close they resemble we use the distance metrics proposed by Kara & Stahovich [17]. The first is a modified Hausdorff distance. If we consider shape two shapes A and B as a set of points, then the modified Hausdorff Distance (MHD) is defined as:

$$MHD(A, B) = \max(h_{mod}(A, B), h_{mod}(B, A))$$

Where

$$h_{mod}(A, B) = \frac{1}{N_A} \sum_{a \in A} \min_{b \in B} \|a - b\|$$

In this case $\|a - b\|$ is the Euclidean distance between the two points. And N_A is the number of points in A. So h_{mod} is the average of the minimum distances between the two sets of points. More intuitively we want to make a point-to-point comparison between the two shapes and establish how close the two shapes resemble.

We also perform a comparison using a distance metric called the Tanimoto coefficient which checks how many points overlap within a certain threshold. In this case we look at both the submitted image and the templates as binary images and compare how many pixels overlap (have the same binary value). Formally the tanimoto coefficient is defined as

$$T(A, B) = \frac{n_{ab}}{n_a + n_b - n_{ab}}$$

Where n_{ab} is the number of overlapping black pixels between the two images, and n_a and n_b are the number of pixels in A and B respectively. We then combine the two calculations by normalizing and averaging the resulting confidences.

Doing a quick time complexity analysis we can see that Tanimoto comparison runs in $O(N_p)$ where N_p is the pixel resolution of the images. If we pick a threshold that is large enough this will run very fast, we do not want it to be too large as we would scarify accuracy. Most likely our time bottleneck is going to happen in the Hausdorff calculation. After re-sampling, we need to calculate the minimum distance for all N_B points, and then do this N_A times to get the average. In our case $N_A = N_B = N$ where N is the number of points after re-sampling. We see that time complexity for comparison is $O(N^2)$ and if we have \bar{T} templates in average and M different shapes we would have an overall complexity of $O(M\bar{T}N^2)$. We do not consider re-sampling as we can save the templates in the re-sampled form so re-sampling will only affect the submitted shape and will only happen once before comparison. The time complexity gives us the intuition that if we want better performance we need to have a smaller number of points after re-sampling but this will affect resolution and therefore accuracy, also a small number of templates will be better but this affects accuracy as well. Finally we want a small number of shapes in our set meaning that as we expand our system to include more shapes, performance might be affected negatively. In the current implementation $N = 64$, $\bar{T} = 10$ and $M = 10$. recognition happens under 500ms in a all of the computers we used for testing.

Agent Intelligence

At the core of the interaction we have a software agent that coordinates interaction between the child and the system. This agent runs in its own thread, and keeps track of the current state of the interaction. The agent has several responsibilities in the system and is enhanced by certain features that make it an interesting virtual tutor. Figure 12 shows a diagram of the interaction flow of the activities performed by the agent.

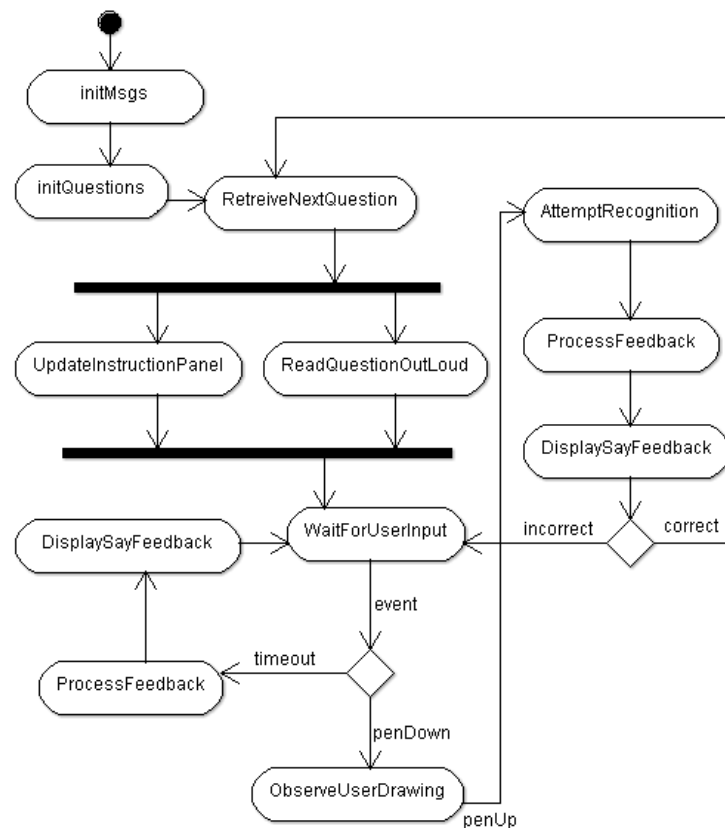


Figure 12 Activity diagram of the interaction flow of the agent

Agent Personality

At startup, the program first loads what we call the *personality* of the agent. The expressiveness of the agent is an important feature that engages kids into interaction and pushes them to work harder to achieve their goals [15]. We give our agent its own personality which will be reflected on the faces shown in the Agent Panel, the answers in the Feedback Panel and its corresponding audio prompts. Most of these features reside in particular system files that can be easily changed and tweaked to adjust the personality of the agent. One of the most relevant features of the agent personality is its emotional states. The agent's emotional state is a discrete emotion that can take any of the following values: NEUTRAL, HAPPY, DISAPPOINTED, OBSERVANT, IMPATIENT, ANGRY, CONFUSED, EXCITED, OR LAUGHING. A particular personality provides the corresponding set of animated images that go with each of these emotions. The visual feedback TAYouKi provides is very important to children as a reinforced method of feedback. Even at this early age they are responsive to facial expressions that are basic to all humans [10]. The visual representation of these emotions for the default personality is shown in Figure 13.



Figure 13. Different expressions of the avatar

The next concern is how to navigate between emotional states based on the current interaction. We do so by taking three main events into account, a correct answer, an incorrect answer or a timeout. We log each of these events, and as we will see later, we can use the history of events to take action on different parts of our system. We can use a state machine to determine the way the emotions change according to these events. As other features of the personality the transitions of this state machine are also loaded from a configuration file so they can easily be modified if needed. In our default

personality we modeled these transitions to be more human-like based on observations of normal human transitions between emotions. The state machine of the default personality is shown in Figure 14.

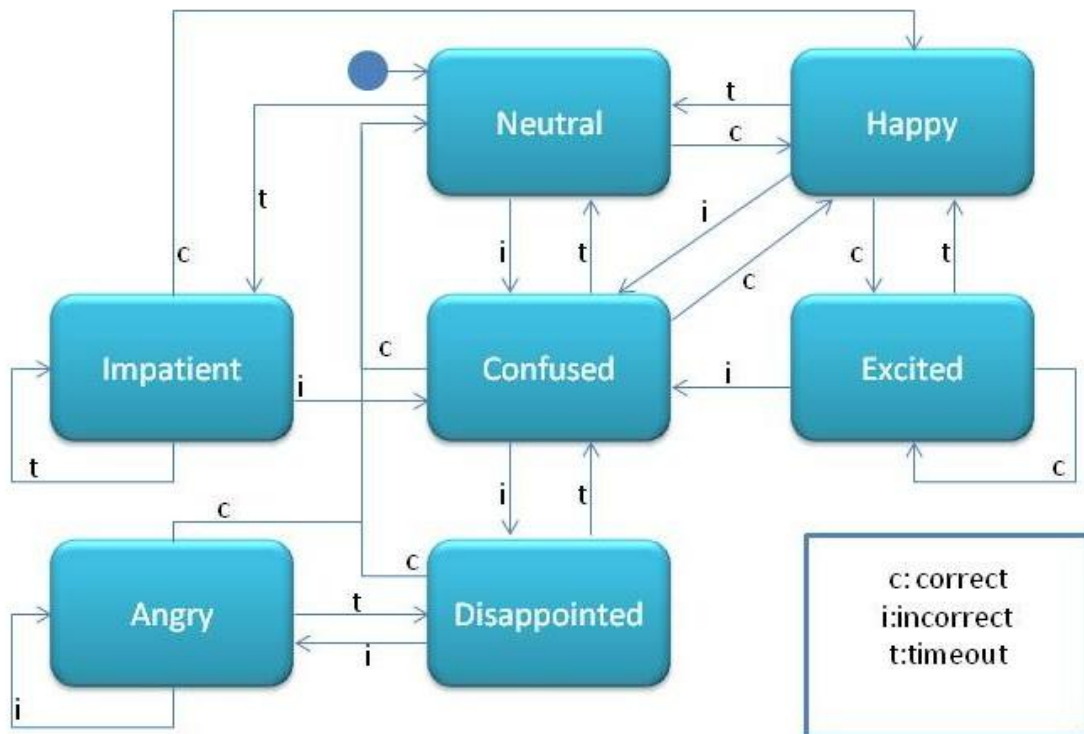


Figure 14. The state diagram that rules the change of emotional state of the agent

Question Navigation

First the agent personality and then the questions are loaded by the system.

Similarly, the text and animated pictures that go with the questions also reside in system files that can be modified and expanded as needed. After the questions are loaded, the

agent iterates through the list of questions, each time reading the instructions aloud and displaying them in the Question Panel. The questions will initially involve drawing basic shapes, like circles, lines and squares but soon they become more complex and include shapes like a star, a house, a sun or certain letters or numbers. Our initial set of questions consists of 10 questions and can be expanded as needed. These questions ask the kids to draw the shapes shown in Figure 10.

Each question is defined by a list of textual instructions and images that will map according to the current *help level*. The help level is a numeric value that indicates how explicitly instructions should be given to the child. The particular semantics of what each number means are determined by the question itself. The only premise is that 0 is the least explicit or most difficult way of posing the question, and each increasing integer will give more explicit help (e.g. 0: Draw a regular four sided contour, 2: Draw a shape with 4 sides of equal length, 3: draw a square like the one in the image, etc.).

Each question file also contains the expected correct answer which the user's answer will be compared against using the corresponding recognizer. The agent is responsible for advancing and selecting the next question in addition to controlling the help level at all times. If the student is performing well, for instance, the next question might have a lower initial help level to make it more challenging. Ideally we could use the scaffolding principles to achieve personalized instruction for the kids that goes according to their current zone of proximal development (learning stage).

Measuring Engagement

One of our objectives as a tutoring system is to make sure we gain and maintain the attention of the kids when we are teaching them new skills. We need some quantitative measure that allows us to know if the kid is engaged with the system. Engagement is not necessarily easy to measure by humans [35], and can even be much more complex to assess by a computer system. We attempt to do so by having a counter keeping track of time between every pen-up and pen-down event. A timeout will occur if the counter reaches a certain value (e.g. five seconds) without user input. We maintain a history of these timeouts along with the number of incorrect and correct questions answered by the user.

Table 2. Variables used for engagement measurement

<i>Symbol</i>	<i>Meaning</i>
N	Total number of events registered
C_i	Number of <i>correct</i> events in the past i events
T_i	Number of <i>timeouts</i> events in the past i events
W	Window size to evaluate engagement

Table 2 shows the main variables we use to measure engagement. As stated in the section *Agent Personality* we log a history of events, such that at any time after beginning the program we will have a list of N recorded events, where for each event e

on the system, $e \in \{correct, incorrect, timeout\}$. Intuitively, to check engagement we want to make sure that the kid has been actively using the system lately, and also we want to make sure that his performance is improving instead of decreasing. More formally, we can define a window of size $W < N$ and analyze the last W events. If in that window all we can see are timeouts, i.e. $T_w = W$ the kid has probably lost attention, so the agent says something such as “Are you there? Let’s keep drawing!”. Similarly, if for the analyzed window the relative score has gone down and there is a significant amount of timeouts, it might mean that the kid is losing interest in the current activity. In terms of variables we define a *global score* as $C_N/N - T_N$ and a *local score* as $C_w/N - T_w$ and *local score* $<$ *global score* then the agent should ask the kid if he is bored and perhaps suggest another question.

Feedback Generation

The feedback generation is perhaps the most visible feature of our system. It takes all the work done by previous modules such as the sketch recognizer and translates them into the final output that goes to the user. It serves as a global module that orchestrates the process of the system. The particular feedback given by the system depends on several factors, such as the previous answer history, its current emotional state, the confidence of the recognition, and the time between inputs. To give accurate feedback, we use these factors to model discrete *interaction states* of the system.

At any given time the *interaction state* can be characterized by

- a) The agent's emotional state
- b) The time that has passed since the last interaction

- c) The amount of help we are giving the kid (help level)
- d) The current question.

To determine the transition between one state and another, we also take into account the event history of the system. To represent time, we use a discrete time counter that increases every second, as representing continuous time can be costly and unnecessary. This counter will be reset by any user event on the workspace, such as drawing a stroke or pressing a button. This counter triggers important events; after it reaches certain threshold, it can trigger recognition automatically or generate a *timeout event*.

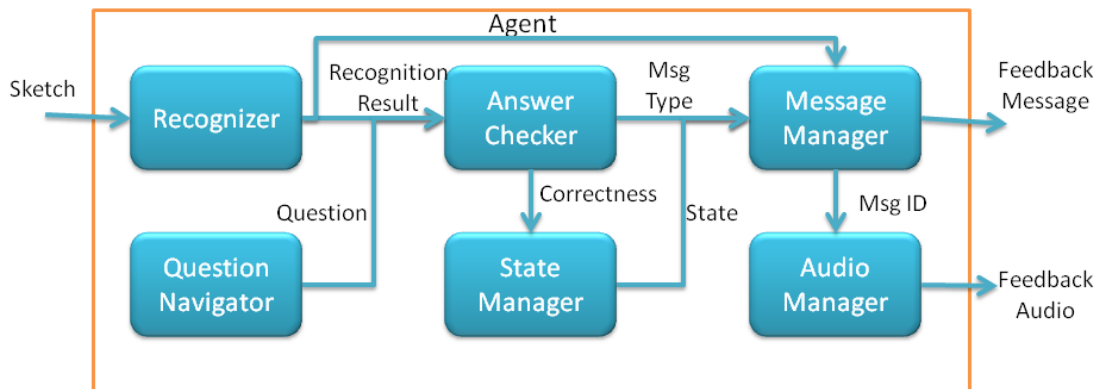


Figure 15. Box diagram showing the system's inputs, processes and outputs

Recognition can be triggered either by a timed event or by an explicit request from the user by clicking on the checkmark button after drawing. This will trigger the process shown in Figure 15. In this process, first, the current sketch is submitted to the

both geometric and visual recognizers. The output of each recognizer is a *Recognition Result* that states the most likely shapes that correspond to the current sketch as well the confidence of that recognition result. The *Recognition Results* combined with the current question and expected shape serves as an input to the answer checker. The answer checker will determine whether the answer is correct. Because we are dealing with children we want to be lenient about recognition, so instead of averaging the result we consider the answer as correct if any of the two recognizers consider it correct. However if one of them is incorrect this will affect the feedback message as we will see next.

Figure 16 shows a diagram of the answer checker.

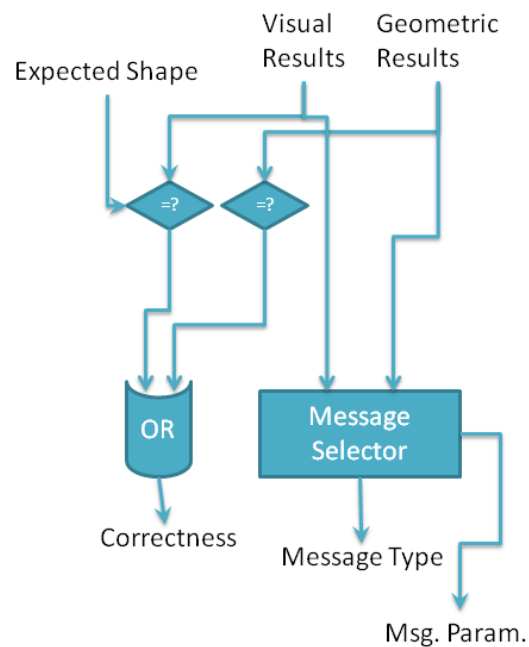


Figure 16. Answer Checker

This correctness will serve to generate the corresponding *correct* or *incorrect* event e which will be logged to the event history and will serve as input to the modules explained in the *Agent Personality* and *Measuring Engagement* sections. The answer checker will also determine which kind of message the system should return. We call this the *Message Type*.

Table 3. Example messages for different message types

<i>Message Type</i>	<i>Pre-Emotion</i>	<i>Example</i>	<i>Post-Emotion</i>
CORRECT	EXCITED	WoW!! You are Amazing!!	EXCITED
CORRECT	DISSAPOINTED	That is better!.	NEUTRAL
INCORRECT SHAPE	HAPPY	Nice circle! Can you draw a square?	CONFUSED
INCORRECT SHAPE	ANGRY	That is not a square, keep trying.	ANGRY
TOO MANY STROKES	IMPATIENT	Easy. Easy...You drew too many lines.	CONFUSED
TOO FEW STROKES	NEUTRAL	Almost there! You are missing something!	CONFUSED
LOST ENGAGEMENT	CONFUSED	My friend! Are you there??	IMPATIENT
GEOMETRICALLY INCORRECT	NEUTRAL	Funny way of drawing, but is a valid House.	CONFUSED
VISUALLY INCORRECT	CONFUSED	Is that a car?? Ok, if you say so.	DISSAPOINTED

Table 3 summarizes various types of messages supplied by the system. An extensive version of all the messages in the system can also be found in Appendix C.

The message handler translates this message into a tailored response corresponding to the current state of the interaction. As a first step, the message handler takes the generic message type and matches it with its current emotional state. For example, two responses to a correct answer may be “Congratulations!” if the agent is happy, or “OK, let's continue” if the agent is neutral or impatient. Additionally, it may become boring after a while to have the same messages repeated over and over, so we give the agent the capability of having multiple synonym messages in the configuration file. Finally, since some of the feedback also needs to be tailored depending on the error we also have message parameters, which are obtained from the Recognition Result. These parameters are used by the translator to fill in the blanks and make particular message instances. For example we can have the following message “*Nice \$foundShape\$! But I expected to see a \$expectedShape\$, try again!*” which can be instantiated into “*Nice **circle!** But I expected to see a **square**, try again!*”

At this point is important to think about the scalability of the content. For each message type τ TAYouKi can express differently depending on its current emotion ε . Moreover because we allow multiple messages for the same mapping of ε and τ we can have p different messages for each mapping. For E different emotions and T different message types we have a total number of parameterized messages M as follows:

$$M = \sum_{\tau=1}^T \sum_{\varepsilon=1}^E p_{\varepsilon\tau}$$

We can simplify this value by taking the average pool size \bar{p} :

$$M \cong \tau\varepsilon\bar{p}$$

Finally we have the parameters which can alter the resulting message in whatever possible permutation of parameters. In the current implementation we do not have more than two parameters for any message type and we do not expect it to grow, each parameter however usually take up to n different values where n is the number of supported shapes. This means we have about Mn^2 possible message instantiations.

Currently we have seven different emotions reachable by our state manager. And we support for nine different message types. The pool size varies depending on how common is a particular state. States with self-transitions such as angry, happy or impatient tend to appear more often and thus have a bigger pool. We have an average message pool of size of 1.7 in our current content. In other words, most of the combinations of a message type and an emotion give two distinct messages, some only give one and some others give more than two. (Resulting in an average of 1.7) This gives us about 107 messages without being instantiated with particular parameters. In the case of text we can easily pass the parameters on demand so we do not need to worry about all the possible number instantiations. However, if we want to record independent audio cues for all possible message instantiations we would need to record up to 10,000 files. This makes us think about possible alternatives.

Our solution was to use a text to speech module. We used an open source alternative called FreeTTS [48] that allowed us to give the final instantiation to this module for it to translate it without worrying about recording 10000 audio files. After the feedback is reproduced on-screen and spoken, the users can proceed to follow the

instructions. Figure 17 shows a state diagram of the rules used by the message manager to change the emotional state of the agent.

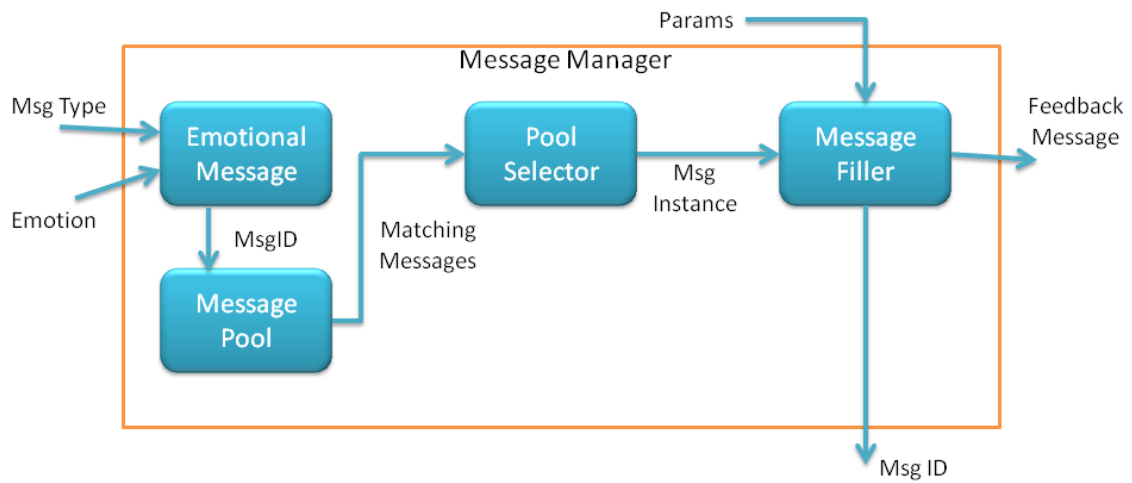


Figure 17. Box diagram showing the sub modules of the Message Manager

INSTRUCTIONAL DESIGN

Our aim in this project is to help solving a real need. In order to do this we need to understand the software as part of a system where students, instructors and learning environment interact in order to reach one or more learning goals. The appropriate selection of content and an evaluation method that can provide valuable feedback is a key component in the success of the project. The process of performing analysis, design, implementation and evaluation of instructional material is often called Instructional Design. There are several approaches to design instruction but a very accepted model to build effective instructional material is the Dick and Carey Model [6]. Figure 18 shows a diagram that summarizes this model of instructional design.

In this section we show how we applied this model to the design of our system in order to build effective instruction. We then explain the evaluation methods we used for this project followed by a discussion of the interactions methods we explored as part of the analysis to build our system.

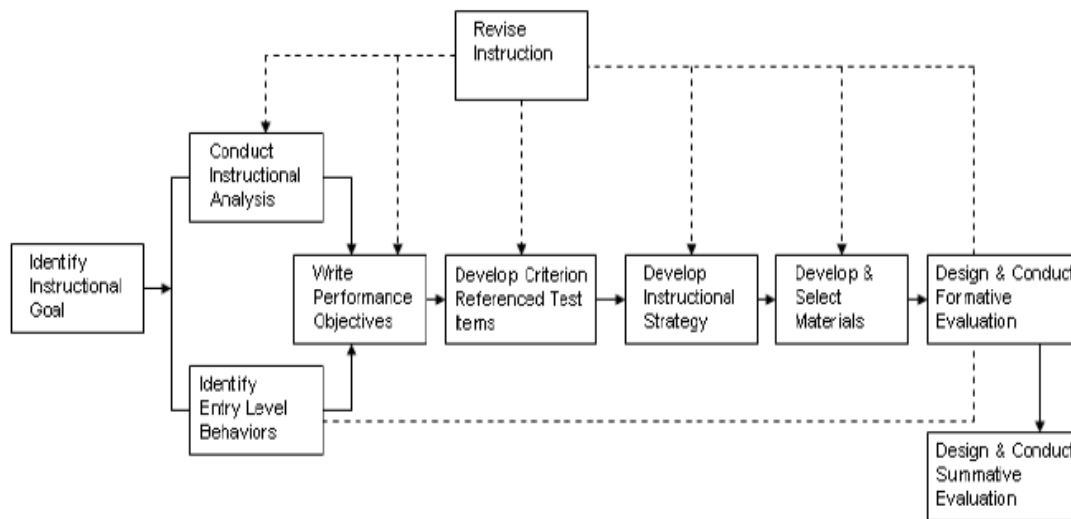


Figure 18 Dick & Carey Model for instructional design

In the following subsections we briefly go through each step of the Dick & Carey Model explaining how we designed our Intelligent Tutoring System, and its evaluation.

Instructional Goal

The first step in the design of instruction is to establish a well defined instructional goal. What are we going to teach to the kids with our system? This is the question that will drive the overall design. Our goal is stated next:

“By receiving basic instructions from a parent or instructor, and accompanied by a computer system, children in the age group 3-5 will be able to identify and draw basic geometric shapes, such as circles, triangles, lines and squares. They will progressively be able to draw more complex shapes based on these primitives. The ultimate goal is that eventually the children will be able to draw all the letters in the Latin alphabet (A-

Z) (upper and lower case) and the numbers (digits) from 0 to 9. Children will be able to reproduce what they learned in regular pen and paper.”

Instructional Analysis

Next we can further analyze our goal by decomposing it into smaller sub goals and required skills. Our system aims to aid in teaching both intellectual and psychomotor skills. Figure 19 shows the goal decomposition as 3 simple steps and their subordinate required skills. The simplicity of the goal is intended, as our users are at a very young age and we do not expect them to perform very complex tasks after instruction is provided. However we do expect them to be able to reproduce what they learned. This means to be able to draw a particular shape on a piece of paper or digital screen when instructed.

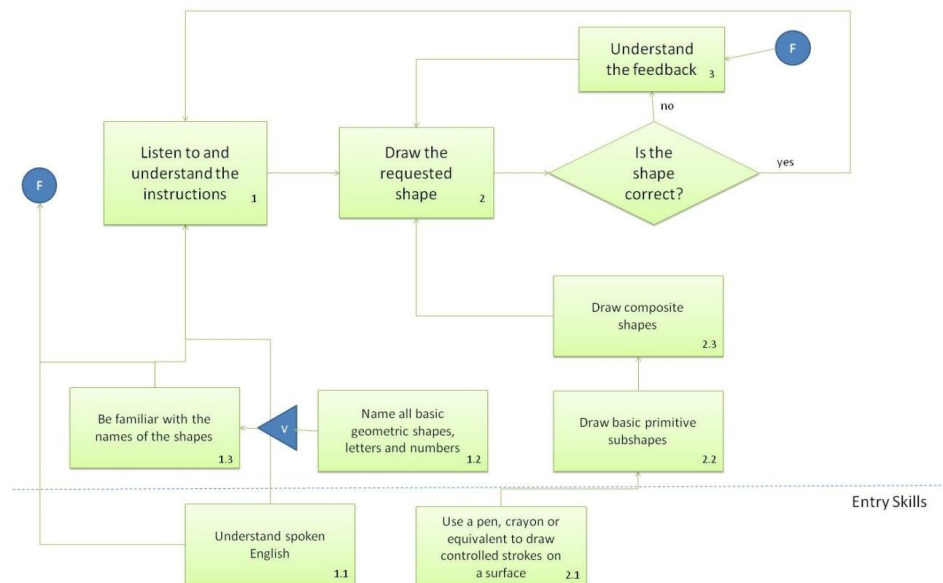


Figure 19. Goal decomposition

Learner Context and Analysis

Before venturing into development we analyzed some of the more relevant characteristics of the learners, the environment where the learners are going to receive the instruction and the environment where the learners are going to perform it. We used different data sources to perform this analysis: early user studies, interviews with subject matter experts (SME) and personal observations. Tables 4-6 describe such analysis.

Learner Analysis

Table 4. Learner analysis summary

<i>Information Categories</i>	<i>Learner Characteristics</i>
1. Entry skills	Students at this age are able to draw basic strokes on paper, students understand spoken English.
2. Prior knowledge of topic area	Most students are interested in drawing as a form of creative expression, they do not yet master the alphabet but many of them can already visually identify some shapes, letters and numbers.
3. Attitudes toward content	Most kids like to draw, not all of them are fully interested in writing and reading at this time.
4. Attitudes toward potential delivery system	Kids are very excited towards computers with sound, animation and fun interaction. Teachers see lots of potential in using a computer tutoring system. Some are a little skeptical, but are willing to try.
5. Motivation for instruction (ARCS)	Peer pressure is important at this age, kids are motivated to learn as they see their friends and bigger kids having the ability to draw beautiful shapes and express with letters and numbers.
6. Educational and ability levels	Students coming to the clinic are very diverse in terms of background and ability. This software is appropriate only to a particular, yet significant, subgroup.
7. General learning preferences	Kids like the feeling of drawing on any surface, they like colors and crayons. Yet they also like the sounds and feedback provided by computers.
8. Attitudes toward training organization	The relationship of kids and instructors is very good in the clinic.
9. General group characteristics	A usual session in the clinic takes 1 hour and each instructor handles groups of 5 students at a time. Kids could greatly benefit from the use of an intelligent tutor that can support the learning experience.

Performance Context Analysis

Note that in this section, the performance context is huge, as the set of skills they learn here will be reflected in many scenarios of the learner's life. However we are concerned and analyze here only their near future where they will need these skills to build more complex ones. (i.e. second grade)

Table 5. Performance context summary

<i>Information Categories</i>	<i>Performance Site Characteristics</i>
1. Managerial / supervisory support	The reading clinic director is very interested in this initiative. Teachers inside and outside of the clinic seem also very interested. Some parents also see this tool as a very powerful method to enforce learning of these key skills.
2. Physical aspects of site	The physical aspects are not very relevant because of the performance of the skills can be made anywhere you can find pen and paper. Usually this will be in a classroom or at home. However, it is important to note that the computers are not required at the site to perform the skill, so the students should achieve a level of mastery that does not require the feedback of the computer to perform correctly.
3. Social aspects of site	Teachers will usually grade the work of the students. Legible handwriting is required in future courses, the ability to draw graphs that use basic shapes is also required in the future. This is usually an individual skill that learners should master without help of teammates. However it will become a key part of communication with them.
4. Relevance of skills to the workplace	This will be a fundamental entry skill in future courses and lessons. Without writing and drawing communication is seized in many ways.

Learning Context Analysis

Table 6. Learning context summary

<i>Information Categories</i>	<i>Learning Site Characteristics</i>
1. Number/nature of sites	The reading clinic currently has some computers. The sketch recognition lab can provide additional laptops with pen input and wacom screens to support drawing.
2. Site compatibility with instructional needs	Class space can be provided to use the software; the instructors can dedicate small chunks of time of 15 minutes to use the software. The training required from the instructors is minimal so this should not represent a problem. Instructors are very willing to help. The classrooms have spare electrical outlets to connect the extra screens and computers.
3. Site compatibility with learner needs	The classrooms in the clinic have enough space to hold the small groups of children that would be using the software at a given time.
4. Feasibility for simulating workplace / future education	The workplace can be simulated using any computer with pen input. Social characteristics should be similar given that the training time for the instructors only takes a couple of minutes.

Performance Objectives

An important part of the instruction is being able to assess if the learners are doing what they are supposed to be doing after finishing the instruction. This is, if they have reached the instructional goal. However an instructional goal is usually defined in the scope of the performance context. Here we need observable proof that the learners have achieved their objectives. We then define a main terminal objective and a subset of performance objectives that are summarized in Table 7. These will allow us assess if the instruction is effective. Assessment instruments are briefly described here and presented in detail in the following section.

Terminal Objective

By receiving basic instructions from a parent or instructor, the learner will be able to identify a set of eight different shapes by saying the name of the shape as they see a picture of it. They will then be able to draw the same set of shapes. The shapes will contain basic geometric shapes, composed shapes and at least one letter or number. The learner will identify the shapes verbally and will draw using regular pen and paper. At least six shapes should be correctly identified and five shapes correctly drawn.

Performance Objectives

Table 7. Performance objectives

<i>Subordinate Skill</i>	<i>Performance Objective</i>
1.1 Name any shape of the group	Given the picture of a shape the student is able to verbally identify the shape by saying its name. Learner should correctly name at least 75% of the shapes.
1.2 Understand spoken English instructions	Given a simple instruction in English that the learner is known to be able to perform, at least 90% of the time the student will always be able to either perform it or clearly state the reasons why he is not doing it.
1.3 Be familiar with the name of the shapes	When randomly asking the student for names of shapes the student is able to say a list of no less than 4 shapes from memory. The shapes either are part of the test group or are commonly known geometric shapes, letters or numbers.
2.1 Use a pen to paint ink on a surface	Given a pen or crayon and a piece of paper, when requested the student can draw ink strokes on it. If given a digital pen and a supported screen the student can do the same.
2.2 Draw basic primitive subshapes	Given a pen and a drawing surface, when instructed the student draws a set of simple primitives. Students must correctly draw at least 75% of the primitives.
3 Understand the feedback	Whenever the computer gives feedback to the students the students take corrective action based on what they heard. Students do this at least 80% of the time.

Assessment Instruments

When using this system as part of the instruction we want to make sure that the children learned what they were supposed to. Usually this is done by using a set of standardized tests in pen and paper during and after the instruction. In our case assessment is very interesting, due to the fact that our software is an assessment instrument on itself. This is, it evaluates the correctness of the kids' drawings as they progress. So the main form of assessment during the instruction is not an external instrument, but is embedded in the instructional material we provide.

However assessment not only happens during the instruction. Entry skills are very important to asses, and since we want instruction to transcend beyond the use of our system it might be good to have this entry skill assessments in the performance environment, this means, without the use of a digital screen. Similarly we want to make sure the learning transfer was effective by allowing the learners to perform in their usual performance context: pencil and paper. Appendix B shows the assessment instruments we used before, and after our evaluation.

EVALUATION

Before venturing into explaining the details on how we evaluated our system, it is important to clarify what we are going to evaluate. As explained in the introduction, we are facing a multi-domain problem, and therefore we are required to have a multi-domain evaluation. Figure 20 shows the domains we integrate in our system that we are going to use as our focal point for the valuation of our system.

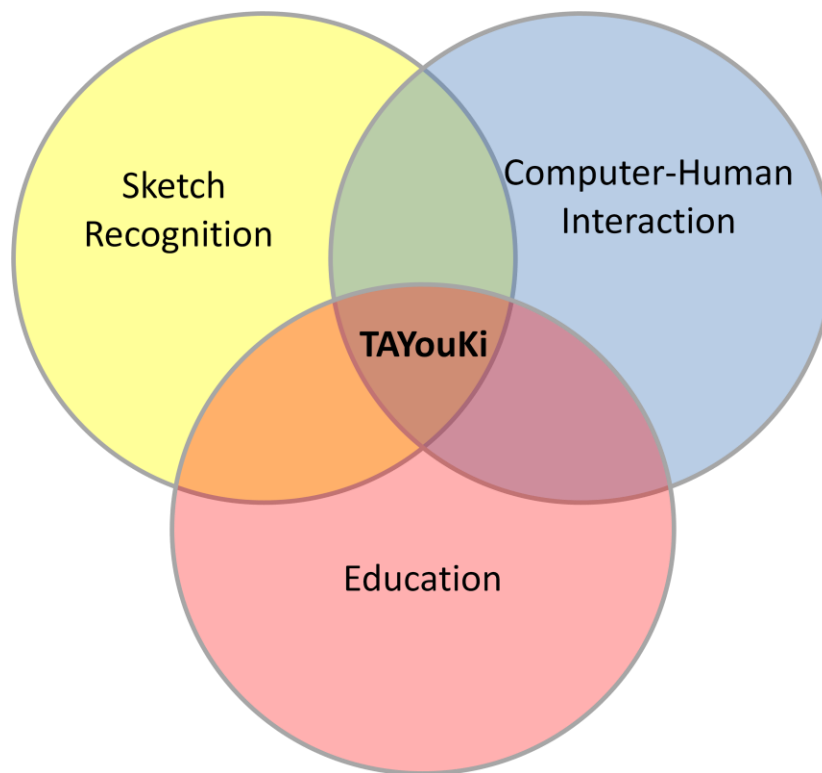


Figure 20 Different domains from which point of view we evaluated our system

We have decided to structure in these three different domains: Sketch Recognition, Computer-Human interaction and Education. In this section we are going to introduce briefly our overall data collection techniques for evaluation and then go in more detail through three concrete user studies. Each of these studies allowed us to gain important insights in one of these domains. Although each study was primarily emphasized towards a particular domain, there is overlapping information and the three studies actually contributed information to all the three domains. However, for the sake of clarity we are going to describe each user study along with the domain it mainly describes, and we will discuss in a more general fashion in the next section.

Evaluation Overview

A common way of testing and comparing a new methodology versus traditional methodologies is to have two control groups testing each method or technology separately. Then, extract quantitative data and treat results in a statistical fashion to infer conclusions out of it. However, the results are only relevant if the groups are sufficiently big and diverse. To obtain a statistically significant group we would need to deploy our system in the real learning context (i.e. classrooms and homes) and get parental permission from all of the kid participants. This is ideal and necessary to have a meaningful *summative evaluation* of a new educational component or tool. However, it is not in the scope of this work to perform this summative evaluation. In summative evaluation we try to determine if a novel instructional component solves an instructional need, and if it does it more effectively than existing solutions. Instead we are interested in *formative evaluation* which will allow us to answer a related question that we consider

more relevant at this stage: How can we make our system better such that it will eventually solve the instructional need?

Because of this formative evaluation approach, we need a more open ended type of evaluation that will give us enough flexibility to give space to suggestions and feedback from the participants and new ideas for improvement. Thus, the evaluation we performed involved a set of case scenarios where we examined the interaction of each participant children in our study in a one to one basis. This allowed us to obtain detailed data about the issues and advantages encountered when using our approach.

On the other hand we also wanted to collect data that would allow us to evaluate the Sketch Recognition, and the CHI aspects of our system in an isolated manner from the educational component. In summary, we had three main sources of data: First we performed a user study with adult participants to collect sketches of our shape set and evaluate recognition accuracy. Then we had a first round of user studies with children using an early prototype where we wanted to detect major issues with the system and the reaction of the children to this type of system. Finally, we conducted a second round of children studies with a more mature prototype that allowed us to gain insights on the instructional value and issues of our system.

Sketch Recognition

Before we ventured into testing with children it was important to know whether recognition was working or not. Low recognition accuracy can easily convert this tool in more of a problem than a solution for the instructor. In particular we established a set of questions that would guide this part of the evaluation:

- What is the accuracy of each recognizer?
- What can be improved?
- Which recognition approach works better for our case?
- Is the accuracy similar for kids?
- What features are most relevant?
- What are the major challenges?

We divide this section in who performed the study, how we collected the data, and how we analyzed it. The following subsections give the details of each of these aspects.

Participants

For this portion of evaluation we wanted a population that already knew what the correct answer should look like. In this case this is simply, an adult population. We were interested in collection sketch data from adults that could easily draw the shapes in our set. We had around 10 participants that were 18 or up and were mostly graduate students from diverse majors and backgrounds.

SOUSA

We relied on an existing data collection tool called SOUSA [29] develop by Jacobson et. al. SOUSA is a web-based data collection tool that enables to create personalized user studies for sketch data collection. We created a user study in this system that requested the participants to draw about 40 sketches distributed in 10 different shapes. Figure 21 shows a screenshot of the SOUSA system and a sample question of our user study.

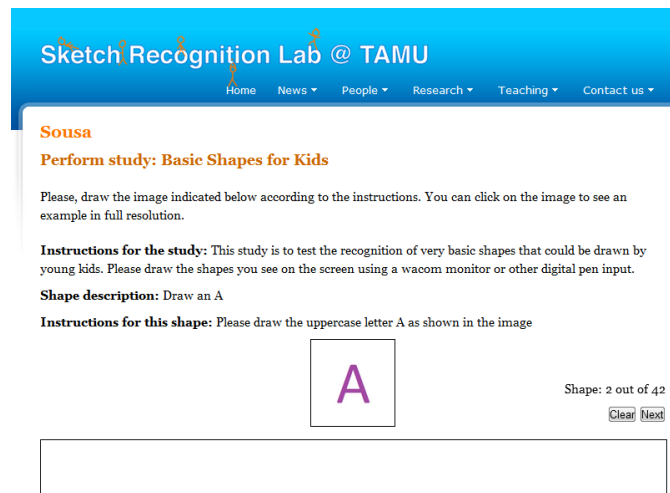


Figure 21. SOUSA is a web-based tool for sketch data collection

We had to trim some of the sketches of participants that mistakenly drew the wrong shape or clicked next on an empty drawing. After trimming we ended up having 432 different sketches about evenly distributed amongst the 10 different shapes. We used this data as the input to evaluate our recognizers. The sample test shapes were drawn differently by users but they were “visually acceptable” meaning that although not perfect a human could match the shape to its corresponding question.

Sketch Analyzer

Our next challenge was to analyze use all this data effectively and efficiently to iteratively improve our recognition rates. We built the *Sketch Analyzer*. This tool allows loading any sketch stored as an xml file using the same data structure that SOUSA uses. The user can visualize the sketch on screen and replay using the time information to reproduce what the learner originally did. Figure 22 shows a screenshot of this tool.

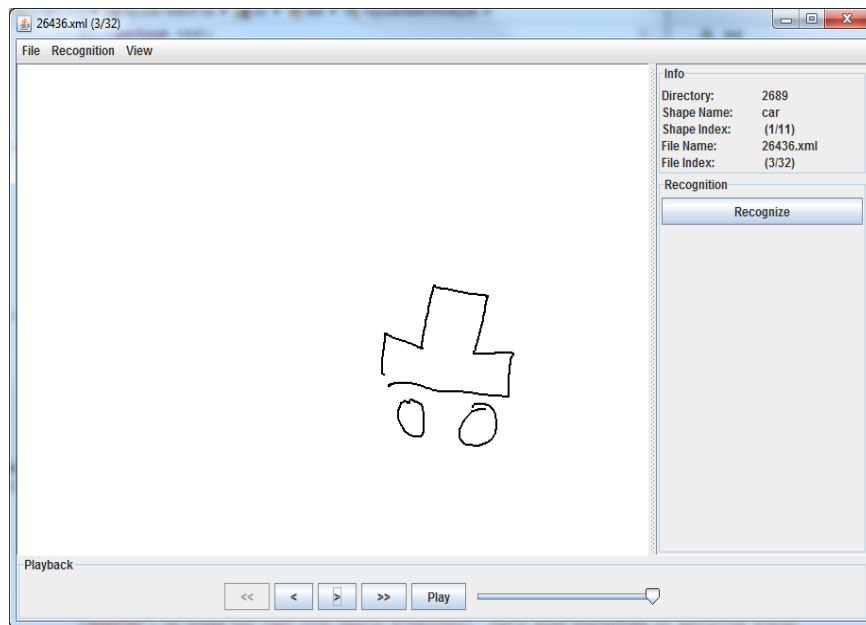


Figure 22. Screenshot of the Sketch Analyzer

More importantly, this tool allows loading a complete folder of data. SOUSA stores each shape sample in a particular folder. The user can load the root folder and navigate between shapes or samples, for each of them the user can attempt recognition by pressing a button. The user can compare the result of recognition with the expected shape for each sketch. Since we are using two different recognizers in our system, the user can select which recognizer is to be evaluated (visual or geometric). This is very powerful for debugging recognition problems but it would result very inconvenient to process each sample one-by-one in a data set like the one we have consisting of over 400 samples. This system provides an additional option of batch processing all of the shapes

and all of the samples at once. The result is a comma separated value file (csv) that contains a confusion matrix and accuracy results.

An interesting side effect of batch processing using the geometric recognizer is that since we had embedded sound feedback for primitive recognition, the batch processor results in a symphony of sounds where we can identify just by listening problems with the recognition. For instance we can see on screen that the analyzer is currently processing a set of spirals. When correctly identified by our low-level recognizer, a spiral will emit a sound that sounds like “*wiii*”. In batch processing we listen a continuous “*wiiwiiwiiwiiwiiwii*” suddenly a “*wiiiiwiiipsh-pshwiii*” we can tell immediately that at least one spiral was incorrectly classified as a poly-line, since the poly-line sounds like “*Psh-psh*”.

A confusion matrix is a convenient representation of the results that allows identifying at a glance both the recognition rates and some potential problems. In a confusion matrix each row represents the sample shape that we were expecting and each column denotes what was actually recognized as. For a cell in the row of shape r and column of shape c we will have the percentage of shapes that were expected to be r and resulted to be recognized as c . Note that if we sum up all the columns in a single row they should add up to 1. The accuracy summary simply represents in a more compact form the diagonal of this matrix. Examples of these tables will be shown in the following section.

Results

We ran our recognizers over the complete data set and we obtained the output shown in Table 8 and Table 9.

Table 8 Confusion matrix for the geometric recognizer

	car	circle	house	letter-a	line	number-1	spiral	square	stickman	triangle
car	0.7742	0.0968	0.0000	0.0000	0.0000	0.0000	0.0323	0.0645	0.0323	0.0000
circle	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
house	0.0000	0.0625	0.8438	0.0000	0.0938	0.0000	0.0000	0.0000	0.0000	0.0000
letter-a	0.0000	0.0000	0.0000	0.9655	0.0345	0.0000	0.0000	0.0000	0.0000	0.0000
line	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000
number-1	0.0000	0.0000	0.0000	0.0000	0.2143	0.7857	0.0000	0.0000	0.0000	0.0000
spiral	0.0000	0.0000	0.0000	0.0000	0.0606	0.0000	0.9394	0.0000	0.0000	0.0000
square	0.0000	0.0000	0.0000	0.0000	0.0566	0.0000	0.0000	0.9434	0.0000	0.0000
stickman	0.0000	0.0000	0.0000	0.0000	0.0645	0.0000	0.0000	0.0000	0.9355	0.0000
triangle	0.0000	0.0189	0.0000	0.0000	0.0189	0.0000	0.0000	0.0000	0.0000	0.9623

Table 9 Confusion matrix for the visual recognizer

	car	circle	house	letter-a	line	number-1	spiral	square	stickman	triangle
car	0.9677	0.0000	0.0000	0.0000	0.0000	0.0000	0.0323	0.0000	0.0000	0.0000
circle	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
house	0.0000	0.0000	0.9688	0.0000	0.0000	0.0000	0.0313	0.0000	0.0000	0.0000
letter-a	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
line	0.0000	0.0000	0.0000	0.0189	0.9811	0.0000	0.0000	0.0000	0.0000	0.0000
number-1	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000
spiral	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000
square	0.0000	0.1509	0.0000	0.0000	0.0000	0.0000	0.1132	0.7358	0.0000	0.0000
stickman	0.0000	0.0000	0.0000	0.0323	0.0000	0.0000	0.0000	0.0000	0.9677	0.0000
triangle	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0189	0.0000	0.0000	0.9811

To facilitate the reading of these results the key elements are summarized in the following tables and graphs. Table 10 gives in a single column the accuracy for each shape for the two recognizers. The Max column is simply the maximum recognition rate between these two. Remember we are considering the answer as correct if any of our recognizers gives a correct result, so our overall recognition accuracy is 98.29% using the data we collected. The confusion columns give an insight on what was the recognizer misinterpreting more frequently (e.g. a car was confused for a circle by the geometric recognizer, and by a spiral by the visual one). Figure 23 shows a bar graph where the accuracy for each shape can be seen for each recognizer.

Table 10 Accuracy summary

	Accuracy			Confusion	
	Geometric	Visual	Max	Geometric	Visual
car	77.42%	96.77%	96.77%	circle	spiral
stickman	93.55%	96.77%	96.77%	line	letter-a
letter-a	96.55%	100.00%	100.00%	line	NA
triangle	96.23%	98.11%	98.11%	circle	spiral
number-1	78.57%	100.00%	100.00%	line	NA
line	100.00%	98.11%	100.00%	NA	letter-a
square	94.34%	73.58%	94.34%	line	circle
spiral	93.94%	100.00%	100.00%	line	NA
house	84.38%	96.88%	96.88%	line	spiral
circle	100.00%	100.00%	100.00%	NA	NA
Average	91.50%	96.02%	98.29%	line	spiral

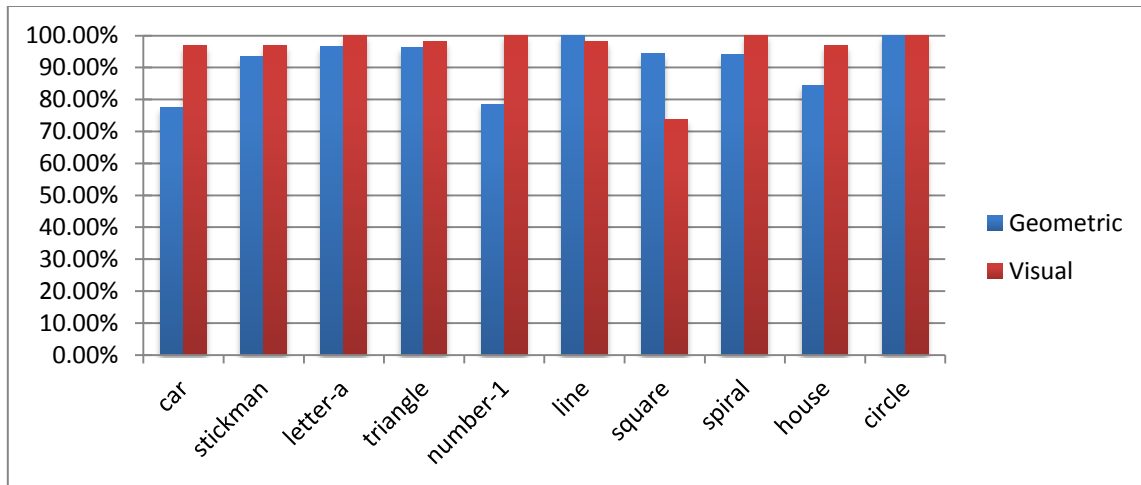


Figure 23 Recognition rates of each recognizer for the sample shape set

Computer Human Interaction (CHI)

In this section we would like to examine how a group of young children whom are potential users of our final product would respond to the presented interaction methods in comparison to traditional educational software, specifically when it comes to instruction on writing and drawing given the constraint of geometric shapes. We conducted a user study for this purpose on an early prototype of our system. Since at this point the recognition was not completely refined, we wanted to isolate the recognition part of the system with the interactivity it provides. Again, we post the guiding research questions below:

- Was the feedback method appropriate and useful? (sounds, pedagogical agent..)
- Do the kids find the software engaging? For how long do they remain engaged?

- What is the children's/teachers' attitude toward the use of digital pens?
- Was the timing appropriate?

Our study at this stage was purposely flexible at this point, meaning we did not constrained our participants to follow strict procedures, but instead became familiar with them and gave them guidance in the process, following pre-established guidelines but giving enough freedom for the participants to contribute new ideas and suggestions. In this process we conducted some pre-instructional activities and then we had a first hands-on experiment between the children and our system. We closed with informal interviews and surveys for both parents and kids. In this section we describe these activities and the results we obtained.

Participants

For this user study we had two children of ages three and four. We contacted their parents and we introduced our system to them. The parents also were considered as participants of our study playing the part of instructors and mediators between the children and our system. We obtained the permission from the parents to perform the study, and we also asked for verbal consent from the kids. Each of our studies was separately observing the interaction of the kids and the system, and had two main participants: the child and his or her parent.

Location Conditions

Our system has wide flexibility in evaluating participants based on location, since our system is not restricted in use to only classroom environments. We desire our system to be available outside the classroom as well, in order to open more opportunities

for children to learn and optionally with accompaniment from their parents. For this reason we do not restrict our testing location to a classroom.

However, in this early stage of evaluation we wanted to have a controlled environment such that we can isolate other hardware or external issues with the data we were trying to collect here. The first evaluation was carried out on campus to provide the participants with the right equipment and a comfortable workspace. We used two versions of pen-enabled Wacom screens, a 21" and a small 12". Each kid had a seat in front of a pen enabled screen besides his or her parent, while the developer sat in a third seat with another monitor connected to the same computer from where he can watch the interaction closely and override the agent if needed.

Pre-Instructional Activities

In our system motivation is a key component because of our target age. Children may not understand the importance of drawing and writing and it is crucial to get them motivated to learn these skills before even trying to begin instruction. Perhaps at this age the best way to motivate children is by serving as a role model. If they see how you and some of their peers can draw some beautiful and interesting shapes they will be motivated to explore by themselves.

We met with our participants (both parents and children) some days before the instruction and became familiarized with them in a more informal scenario by conducting a series of pre-instructional activities described below. We repeated some of these activities on the actual day of the study in order to serve as a warm-up and better assess the entry skills of our participants.

Step 1: Breaking the Ice

This might sound trivial but is a crucial part of instruction in our case. Children need to feel related to the instructor somehow; the instructor needs to gain their trust to begin with. They need to believe what the instructor says; we base several things under this assumption. There is no magical unique formula for this and it will really depend on the kid. However there are certain things that might help. For example bringing toys and crayons to the learning context is important so they can play with you. It is also important to have prepared some simple questions or comments that might take their shyness away. For example: “My favorite color is blue, which one is your favorite?” “Do you like crayons?”

Step 2: Showing Them How Cool It Is to Draw

Once you have their attention, you need to show them how interesting it is to draw. This is done simply by drawing and allowing them to draw with you. No restrictions, no constraints, allow them to be creative and express whatever they want to draw. Initially this can be done in paper, but soon it will be important to allow them to do this on the computer screen to get them used to it. It might also be engaging for them to see cool sounds and animations on screen.

Step 3: The Importance of Geometric Shapes

Now they are used to draw on paper and on the screen, we need to begin giving some structure to shapes. Why not drawing just scribbles? Show them how you can convey meaning in the drawings by drawing well known geometric shapes, such as a square and a triangle, and how can you combine these to build more complex shapes.

After conducting these previous activities we had the children more engaged and interested in the system as well as their parents, such that attitude components would not affect our data significantly.

Wizard of Oz Experiment

Because these tests were ran almost in parallel with the first one, we wanted to isolate the recognition effectiveness with the interactivity and user interface methods. We developed a Wizard of Oz experiment [18] to overcome this problem. This is: we emulated the intelligence of the agent and the recognition by having a human controlling the answers behind the scenes. We thought this was a good opportunity to gather interesting evaluation data, so instead of full agent control or full human control we had a hybrid. We had an alternate window where we had the capability of inputting messages and overwrite anything the agent was about to say, as well as the correctness of the answer of our young user. In case things got mistakenly recognized by the agent we were able to fix it on the run by overriding his decision. Figure 24 shows a screenshot of the system when using the experiment. In this case the figure drawn by the learner was not recognized or approved by our system. However, using an external window the instructor was able to override the decision of the agent, as well as the message. Note that in our particular study we were running the test using dual monitors, such that the children were not able to see this second window.

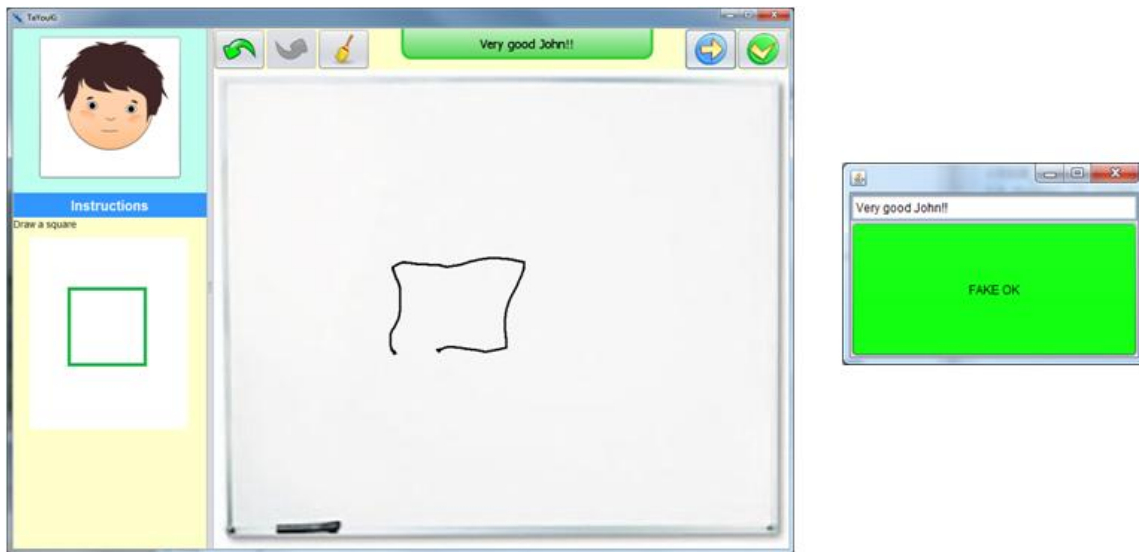


Figure 24 Wizard of Oz experiment. A side window overrides the agent's decisions and messages

Results

The first child user was almost three years old, while the second was just above four. We will call them User 1 and User 2 respectively. User 1 was a little shy at the beginning, but after following the pre-instructional activities the user was very motivated and eager to continue. The use of colors engaged this user very much. He liked to play with all sorts of colors, when provided with crayons to draw. The shapes he drew on a blank paper were mostly slanted lines over tracing each other. When requested to draw particular shapes like a triangle or circle he drew curved lines making 2 over-traced circles. We then moved on to the use of the actual the screen using a commercial paint program; the strokes on screen were fairly similar to those on paper. Finally we moved to the TAYouKi prototype, the user seemed deeply engaged with the fact there was "a little boy" on screen. Sounds were immediately appealing. Once he started drawing

feedback messages started appearing with the corresponding voice message. Soon we had to make use of the *Wizard* button to override recognition, as most of his shapes were not properly accepted. In his case, most of the time it was because his drawing did not corresponded to what he was asked for.

User 2 was a bit more mature because of her age. Interaction went in a similar fashion, except that in this case the drawings on screen were more appropriate, and we made less use of the wizard button. However we did noticed some cases where poor recognition caused us to overwrite the agent's assessment. She seemed very engaged in the activity at all times, and was fully aware of the feedback and the agent's emotional response.

Education

The final lens we used to evaluate our system was the educational component. Our system should aim to improve on certain facets of traditional forms of instruction such that it nicely complements existing methods, while not reducing the quality of instruction as well as easing the workload burden on instructors. We need to evaluate if the integration of the CHI and Sketch recognition components serve as useful instructional material. The research questions that guided this part of the evaluation were:

- Are kids reaching one of the defined instructional goals by using our system?
- Does the digital pen interaction resemble enough to regular pen/paper ?
(Delivery method and transfer learning)

- Were the feedback messages appropriate and useful?
- Where the questions (shapes) relevant and easy to understand?

Participants

For this user study we had three children of ages three, four and six different from those of the first study. Once more, we contacted their parents and introduced our system to them and asked for the according permissions. In this evaluation, we wanted to be a little more structured than in our first study with kids. So the parents played a role of evaluators besides the previous role of mediators of the system. This doesn't mean we were not open to comments and suggestions, but we had a more rigid structure in the processes and surveys. Each of the studies was performed separately observing the interaction of the kids and the system.

Pre-Testing

After conducting the pre-instructional activities mentioned in the previous section, each participant kid was asked to fill in a Pre-Test to make sure of the knowledge of the participant previous to any interaction with the software. Each participant kid was given crayons and paper. Unmarked white paper was used for pre-instructional activities and motivation. Then they were introduced to geometric shapes using the material that can be found in Appendix B. We followed a structured procedure interacting with each kid in a playful fashion in order to answer the Pre-Test Form in Appendix B for each of the participant kids.

These activities had the objective to serve both as an assessment of the entry-skills as well as a pre-test that assessed the skills included in the instruction to better guide the children in the process. In a more extensive study in terms of number of participants it would also serve for statistical comparison of learning.

Activity Log

After they have played for some time with crayons and paper, we allowed them to play with our prototype system. This time the interaction we tried to avoid some more the intervention of the developers and parents to see the effectiveness of the child-computer interaction. We closely observed the interaction at all times, but many things may happen that are hard to capture or recall that are valuable for analysis. To aid in this problem we introduce another important component of our system, which is the *activity log*. As explained in the implementation section the interaction of the kids can be atomized into particular events that can be triggered explicitly by the user, or by system events such as time or pen activity.

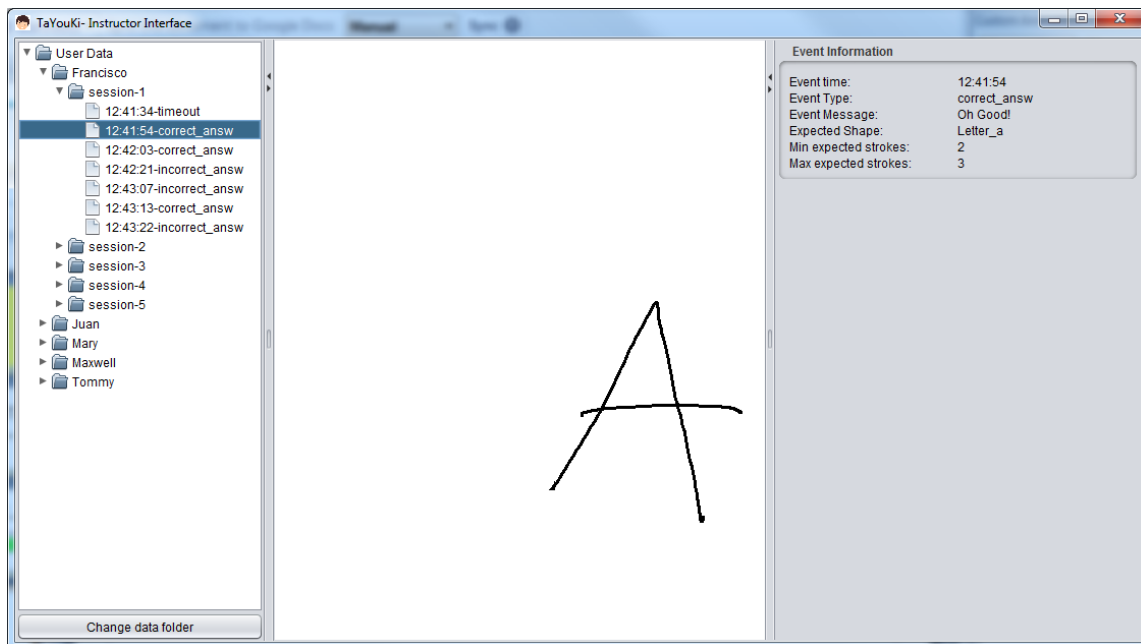


Figure 25 Instructor interface for the activity log

Figure 25 shows a screenshot of the activity log as viewed by the instructor. Each user has its own folder that contains one or more activity sessions. In each activity session we find a chronological history of the different events that occurred in the system while the kids were interacting with it. The instructor can see what triggered each event, and what the agent responded at all times. For example the instructor can see the drawing of a square drawn by the student when he was expected to draw a circle and what the agent responded. The instructor can follow the progress of the learner and see how he reacted to particular feedback.

Post-Testing

Following the completion of the interaction with the system, we provided the children again with crayons and paper. The original idea was to perform a summative

assessment to determine if the performance objective was met or not. However the period of time the users interacted with the system was limited. And the instructors (parents) were not familiarized with the tool to include it as part of structured instruction.

Instead of proportioning a tool for summative assessment and evaluation of our system, we distributed questionnaires to the parents with a variety of Likert scale questions that address relative factors such as ease of use, time of completion of specific instructional tasks, perceived effectiveness of learning of the children, as well as freeform questions expressing their opinions of diverse factors of the system and comparisons to traditional instruction for the targeted lessons. We tried to answer these questions in the presence of the participant child so the parent would serve as mediator for his answers. E.g. “Did you like the little boy?” “Did you like the pshh pshh sounds he made?” A sample post-test questionnaire can be seen in Appendix B.

Results

Results for this part of the evaluation were somewhat more anecdotal and will be discussed in detail in the discussion section. However the reader is encouraged to take a look at Appendix A, which contains some of the user logs, which show the interactivity of the system and provide rich and valuable information which can be analyzed in detail.

DISCUSSION

There is plenty to discuss at this point, and a random brainstorm of ideas can become uneasy to read in this section. We decided to guide this discussion using the same scope domains and research questions we proposed in our evaluation section. We continue with some closing remarks in the conclusion section.

Sketch Recognition

What Is the Accuracy of Each Recognizer?

We saw in the evaluation section that for the data we collected using SOUSA we had average accuracies of above 90% for both recognizers. An average accuracy gives is a good glance, but we lost information by aggregating this information which is very valuable to look in detail. In particular we can see in Table 10 that simple primitives such as the circle or line get perfectly recognized by both recognizers, while more complex shapes such as the car are misrecognized more often.

Which Recognition Approach Works Better for Our Case?

We saw in Table 10 that the vision-based recognizer performed better for the shape set we propose. The geometric recognizer followed close by. But more importantly we need to note how the recognizers complement each other. In shapes such as the line or the square these are very clearly defined in a geometric perspective, so the geometric recognizer showed better accuracies. We saw in our case that there is no single better approach; the best thing to do is to take both recognizers into account. And if we do this we need to be careful about how we blend this information, a simple

average of the accuracies would be misleading. We process the results separately and use this to give more tailored feedback.

What Can Be Improved?

We have not reached perfection in any of our recognizers. And to be practical it is not reasonable to think we will. Even human perception fails on some shapes and learners always come up with new ways of drawing. Although we have noticed some points that can be fixed with more work, such as the low level treatment of closed shapes, we think that a more important issue at hand is how to fail when we do. A system that fails gracefully still provides the sense of intelligence and can be a powerful instructional tool. We want to focus the reader attention to the confusion columns in Table 10. Note that the geometric recognizer often misinterprets complex shapes by one of its composing primitives. Contrary the visual recognizer confuses more randomly with shapes of similar bounding box. We can think that the information we get from a geometric recognizer will allow us to fail more gracefully than the visual misrecognition. For example if we are expecting a car, and the recognition fails, meaning that the submitted shape is visually acceptable but the system says is incorrect, it is more natural to say “I cannot see a car yet, but I see a circle”... that “ this looks more like a spiral”

Is the Accuracy Similar for Kids?

To answer this question we do not possess a large enough number of participants to make a significant quantitative comparison. However the user studies did give us some important insights about this. An important one is that the answer to this question will vary greatly depending on the age level. For the six-year old, the accuracies mapped

one-to-one with those of the adults. It was actually better than the reported accuracy. The four-year olds had a slightly lower recognition rate, because even when they drew something an instructor might classify as correct, it was still very messy for the system, with high rotation and proportion variance and disconnected primitives and wiggly strokes. However it actually amazed us how many times the system accepted some of these shapes which were fairly messy. Three-year olds did not provide us with significant data to talk about accuracy of recognition. They barely drew the simplest shapes and the complex ones were mostly scribbles.

What Features Are Most Relevant?

We found out that there are indeed some features that seem to be common in younger kids that disappear over-time. This is particularly useful for three-year olds and should be studied further. We found that these young learners tend to draw longer line segments (high total stroke length), also high density points and a higher average speed. Geometrically it was mostly over tracing lines what they could draw. The sketches had little or no curvature.

What Are the Major Challenges?

As we said before it is unrealistic to think about perfect recognition when we have such a subjective domain. Even in adults it is a challenging task to give semantic to some drawings. We consider our main challenge is how to make the system aware of its own limitations such that the feedback is the most useful at all times. If the system recognizes a circle with 40% confidence it might be better to say “this reminds me of a

circle, but I am not sure. . . . Can you draw it again?” that mistakenly label a square as a circle and give the wrong message to the learner.

Computer Human Interaction

Was the Feedback Method Appropriate and Useful? (Sounds, Pedagogical Agent..)

Children found the sounds very appealing. They smile with them and try to mimic them. We also found that facial expressions make the software more personal for the kids. We infer this because they smile back with pride when the agent smiles. Or say phrases such as “*Why is he mad at me?*”, “*How does he know my name?*”

Do the Kids Find the Software Engaging? For How Long Do They Remain Engaged?

We found that this very much depends on the age and skill level of the user. Out of the few children we tested with, we observed it is more engaging for ages four than to three or six. Both four year olds remained engaged during the complete time of the activity, actively drawing and responding to feedback. Three-year olds, were also very engaged at first when they approached the system. Especially the sounds seemed to be very appealing for them. However they soon became frustrated that they could not draw any of the shapes to make the agent happy. The six year old was no exception when first approaching the system. She was very happy and engaged with it for the first few minutes. But she went through all the shapes in a few minutes and then repeated them a couple of times, but she was bored after that expecting more challenge.

What Is The Children's/Teachers' Attitude Toward the Use of Digital Pens?

Pen interaction was preferred over mouse and keyboard. This was a consensus we had with their parents. All of them agreed this was more natural for the kids than using the keyboard at this point.

Was the Timing Appropriate?

This was very challenging, and during our first user study it was completely inappropriate, becoming into an annoying boy that gave warnings and complained every five seconds, often interrupting a work in progress by the learners. However when we first tried to fix it became too sparse giving the impression it gave no feedback at all. We attempted to fix this by adding to important features: an explicit feedback request button and a *patience* variable that can be set by parents or instructors and would determine how often feedback is provided. We found that in general timing is challenging to fix at a universal point that will fit everyone, a fine tune of the patience was likely required.

Education

Are Kids Reaching One of the Defined Instructional Goals by Using Our System?

This is the ultimate question we would like to answer. This would be our summative evaluation that will determine if our system should be used or not compared to alternative methods. However, at this point our system is still in the road of improvement and attempting to answer this question under the current progress might be harmful, as we can be discarding a potentially powerful solution because of its lack of maturity. This said, we can use our qualitative data to say in an anecdotal fashion that this tool got kids engaged and they were encouraged to draw in order to see the happy

face of the agent. In our small sample set of participants this was the case and we think that together with additional tools it can conform part of a robust solution for an instructional need.

Does the Digital Pen Interaction Resemble Enough to Regular Pen/Paper ? (Delivery Method and Transfer Learning)

The digital pens, especially Wacoms with a back screen where intuitive and easy to use for kids, as they went from paper to screen almost transparently. However, some felt that a step further was still needed as the pens we used still did not have some of the affordances and characteristics of crayons or other real objects. Particularly when testing with resistive screens such as the Toughbook, some kids were unable to continuously input the required pressure needed to draw.

Were the Feedback Messages Appropriate and Useful?

We noticed that many of the messages were not necessarily being paid attention by the kids, especially the young ones. However the low level sounds, audio fx and facial expressions were noticed and celebrated by all of them. . The parents of the kids also commented that at some points the messages were not the most convenient to encourage motivation. The agent went “mad” too quickly causing more plain / less joyful messages. User two asked at some point “Why is he getting mad at me mom? I drew a house as he asked” which let us understand they were receptive to the emotional state of the agent which is what we were looking for, but that in this case it provided a somewhat negative effect due to the tight recognition.

Where the Questions (Shapes) Relevant and Easy to Understand?

The shape set was limited in this prototype, it was too complex for three-year olds, too easy for six-year olds. A priority in future work would be to expand this content so this is not an issue, specifically adding more shapes accompanied by a cognitive developmental rating. Sample shapes would include simpler shapes such as simple scribbles, shape tracing, and shapes to fill in. The system would then be able to automatically update to the correct developmental shape level.

General Discussion

The formative evaluation was a key component of our system. The decision of having a Wizard of Oz method at this stage was a very good decision. At this point we realized many weaknesses in the recognizer that gave incorrect feedback messages even though the answer was visually *fairly close*. Fortunately we could overwrite the agent's decision on runtime and continue with the study. Similarly we caught some weaknesses in the interactivity at this point.

In a future version of the system we could add server support for recording daily activities and more than one “mini-game” that can be scored, driving the children towards particular goals set by the instructor. All these mini-games would share the presence of an avatar and the use of sketch-based capabilities. In order to scale the development robustly and efficiently, a potential development approach would be to build a common API for the development of new games and the inclusion of extra content.

CONCLUSIONS

In this project we have developed a prototype of an intelligent tutoring system. This system employs different sketch recognition techniques to aid in teaching drawing skills for early learners. The interactivity of five target learners with the system was closely analyzed to identify its main strengths and weaknesses. These case studies served to gain important insights regarding the research challenges in different domains (CHI, Sketch Recognition, and Education).

We found out that when dealing with children we want to be loose enough about correctness to keep them encouraged, yet give them feedback on whatever they can improve (as opposed as in what they did wrong). We also need to handle multiple interpretations and use different recognizers to consider scenarios that are very seldom in adults but more frequent on children. In our system we developed and used a geometric recognizer of a set of 10 symbols that had the added value of giving low-level feedback. This was appreciated and used by the children we tested the system with. We also counted with a pedagogical agent with sound and facial expressions which brought the user closer to the system.

We performed formative evaluation for a tool that has great potential as an aid to teach drawing skills. In order to do this we built a prototype that serves as proof of concept and allows us to identify ways to address major cognitive differences that result. We found for instance that the core drawing development happens during a small time window. Children make a lot of progress from age three to six and this implies that the

content must adapt to the current user. And we also found that timing and content of the feedback needs to be treated with care as it might easily become annoying or discourage the children.

In summary, we built a prototype of a system that is in its early stages but that already shows great potential as a teaching tool that can effectively and efficiently help literacy and drawing instructors in the complex task of teaching children the required psychomotor skills that are so fundamental for their further development. This project was an important milestone towards that goal as we gained important insights and established the foundations for a novel solution for a real instructional need.

REFERENCES

1. Alvin, A., De la Cruz, R., Fonseca, D., and Samonte, M. Basic Handwriting Instructor for Kids Using OCR as an Evaluator. *Networking and Information Technology (ICNIT), 2010 International Conference on*, IEEE (2010), 265–268.
2. Anning, A. Learning to Draw and Drawing to Learn. *International Journal of Art at Design Education* 18, 2 (1999), 163-172.
3. Bear, D.R. “Learning to Fasten the Seat of My Union Suit Without Looking Around ”: The Synchrony of Literacy Development. *Theory Into Practice* 30, 3 (1991).
4. Chaiklin, S. The Zone of Proximal Development in Vygotsky’s Analysis of Learning and Instruction. *Vygotsky’s Educational Theory in Cultural Context*, (2003).
5. Demirbilek, M., Yilmaz, E., and Tamer, S. Second Language Instructors’ Perspectives About the Use of Educational Games. *Procedia - Social and Behavioral Sciences* 9, (2010), 717-721.
6. Dick, W., Carey, L., and Carey, J.O. *The Systematic Design of Instruction*. Upper Saddle River, NJ, 2009. Pearson.
7. Dinesha, V. NeuronDotNet. 2008. <http://sourceforge.net/projects/neurondotnet/>.
8. Dixon, D. and Prasad, M. iCandraw: Using Sketch Recognition and Corrective Feedback to Assist a User in Drawing Human Faces. *in CHI’10 Proc. of the 28th International Conference on Human Factors in Computing Systems*, (2010).
9. Dyson, A. The Emergence of Visible Language: Interrelationships Between Drawing and Early Writing. *Annual Meeting of the American Educational Research Association*, (1983).
10. Ekman, P. Facial Expression and Emotion. *The American Psychologist* 48, 4 (1993), 384-92.
11. Eran, D. Newton Lessons for Apple’s New Platform. *Roughlydrafted.com*, 2006. <http://www.roughlydrafted.com/RD/Q4.06/600D65E6-A31E-45CA-AFC5-42BC253F5337.html>. Accessed Dec. 18, 2011
12. Fleetwood, M.D., Byrne, M.D., Centgraf, P., Dudziak, K.Q., Lin, B., and Mogilev, D. An Evaluation of Text-Entry in Palm OS - Graffiti and the Virtual

- Keyboard. *in Proc. of the Human Factors and Ergonomics Society Annual Meeting*, Human Factors and Ergonomics Society (2002), 617-621.
13. Hammond, T. and Davis, R. LADDER, A Sketching Language for User Interface Developers. *Computers & Graphics* 29, 4 (2005), 518-532.
 14. Hartman, L. and Kabanza, F. An Intelligent Tutor for Tele-Robotics Training. *AIAA Proc.*, (2009), 1-6.
 15. Hubal, R. A Mixed-Initiative Intelligent Tutoring Agent for Interaction Training. *presented at the Intelligent User Interface Conference*, (2001).
 16. Johnson, G. and Do, E.Y.-L. Games for Sketch Data Collection. *in Proc. of the 6th Eurographics Symposium on Sketch-Based Interfaces and Modeling - SBIM '09*, (2009), 117.
 17. Kara, L.B. and Stahovich, T.F. An Image-Based Trainable Symbol Recognizer for Sketch-Based Interfaces. *in AAAI Fall Symposium Series 2004: Making Pen-Based Interaction Intelligent and Natural*, (2004), 99-105.
 18. Kelley, J.F. A Natural Language Program Developed With the OZ Paradigm: Implications for Supercomputing Systems. *First International Conference on Supercomputing Systems*, (1985).
 19. Khine, M. Core Attributes of Interactive Computer Games and Adaptive Use for Edutainment. *Transactions on Edutainment I*, (2008).
 20. Kim, H. and Roh, Y. An Immersive Motion Interface With Edutainment Contents for Elderly People. *Motion in Games*, (2008), 154-165.
 21. LaViola, J.J. An Initial Evaluation of MathPad2: A Tool for Creating Dynamic Mathematical Illustrations. *Computers & Graphics* 31, 4 (2007), 540-553.
 22. McKinnon, I. Children's Music Journey: The Development of an Interactive Software Solution for Early Childhood Music Education. *Computers in Entertainment (CIE)* 3, 4 (2005), 1-10.
 23. Meyer, A. Pen Computing: A Technology Overview and a Vision. *ACM SIGCHI Bulletin*, (1995).
 24. NCES, N.C. for E.S. NAEP Validity Studies: Computer Use and Its Relation to Academic Achievement in Mathematics. 2003. nces.ed.gov/pubs2003/200315.pdf. Accessed Nov. 11, 2011

25. NCES, N.C. for E.S. Public Schools and Instructional Rooms With Internet Access, By Selected School Characteristics. 2005, Table 413. http://nces.ed.gov/programs/digest/d07/tables/dt07_413.asp. Accessed Nov. 11, 2011
26. Narvaez, L., Brimijoin, K., and Tomlinson, C.A. *Differentiation at Work, K-5: Principles, Lessons, and Strategies*. Corwin, Thousand Oaks, CA 2010.
27. Paulson, B., Eoff, B., Wolin, A., Johnston, J., and Hammond, T. Sketch-Based Educational Games: Drawing Kids Away From Traditional Interfaces. *in Proc. of the 7th International Conference on Interaction Design and Children*, ACM (2008), 133–136.
28. Paulson, B., Wolin, A., Johnston, J., and Hammond, T. SOUSA : Sketch-based Online User Study Applet. *in Eurographics 5th Annual Workshop on Sketch-Based Interfaces and Modeling*, (2008), 81-88.
29. Paulson, B. and Hammond, T. PaleoSketch: Accurate Primitive Sketch Recognition and Beautification. *in Proc. of the 13th International Conference on Intelligent User Interfaces*, ACM (2008), 1–10.
30. Pereira, J., Carriço, L., and Duarte, C. Improving Children’s Writing Ability. *in Proc. of the 13th International Conference on HumanComputer Interaction Part IV Interacting in Various Application Domains 5613*, (2009), 186-195.
31. Plamondon, R. and Srihari, S.N. On-Line and Off-Line Handwriting Recognition : A Comprehensive Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* 22, 1 (2000), 63-84.
32. Plimmer, B. and Apperley, M. Computer-Aided Sketching to Capture Preliminary Design. *Australian Computer Science Communications*, Australian Computer Society, Inc. (2002), 9–12.
33. Press, L. Dynabook Revisited - Portable Computers Past, Present and Future. *Communications of the ACM* 35, 3 (1992), 25-31.
34. Rapeepisarn, K., Wong, K.W., Fung, C.C., and Depickere, A. Similarities and Differences Between Learn Through Play and Edutainment *.in Proc. of the 3rd Australasian Conference on Interactive Entertainment*, Murdoch University (2006), 28-32.
35. Reeve, J., Jang, H., Carrell, D., and Jeon, S. Enhancing Students’ Engagement by Increasing Teachers' Autonomy Support. *Motivation and Emotion* 28, 2 (2004), 147-169.

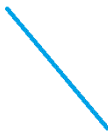

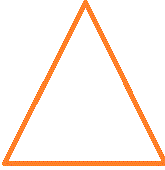






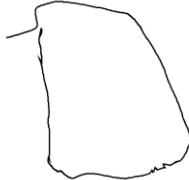
36. Rubine, D. Specifying Gestures by Example. *ACM SIGGRAPH Computer Graphics* 25, 4 (1991), 329-337.
37. SAMSUNG. Samsung Galaxy Note. 2012. www.samsung.com/galaxynote. Accessed Feb. 11, 2012
38. Sezgin, T.M., Stahovich, T., and Davis, R. Sketch Based Interfaces : Early Processing for Sketch. *in Proc. of PUI-2001. NY, (2001)*.
39. Shiratuddin, N. and Landoni, M. Evaluation of Content Activities in Children's Educational Software. *Evaluation and Program Planning* 25, 2 (2002), 175-182.
40. Smart Technologies. Smart Board. 2012. <http://smarttech.com/smartboard>. Accessed March. 15, 2012
41. Subrahmonia, J. Pen Computing: Challenges and Applications. *Pattern Recognition, 2000.*, (2000), 60-66.
42. Sutherland, I. Sketchpad: A Man-Machine Graphical Communication System. *in Proc.-Spring Joint Computer Conference, Michigan, (1963)*, 329–346.
43. Tsuguya, O., Takahiko, K., and Toshiaki, O. Position Detecting Apparatus. U.S. Patent 1989.
44. Valentine, S., Smith, A., and Hammond, T. A Sketch Comparison Technique for Use in Sketch-Based Tutoring Systems. *2011 Intelligent User Interfaces (IUI 2011) Workshop on Sketch Recognition, (2011)*.
45. Vanlehn, K., Lynch, C., Schulze, K., Shapiro, J., Shelby, R., et al. The Andes Physics Tutoring System: Lessons Learned. *International Journal of Artificial Intelligence in Education* 15, 3 (2005), 147–204.
46. Varley, P. and Company, P. Automated Sketching and Engineering Culture. *VL/HCC Workshop: Sketch Tools for Diagramming, September (2008)*, 83-92.
47. Vygotsky, L.S., Cole, M., John-Steiner, V., Scribner, S., Souberman, E., and Wertsch, J.V. Mind In Society: The Development of Higher Psychological Processes. *American Anthropologist* 92, 1978, 956-957. <http://psycnet.apa.org/psycinfo/1979-28227-000>.
48. Walker, W., Lamer, P., and K. wok, P. FreeTTS. 2005. <http://freetts.sourceforge.net/>. Accessed Nov. 28, 2011

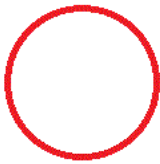
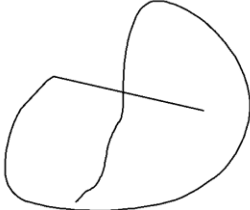
49. Wang, D. and Li, J. Usability Evaluation of Children Edutainment Software. *Usability and Internationalization. HCI and Culture*, (2007), 622-630.
50. Wang, D., Ying, T., Xiong, J., and Wang, H. A Pen-Based Teaching System for Children and Its Usability Evaluation. *Human-Computer Interaction.*, (2009), 256-265.
51. Wobbrock, J. and Wilson, A. Gestures Without Libraries, Toolkits or Training: A \$1 Recognizer for User Interface Prototypes. *in ACM Symposium on User Interface*, (2007).
52. Wolin, A., Eoff, B., and Hammond, T. Shortstraw: A Simple and Effective Corner Finder for Polylines. *5th Annual Workshop on Sketch-Based*, (2008).
53. Yang, F.J. The Ideology of Intelligent Tutoring Systems. *ACM Inroads 1*, 4 (2010), 63–65.
54. Yang, H.-C. The Developmental Characteristics of Four- and Five-Year-Old Pre-Schoolers' Drawing: An Analysis of Scribbles, Placement Patterns, Emergent Writing, and Name Writing in Archived Spontaneous Drawing Samples. *Journal of Early Childhood Literacy 6*, 2 (2006), 145-162.

APPENDIX A: USER LOGS

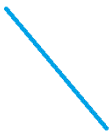

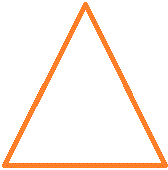



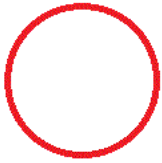
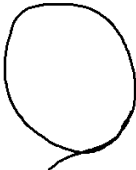

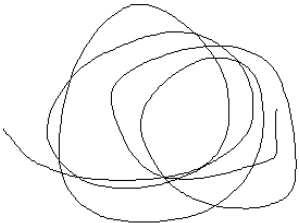
In this appendix we summarize some of the interactions the children had with our system during the user studies.



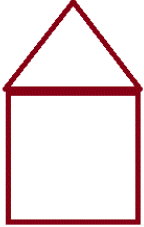
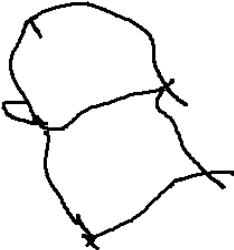
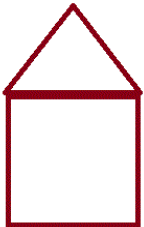

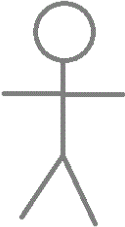

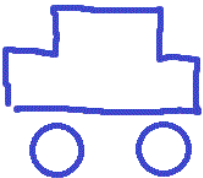
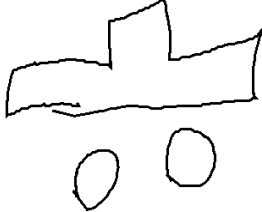
User 1 – Age 3





<i>Event</i>	<i>Expected</i>	<i>Drawn</i>	<i>Message</i>
CORRECT			Very Good!
INCORRECT			I see lines, but not a triangle
INCORRECT			That is not a square
INCORRECT			Mhh, not quite right!
WIZARD			Much Better!

INCORRECT			Mhh, this is not a circle!
-----------	---	--	----------------------------



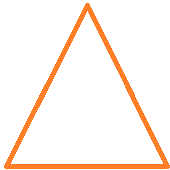


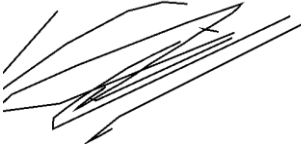
User 2 – Age 4

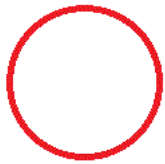



<i>Event</i>	<i>Expected</i>	<i>Drawn</i>	<i>Message</i>
CORRECT			Very Good!
INCORRECT			You have drawn many lines, but that is not a triangle
CORRECT			Good!
CORRECT			Great!!
INCORRECT			Oops Let's Try again!!

CORRECT			WoW!! You are Amazing!!
CORRECT			Not quite right, keep trying!
WIZARD_INTERRUPT			Much better!!
INCORRECT			Great progress, but it seems like you need to draw more lines!
SKIP			
CORRECT			Great!



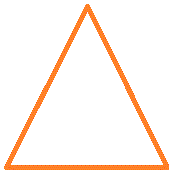


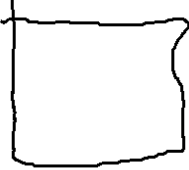
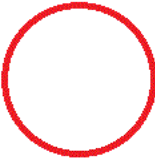

INCORRECT			You must be near, finish your drawing!
CORRECT			Oh! There you go!



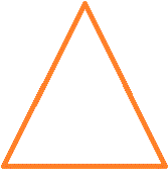

User 3 – Age 3

<i>Event</i>	<i>Expected</i>	<i>Drawn</i>	<i>Message</i>
CORRECT			Very Good!
INCORRECT			This does not look like a triangle, keep trying.
SKIP			
CORRECT			This does not look like a square, keep trying!
SKIP			

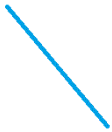
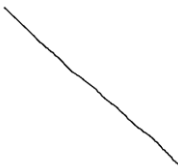

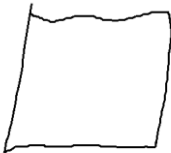
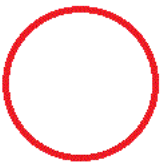
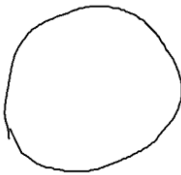


CORRECT			Not quite right, try again!
SKIP			
CORRECT			Ohh you must be near but I asked for a spiral

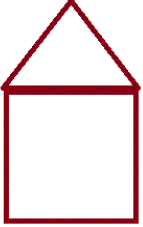
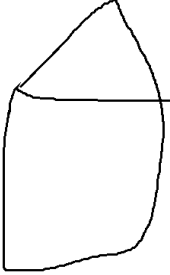
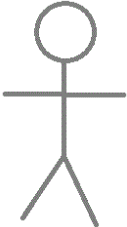

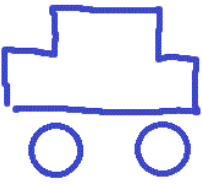





User 4 – Age 4

<i>Event</i>	<i>Expected</i>	<i>Drawn</i>	<i>Message</i>
CORRECT			Very Good!
CORRECT			Yes!! Doing Great!
CORRECT			Fantastic!! You are very smart!!
CORRECT			Wow!! You are so intelligent!!

INCORRECT			Ohh you must be near but I asked for number one
SKIP			
CORRECT			Very Good!

User 5 – Age 6

<i>Event</i>	<i>Expected</i>	<i>Drawn</i>	<i>Message</i>
CORRECT			Very Good!
CORRECT			Yes!! Doing Great!
CORRECT			Wow!! You are so intelligent!!
CORRECT			WoW!! You are Amazing!!

CORRECT			Fantastic!! You are very smart!!
INCORRECT			Great progress, but it seems like you need to draw more lines!
SKIP			
CORRECT			Great!
INCORRECT			You must be near, finish your drawing!
CORRECT			Oh! There you go!

APPENDIX B: ASSESSMENT INSTRUMENTS

Pre-Test Form

Please perform the following activities prior to using the software. You can use regular pen and paper or a pen-inputted computer and a commercial canvas such as Microsoft Paint. First show them the shape cards in the assessment materials section, either printed or on the screen. Then, have them to get used to drawing as a warm up. While you are doing this, please answer the following questions:

1. The learner seems to understand the given instructions such as “Grab that pen”, “Try again”, “Draw your favorite shape on the paper”

Yes___ No___

2. When you ask the student for names of any geometric shape, letter or number, the student named the following:

1. _____

2. _____

3. _____

4. _____

3. When you show the learner the shape cards, he/she verbally named the following images accordingly. (Check all that apply)

Line

Circle

Triangle

Square

Spiral

Car

Letter A

Number 1

House

Person/Boy

4. Given a pen or crayon and a piece of paper, the student can draw ink strokes on it when requested. If available he/she can do the same with a digital screen and a stylus. (Check all that apply)

Pen & Paper Stylus

5. When requested, the learner correctly draws the following shapes: (Please score using your own visual perception the drawing of each shape in a scale from 1 to 5, being 5 a nearly perfect shape, 1 a completely unrelated drawing)

	Score				
Line	1	2	3	4	5
Circle	1	2	3	4	5
Triangle	1	2	3	4	5
Square	1	2	3	4	5
Spiral	1	2	3	4	5

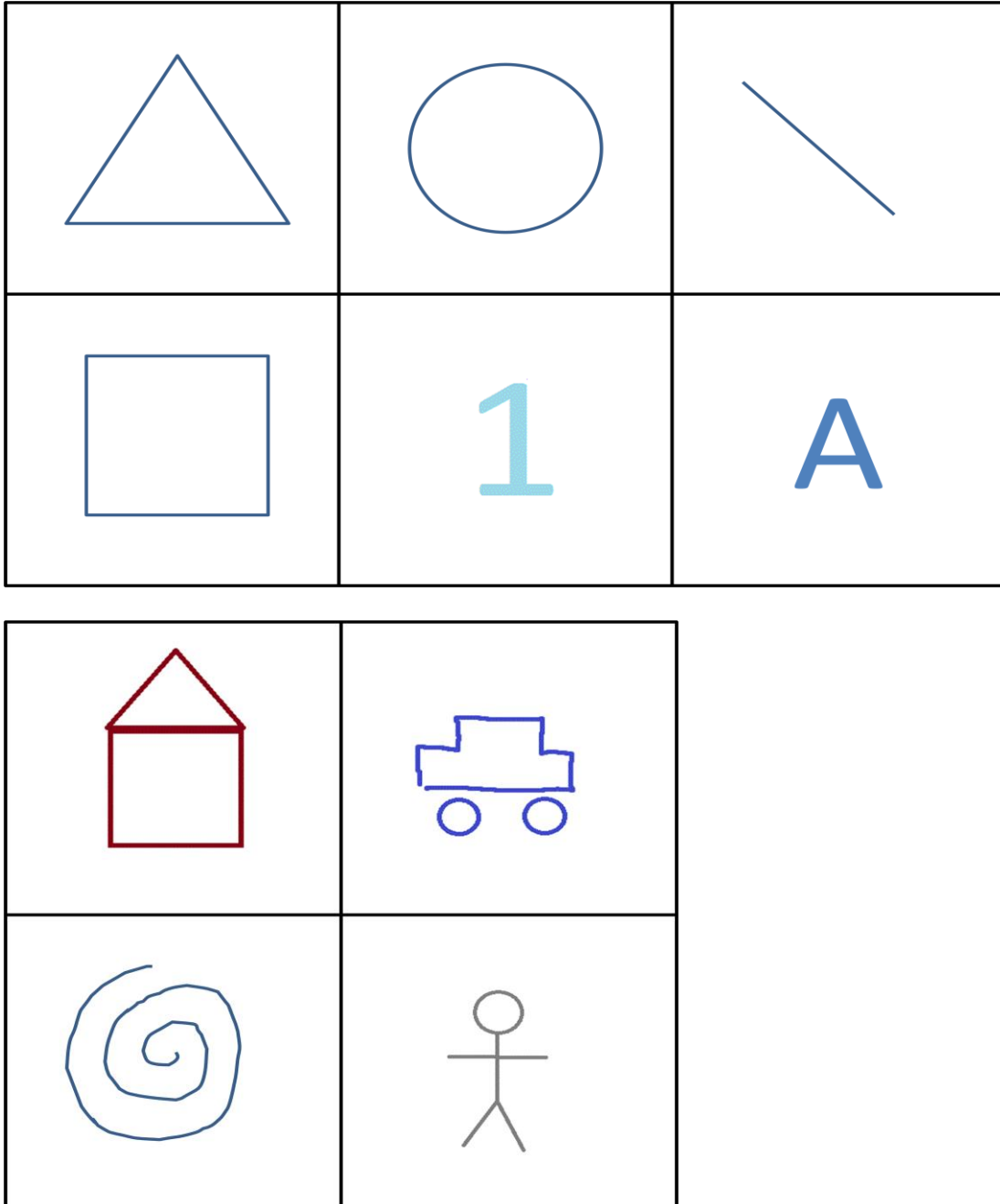
6. In the cases where the score was below 5(if any) please asses how receptive the learner was to your feedback in a scale from 1 to 5 (1 no receptive at all, will not listen to any instructions, 10 very receptive took proper corrective action)

Receptiveness to feedback

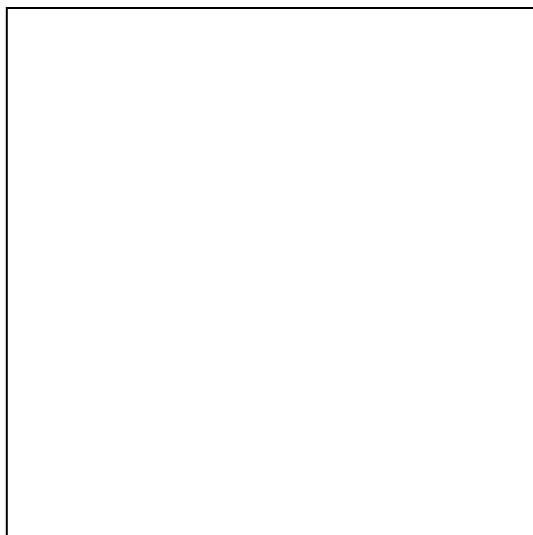
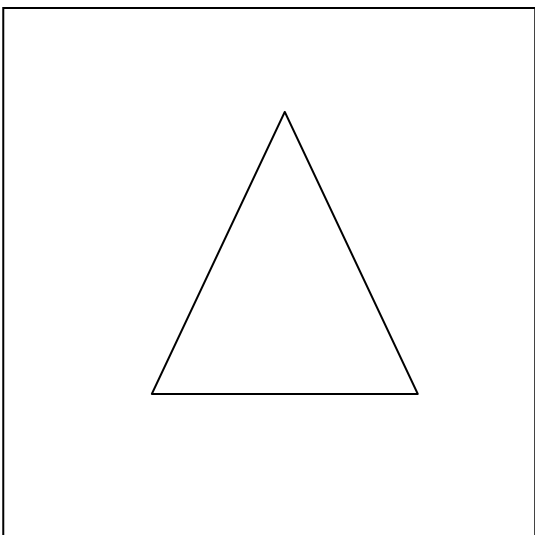
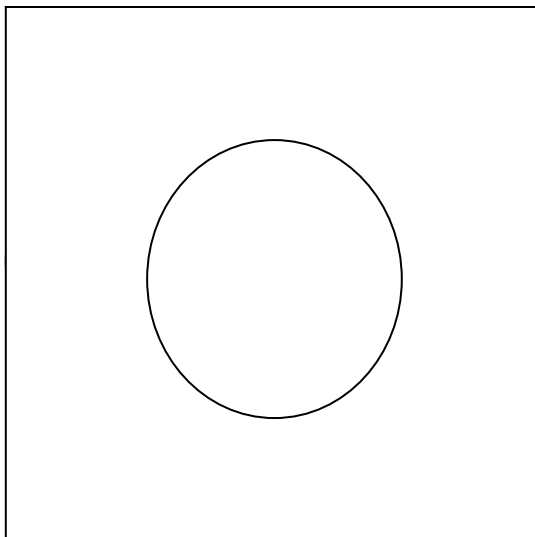
1	2	3	4	5
---	---	---	---	---

Assessment Materials

Printable shapes to show to students



1. Draw the figure on the left on the box in the right



Nickname of the student _____
Post-Test Form

Please fill out this form after using the software, based on your experience and that of the kid.
 For each item identified below, please circle the number to the right that best fits your
 judgment. Being 1 the lowest score (completely disagree) and 5 the highest (completely agree)

Description/Identification of Survey Item	Scale				
1. The instructor is experienced with computers	1	2	3	4	5
2. The software is easy to understand and use for the instructor	1	2	3	4	5
3. The software is easy to understand and use for the children	1	2	3	4	5
4. The children were engaged with the software	1	2	3	4	5
5. The software can help the children learn	1	2	3	4	5
6. The children seem to prefer this kind of interaction over traditional mouse and keyboard	1	2	3	4	5
7. The children retain attention for longer periods when using the system compared to similar pen and paper activities.	1	2	3	4	5
8. The feedback messages of the system were understood by the kid	1	2	3	4	5
9. The feedback messages of the system were appropriate for the kids	1	2	3	4	5
10. The feedback timing was appropriate	1	2	3	4	5
11. The kid found the animated character on screen appealing.	1	2	3	4	5
12. The sounds played by the system depending on the primitive (line, circle...) were engaging and useful for instruction.	1	2	3	4	5
13. The shapes that were correctly drawn by the learner were correctly recognized by the system	1	2	3	4	5
14. The overall system serves its purpose as an educational aid for the instructor in teaching shapes, letters and numbers	1	2	3	4	5

Please leave us your additional comments and ideas.

APPENDIX C: FEEDBACK MESSAGES

Here we show the messages TAYouKi uses for feedback in this prototype version. Note that these messages can be easily expanded or edited using any text or XML editor.

LOST_ENGAGEMENT

-NEUTRAL

*Hello! Hello! Are you there??

-HAPPY

*What happened buddy? Are you there??

-DISSAPOINTED

*Are you there? Am I talking alone?

-IMPATIENT

*Hello! Is my little friend there??

-ANGRY

*It seems you left me alone

-CONFUSED

*My friend! Are you there??

-EXCITED

*Oh! come on!!! we were having fun!

CORRECT

-NEUTRAL

*Very Good!

-HAPPY

*Yes!! Doing Great!

-DISSAPOINTED

*That is better!

-IMPATIENT

*Oh Good!

-ANGRY

*Ok...there it is!!

-CONFUSED

*Oh! There you go!

-EXCITED

*Wow!! You are Amazing!!

*Fantastic!! You are very smart!!

*Wow!! You are so intelligent!!

VISUALLY_INCORRECT

-NEUTRAL
 *Looks funny, but is a valid \$expectedShape\$

-HAPPY
 *You are so creative! that is a \$expectedShape\$ that looks like a \$foundShape\$

-DISSAPOINTED
 *I will pass this one despite how it looks

-IMPATIENT
 *Ok, lets do the next

-ANGRY
 *Not what I was expecting, but lets move on

-CONFUSED
 *Is that a \$expectedShape\$?? Ok, if you say so

-EXCITED
 *Genius, it looks like a \$foundShape\$ but is a valid \$expectedShape\$

GEOMETRICALLY_INCORRECT

-NEUTRAL
 *Funny way of drawing, but is a valid \$expectedShape\$

-HAPPY
 *You are so creative! Interesting way of drawing a \$expectedShape\$

-DISSAPOINTED
 *I will pass this one despite how you drew it

-IMPATIENT
 *Ok, lets do another one

-ANGRY
 *Not the \$expectedShape\$ I was expecting, but lets move on

-CONFUSED
 *Is that a \$expectedShape\$?? Ok, looks like it

-EXCITED
 *Genius, it looks like a \$foundShape\$ but is a valid \$expectedShape\$

INCORRECT_SHAPE

-NEUTRAL
 *Nice \$foundShape\$! Can you draw a \$expectedShape\$?

-HAPPY
 *Oops! lets try again

-DISSAPOINTED
 *That is not a \$expectedShape\$, keep trying

-IMPATIENT

*Hey! You are back! but that is not a \$expectedShape\$.
Try Again.

-ANGRY

*I asked you to draw a \$expectedShape\$. I am sure you
can do it

*I know you can do it better

*That looks like a \$foundShape\$. I asked for a
\$expectedShape\$

-CONFUSED

*Ehh... that seems like a \$foundShape\$

-EXCITED

*Ohh, you must be near but you can do it much better

TOO_MANY_STROKES

-NEUTRAL

*too many lines my friend

-HAPPY

*Oops! You have drawn too many lines, lets try again

-DISSAPOINTED

*Easy Cowboy! you have drawn too many lines!

-IMPATIENT

*Easy. Easy...You drew too many lines

-ANGRY

*NO. NO. my little friend! You have drawn too many
lines

-CONFUSED

*Uhhh... seems like you have drawn too many lines

-EXCITED

*Ohh you must be near, but you have drawn too many
lines, erase some

TOO_FEW_STROKES

-NEUTRAL

*Almost there! you are missing something!

-HAPPY

*Oops! You have drawn few lines, keep trying. I know
you can

-DISSAPOINTED

*too few lines, lets do it right

-IMPATIENT

*Hey buddy... seems like you have drawn too few lines

-ANGRY

*NO. no. my little friend!! You have drawn too few
lines

-CONFUSED

*Uuhmmm!!... seems like you need to draw more lines
-EXCITED
*Almost there my friend. You must be near. Try your
best
EMPTY_SKETCH
-NEUTRAL
*You are just awesome. Can you draw something for me!
-HAPPY
*Hey! Remember to draw something first
-DISSAPOINTED
*Lets have fun, but you should draw something first
-IMPATIENT
*Come on buddy! It will be funny. Just draw something
-ANGRY
*Is this a drawing of the invisible man? Draw
something for me!
-CONFUSED
*Well! I do not see anything. Can you can draw
something?
-EXCITED
*Draw your beautiful art before pressing the button

VITA

Francisco Alfonso Vides Cerón is a graduate researcher at the Sketch Recognition Lab at Texas A&M University. He was born in Bogota, Colombia where he received a double major from Los Andes University: A bachelor of science in electrical engineering and a bachelor of science in computer science. With this work, he is completing a Master's degree in computer science at Texas A&M University. His research interests are in Computer-Human Interaction (CHI), Artificial Intelligence (AI), and Intelligent Tutoring Systems (ITS).

Francisco may be reached at the Sketch Recognition lab in Texas A&M University. Teague 327 3112 TAMU, College Station, Texas, 77840 or emailed at pacovides@gmail.com