

Virtual laboratories in education

by

Roy Eli Kfir

Abstract

Virtual laboratories in education

by

Roy Eli Kfir

Submitted in partial fulfillment of the requirements for the degree Magister Scientiae
in the Faculty of Natural and Agricultural Sciences

University of Pretoria

Pretoria

October 2002

Supervisor: Dr. V. Lubon
Department of Computer Science

Submitted in partial fulfillment of the requirements for the degree Magister Scientiae

1 708.200 2995
NEW 1200182

Virtual laboratories in education

by

Roy Eli Kfir

Abstract

This thesis focuses on the fields of education and computer graphics. Teaching and learning with computers is becoming an important part of the classroom scenario. For educational purposes virtual reality has been proposed as a technological breakthrough that holds the power to facilitate learning. However, virtual reality applications produced for educational purposes are usually of fixed implementations that require programming and technical background. They are also limited for use with expensive virtual reality equipment.

The Intelligent Tiles framework developed in this study attempts to overcome many of the above-mentioned limitations. It is a generic and adaptable framework that supports teacher driven development of virtual laboratories. By using this framework, virtual laboratories can easily be authored and be used in the education of young learners in teaching topics in earth science, such as ecosystems. The simulation of these virtual laboratories allows young learners to explore, understand and gain mathematical skills such as counting, sorting and classification as well as learning ecological concepts and relationships between different animals and plant life. The framework also has the advantage of being affordable, since it is developed for a desktop virtual reality solution, considered the least expensive and most widely used of all virtual reality platforms.

This thesis shows that by using the Intelligent Tiles framework a user-friendly, easily employable and affordable educational tool can be developed, empowering teachers and making learning more fun and effective.

Thesis supervisor: Dr. V. Lalioti

Department of Computer Science

Submitted in partial fulfilment of the requirements for the degree Magister Scientiae

Virtual laboratories in education

deur

Roy Eli Kfir

Opsomming

Hierdie skripsie fokus op die gebiede van opvoedkunde en rekenaar grafieka. Onderrig met behulp van 'n rekenaar is besig om 'n belangrike deel van die klaskamer te word. Vir opvoedkundige doelwitte is virtuele realiteit voorgestel as 'n tegnologiese deurbraak wat gebruik kan word vir gefasiliteerde onderrig. Die virtuele realiteit programme wat vir opvoedkundige doelwitte geproduseer is, benodig gewoonlik programmering en tegniese agtergrond. Hulle is ook beperk tot die gebruik van duur virtuele realiteit toerusting.

Die "Intelligent Tiles" raamwerk wat gedurende hierdie studie ontwikkel is, probeer baie van die bogenoemde beperkinge oorkom. Dit is 'n generiese en aanpasbare raamwerk wat onderwyser-gedrewe ontwikkeling van virtuele laboratoriums ondersteun. Deur hierdie raamwerk te gebruik, kan virtuele laboratoriums maklik ontwikkel word om gedurende die opvoedkunde van jong leerders in onderwerpe soos wetenskap, soos byvoorbeeld ekosisteme, gebruik te word. Die simulاسie van hierdie virtuele laboratoriums laat jong leerders toe om te eksperimenteer, wiskundige tegnieke soos om te tel, te sorteer en te klassifiseer aan te leer en om die ekologiese konsepte en die verhoudings tussen verskillende plante en diere te leer. Een voordeel is ook dat die raamwerk bekostigbaar is, aangesien dit ontwikkel is vir 'n "desktop" virtuele realiteit oplossing, wat beskou is as die goedkoopste en die virtuele realiteit platform wat die meeste gebruik word.

Hierdie skripsie illustreer dat die gebruik van die "Intelligent Tiles" raamwerk 'n gebruikersvriendelike en bekostigbare opvoedkundige hulpmiddel tot gevolg kan hê wat onderwysers kan in staat stel om opvoedkunde prettig en meer effektief te maak.

Skripsie toesighouer: Dr. V. Lalioti

Department Rekenaarwetenskap

Ingedien ter gedeeltelike vervulling van die vereistes van die graad Magister Scientia

Acknowledgements

I wish to thank my supervisor Prof. V. Lalioti for suggesting the topic of this study, and her dedicated supervision throughout. My colleagues and friends, especially Mardé Greef, James Pun and Kenny Fei, and my family for their inspiration, support and encouragement.

The CSIR, the University of Pretoria and the National Research Foundation are acknowledged for their financial support.

Figure 1.	CommercialDesk [19]	21
Figure 4.	Illustration of the CAVE system [19]	21
Figure 7.	Students interacting with the Field [30]	29
Figure 8.	A scene from the Field [30]	30
Figure 9.	Scenes from the Aiyun VR Pond-Eco-System simulator [27]	31
Figure 10.	A scene from the Kewal 2 arch Project [22]	31
Figure 11.	The 3D Editor for Traffic Playground [28]	32
Figure 12.	The iTiles framework	36
Figure 13.	iTiles World Flow summary	40
Figure 14.	A world transformation (dotted lines indicate optional properties)	40
Figure 15.	Tile elevation transformation	41
Figure 16.	World object transformation	41
Figure 17.	Movement force of a dynamic world object (central line indicate optional properties)	41
Figure 18.	Positive Force of a dynamic world object (both left & right indicate optional properties)	41
Figure 19.	Constant and incremental positive forces (where left towards 'L' refers to force strength)	41
Figure 20.	A dynamic world object's negative force	41
Figure 21.	A character facing north, with herbivorous state (state with state depth 2)	41
Figure 22.	The dynamic world object state machine	41
Figure 23.	iTiles widgets classes' visual representation	41
Figure 24.	The tile merging process	42
Figure 25.	The grass, sand and water tile classes' sector	42
Figure 26.	Fit tree	42
Figure 27.	Broad tree	42
Figure 28.	Palm tree	42
Figure 29.	Willow tree	42
Figure 30.	Duck	42
Figure 31.	Elephant	42
Figure 32.	Pig	42
Figure 33.	Mouse	42

List of Figures

Figure 1.	Desktop VR [20]	18
Figure 2.	VR products developed by 5DT: The HMD 800 and 5DT Data Glove 5 [22]	19
Figure 3.	The ImmersiaDesk [19]	20
Figure 4.	Illustration of the CAVE system [19]	21
Figure 5.	A scene from NICE [29]	29
Figure 6.	A selection of Quickworlds [31]	29
Figure 7.	Students interacting with the Field [30].....	30
Figure 8.	A scene from the Field [30].....	30
Figure 9.	Scenes from the Argus VR Pond-Eco-System simulator [27].....	31
Figure 10.	A scene from the Round Earth Project [32].....	31
Figure 11.	The 3D Editor for Traffic Playground [28].....	32
Figure 12.	The iTiles framework.....	36
Figure 13.	iTiles World Flow summary.....	40
Figure 14.	A world transformation (dotted lines indicate optional properties).....	41
Figure 15.	Tile element transformation.....	41
Figure 16.	World object transformation.....	42
Figure 17.	Movement force of a dynamic world object (dotted lines indicate optional properties).....	44
Figure 18.	Positive Force of a dynamic world object (dotted lines indicate optional properties).....	45
Figure 19.	Constant and incremental positive forces' metric unit features explained as fuzzy logic (The unit 'wb' refers to world beat and 'f' refers to force strength)	47
Figure 20.	A dynamic world object's negative force	48
Figure 21.	A character facing north, with herbivore vision type with vision depth 2.....	51
Figure 22.	The dynamic world object state machine.....	52
Figure 23.	iTiles widgets classes visual representation.....	66
Figure 24.	The tile merging process.....	69
Figure 25.	The grass, sand and water tile element textures.....	71
Figure 26.	Fir tree.....	72
Figure 27.	Broad tree.....	72
Figure 28.	Palm tree.....	72
Figure 29.	Willow tree.....	72
Figure 30.	Duck.....	73
Figure 31.	Elephant.....	73
Figure 32.	Pig.....	73
Figure 33.	Mouse.....	73

Figure 34.	Turtle.....	74
Figure 35.	Dog.....	74
Figure 36.	The iTiles Ecosystem Virtual Laboratory introductory screen....	76
Figure 37.	The iTiles main menu.....	77
Figure 38.	A screenshot of the iTiles Workbench tool in tile authoring mode.....	79
Figure 39.	A screenshot of the iTiles Workbench tool in world object authoring mode.....	81
Figure 40.	Information stored for an iTiles Workbench file.....	81
Figure 41.	The main screen of the iTiles World Flow tool.....	83
Figure 42.	Information stored for an iTiles World Flow file.....	84
Figure 43.	A screenshot of the iTiles Virtual World tool.....	85
Figure 44.	Third person view.....	87
Figure 45.	Virtual identity view.....	87
Figure 46.	Inner vision view.....	87
Figure 47.	Authoring the base terrain to look like an arid African savannah.	89
Figure 48.	Populating the African savannah with elephants and trees.....	90
Figure 49.	The iTiles world at the start of the simulation.....	92
Figure 50.	Scenes from the simulation of the Drought in Africa world.....	93
Figure 51.	Thirsty elephants drinking the last bit of water.....	93
Figure 52.	iTiles Class Diagram.....	105
Figure 53.	Dynamic world objects list (iTiles World Flow tool screen).....	109
Figure 54.	Static world objects list (iTiles World Flow tool screen)	109
Figure 55.	Tile elements list (iTiles World Flow tool screen)	110
Figure 56.	Positive forces list of a dynamic world object (iTiles World Flow tool screen).....	110
Figure 57.	Positive force of a dynamic world object - Page 1 (iTiles World Flow tool screen).....	111
Figure 58.	Positive force of a dynamic world object – Page 2 (iTiles World Flow tool screen).....	111
Figure 59.	Positive force of a dynamic world object – Page 3 (iTiles World Flow tool screen).....	112
Figure 60.	Positive force of a dynamic world object – Page 4 (iTiles World Flow tool screen).....	112
Figure 61.	Movement forces list of a dynamic world object (iTiles World Flow tool screen).....	113
Figure 62.	Movement force of a dynamic world object (iTiles World Flow tool screen).....	113
Figure 63.	World object transformations list of a static world object (iTiles World Flow tool screen).....	114
Figure 64.	World object transformation of a static world object (iTiles World Flow tool screen).....	114
Figure 65.	Tile element transformations list of a tile element (iTiles World Flow tool screen).....	115

Figure 66.	Tile element transformation of a tile element (iTiles World Flow tool screen).....	115
Figure 67.	Negative forces list of a dynamic world object (iTiles World Flow tool screen).....	116
Figure 68.	Negative force of a dynamic world object (iTiles World Flow tool screen).....	116

List of Tables

Table 1:	Tile attribute list of world objects.....	74
Table 2:	Dynamic world object vision properties.....	74
Table 3:	World sounds.....	75
Table 4:	iTiles classes description.....	107

Contents

ACKNOWLEDGEMENTS	I
LIST OF FIGURES	II
LIST OF TABLES	IV
1. INTRODUCTION	1
1.1 WHAT IS A VIRTUAL LABORATORY?	1
1.2 THESIS FOCUS	1
1.3 THESIS LAYOUT	2
2. BACKGROUND	4
2.1 TECHNOLOGY IN EDUCATION.....	4
2.2 TEACHING AND LEARNING WITH COMPUTERS.....	6
2.2.1 LEARNING THEORIES	7
2.2.2 LEARNING WITH COMPUTERS.....	9
2.2.3 PEDAGOGIES FOR COMPUTERS	10
2.3 LEARNING SOFTWARE	13
2.4 VIRTUAL REALITY	16
2.4.1 VIRTUAL REALITY SYSTEMS	17
2.4.1.1 Desktop VR.....	18
2.4.1.2 Immersive VR.....	19
2.4.1.3 Projection based VR	20
2.5 VIRTUAL REALITY IN EDUCATION.....	21
2.5.1 EDUCATIONAL VIRTUAL REALITY APPLICATIONS.....	22
2.5.2 PEDAGOGIES AND LEARNING WITH VIRTUAL REALITY.....	23
2.5.2.1 Concerns and factors influencing the use of Virtual Reality in education.....	26
2.5.3 VIRTUAL LABORATORIES	27
2.5.3.1 Virtual laboratories in education.....	28
2.5.3.2 Techniques for evaluating virtual laboratories.....	32
2.5.3.3 Development of a virtual laboratory	33

2.5.3.4	Major problems of virtual laboratories	33
2.6	SUMMARY	34
3.	THEORETICAL APPROACH	35
3.1	THE INTELLIGENT TILES VIRTUAL LABORATORY FRAMEWORK.....	35
3.1.1	INTRODUCTION TO THE FRAMEWORK	35
3.1.2	AIMS OF THE ITILES FRAMEWORK	37
3.2	COMPOSITION OF AN ITILES WORLD	38
3.3	BEHAVIOUR OF AN ITILES WORLD.....	39
3.4	TRANSFORMING AN ITILES WORLD.....	41
3.5	SIMULATING BEHAVIOUR FOR AN ITILES WORLD.....	43
3.5.1	MOVEMENT FORCES.....	43
3.5.2	POSITIVE FORCES	44
3.5.3	NEGATIVE FORCES	48
3.6	MOVEMENT OF CHARACTERS IN THE SIMULATION.....	49
3.6.1	CHARACTER VISION AND NAVIGATION	49
3.6.2	THE DYNAMIC WORLD OBJECT STATE MACHINE.....	51
3.6.3	FEASIBILITY RULES	53
3.7	LEARNING WITH AN ITILES VIRTUAL LABORATORY.....	53
3.7.1	LEARNING THROUGH INTERACTION	54
3.7.2	LEARNING WITH WORLD TRANSFORMATIONS	55
3.7.3	LEARNING WITH WORLD FORCES	55
3.7.4	EDUCATIONAL ADVANTAGES OF THE ITILES FRAMEWORK	57
3.8	SUMMARY	59
4.	IMPLEMENTATION	60
4.1	OVERVIEW OF IMPLEMENTATION.....	60
4.1.1	APPLICATION PROGRAMMING INTERFACES (APIs).....	61
4.1.2	HARDWARE AND SOFTWARE USED IN IMPLEMENTATION	62
4.1.3	PROGRAMMING APPROACH	62
4.1.4	USER INTERFACE DESIGN	63

4.2	IMPLEMENTING THE ITILES ECOSYSTEM VIRTUAL LABORATORY ..	64
4.2.1	iTILES SYSTEM MANAGEMENT	64
4.2.2	ABSTRACTING GLUT FUNCTION CALLS	64
4.2.3	2D GUI	65
4.2.4	TEXTURES	67
4.2.5	iTILES SYSTEM INTERFACES.....	67
4.2.5.1	Tile set interface.....	67
4.2.5.2	World object interface.....	67
4.2.5.3	World sound interface.....	68
4.2.6	COLLISION DETECTION.....	68
4.2.7	TILE MERGING	68
4.2.8	CAMERA	69
4.3	OVERVIEW OF THE ITILES ECOSYSTEM VIRTUAL LABORATORY.....	70
4.3.1	POPULATING iTILES SYSTEM INTERFACES	70
4.3.2	THE INTRODUCTORY SCREEN AND MAIN MENU	75
4.3.3	THE iTILES WORKBENCH TOOL	77
4.3.3.1	Tile authoring mode.....	78
4.3.3.2	World object authoring mode	79
4.3.4	THE iTILES WORLD FLOW TOOL	82
4.3.5	THE iTILES VIRTUAL WORLD TOOL.....	84
4.4	SUMMARY	88
5.	DROUGHT IN AFRICA	88
5.1	IDENTIFYING LESSON OBJECTIVES	88
5.2	AUTHORING	89
5.3	SPECIFYING BEHAVIOUR	90
5.4	SIMULATION	92
5.5	SUMMARY	94
6.	CONCLUSIONS AND FUTURE WORK	95
6.1	CONCLUSIONS	95
6.2	FUTURE WORK.....	96

BIBLIOGRAPHY	99
APPENDIX A: iTiles Class Library	104
APPENDIX B: iTiles World Flow tool screens	108
APPENDIX C: iTiles Ecosystem Virtual Laboratory manual	117

"If we teach today as we taught yesterday, we rob our children of tomorrow."
John Dewey (1916)

Introduction

1.1. What is a virtual laboratory?

A laboratory is a place that provides the environment for experiments, observation or practice in a field of study (1). Its main objective is to help students gain practical skills in a medium, the flow chart, notes of a laboratory. The emergence of a virtual laboratory provides a new world with an it teacher's expertise to help students to learn in ways that were not possible until recently. Virtual laboratories, and other simulation-based learning, are doing "virtually good" in an effort to realize the goal of "learning by doing" from education principles. A virtual laboratory is a computer-generated environment, however, which encourages the learner from the field of study to interact played on educator and learners. In this regard, the virtualization of a real world laboratory is still in progress.

1.2. Thesis focus

This thesis focuses on the research of young learners. This research, which is a study of using computers as a medium of education, includes a review of the literature on learning with computers, current perspectives in education, and the use of virtual reality and applications of virtual laboratories as a medium. The main focus of the thesis is to present a generic user-adaptable framework for the teacher to design virtual laboratories to be used in the education of young learners.

Chapter 1

“If we teach today as we taught yesterday, we rob our children of tomorrow”

- John Dewey [15]

Introduction

1.1 What is a virtual laboratory?

A laboratory is a place that provides the opportunity for experimentation, observation or practice in a field of study [1]. By constructing a virtual laboratory using virtual reality as a medium, the above characteristics of a laboratory are augmented by a virtual simulation. A virtual laboratory provides a user with an interactive experience that facilitates the learning process in ways that were not possible up to this time. Virtual laboratories can be classified as simulation-based, learning by doing. ‘Simulation’ from virtual reality principles, and ‘learning by doing’ from education principles. A virtual laboratory is comparable to a virtual environment, however what distinguishes the former from the latter is that more emphasis is placed on education and learning. In this thesis the utilisation of virtual laboratories in education is addressed.

1.2 Thesis focus

This thesis focuses on the education of young learners. It discusses the educational principles of using computers as a medium of education, includes a focus on theories of teaching and learning with computers, current perspectives in educational technology, with an emphasis on virtual reality and applications of virtual laboratories in education. The main focus of the thesis is to present a generic and adaptable framework for the teacher driven development of virtual laboratories to be used in the education of young learners.

1.3 Thesis Layout

Chapter 2 discusses the background themes relating to virtual laboratories, such as the impact of introducing technology into an educational setting, and the learning theories and practices that underlie the educational use of computers. The types of software that have been developed for learning are also discussed. Here the focus is drawn to simulation type software, presenting virtual reality and applications in education. The chapter highlights pedagogies and learning with virtual reality, and also investigates the concerns and factors influencing the use of virtual reality in education. It discusses virtual laboratories in education by presenting constructive examples, and explores the requirements needed for the development of virtual laboratories. Techniques for evaluating, and major problems of virtual laboratories are also presented.

In Chapter 3, the theoretical approach of the Intelligent Tiles framework is presented. The chapter discusses how this framework is used for the development of a virtual laboratory, and how this virtual laboratory is used in an educational setting. The composition of a virtual environment produced with this framework is discussed. The chapter also focuses on the simulation of the virtual laboratory, presenting an approach for the behavioural and movement aspects of virtual characters, and the visual and auditory transformations of the virtual environment. How young learners can interact with and observe the simulation of the virtual laboratory is also addressed. The educational advantages of the Intelligent Tiles framework are highlighted.

In Chapter 4 the implementation of a virtual reality application implemented using the Intelligent Tiles framework is discussed. Chapter 4 presents this application, called the iTiles Ecosystem Virtual Laboratory, used for the authoring and simulation of a virtual laboratory. The development process, application program interfaces and the object oriented programming approach are presented. Implementation specifics such as texture management, 3D models, collision detection, user interface design, sound management, tile merging and camera smoothing are highlighted. The chapter also presents the tools of the implemented system. How tools are used for authoring, specifying behaviour and simulation of a virtual laboratory is discussed. These tools are presented in terms of usability and implementation, with screenshots.

Chapter 5 presents how the iTiles Ecosystem Virtual Laboratory application is used for the development of a virtual laboratory. The chapter identifies the lesson objectives for this virtual laboratory, called Drought in Africa. It discusses how the different tools of the iTiles

Ecosystem Virtual Laboratory are used, and how these lesson objectives are embedded in the authoring process. The relationships and behaviour of virtual characters specified are discussed. The chapter also discusses how the simulation of this virtual laboratory can address pedagogies in observation, mathematics and classification and offline activities. The ecological relationships and occurrences taught are also discussed.

Chapter 6 provides conclusions of the dissertation and discusses future work with regard to the Intelligent Tiles framework and iTiles Ecosystem Virtual Laboratory application.

Appendix A provides the class library used in implementation of the iTiles framework

Appendix B presents screenshots from the iTiles World Flow tool.

Appendix C presents the manual and user's guide of the iTiles Ecosystem Virtual Laboratory.

Background

This chapter focuses on impact of technology in the education of young learners. In the first section of this chapter, the field of technology in education is explored, introducing research on the use of computers in the classroom and the impact the technology has on education. It then presents the influence on teaching and learning with computers and introduces the types of software developed for learning. Virtual reality is presented in the second half of the chapter, providing a brief introduction to virtual reality systems. Virtual reality in education is then explored, and the impact it has had on education is discussed in terms of application areas, advantageous attributes and concerns of its use. The chapter then focuses on the approach of virtual laboratories, presenting examples, techniques for evaluating them and major problems encountered with existing virtual laboratories.

2.1 Technology in Education

In a traditional instructional learning environment, students are expected to learn by assimilation, by listening to a lesson presented by a teacher or reading material on that subject matter. Technology provides another medium of learning whereby the computer has achieved groundbreaking results and opened up new avenues for learning. The advent of technology is not a simple exchange of old for new. Technology can change the way children think, what they learn, how they interact and how we assess them [17]. Gaining confidence in their ability to control technology is a valuable lesson children learn from computers [14]. Very young

Chapter 2

“We should compare the computer not to books, but to a blank sheet of paper, a notepad, an artist’s canvas, or a blackboard. The computer may be a tool, but the act of computing itself is a medium for thought”

- B. Hokanson, S. Hooper [10]

Background

This chapter focuses on impact of technology in the education of young learners. In the first section of this chapter, the field of technology in education is explored, introducing research on the use of computers in the classroom and the impact this technology has on educators. It then presents the influence on teaching and learning with computers and introduces the types of software developed for learning. Virtual reality is presented in the second half of this chapter, providing a brief introduction to virtual reality systems. Virtual reality in education is then explored, and the impact it has had on education is discussed in terms of application areas, advantageous attributes and concerns of its use. The chapter then focuses on the approach of virtual laboratories, presenting examples, techniques for evaluating them and major problems encountered with existing virtual laboratories.

2.1 Technology in Education

In a traditional instructional learning environment, students are expected to learn by assimilation, by listening to a lesson presented by a teacher or reading material on that subject matter. Technology provides another medium of learning whereby the computer has achieved groundbreaking results and opened up new avenues for learning. The advent of technology is not a simple exchange of old for new. Technology can change the way children think, what they learn, how they interact and how we assess them [17]. Gaining confidence in their ability to control technology is a valuable lesson children learn from computers [14]. Very young

children have shown comfort and confidence in using software that requires single-key presses [17]. They can turn on a computer, follow pictorial direction, and use situational and visual cues to understand and reason about their activity. With changes in hardware, children with physical and emotional disabilities can also use a computer with ease. Besides enhancing their mobility and sense of control, computers can help improve children's self-esteem [17]. Research findings presented in [16] have the following positive findings on computer use in the classroom:

- Students learn more in less time when they receive computer-based instruction.
- Students like their classes more and develop more positive attitudes when their classes include computer-based instruction.
- Students in technology rich environments experienced positive effects on achievement in all major subjects.
- Students in technology rich environments showed increased achievement in preschool through higher education for both regular and special needs children.
- Students' attitudes toward learning and their own self-concept improved consistently when computers were used for instruction.

However, the impact of computers on education due to wide societal use outside the classroom should be considered. Computer use coupled with a societal infatuation with fast-paced, non-linear media in general, may develop a haphazard, hypertext structured thought process, while the traditional educational system is based on reading and writing that fosters the development of a linear, logical thought process that is valued and integrated into the structure of society [10].

Research indicates that as schools and classrooms gain greater access to greater numbers of computers, to more sophisticated software, and to connections to the Internet and the World Wide Web, one of the key issues to be addressed is the need for teachers to become familiar and understand the advantage of the use of such technology. Teachers not only must become familiar with the possibilities for learning and for support promised by these advances but also must help children learn about computers and learn about using computers [6]. This is of utmost importance since busy classroom teachers do not want another instructional tool forced on them without appropriate provisions for training, preparation and implementation [9]. As technology becomes more accessible to early childhood programs, childhood educators have a responsibility to examine its impact on children and prepare themselves to use it for all children's benefit [15].

Without appropriate training, teachers may regard a computer simply as a tool and use it in a limited manner, i.e. only to deliver instruction to students. Such an approach uses only part of the capabilities of a computer (the communicative or representative functions) and fails to acknowledge its generative potential [10]. One important key to the effectiveness of computer technology is its interactive quality, which allows children to get involved with the content as they manipulate this medium [12]. The manner in which educators use computers is guided by their theoretical understanding of education [10]. Therefore it is essential to give teachers time and support to build confidence and competence in the pedagogy of computer use [11]. Many teacher education programmes are beginning to integrate technology experiences into professional education courses [6]. Teacher education programs should not only teach pre-service teachers how to use hardware and software but also teach them how to incorporate computers into their teaching strategies and activities [6].

There are a number of advantages of incorporating computers into teaching strategies. In the practice of teaching with computers, teachers can take on the role of facilitators or coaches, tailoring their assistance to the needs of individual children [12]. Differences in learning styles are more readily visible at the computer where children have the freedom to follow diverse paths towards the goal. Observing the child at the computer provides teachers with a “window into a child’s thinking process”. This is particularly valuable with special children, as the computer seems to reveal their hidden strengths [17]. In summary, the use of computers in children’s education has great value for children (particularly for those with learning difficulties or for those who are gifted or talented), by serving to [11]:

- Engage children’s attention.
- Individualise instruction (a practice of one computer for every child).
- Provide access to learning experiences difficult to provide for by other means.
- Give learners control over their learning.
- Provide opportunities for, and the means of, communication.

2.2 Teaching and Learning with Computers

Computing in education is still in its infancy and has great potential to enrich the learning environment. The role that educational technology plays in the process of education has been addressed by many recent research studies. This section addresses the adaptation of existing learning theories to computer usage, child learning with computers, and pedagogies for educating with computers.

2.2.1 Learning theories

Learning theories embedded in the way children are taught are crucial to successful learning with a computer. Learning theories have been discussed in many research studies. The following is a description of learning principles, theories and concepts commonly used to explain how learning occurs with children.

The general principles of learning presented in [13] are:

- Attention is a prerequisite for learning.
- What is learned persists if it is practised.
- People forget things they do not use.

The research into learning theories and concepts has provided a basis for a pedagogic framework for best practices on using computer technologies. However, to fully account for theories of how children learn is far beyond the scope of this thesis, and for this reason, general principles that apply specifically to learning, when learning is supported by technology, are addressed. Learning theories and concepts used in computer learning include the constructivist paradigm, experiential learning, feedback and reinforcement learning, and incidental learning.

Constructivist paradigm

There is really only one way to learn how to do something and that is to do it. If you want to learn how to tie your shoelaces, you must have a go at doing it. Learning a skill means eventually trying your hand at the skill, and when there is no real harm in simply trying we can allow novices to “give it a shot”. This concept is called *learning-by-doing*.

In a constructivist view of learning, the idea is that students are better able to master, retain and generalise new knowledge when they are actively involved in constructing the knowledge through *learning-by-doing* [26].

Experiential learning

Students learn best when they enjoy a rich, often multi-modal, experience of the educational material [13]. Students need to experience concepts and principles contained in the content as much as and as directly as possible. This can be achieved by:

- Increasing intrinsic motivation by giving the learner ownership or control of an experience.

- Achieving mental and physical engagement by appropriate amounts of a challenge and a cohesive narrative framework for the learning experience.
- Learning experiences should be valid within a social context and involve group interaction.

The key distinction to experiential learning is that it addresses the needs and wants of the learner [11].

Feedback and reinforcement learning

Feedback and reinforcement are two of the most pivotal concepts in learning. Feedback involves providing learners with information about their responses whereas reinforcement affects the tendency to make a specific response again. Feedback can be positive, negative or neutral; reinforcement is either positive (increases the response) or negative (decreases the response) [11].

Feedback is almost always considered external while reinforcement can be extrinsic (external) or intrinsic (i.e. generated by the individual). Information processing theories tend to emphasise the importance of feedback to learning since knowledge of results is necessary to correct mistakes and develop new plans. On the other hand, behavioural theories focus on the role of reinforcement in motivating the individual to behave in certain ways. One of the critical variables in both cases is the length of time between the response and the feedback or reinforcement. In general, the more immediate the feedback or reinforcement, the more learning is facilitated. These principles are often used in drill and practice software (see Section 2.3).

Incidental learning

For young learners in particular, play is recognised to be a very powerful learning medium [11]. Not only does play provide a focus for building valuable skills in computer use but it can also provide a means of incidental learning. In particular, playing with computers can be characterised by children building:

- *Intrinsic motivation.* Some software provides for high levels of motivation in learners, where children are motivated by the task presented in the software and undertake a learning experience for its own sake, and often for extended periods of time.
- *Attending to the means rather than the end.* When children are involved in exploring and experimenting with software, they are often doing so in a context where there is no

danger of 'being wrong', and are able to build self-esteem and confidence in their abilities to learn successfully.

- *Demonstrating non-literal behaviours.* Role-playing, pretence and imaginative thinking are often hosted in software such as adventure games and simulations.

2.2.2 Learning with computers

By presenting concrete ideas in a symbolic medium, the computer can help bridge the two for young children. Research shows that what is “concrete” for children is not what is “physical”, but what is *meaningful*. Computer representations are often more manageable, flexible and extensible. While the experience and opportunities children have using technology are diverse they all share one common thread: each challenges children to explore, discover and learn.

Given the capabilities of a computer, one may ask has it affected children’s learning?

The effective usage of computer technology may enhance children’s learning. Although, some suggest that the initial optimism regarding the benefits of this technology may have been unrealistic. Others criticisms maintain that computers may cognitively de-skill students by eliminating schoolwork that may require mental effort, as the computer now routinely performs many activities once performed in students’ minds and consequently the skills previously developed and practised are lost [10]. However, contradictory to the above, computers appeal to a child’s imagination and natural curiosity, and capitalising on them can provide many unique opportunities for learning. Young children are eager to learn, willing to take risks, and capable of using a wide variety of learning resources. When children utilise technology, they explore, create, communicate and collaborate with others [12].

In [11], general themes that have emerged from literature that help provide a framework in which information technology, or computers, are best placed to engage learning are:

- *Active learning.* Students learn best when they are actively constructing knowledge by manipulating, creating, experimenting, etc.
- *Personal learning.* Students learn best in a context that they can identify personally with and/or that they own.
- *Individualisation.* Research findings clearly demonstrate that learners each learn differently. Differences in learning can be partly related to cultural influences but can also be accounted for by reference to styles of learning.
- *Gender equality.* It seems that girls are often less likely to make use of computers for a host of social, psychological, and environmental reasons. In this situation, however, there

are well-defined strategies that teachers can implement to support girls' use of information technologies.

- *Cooperative learning.* Cooperative learning is seen as a powerful instructional tool or method, to encourage active engagement by students in learning and to enable learners to build both affective (e.g. team work) and cognitive skills (e.g. articulation). Cooperative learning facilitates group acceptance of common goals, as well as a framework within which to support students to achieve these goals.
- *Learning strategies.* The use of learning strategies is increasingly being recognised as an effective means of producing significant improvements in cognition and motivation to learn.
- *Contextual or situated learning.* Learning is made more meaningful and is acquired more effectively, if situated in real world contexts, rather than in abstract and academic settings. For example, if we wish to learn basic number skills, we are more likely to learn these skills better if they are practised in a familiar and meaningful context, such as buying groceries in a supermarket.

Research studies on the impact of education technology on student achievement presented evidence that learning technology is less effective or ineffective when the learning objectives are unclear and the focus of the technology is diffused [16]. Therefore the way media is used greatly affects its educational potential [10].

There exist links between learning theories, learning with computers and teaching practices. In the next section, the pedagogic applications of computers are presented, which form the links between these topics.

2.2.3 Pedagogies for computers

Do we teach with computers or do students learn with computers?

It is not enough to sit children in front of a computer and let chaos reign supreme. Educators and adults need to guide children, helping them to appropriately use the technology to create, communicate, collaborate and explore. In general, it is the pedagogy that provides for learning in students, not the computer or software alone – without an appropriate pedagogy, computer use cannot provide for any planned, significant learning outcome in students [11]. The computer can offer unique opportunities for learning through exploration, creative problem solving, and self-guided instruction. Effectively integrating technology into a curriculum demands effort, time, commitment and sometimes even a change in one's beliefs [17].

In [11], the following pedagogies are mentioned, which involve the use of one or more computers and associated software, examining theoretical frameworks for education as they guide computer use:

- *Rotational use of computers.* Learners are rotated around a computer in an instructional system where a limited number of computers are situated in a single learning area or classroom. Often this rotation is guided by the use of a roster, controlled by a strict timetable or directly by the teacher, or sometimes with learners themselves determining the length of time they each spend at the computer.
- *The computer station.* A planning strategy used most often in primary schools, where the computer becomes a station alongside a number of other stations (e.g. a reading station, writing station, resource station, painting station). Students might rotate around these stations or perhaps choose a station to work in according to need. This strategy is often practised to make effective use of limited resources, as well as to provide students with choice and decision-making about learning activities.
- *Needs-only basis.* A planning strategy that sees the use of a computer primarily as a support resource in the classroom, and where students are encouraged to use the computer(s) according to need – so, for example, if a student had a need to use the classroom computer as a word processor, he/she would be able to do so; or if another student wished to use the computer as a means of accessing the Internet for locating specific information, he/she would be able to do so. As a planning strategy, deploying one or more classroom computers on a 'needs only basis' often leads to a small number of children making exclusive use of the computer (i.e. those with the appropriate skills and confidence).
- *Computer as reward.* The computer has long been used as a reward system for children - perhaps to reward early finishers of 'traditional' (i.e. pen and paper) work; or to reward children who complete work beyond their normal standards. In other words, teachers who employ this pedagogy are using the computer as an extrinsic reward, as a motivational strategy; or perhaps as a classroom management strategy.
- *Computer use on contract.* A contract programme is usually initiated as part of a student management strategy, where individual children are presented with contracts of work drawn up by a teacher, each contract describing exactly what an individual student should complete, perhaps in terms of tasks as well as behaviours. These contracts may involve the substantial use of a computer, since computer aided learning (CAL) can provide well for individualised learning. The goal of most contract approaches is to involve students in a negotiation of work, as well as in deciding on the behaviours that should be reinforced

and rewarded as a result of completing the contract. This approach, from the point of view of learning, is intended to provide students with experience of negotiation, self-responsibility and goal setting.

- *Computer as electronic blackboard.* The classroom computer is often employed, at all levels of education, as an electronic blackboard. That is, it is used to demonstrate a concept, event or phenomenon by illustration or description. Different types of software can be used in this context, since the pedagogy for this use of the computer centres on the dominant and mediating role of the teacher in motivating, questioning, explaining to and reinforcing students – the computer is simply employed to demonstrate, illustrate and describe. Perhaps this mode of computer use is best employed when using software that demonstrates a concept; event or phenomenon that is difficult to reproduce by other means.
- *Integrating the computer.* Integration is a concept and pedagogy more common to primary rather than secondary educators. It is based on the notion of employing the use of a computer meaningfully in a number of different curriculum areas or subjects, perhaps in a thematic or topic approach to curriculum design. For example, if an organising theme for a class curriculum is taken to be Fairy Stories, the computer may be used by students to word process a description of a fairy story, take part in a computer based adventure centred on a fairy story, and perhaps develop a 'light painting' illustrating aspects of a fairy story using a paint program. There are plenty of ideas for and examples of, computer integration of this sort. In many examples of this approach, the organising theme is taken from a software package itself; very often the software that provides the theme is an adventure or simulation. In other situations, the computer can be integrated almost totally within a set curriculum, without reference to a thematic or topic approach, simply by employing the computer meaningfully and flexibly to both support and extend learning activities. Teachers, due to the difficulties of resource allocation and management, however, rarely attempt this form of integration, as it also demands exceptional planning and integration skills on the part of the teacher.
- *Computer as surrogate teacher.* As a surrogate teacher, the computer is used to set tasks for a student, as well as monitor and evaluate a student's performance in that task. In this sense, the computer carries out a number of roles assumed by a human teacher. Software used for this approach to computer use is often of the drill and practice type.
- *Computer as cognitive tool.* A cognitive tool is usually characterised by a generic item of software, such as a spreadsheet, which provides for knowledge construction and modelling. They are content-free and therefore appropriate to a range of subject or domain areas. Cognitive tools are defined as mental and/or technological devices that

support, guide and extend the thinking processes of their users. Just as a convection oven supports the cooking process, cognitive tools support thinking and learning processes. The computer, in the shape of a cognitive tool, allows the learner to externalise their thinking, to enrich it, manipulate it and change it, all by interacting with one or more conceptual models on the computer. The nature and use of cognitive tools is closely aligned with the concept of cognition as mental models.

- *Role of the teacher.* What role(s) should a teacher play when children are using computer technologies? Well, as with many such 'open' questions, there is not just one answer. Teachers may wish to play a diverse and changeable pattern of roles within or between lessons, just as they might do in a non-computer based teaching context. The roles that a teacher might assume in a computer-based classroom depend on a range of influencing factors, including the type of software in use, the nature and objectives of the learning activities and the instructional strategies employed (e.g. group or individual work). The predominant or 'ideal' role of a teacher when using educational software could be considered a facilitating or guiding one. However, when using some types of instructional software the teacher may be excluded as the computer takes over. It is therefore important to consider how a teacher's role might change in line with different teaching architectures.
- *The computer in a gender equal classroom.* Gender equity is a major issue in computer assisted learning, and various, well-documented findings exist in this area of home and school computer use. For example, there are a well-defined set of teaching strategies that can be implemented to provide for a gender equal classroom in relation to computer use, as well as actions of a more general nature that apply to computer use at both school and home. Indeed, the gender bias of the computer-game market appears to contribute to girls' lack of interest in computer games and inhibits their access to computer technology.

Expanding on the pedagogy of 'integrating the computer', research shows that computer activities yield the best results when coupled with suitable off-computer activities [17]. For example, children who are exposed to developmental software alone show gains in intelligence, non-verbal skills, long-term memory and manual dexterity. Those who also work with supplemental activities, in comparison, gained in all of these areas and in addition also improved their scores in verbal, problem solving and conceptual skills.

2.3 Learning Software

An important question regarding educational computer use continues to be how computers might improve learning. Learning with computers is often referred to as computer-assisted learning (CAL). CAL is directly dependent on the underlying computer software that drives

it. Computer aiding software and tools aiding in education have been developed for both individualised instruction and for small groups (mainly due to resource constraints, such as the scarcity of computers). Computer software may carry objectives for learning, where sometimes this may be clear and stated, but often this is implicit and open to interpretation. More explicit objectives in software are usually cognitive in nature and are concerned with students building content knowledge, or perhaps with practising skills with greater accuracy [11]. Software for learning can be pre-developed programmatically or authored using authoring software. How software can be authored or created is discussed next, followed by a classification of the types of computer learning software.

Authoring software

According to Lycos' Tech Glossary [2], an authoring tool, also known as authorware, is a program that helps to develop a multimedia application. Authoring tools usually enable you to create a final application merely by linking together objects, by defining the objects' relationships to each other, and by sequencing them in an appropriate order. Authors with the use of authoring tools can produce attractive and useful graphics applications. Typically, authoring tools require less technical knowledge of the programming domain and most authoring systems also support a scripting language for more sophisticated applications.

An authoring tool that requires little technical know-how would provide a teacher with a simple and quick mechanism to deliver a computer-oriented lesson. However, this depends on the type of authoring tool and the learning objectives. Students can also take on the role of authors. In [16], research of the Learning and Epistemology Group at MIT is presented where they have employed learning by design principles to educational technology by having students (children) become creators and designers of educational software. Here children learn through design activities by programming computers to create applications that other children use and learn from.

Types of learning software

Various types of learning software have been developed for different types of computer learning. Educational software can be characterised not only as a teaching and learning resource, but also as a teaching strategy; that is, a software program carries with it explicit and/or implicit strategies for its use (i.e. instructional strategies) [11]. The types of learning software presented here have been designed to facilitate language development, mathematical skills and higher-order thinking. These are drill and practice software, development software and simulation and discovery-based software.

Drill and practice software has been designed for individual use for practising and improving various skills [12]. This type of software can improve the reading readiness skills of preschoolers and of young learners that show signs of potential reading difficulties. Using this software can also help young children develop competence in counting and sorting. The gains of using this software are directly linked to the amount of time spent using it [17].

Development software enables children to progress and learn at their own pace [12]. With development software the computer is used as an exploration and creativity tool, and therefore not limited to teaching or learning a particular curriculum [7]. It is important to note that Seymour Papert, a founder of the use of computers in education is the inventor of one of the most known development software (Logo). Logo is a procedure-oriented computer programming language developed for young children to 'teach' the computer how to perform tasks and build an understanding of concepts typically related to mathematics [11]. Logo allows the creation of pictures with geometric shapes and children have demonstrated growing knowledge and competence in working with concepts such as symmetry, patterns and spatial order. Research studies on Logo include a study where students have employed Logo in order to design software to teach fractions to younger students [16]. These students were required to structure their computer programs, maintain connections between content and functionality, and design the user interface and activities as well as consider different ideas about how to teach fractions to younger students. This study has indicated that:

- Students who designed fraction software for other students using Logo learned fractions better than students taught fractions using conventional methods.
- Students who used Logo to design software learned Logo better than students who received Logo programming instruction only.

Simulation and discovery-based software is another classification of learning software. A simulation is described as the process of imitating a real phenomenon with a set of mathematical formulas [2]. Advanced computer programs can simulate weather conditions, chemical reactions, atomic reactions, and even biological processes. Simulations can be used as a means of engaging learners in 'learning by doing', or experiential models of learning (see Section 2.2.1). Simulations are also a way of providing goal-based approaches to teaching and learning [11]. Discovery-based software encourages this and allows ample room for free exploration, which is considered more valuable in this regard [17]. In [11], virtual reality is described as an extension of a CAL simulation, where real life can be simulated in such a way that the simulation appears real to the learner experiencing it. Virtual reality is discussed in the next section.

2.4 Virtual Reality

What is virtual reality? Virtual reality is a computer generated synthetic environment that provides the illusion to the user as if it were real. The term virtual reality (VR) is broadly used, and has many definitions and synonyms. Examples of such synonyms are: virtual worlds, virtual environments, cyberspace, synthetic environments, simulator technology and artificial reality. The term virtual reality is sometimes used more generally to refer to any virtual world or virtual environment represented in a computer, even if it is just a text-based or graphical representation [2]. In literature ‘virtual world’ and ‘virtual environment’ are often interchangeable. The virtual world is composed of objects that can have geometry, hierarchy, scripts and other attributes. These objects can be positioned and oriented at a location in space (2D or 3D), and by applying translation and rotation operations these attributes can be changed [18].

There are two main types of virtual reality:

- *Non-immersive VR*. In non-immersive VR a standard computer or television monitor is used to display the virtual world.
- *Immersive VR*. In immersive VR, the user is immersed inside the virtual world for a more realistic experience and has been referred to as a technology that literally “envelopes one in a surrogate existence”. However this depends on the virtual reality system being used (see Section 2.4.1).

In [18], four key aspects of a virtual reality application are discussed:

- *Input process*. The input processes of a VR application control the devices used to input information into the computer. There is a wide range of possible input devices: keyboard, mouse, trackball, joystick, 3D and 6D position trackers (e.g. glove, wand, head tracker and body suit). Ideally these technologies should provide three measures for position (X, Y, Z) and three measures of orientation (roll, pitch, yaw). Interaction with a VR application involves focussing on the user and examining the style of interaction that can take place between the user and the virtual environment. Interactivity involves moving objects as well as yourself around in the VR environment [4].
- *Simulation process*. The core of a VR application is the simulation system. This is the process that knows about the objects and various inputs. It handles the interactions, the scripted object actions, simulations of physical laws (real or imaginary) and determines the world status. This is basically a discrete process that is iterated once for each time step or frame.

- *Rendering process.* The rendering processes of a VR application create the sensations that are output to the user. There are separate rendering processes for visual, auditory and haptic (touch/force) systems. The *visual rendering process* is most common and has a long history in the world of computer graphics and animations, and is often referred to as the rendering pipeline since a series of sub-processes are invoked to create each frame. A major consideration for graphics rendering is the frame generation rate, which is typically greater than 20 frames per second, as this is the minimum rate at which the human brain will merge a stream of still images and perceive a smooth animation. A VR application is greatly enhanced by the inclusion of an *auditory rendering process*, which may produce mono, stereo or 3D audio. A user's presence is greatly improved by the inclusion of such an audio component. The task of a *haptic rendering process* is the generation of touch and force feedback information.
- *World database.* The storage of information on objects and the world is the major part of the design of a VR application. The primary things that are stored in the world database are the objects that inhabit the world, scripts that describe the actions of those objects or the user (in terms of haptics), lighting, program controls and hardware device support.

The way users see a virtual reality system is referred to as their *perspective* on the virtual world. Because of the flexibility and user control within a virtual environment, the number of possible perspectives a user can find is endless. The ability to use different perspectives, or frames of reference of the virtual world, may be useful for highlighting different patterns and relationships in abstract information. There are two basic kinds of frames of reference: the exocentric and egocentric frames. The first provides the user with a view of a phenomenon or a space from the outside looking in. The latter provides the same view but from within the phenomenon or the space itself. Another view is the bicentric frame of reference, which allows users to alternate between the exocentric and the egocentric frames of reference. By being able to change his/her frame of reference a user will influence mastery in a positive manner, where the egocentric view supports local information while the exocentric view sheds light on information on a larger scale. [34]

2.4.1 Virtual reality systems

In order to understand the use of virtual reality, it is important to understand the technology hardware used to present this medium to the end-user. This section provides a relatively simple description of virtual reality hardware and is intended to describe VR systems that are mentioned in this thesis.

In general, factors shared by most of these emerging and existing technologies are important, such as capabilities, cost, availability, size and performance. These factors can play a major role in assisting educators in selecting affordable and accessible virtual reality hardware for educational purposes, since there are presently very few cost effective technology solutions available. Because of the different types of VR, VR hardware can also be categorised by the type of the immersion that can be provided.

2.4.1.1 Desktop VR

Desktop virtual reality (see Figure 1), a less elaborate computer-created environment is processed on a personal computer and usually experienced on standard CRT monitors. The interaction with the 3D environment is usually done with the aid of a mouse, keyboard and other specialised VR hardware such as a data glove, or spaceball input system. Desktop VR is sometimes called a “Window on a World” (WoW), a window through which one beholds a virtual world [18]. While not very immersive, desktop VR worlds can reflect the textures of the objects portrayed, are interactive and have fully navigable capabilities [4]. This form of VR lacks any feeling of immersion on the part of the user, however stereo modes may be enabled, allowing the user to wear shutter glasses, increasing the 3D perception. Desktop VR is considered the most common and least expensive form of VR, and the most popular platform for educational VR applications, and it is expected to grow in popularity with faster, more-powerful and more affordable desktop computers [9, 26]. There are a number of new fast graphic and video accelerator cards available for the PC, which can deliver cutting edge graphics performance, presenting dynamic, realistic 3D worlds and characters. These cards are referred to as graphics processing units (GPUs) and can now offer a viable alternative to expensive desktop VR systems such as Silicon Graphics workstations.



Figure 1. Desktop VR [20]

2.4.1.2 Immersive VR

Stereo vision that is often included in an immersive VR system, is produced by creating two different images of the world, one for each eye. The images are computed with the viewpoints offset by the equivalent distance between the eyes. There are a large number of technologies for presenting these two images [18].

LCD shutter glasses

Liquid crystal shutter glasses have been designed to shut off alternate eyes in synchronization with a display that sequentially displays the two images computed for stereo vision.

Head mounted display (HMD)

In HMD VR, the user “wears” a stereo display (see Figure 2), much like a pair of glasses that provides a view into the virtual world [21]. The goal in the technology is to provide the widest field of view at the highest quality and with the least weight at a reasonable cost. The main limitation with the technology is that the eyes are covered by the display. New techniques are however addressing this, for example the field of augmented reality (AR) where the computer augments the user’s view of the physical world with additional information, by superimposing the physical world onto the computer generated virtual world.

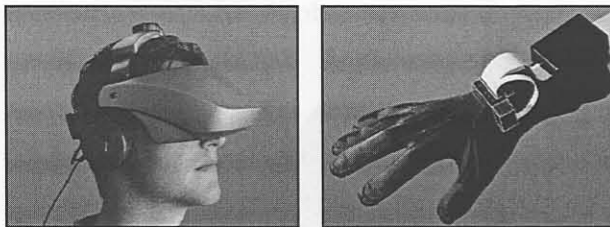


Figure 2. VR products developed by 5DT: The HMD 800 and 5DT Data Glove 5 [22]

Interaction devices

Interaction devices are hardware used as a user’s input into the VR system to manipulate or interact with the virtual environment. The simplest control hardware is a conventional mouse, trackball or joystick. While these are two-dimensional devices, creative programming can use them for 6D controls. There are a number of 3D and 6D devices available (called spaceballs) on the market containing extra buttons and balls used to control not just the XY translation of a cursor, but its Z dimension and rotations in all three dimension. Another common interaction media is the instrumented data glove (see Figure 2), which is outfitted with sensors on the fingers, as well as an overall position/orientation tracker. The glove concept has been extended to full body suits with position and bend sensors, used for capturing motion for applications such as character animation. Some interaction devices include a form of force

and haptic feedback mechanisms, which are triggered as a result of interaction with the virtual environment. [18]

One of the biggest problems in interaction devices is that of latency in position tracking, the time required to make the measurements and reprocess them before input to the simulation engine [18]. Head and body gear are considered restrictive and uncomfortable [5]. New trends in interaction devices try and eliminate such restrictive media, and include recognition of natural human interaction mechanisms such as gesture and voice.

2.4.1.3 Projection based VR

The following projection based VR systems have been used in the research area of VR in education.

ImmersaDesk™

The ImmersaDesk, was developed at the Electronic Visualization Laboratory at the University of Illinois at Chicago USA, and is manufactured by Fakespace Systems (<http://www.fakespacesystems.com>), is a 1.27m by 1.7m rear-projected video system with head- and hand-tracking, employing lightweight shutter glasses to present a stereoscopic display (see Figure 3 for an illustration of the ImmersaDesk) [23]. The ImmersaDesk is used in educational VR applications as the display can support a small group of children simultaneously [30].



Figure 3. The ImmersaDesk [19]

CAVE™

In this class of VR, the user functions within a room on which one or more of the surfaces (walls, floor, ceiling) is the display [21]. The CAVE (CAVE Automatic Virtual Environment) was developed at the Electronic Visualization Laboratory, University of Illinois at Chicago, USA (see Figure 4 for an illustration).

The CAVE is a projection-based VR system that surrounds the viewer with 4 screens. The screens are arranged in a cube made up of three rear-projection screens for walls and a down-projection screen for the floor; that is, a projector overhead points to a mirror, which reflects the images onto the floor. A viewer wears stereo shutter glasses and a six-degrees-of-freedom head-tracking device. As the viewer moves inside the CAVE, the correct stereoscopic perspective projections are calculated for each wall. A second sensor and buttons in a wand held by the viewer provide interaction with the virtual environment. [24, 25]. Objects in the virtual scene do not just appear on the CAVE walls and beyond, they can appear to enter into the physical space of the CAVE itself, where the user can interact with them directly [21].

Limited access to systems for research and development resulting from high costs reduces the number and range of potential users [5]. However, with the pace of low-end PC platforms, these systems are now rapidly increasing in numbers and numerous installations exist around the world.

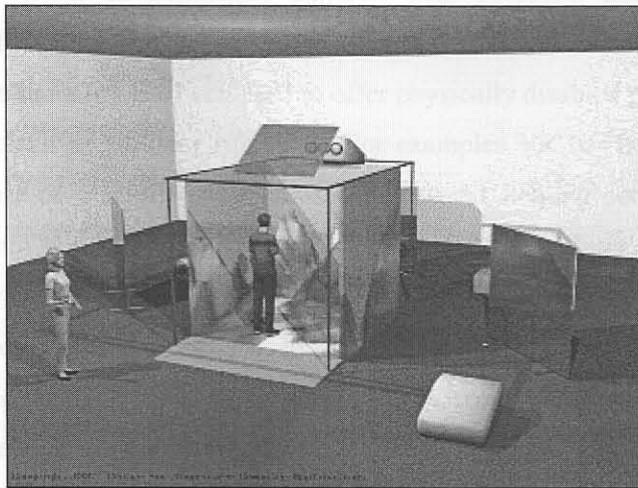


Figure 4. Illustration of the CAVE system [19]

2.5 Virtual Reality in Education

For educational purposes, virtual reality has been proposed as a technological breakthrough that holds the power to facilitate learning. In this section, virtual reality is explored from an educational perspective. The applications of virtual reality in various education subjects are presented, followed by highlighting attributes of virtual reality useful for the pedagogy of, and for learning with virtual reality. The concerns and factors influencing the use of virtual reality in education are then presented.

2.5.1 Educational virtual reality applications

Virtual reality may be used as a means of enhancing, motivating and stimulating students' understanding of certain events, especially those for which the traditional notion of instructional learning have proven inappropriate or difficult. Virtual reality has many applications in various disciplines such as applications designed to for the purposes of training, such as vehicle simulators, medical and military training. In this section however, the applications of VR in the education of young learners is addressed. Virtual reality has been incorporated into many curriculum objectives in the educational domain and the list of educational subjects covered by the use of VR is quite broad. A range of VR applications categorised by educational subjects is presented below, followed by how pedagogic support is embedded in these VR applications.

Special education and life skills

With the use of virtual environments, children with profound and multiple learning difficulties have the chance to access areas of the real world which they might never experience [4]. Virtual reality has been used to offer physically disabled persons the sensation of movement and teach preparatory life skills. For examples VR has been employed in the simulation of the use of a wheelchair in a simulated world, helping people gain experience before venturing into the dangerous real world. With VR technology it is not difficult to imagine a person with a physical disability competing with an able-bodied challenger on even ground in the world of virtual reality [5]. In the training and rehabilitation of young bicycle users, Carnegie Mellon University has developed a Virtual Bicycle projection based VR application, where a user can encounter various accident scenarios [26].

History and world cultures

Exploring of other cultures and ways of living has been achieved using collaborative virtual environments to communicate with other children in other parts of the world. In [4], research on the Vivid Group is presented on a Virtual Cities experiment. A three-way satellite feed system allows children from around the globe to interact with one another in a shared virtual environment. VR environments have also been used to present historical events and ancient cultures. In [36], VR is used to present a cultural heritage application of a once lost Southern African community called Cato Manor, where a user can take on the virtual identity of a young Zulu boy in experiencing the virtual environment. In learning about ancient civilisations, a Desktop VR application of a Greek villa has been developed by Sheffield Hallam University in the United Kingdom for learning about Ancient Greece [26].

Science and Mathematics

VR has been used in exploring chemistry, where three-dimensional models help in understanding the shapes and properties of complex molecules, and in teaching the laws of physics. The effects on physical forces like gravity can be experimented visually, by altering the strength of such forces and therefore creating alternate worlds that violate physical laws of our universe [4]. In [33], three ‘ScienceSpace’ worlds have been developed for learning complex and abstract scientific concepts: ‘NewtonWorld’ for learning about Newton’s Laws of Motion and the conservation of kinetic energy and linear momentum; ‘MaxewellWorld’ to explore the nature of electrostatic forces and fields, aiding students to understand the concept of electric flux and discover Gauss’ Law; and ‘PaulingWorld’ for learning about the composition of molecules. In teaching algebraic equations, VR has been employed where students could become parts of the equation, putting pieces of the equation into equilibrium to solve mathematical problems [4].

Biology and Earth Science

The Georgia Institute of Technology, Graphics Visualization and Usability Center has developed a Virtual Gorilla Exhibit which allows a user to explore a representation of a gorilla exhibit at a zoo, in understanding a zoo habitat, gorilla behaviours and social interaction [26]. VR has also been used to teach the ‘building blocks’ of earth system science, involving the physical, biological and chemical processes, which together make up the earth system [13]. Subjects such as the life cycle of plants and animals in biology, the weather and water cycle, and volcanism and the movement of tectonic plates in geology have been taught with VR applications (see Section 2.5.3.1 for examples thereof). In learning the principles of human cell biology, ERG Engineering, Inc. have developed a Cell Biology VR application for use with a HMD, allowing the user to build cells from different types of organelles [26].

2.5.2 Pedagogies and Learning with Virtual Reality

A number of characteristics of virtual reality hold the potential to enhance the effectiveness of using computers as an educational medium. VR facilitates new kinds of learning experiences that are highly perceptual in nature, and which enable students to be immersed within a phenomenon visually, auditory and haptically. The Virtual Reality and Education Laboratory (VREL) at East Carolina University claim virtual reality has the potential to change the way we learn [9].

Research by Andolesk indicates that virtual reality offers enormous educational possibilities. He says: “Virtual worlds engage students cognitively and affectively and the interactions in

these worlds are intuitive. VR is regarded as a highly promising simulation tool, a computer-human interface, a communications medium, and an artistic medium. Applied to instruction, virtual reality has the potential to revolutionize teaching and learning processes. It is for this reason that educators must be aware of this new educational technology” [4]. Dede, Salzman, Loftin and Ash (1999) state: “Our studies are exploring new ideas about the nature of learning based on the unique capabilities that virtual reality provides” [8].

The following highlights a number of attributes and consequences of using virtual reality, which present advantages of using VR as a medium of instruction:

3D

Traditional methods of displaying and visualising models and data are two-dimensional to their nature even through they seek to describe a reality that is often three-dimensional. VR allows students not only to visualise models and data in a more appropriate three-dimensional context, but also to interact with the models and take on several different points of view, changing the models’ relative sizes as well as the perspective from which the users experience the models. Virtual reality improves students recognition and manipulation of 3D shapes, improves students motivation and general class performance [26].

Exploring

VR allows for the exploration of real things that, without changes to scale in size and time, could not otherwise be effectively examined, such as molecular structures. Exploring existing places that students would not otherwise have access to, e.g. travel to distant locations, such as the surface of Mars.

Interaction

VR allows for interaction with an environment in non-realistic ways, for example having extraordinary powers and flying around a generated environment [4]. VR also allows for simulating reality for human communication. With a simulated environment, a virtual meeting room can be created with parties interacting together in the virtual room although separated by a great geographical distance. The people present in the meeting can be represented in the virtual environment as virtual characters, also called avatars. Virtual worlds may be constructed to include cues, prompts, reinforces, and feedback delivered through visual, auditory, or even haptic modalities [5].

Understanding

Unique to virtual reality is the ability to make abstract concepts concrete [5]. VR allows for changes in the relative sizes of the user and the objects of the virtual environment. Parameters that cannot normally be seen, such as radiation beams or sound frequencies, can be seen, heard, and even felt in virtual worlds [5]. Representations of abstract concepts like mathematical functions or scientific structures to enable learners to physically interact with them and/or view them from different perspectives, improves learners understanding. Learners may be able to construct mental models of phenomena that have no counterpart in their everyday experience [33]. Virtual reality may be used for creating places and things with altered qualities, such as Earth during an ice age. Exposing students to such experiences trigger insights that lead to a better understanding of the phenomenon as a whole. In [4] it is mentioned that when dealing with abstract concepts through VR simulations, students can develop and retain cognitive skills much easier, hence removing barriers to learning for disadvantaged and gifted students alike.

Economics and safety

In situations where it is too dangerous or expensive to allow students to take on the roles they want to learn, VR can provide realistic experiences through simulations. Simulator training offers invaluable, true-to-life experience without risking the health and safety of the learner or the destruction of expensive equipment [4, 5].

Experience

VR allows for experiences that are not possible in the real world, as it allows for changes in the relative sizes of the user and the objects in the virtual environment. At one extreme, the user could interact with and even step into atoms and electrons, while at the other extreme acquire a sense of distance in the universe by visualising planets and moons. The creation and visualisation of representations of objects and events that have no physical form in the real world are achieved with VR. Virtual reality provides for both individualised and collaborative/cooperative-learning environments, where one can experience communication mechanisms not possible in everyday life. In cognitive science, a whole area of research is striving to enhance collaborative and situated learning [7].

Immersive factor

Immersive VR makes multi-sensory cues to interact with the user, which allows the designer of the virtual environment to use interface designers to present information that is not available to human senses in a direct and clear manner. For instance, variations in the

intensity of sound may be used to indicate the current level of radiation, and different places could be given different colours that correspond to the current temperature in that area.

The above list is not at all exhaustive, and there are many more advantages of using VR in education. Dede, Salzman, Loftin and Ash (1999) state: “Our research suggests that such immersive, multi-sensory experiences enhance students’ abilities to conceptualise and integrate complex, abstract scientific ideas” [8].

By allowing students to experience various activities in a virtual world, these worlds are intended to help these students learn basic skills that will help in their daily lives [26]. But one may ask what are the pedagogical capabilities of these educational systems? What children can learn from a virtual reality system depends on the extent to which pedagogy is embodied in a virtual world, and the extent of teacher support that is provided.

In [26], pedagogical support in virtual worlds are categorised as the follows pedagogical paradigms which support constructivist learning:

- *No pedagogical support.* This is the simplest type of pedagogy and involves minimal interaction and a simple walkthrough of a virtual world to support the educational objectives. In cases where awareness of VR technology and its possible applications is provided, no pedagogy is needed.
- *Experiential.* With experiential learning, students need to experience the concepts and principles contained in the content as much and as directly as possible. This pedagogy promotes that the learner has control over, or personal involvement in the experience, that there should be an appropriate amount of a challenge for a student, and learning experiences should involve more than just one person, directly or indirectly, establishing group interaction.
- *Guided-inquiry.* In the guided-inquiry paradigm, a student’s interaction with the virtual world is guided. Here the pedagogical support is not embedded in the virtual world, but can be provided by other means, for example textual material or a set of questions to be answered about the virtual world.

2.5.2.1 Concerns and factors influencing the use of Virtual Reality in education

The following section presents the concerns and factors influencing the use of VR in education.

The use of VR itself

The attitudes of parents, teachers and children towards computers are a greatly influencing factor of using computers in the classroom and at home. Computer literacy of parents, teachers and school children is an important factor limiting the use of VR in education. Ease of use of the virtual reality application, in terms of both VR hardware and software is another contributing factor.

Pedagogical grounding

Virtual reality technology relies on software applications that do not necessarily have pedagogical grounding [6]. Teachers need to know how to incorporate computers into their teaching strategies and activities in addition to knowing how to use the software [6].

Economics of VR hardware

Should virtual reality be used in situations where it is neither appropriate nor cost-effective? Current VR hardware is expensive and is consequently limited to situations with special funding such as academic institutions and research environments. In a developing country such as South Africa there are hardware constraints in a public school setting. Desktop VR could be a viable alternative.

Ergonomics of VR hardware

Since VR is a relatively new field, the ergonomics of young children using VR has not been addressed. VR hardware has not been designed for use with young children and sometimes shutter glasses are too big, and the devices are too big to fit in the user's hand. Some cases of simulator sickness have also occurred. Also, how a child sits at a computer, in the case of desktop VR, determines whether they avoid eye, shoulder, or back strain [14]. Children need to be advised on such ergonomics, and computer hardware may require investment (e.g. a child may have trouble with an adult sized mouse, where a mouse designed to fit a child's hand will need to be purchased).

2.5.3 Virtual laboratories

Virtual laboratories can be described as virtual environments for educational purposes. Virtual laboratories can be regarded as an extension of CAL software for learning (see Section 2.3). In this section, examples of virtual laboratories in education are presented, including techniques for evaluating virtual laboratories, a guide for the development of a virtual laboratory and the major problems of virtual laboratories.

2.5.3.1 Virtual laboratories in education

In a recent report by the Institute for Defense Analysis, Christine Youngblut has comprehensively surveyed work in virtual environments, citing approximately 50 VR-based learning applications and 35 studies, which include desktop virtual environments [26]. There are currently very few VR-based learning environments designed for young children [32].

Most of these virtual laboratories have been developed for studying the effectiveness of VR on the individual learner, and on whether their use can be successfully aligned with a school's goals, curricula, practices and culture.

The virtual laboratories examples in this section have been presented by looking into the following criteria (where the information has been available or applicable):

- *Pedagogy*. The educational content and pedagogies employed. Educational advantages of the virtual laboratory.
- *Technology*. The VR technology used.
- *Authoring tool*. The virtual laboratory has an authoring tool.
- *Industry Classification*. Has been developed by the research or commercial arena.
- *Classroom*. Has any investigation of the virtual laboratory been conducted on young children in or outside of a classroom environment?
- *Teacher driven*. Teachers' influence in development or a teacher driven system.
- *Collaboration*. Collaborative or individual learning experience.

NICE [26, 29]

NICE (Narrative, Immersive, Constructionist/Collaborative Environments for Learning in Virtual Reality) is a setting for a virtual island where children can search for empty space and build their own ecosystems (see Figure 5). Symbolic representations of various environmental changes are used to facilitate children's understanding of complex ecological interrelationships. Designed to work in the CAVE (see Section 2.4.1.3), and related project-based VR hardware, NICE allows groups of children to learn together in both the same physical location, as well as remotely located sites.

Figure 5. A colony of Quackwicks [31]

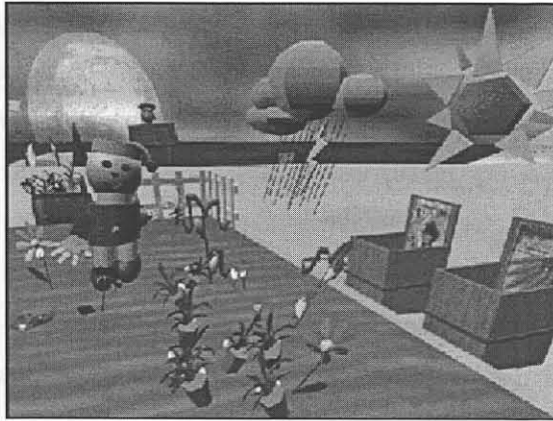


Figure 5. A scene from NICE [29]

QuickWorlds [31]

The QuickWorlds program is intended to provide a fast-turnaround mechanism for teachers who would like to make virtual models available to their students as part of the regular learning program. The models that have been developed range from simple static models such as a wood ant, and the interior of the Earth, to more complicated dynamic models such as the volcano, iceberg, human heart, and solar system (see Figure 6).

The QuickWorlds program has addressed working closely with teachers, as they are responsible for structuring the learning experience and guide the actual lessons using the models. The VR equipment used in QuickWorlds is the ImmersaDesk (see Section 2.4.1.3), which has been installed at the school for the study of how VR may benefit conceptual learning. An example of the pedagogy employed is the use of the heart model by the physical education teacher as reinforcement learning in teaching the function of the human heart.

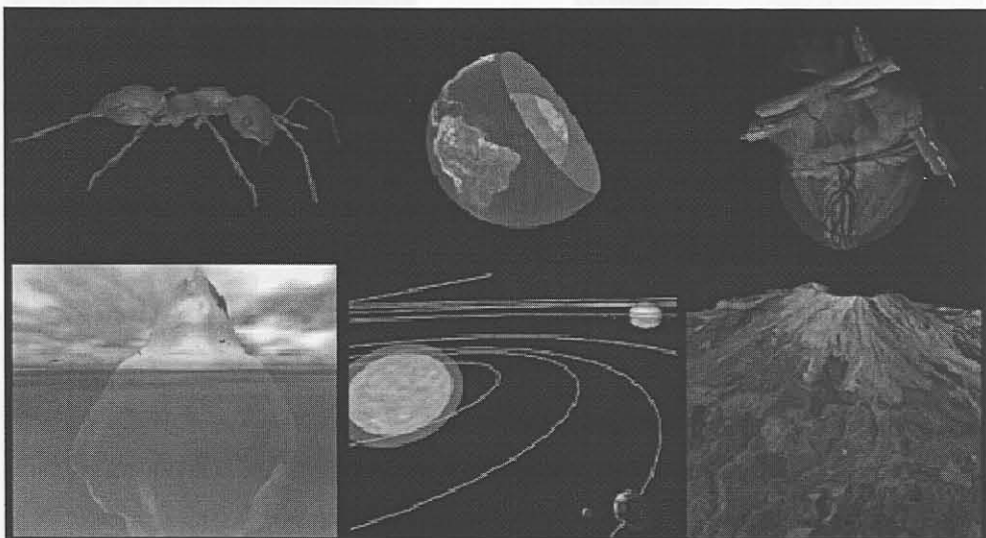


Figure 6. A selection of Quickworlds [31]

First-Person Science Inquiry in Virtual Ambient Environments [30]

Virtual ambients are simulated synthetic environments, which have been designed to support science inquiry learning among elementary school students. Hence users may observe the phenomena in the virtual environment, but cannot affect the course of the underlying simulation.

In a virtual ambient, the Field (see Figure 8), students collaboratively explore a large “natural” terrain populated by up to eight different plant types [30]. The Field is configurable and has an authoring tool in the form of a standalone Java application that allows for selecting a plant type and clicking on the desired location. The system has been implemented on an ImmersaDesk (see Section 2.4.1.3), for the reasons of providing a wide visual field-of-view, supporting collaborative investigations and maximizing the sense of immersion to enhance authenticity (see Figure 7).

The system was tested at a school that places a strong emphasis on environmental awareness. The following is the pedagogic approach used to present sixth grade students to the mathematic concept of co-occurrence using the Field virtual ambient:

- *Scouting.* Writing general observations and taking snapshots of findings.
- *Preliminary discussion.* Presentations of findings from the scouting exercise.
- *Exploration and data collection.* Students take on different roles: navigator, driver, data announcer and data recorder.
- *Follow-up discussion.* Here the teacher teaches the mathematical concept of co-occurrence.

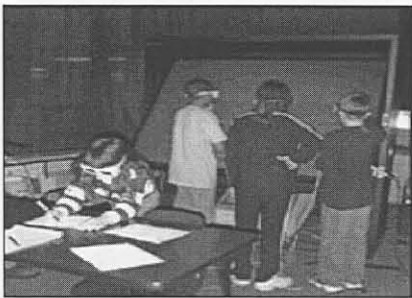


Figure 7. Students interacting with the Field [30]

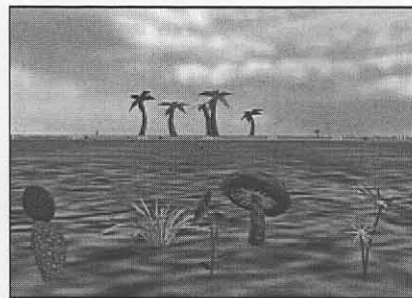


Figure 8. A scene from the Field [30]

Pond-Eco-System Simulator [27]

Argus Virtual Reality International has developed a Pond-Eco-System simulation (see Figure 9). From [27], the following is stated about the simulator: “Here biology is taught through active simulation, where students can interact with their subject. Students can take the role of the different types of life, of flying, swimming or crawling through the virtual environment. Simulated insects operate, as in life, with their own independent goals along with their own pathways”.

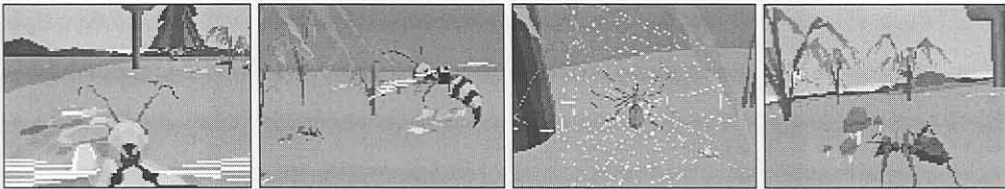


Figure 9. Scenes from the Argus VR Pond-Eco-System simulator [27]

The Round Earth Project [32]

In [32], collaborative virtual reality technologies are used to support pedagogical strategies in teaching children that the Earth is spherical (see Figure 10). This project involved the VR technologies of both an ImmersaDesk and a CAVE system. The pedagogy employed here was the process of children taking on different roles: an astronaut that explores the surface of an asteroid (in a CAVE system), and a mission controller, views and guides the action performed by the astronaut on that same asteroid as seen from an orbital distance (on the ImmersaDesk). Another pedagogy employed was to take control of a spaceship, from the time of lift-off, orbiting the Earth from a distance, and re-surfacing.

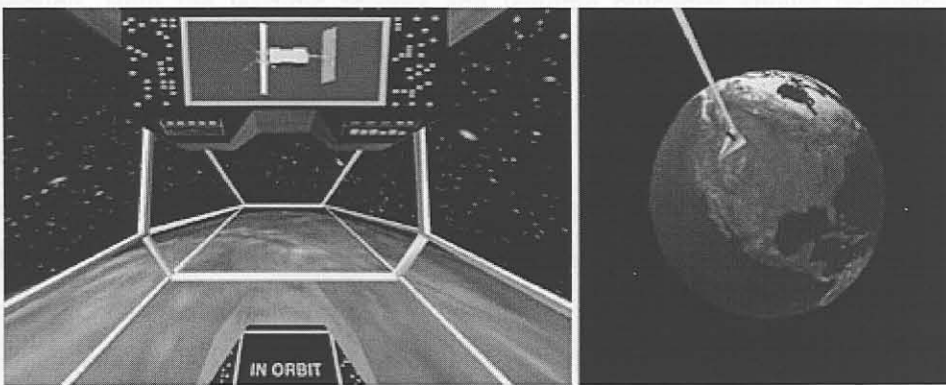


Figure 10. A scene from the Round Earth Project [32]

3D Traffic playground [28]

In [28], a 3D Editor for Traffic Playground was developed, for teaching children traffic rules. In this application, the world is defined by how traffic squares composed of road elements are

placed in the editor (see Figure 11). Their orientation and placement directly influence the collision detection model and the flow of movement of traffic within the world. Using the system, it is possible to try basic responses to various traffic situations that are simulated by the system and presented to the user in a simple way.



Figure 11. The 3D Editor for Traffic Playground [28]

2.5.3.2 Techniques for evaluating virtual laboratories

Dede, Salzman, Loftin and Ash (1999) [8] have investigated four issues that are critical to evaluating virtual reality worlds:

- *The learning experience.* The VR experience can be characterized along several dimensions. Focus on the participants' subjective judgements of usability, simulator sickness, immersion, meaningfulness of models and representations, and motivation can be addressed.
- *Learning.* Both the learning process and learning outcomes should be addressed in learning. To access learning outcomes, the mastery of concepts at both the “descriptive” and “causal” levels using multiple measures (e.g. conceptual, two-dimensional, and three-dimensional understanding) can be examined.
- *The learning experience versus learning.* It is important to contrast and understand the relationship between the experience and learning and to identify when the VR experiences help or hinder learning.
- *Educational utility.* This contrast centres on whether, for particularly complex and abstract domains, the medium is a better (or worse) teaching tool than other pedagogical approaches. The quality and efficiency of learning among different alternatives of varying cost, instructional design, and teaching strategy should be compared. In particular, learning outcomes should be compared to less-complex technology-based scientific modelling approaches, such as two-dimensional “microworlds”.

Pantelidis and Auld of the Virtual Reality and Education Laboratory (VREL) at East Carolina University, US say: "...virtual reality is an instructional tool, one of many that effective instructors use. The choice of an instructional tool depends on the specific lesson objectives, the student(s), and the teacher – the decision to use VR is the instructor's" [9].

2.5.3.3 Development of a virtual laboratory

Any virtual laboratory that is being developed needs contributions from education, from computer scientists and knowledge of the taught domain.

Virtual laboratories may be abstract worlds, or may have been designed to meet specific school curriculum objectives. When developing a virtual laboratory it is important to form technological relationships with elementary students and elementary teachers while working in classrooms. Collaboration with teachers is a necessity. Developers also need to understand the basics of elementary teaching.

The main area of software engineering interested in education is reusability, because of the enormous efforts necessary to develop actually used intelligent systems, and because most efforts in that direction start from scratch [7]. Lelouche states that in using software engineering principles, it is important that there must be techniques and tools to design/develop more cost effective systems [7].

Where will virtual laboratories be in ten years from now? What new and special educational applications will be available to teachers and students? Dede, Salzman, Loftin and Ash (1999) [8] discuss development of virtual laboratories of the future: "Within the next decade, the video game industry will develop devices capable of multisensory immersion ubiquitously available in rich and poor homes, urban and rural areas. To compete with the captivating but mindless types of entertainment that will draw on this power, educators will need beautiful, fantastic, intriguing environments that also foster deep and effective learning". They also say this research will produce another important outcome, "a deeper understanding of human learning".

2.5.3.4 Major problems of virtual laboratories

In realising the factors influencing the use of virtual reality in education and the points discovered in the development of a virtual laboratory, the following list includes major problems involving the implementation of virtual laboratories:

- Not developed on any teaching framework, or do not support the school systems.
- Not developed as re-configurable systems.
- May be very technical.
- Portability on different operating systems is not addressed.
- Most virtual laboratories have been built for expensive virtual reality systems (e.g.: CAVE) which are available only to research institutions or funded research organisations. Immersive educational environments are being developed using high-end equipment. They are consequently limited to situations with special funding such as academic institutions and research environments.
- Developing country issues are not addressed and this educational technology medium has until now only benefited the developed world.
- Not educationally viable and do not address educational needs.
- Virtual laboratories share problems of the concerns and factors influencing the use of virtual reality in education, since they rely on virtual reality as their medium.
- Pedagogy. The relationship between the development and use of new interactive technologies, and traditional pedagogic theory and practice in schools, is becoming somewhat incompatible – is it a case of pedagogy not keeping up with technology; or of technology not delivering what practice and theory in pedagogy demands? [11]

2.6 Summary

In this chapter the field of technology in education was discussed. Technology such as computers and virtual reality were explored from an educational perspective in terms of teaching and learning. The main focus of this thesis in terms of virtual laboratories in education was explored, introducing examples of virtual laboratories and including techniques for evaluating and developing them. The next chapter discusses the theoretical approach that was followed for building the iTiles framework. It presents an approach that can cater for some of the major problems of current virtual laboratories.

3.1.1 Introduction to the framework

Virtual laboratory solutions today are beginning to present students with a kind of interactive encyclopaedia and are becoming more accepted for their benefit in the classroom [39].

Chapter 3

“The ability to form ‘abstract’ concepts is probably the basis of man’s ability to reason”

- Edward de Bono [37]

Theoretical approach

In this chapter, the theoretical approach and underlying concepts of the Intelligent Tiles (iTiles) virtual laboratory framework are presented. Firstly, the chapter introduces the iTiles framework and secondly the aims of the framework are presented. Next the key theoretical concepts of the iTiles framework are discussed: the components, and behaviour of an iTiles world. These concepts are explained in more detail in the sections that follow: transforming an iTiles world, simulating behaviour for an iTiles world and the movement of characters in the simulation. The chapter ends discussing how young learners can learn with an iTiles virtual laboratory and educational advantages of the iTiles framework are highlighted.

3.1 The Intelligent Tiles virtual laboratory framework

The main research of this thesis involves the creation of a framework that can be used for the design and development of a virtual laboratory, aimed for use in the education of young learners. The virtual laboratories that can be developed through the use of this framework are ecosystem type virtual environments used in teaching biological subjects such as earth science. These virtual environments are worlds consisting of a terrain inhabited by a set of virtual life forms.

3.1.1 Introduction to the framework

Virtual laboratory solutions today are beginning to present students with a kind of interactive encyclopaedia and are becoming more accepted for their benefit in the classroom [39].

3.1.2 Aims of the iTiles framework

The Intelligent Tiles (iTiles) framework, originally introduced in [39], is a method and approach that can be used for the development of a teacher driven virtual laboratory. The framework forms a foundation for a virtual environment that allows for deviations, and accommodates extensions to the framework. Being a framework for a virtual laboratory, the framework attempts to cover both virtual reality and educational aspects involved.

The main principle of the iTiles framework is the use of tiles to author a virtual environment. The virtual environment is simple, without much complexity and little technical know-how and understanding is needed for it to be applied as a teaching aid. The virtual environment produced using the concepts in the iTiles framework is called an **iTiles world**.

The use of a virtual laboratory produced with the iTiles framework, in an educational setting, involves three simple steps (see Figure 12):

1. Author the iTiles world.
2. Specify the behaviour of the world objects that inhabit the authored iTiles world.
3. Present a simulation of the authored iTiles world to young learners to interact and learn from using an experiential/learning-by-doing pedagogy.

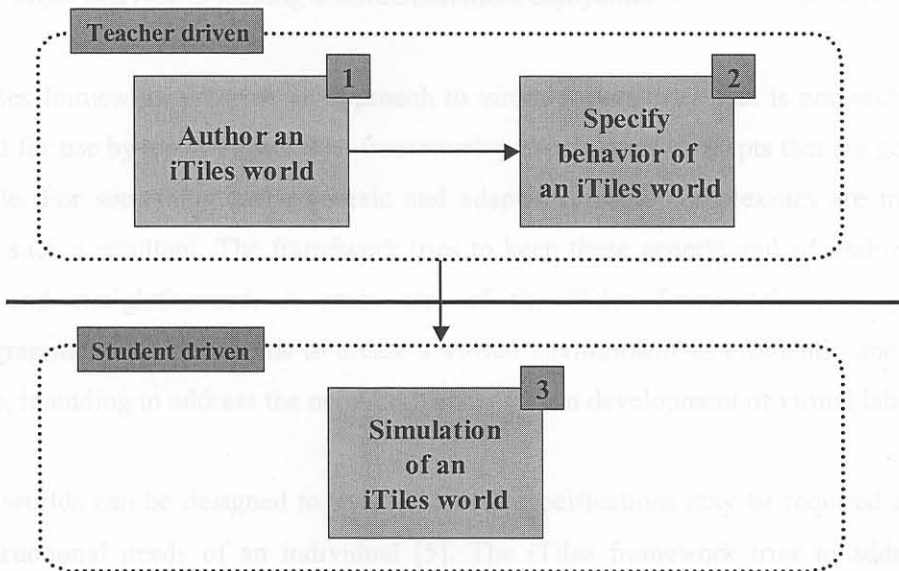


Figure 12. The iTiles framework

Steps 1 and 2 are teacher driver, while step 3 is for students. More advanced students could perform steps 1 and 2.

3.1.2 Aims of the iTiles framework

The main objective of the iTiles framework is to aid the education of elementary school children by facilitating the teacher driven development of virtual laboratories. In addressing the major problems involving current virtual laboratories, the iTiles framework tries to address the problems through the following aims and objectives:

The iTiles framework makes use of virtual environments and multimedia, to support science and technology education and aims to advance the educational use of virtual laboratories. It facilitates the learning process by helping children make sense of the world they live in, in ways that were not possible up to this time, which are uniquely possible by using educational technology as a medium. The framework addresses the current cost and limited accessibility of immersive VR systems, by allowing development of virtual laboratories on a platform of broader use (e.g. Desktop VR). The framework also involves virtual reality concepts, both audio and visual elements, and is aimed for the development of a multimedia and interactive experience. It considers important visualisation and interactivity factors that play a major role in the user experience of a virtual environment. Not only does the framework support visualisation, but it also incorporates sound, which presents a more immersive world for the user to explore. Herewith a world can “come to life” in a more advantageous way since sound plays an important role in making a simulation more enjoyable.

The iTiles framework presents an approach to virtual laboratories that is non-technical and intended for use by a non-expert. The framework presents some concepts that are generic and adaptable. For something to be generic and adaptable, some complexities are in order to achieve such a resultant. The framework tries to keep these generic and adaptable qualities simple and straightforward. A main aim of the iTiles framework is to provide a nonprogrammer with the means to create a virtual environment as efficiently and easily as possible, intending to address the need for teacher driven development of virtual laboratories.

Virtual worlds can be designed to meet whatever specifications may be required to address the instructional needs of an individual [5]. The iTiles framework tries to address these instructional needs by giving a teacher the creative freedom to author the environment for a lesson and specify the behavioural aspects of the environment. If learning is made more interesting and fun, students may remain engaged in an activity for longer periods of time. Research to date indicates that virtual worlds of colour, shape, sound, and feel should amplify the powers of the mind to see complex sets of data and to absorb, manipulate, and interpret

information more quickly and completely [4]. Using the iTiles framework, it is aimed that such virtual worlds be produced.

3.2 Composition of an iTiles world

To understand the composition an iTiles world, the components of the virtual environment are explained in this section. The **components** of an iTiles world are **tiles**, and **world objects**.

The **tile components** form the basis of an iTiles world. Each tile is square in dimension and is associated with an element type. These element types are analogous to geographical terrain types that appear in the real world, for example grass and sand. A tile's element type is visually represented by a unique texture or a specific colour. For any iTiles world a tile set needs to be considered with unique tile element types being specified. Tiles elements are grouped together as a two dimensional (2D) matrix layer of tiles, to form the base terrain of an iTiles world. The size of the world is variable and is dependant on the number of rows and columns specified for the matrix when an iTiles world is authored. The visual representation of the world is established by presenting this flat 2D base terrain in a three dimensional (3D) environment. By applying a 2D tile system in a 3D environment, there are many advantages gained. 2D tiled worlds can only be scrolled and there is usually a fixed viewpoint. In 3D, a feeling of depth is achieved since the world can be zoomed in and out, and can also be rotated on an axis for a different point of view [39].

The **world object components** of an iTiles world are representations of real world and imaginary objects that populate the base terrain of the world, and exist on top of individual tiles. In an iTiles world each tile entity of the base terrain may either be free, or may be occupied by an individual world object. World objects are visually represented by 3D models. These 3D model representations may require scaling in order to be placed on top of tiles in proportional dimensions. Therefore each world object has a minimum and maximum scale specified. A world object may face one of eight different directions on a tile, as it can face any of the tile's four edges or corners.

The world object components have been divided into two categories: static and dynamic world objects. **Static world objects** are immovable world objects that have fixed positions in the environment. An example of a static world object is a tree or a rock. **Dynamic world objects** are world objects that can move in the environment. Dynamic world objects are analogous to virtual **characters**, and will be referred to as either characters or dynamic world objects throughout this and the following chapters of this thesis/dissertation.

World objects are placed on top of tiles based on their **tile attribute list**. This tile attribute list is a list of the tile elements that a world object may be placed on in the environment. In the case of dynamic world objects this list also determines what tile elements the dynamic world objects may move on. For example, a dynamic world object with sand and water tile elements in its tile attribute list can only move on tile elements with the property of sand or water. This tile attribute list also determines the collision detection model, since dynamic world objects will not be able to move on tile elements not listed in their tile attribute list.

An iTiles world does not only come to life by characters moving in the world, but also by the sound they emit and behaviour they exhibit when interacting with the world, other characters and static world objects. The sound features of the environment and behaviour of world objects are presented next.

3.3 Behaviour of an iTiles world

Simulation of a virtual environment involves the animation and movement of the characters that inhabit that virtual environment. Movement can be a result of behaviour and internally defined attributes, such as the relationships of a character with other components of the virtual environment. The **iTiles World Flow** of an iTiles world can be described as the forces of nature, or a specification of a set of rules by which an authored iTiles world must adhere. It is needed for the simulation of the motion and behaviour of characters, for governing the flow of movement of characters, and defining the procedure for the interaction of characters. The properties associated with other components of the environment, such as static world objects and tiles are also specified.

Each iTiles world requires an iTiles World Flow to be defined, since the iTiles World Flow is used as input to achieve the simulation of that iTiles world. Figure 13 illustrates the key iTiles World Flow concepts that are discussed in this chapter. These concepts are:

- *World transformations.* World transformations define visual and auditory changes that can occur in the simulation of an iTiles world. The way a tile element can transform is defined by a tile transformation. Both static and dynamic world objects may have world object transformations defined, indicating how they may transform in the simulation of the iTiles world.
- *World forces.* World forces define the relationships of a dynamic world object with other components of the virtual environment, in terms of the dynamic world object's behaviour and sound emitted. There are forces of attraction called positive forces, forces of

repulsion called negative forces, and forces that represent the relationship of a character's movement on tile elements, called movement forces. Dynamic world objects may have one or more positive forces, negative forces or movement forces specified. Positive forces and movement forces may have an action property defined. When these actions are executed they can trigger world transformations to occur.

3.4 Transforming an iTiles world

These iTiles World Flow concepts are presented in more detail in the sections that follow. World transformations are presented in Section 3.4, followed by a presentation of world forces in Section 3.5

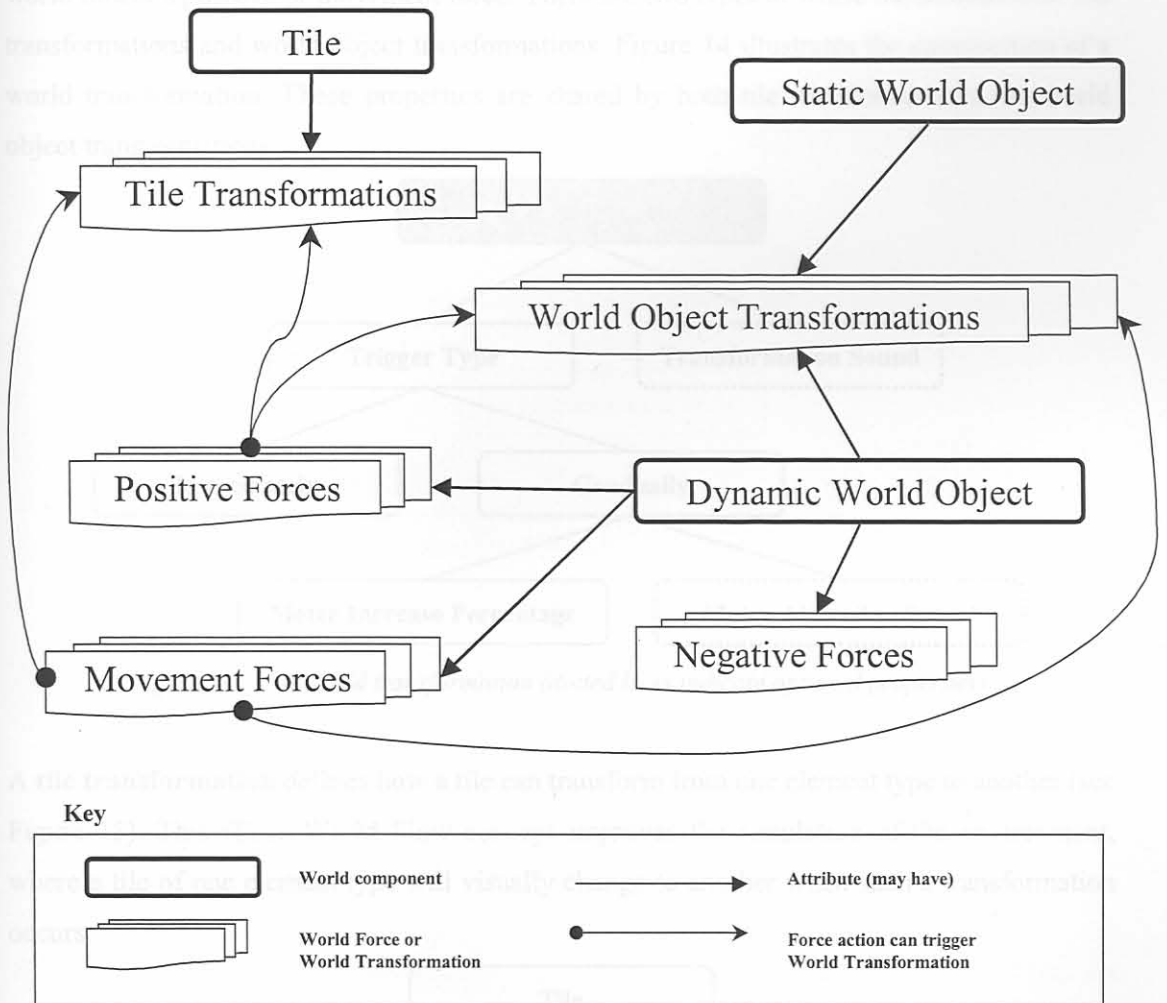


Figure 13. iTiles World Flow summary

In the simulation of an iTiles world, movement is dependent on time. The **iTiles world beat** is a unit of measurement of time used in the simulation of an iTiles world. A single world beat represents one time unit and could be any amount of real time. By changing the amount of time that a world beat represents, the simulation can be slowed down or hurried up. The iTiles World Flow concepts are time dependent and their time attributes are measured in world

beats. Animating the movement of characters in the simulation is related to simulation time, so that it is not dependent on the processing power of the underlying computer hardware. The movement of characters is then performed every iTiles world beat. The movement of characters is discussed in more detail in Section 3.6.

3.4 Transforming an iTiles world

An iTiles **world transformation** represents a visual change or an auditory event that can occur in the simulation of an iTiles world. World transformations are associated with tile elements, world objects and sounds, and are triggered to occur by an action of a dynamic world object’s positive or movement force. There are two types of world transformations: tile transformations and world object transformations. Figure 14 illustrates the composition of a world transformation. These properties are shared by both tile transformations and world object transformations.

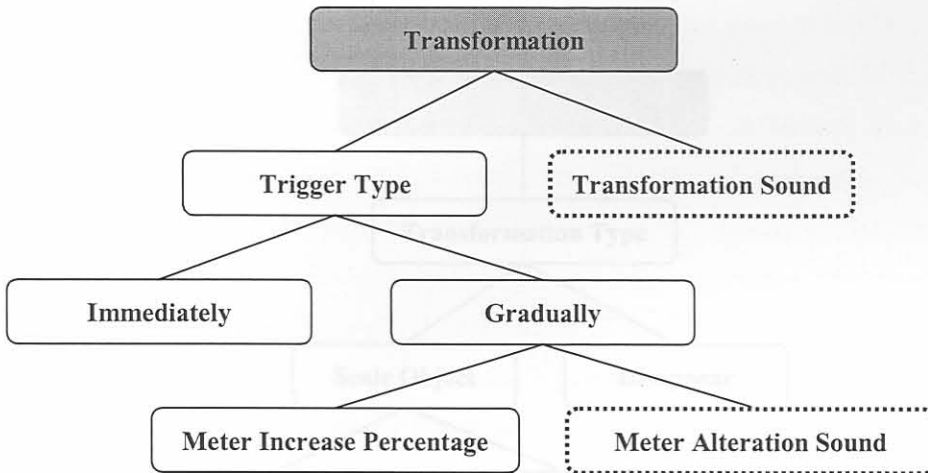


Figure 14. A world transformation (dotted lines indicate optional properties)

A **tile transformation** defines how a tile can transform from one element type to another (see Figure 15). This iTiles World Flow concept improves the simulation of the environment, where a tile of one element type will visually change to another when such a transformation occurs.

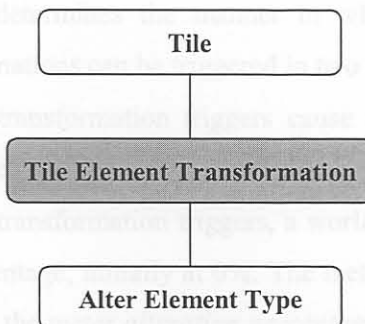


Figure 15. Tile element transformation

A **world object transformation** defines how a world object may transform in the simulation. Three types of transformations exist for a world object (see Figure 16):

- *Scale up.* World objects can transform to be scaled up by a certain percentage. In order to retain visual conformation, world objects will continuously scale up until their maximum scale is reached.
- *Scale down.* World objects can transform to be scaled down by a certain percentage. In order to retain visual conformation, world objects will continuously scale down until their minimum scale is reached.
- *Disappear.* World objects can be removed from the simulation. Once a world object has disappeared from the world, it shall be removed from the tile it is currently on, from the simulation and not reappear.

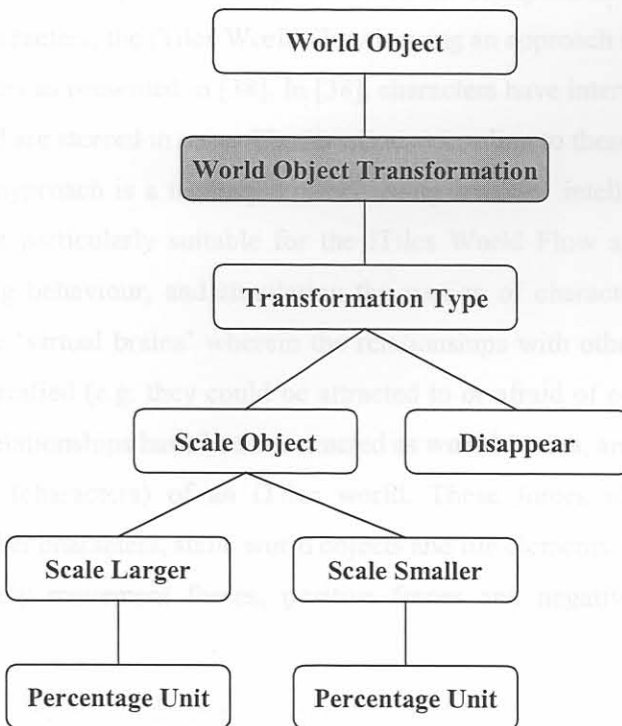


Figure 16. World object transformation

A **transformation trigger** determines the manner in which world transformations are influenced to occur. Transformations can be triggered in two ways:

- *Immediately.* Immediate transformation triggers cause world transformations to occur immediately when triggered.
- *Gradually.* With gradual transformation triggers, a world transformation contains meter, which is stored as a percentage, initially at 0%. The meter's value can be increased by a certain percentage, called the *meter alteration percentage*, specified with the influencing alteration percentage must be specified, indicating the percentage by which the meter of these

triggering force. The world transformation only occurs when the meter reaches full capacity at 100%.

World transformations can have an audio representation where a sound clip may be played when the transformation occurs (called the *transformation sound*). With world transformations that have gradual transformation triggers, a sound clip may be specified that is played when the meter is influenced (called the *meter alteration sound*).

3.5 Simulating behaviour for an iTiles World

Sophisticated algorithms and AI techniques are required to define the behaviour and movement of a virtual character. The advantage of using these techniques is that the number of possible outcomes is nearly unlimited, given a rich set of inputs. In achieving movement and behaviour of characters, the iTiles World Flow is using an approach similar to the method of steering behaviours as presented in [38]. In [38], characters have internal forces that define their movement, and are steered in a specific direction, according to these internal forces. The steering behaviour approach is a technique of achieving artificial intelligence with minimal intelligence. This is particularly suitable for the iTiles World Flow approach. In order to achieve this steering behaviour, and simulating the motion of characters, characters of an iTiles world require ‘virtual brains’ wherein the relationships with other components in the environment are specified (e.g. they could be attracted to or afraid of certain components in the world). These relationships have been abstracted as **world forces**, and are only applicable to moving objects (characters) of an iTiles world. These forces identify a character’s relationship with other characters, static world objects and tile elements. There are three types world forces, namely movement forces, positive forces and negative forces, which are discussed below.

3.5.1 Movement forces

The movement force of a character (see Figure 17) represents how the movement of that character affects the tile elements it is moving on. This movement force may also reflect the manner in which the character is influenced moving on a certain tile element type. Movement forces may be specified for any of the tile element types appearing in a character’s tile attribute list. A movement force has an action, which when executed can trigger a tile transformation of the tile moved on to occur. This action may also trigger a world object transformation that belongs to the character to occur. If any of the trigger types of the world transformations specified in the movement force’s action are of the gradual type, a meter alteration percentage must be specified, indicating the percentage by which the meter of these

world transformations will increase. The action of a movement force is executed when the character moves on or to a tile with a movement force specified for that tile element type. As an example of a movement force, consider a movement force of a dog character on grass. When the dog moves on a patch of grass this can trigger the tile transformation of grass to sand in a gradual manner with a meter alteration percentage of 15% for the gradual transformation trigger type. The more the dog character moves on that patch of grass, the faster it will transform to sand. A movement force may also have an audio representation called a *movement sound*. The movement sound is played when a character moves on or to a tile with a movement force specified for that tile element type. For example, a water splashing sound can be played as a duck moves on water.

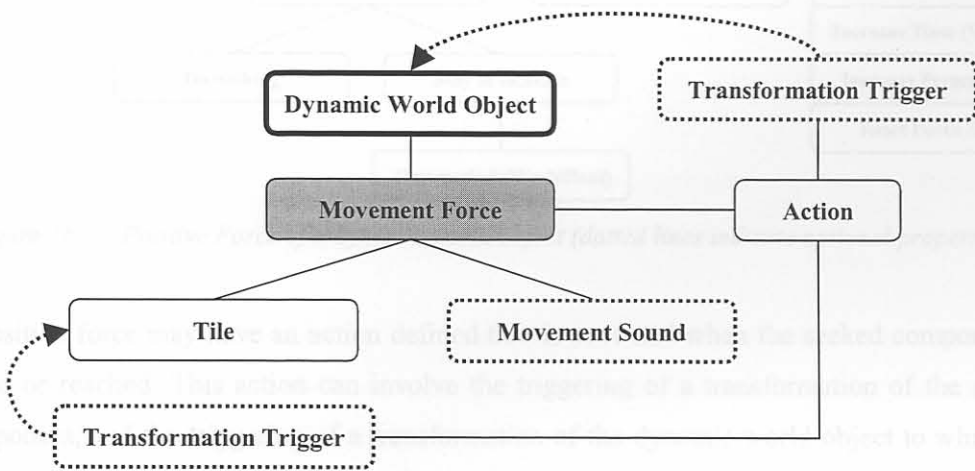


Figure 17. Movement force of a dynamic world object (dotted lines indicate optional properties)

3.5.2 Positive forces

A positive force of a dynamic world object (see Figure 18) defines a dynamic world object's force of attraction towards a component of the environment (i.e. a dynamic world object can have a positive force towards a static world object, a dynamic world object, or a tile element type). A positive force is expressed in terms of a force strength (in percentage), which indicates the strength by which the dynamic world object seeks the component to which it is attracted.

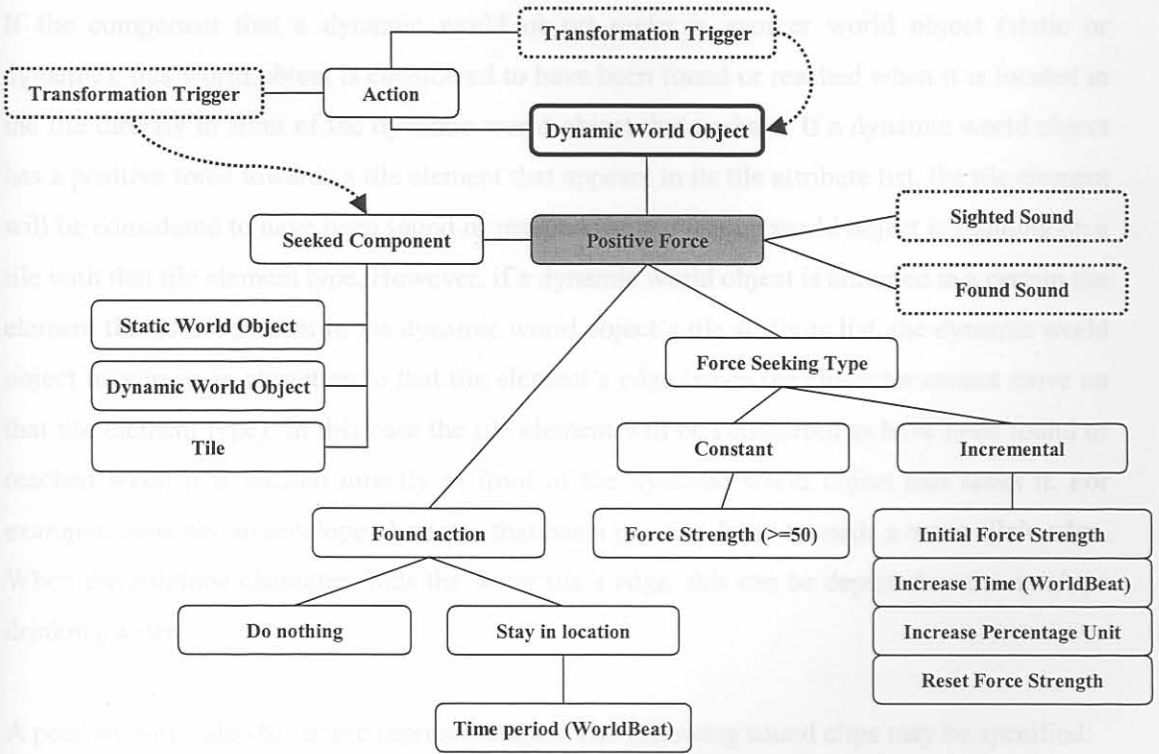


Figure 18. Positive Force of a dynamic world object (dotted lines indicate optional properties)

A positive force may have an action defined that is executed when the sought component is found or reached. This action can involve the triggering of a transformation of the sought component, and the triggering of a transformation of the dynamic world object to which the positive force belongs. This action therefore defines how the positive force has an influence on the sought component and on the character to which the force belongs. When a dynamic world object reaches or finds the sought component, there are two types of ‘found actions’ that may occur, which precede an action’s execution:

- *Do nothing.* The action will therefore execute immediately
- *Stay in location.* Specifying that a dynamic world object stay in the location for a specified number of world beats indicates a time period that will occur before the positive force’s action is executed. Here the dynamic world object will remain frozen in the same location for the number of world beats specified when the sought component is found. If the sought component is another dynamic world object, it too will be frozen for that time period (i.e. a dynamic world object can freeze another dynamic world object). The action will only be executed once this time has elapsed. Once the action has been executed the dynamic world object will be unfrozen and continue as normal in the simulation. If the sought component is another dynamic world object, this dynamic world object would too be unfrozen.

If the component that a dynamic world object seeks is another world object (static or dynamic), this world object is considered to have been found or reached when it is located in the tile directly in front of the dynamic world object that seeks it. If a dynamic world object has a positive force towards a tile element that appears in its tile attribute list, the tile element will be considered to have been found or reached if the dynamic world object is standing on a tile with that tile element type. However, if a dynamic world object is attracted to a certain tile element that is not present in the dynamic world object's tile attribute list, the dynamic world object may have an attraction to that tile element's edge (since the character cannot move on that tile element type). In this case the tile element will be considered to have been found or reached when it is located directly in front of the dynamic world object that seeks it. For example, consider an antelope character that has a positive force towards a water tile's edge. When the antelope character finds the water tile's edge, this can be depicted as the antelope drinking water.

A positive force also has audio representations. The following sound clips may be specified:

- *Sighted sound.* A sound clip may be played when a character sees the sought component directly in front of it.
- *Found sound.* A sound clip may be played when a character finds or reaches the sought component. Only the found sound (not the sighted sound) is played when the character finds or reaches the sought component.

There are two types of positive forces which influence how the force strength (the level of attraction towards the sought component) may change: constant seeking and incremental seeking. A **constant seeking positive force** has a constant force strength specified between 50% and 100%, which remains unchanged throughout the simulation. Figure 19a shows the fuzzy logic interpretation of this force strength scale. With an **incremental seeking positive force**, the force strength increases linearly over time (i.e. the force gets stronger over time) by a specified percentage unit. This positive force resets to a certain percentage once the sought component is found and the positive force's action has been executed. An incremental seeking positive force has the following attributes (see Figure 19b, 19c and 19d for the fuzzy logic interpretation of these attributes):

- *Initial force strength.* The force strength that will be allocated to a positive force at the start of the simulation. An initial force strength of 50% is considered of weak strength, whereas a force strength of 100% is considered the strongest.
- *Increase time unit.* This is the amount of time that will pass till the force strength will increase by the increase percentage unit. This time unit is measured in world beats.

- *Increase percentage unit.* The percentage by which the force strength will increase. The more force strength specified for the increase percentage unit, the stronger the resultant force strength will become.
- *Reset force strength.* This is the force strength to which the positive force will reset once the sought component is reached or found. A force strength between 0% and 50% is considered to have no strength. A force strength of 50% is considered of weak strength, whereas a force strength of 100% is considered the strongest.

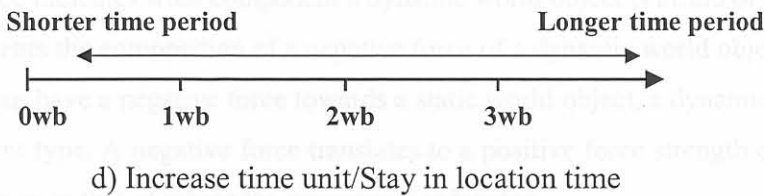
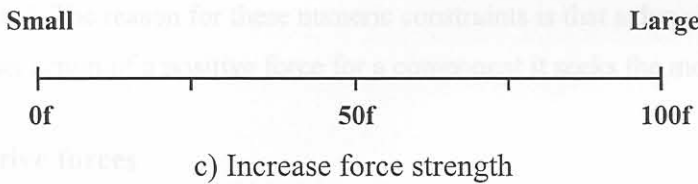
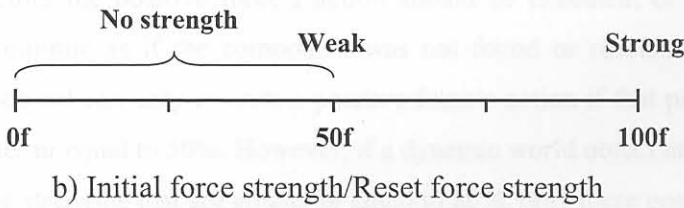
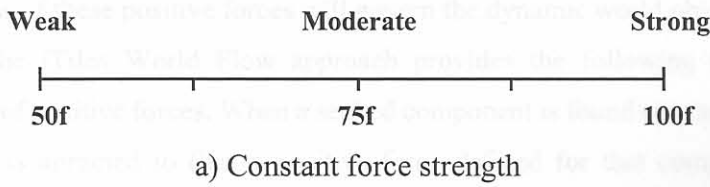


Figure 19. Constant and incremental positive forces' metric unit features explained as fuzzy logic
 (The unit 'wb' refers to world beat and 'f' refers to force strength)

A positive force's incremental force-seeking type can best be explained by using a metaphor of hunger. Consider a person (a dynamic world object) that is a little hungry (initial force strength). Some time passes (increase time) and he/she gets a little hungrier (increase percentage unit). Now in order for the person to satisfy their hunger they start looking for food (the sought static world object). Eventually the person finds some tasty food in the kitchen and starts eating, which takes some time, as he/she is really hungry (found action type of stay in location for time period). After this time has passed, the person is no longer hungry (reset force strength), the food has disappeared and the person has a full stomach (This

positive force's action could specify that when executed the food should disappear by triggering the food static world object's world object transformation of disappearing, and the person character should grow larger by triggering the person character's world object transformation of scaling up). Such a cycle can be continuously repeated for characters in the simulation.

Forceful Components

A dynamic world object can be attracted to various components in an environment, and its level of attraction or force strength to those components can vary during the simulation. The force strengths of these positive forces will govern the dynamic world object's movement and behaviour. The iTiles World Flow approach provides the following mechanism for the prioritisation of positive forces. When a sought component is found or reached that a dynamic world object is attracted to (has a positive force defined for that component), it must be determined whether the positive force's action should be executed, or the dynamic world object should continue as if the component was not found or reached at all. Therefore a dynamic world object can only execute a positive force's action if that positive force's force strength is greater or equal to 50%. However, if a dynamic world object has any other positive forces with force strengths that are greater or equal to 80%, only these positive forces' actions may be executed. The reason for these numeric constraints is that a dynamic world object will only execute an action of a positive force for a component it seeks the most.

3.5.3 Negative forces

A negative force indicates what component a dynamic world object is afraid of or repelled by. Figure 20 presents the composition of a negative force of a dynamic world object. A dynamic world object can have a negative force towards a static world object, a dynamic world object, or a tile element type. A negative force translates to a positive force strength of 100% in the opposite direction to that of where the component is located.

Consider two dynamic world objects: $character_1$ and $character_2$. If $character_1$ is afraid of $character_2$, then a sound clip (called *caught sound*) may be specified that is played when $character_2$ finds or reaches $character_1$. For example, if $character_1$ were an elephant and $character_2$ were a mouse, the *caught sound* could be specified as an elephant trumpeting when the mouse reaches the elephant. A *sighted sound* may also be specified for a negative force, which is played when a character sees the feared component directly in front of it.

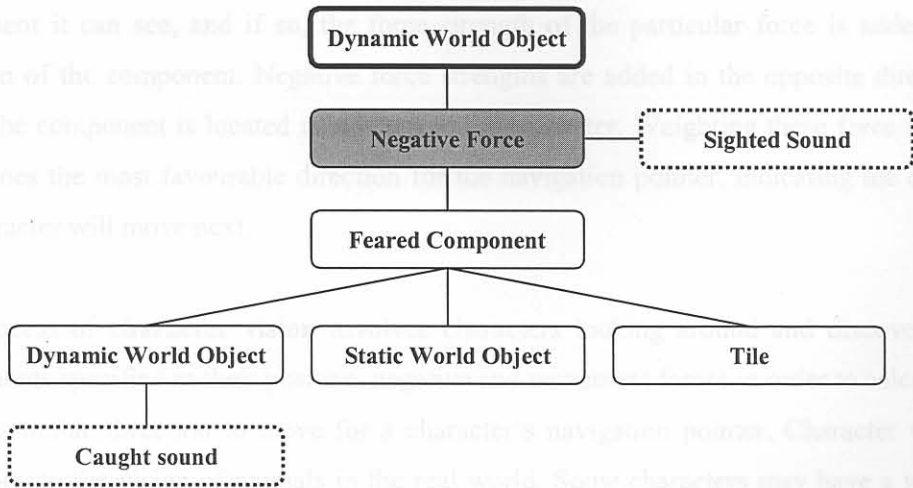


Figure 20. A dynamic world object's negative force

3.6 Movement of characters in the simulation

Characters' movement in an iTiles world can be described as 'robot-like' since characters can either change the direction they are facing (in 45° angles) or move to an adjacent tile if permitted by their tile attribute list, in the direction they are facing. Characters navigate in the world in terms of compass navigation. The 45° angles that a character can be facing relate to the compass directions north, south, east, west, northeast, southeast, southwest and northwest. In performing a check whether the tile that a character wishes to move to is free (i.e. there is no world object on top of it) collision detection mechanisms are put in place. Characters should also be intelligent in that they know the location of the edge of the world.

3.6.1 Character vision and navigation

In the simulation of an iTiles world, the characters in the world 'come to life' through their movements. The world forces described in Section 3.5 govern the movement of characters. Positive and negative forces are used for calculating a favourable direction of movement for characters. These forces of characters need to be taken into account at every world beat as the force strength of a character's positive force can change. A character's behaviour is also dependent on the position it is located in the world and the world objects and tiles it can see. For calculating the most favourable direction to move, characters have 'sight' in terms of **character vision** and a **navigation pointer** to guide them in the right direction.

A character's **navigation pointer** is used in order to determine the best favourable direction to move. This navigation pointer is calculated when a character looks around and determines it's relation to the tiles and the world objects on those tiles that it can see. During the process of character vision a character determines if it has a positive or negative force towards the

component it can see, and if so, the force strength of the particular force is added in the direction of the component. Negative force strengths are added in the opposite direction to which the component is located in relation to the character. Weighting these force strengths determines the most favourable direction for the navigation pointer, indicating the direction the character will move next.

The process of **character vision** involves characters looking around and discovering the components specified in their positive, negative and movement forces in order to calculate the best favourable direction to move for a character's navigation pointer. Character vision is analogous to the vision of animals in the real world. Some characters may have a wide and short depth of vision such as a herbivore, or a far and long depth of vision such as a carnivore. The **vision type** of a character specifies how a character sees their world. It takes into account the direction the character is facing and other directions in relation to the direction the character is facing, in order to identify what tiles the character can see. **Vision depth** is a specification on how far a character can see. This is the amount of tiles a character can see in a particular direction defined by its vision type and is expressed in terms of tile units. Vision types such as carnivore vision and herbivore vision can be defined. For example, Figure 21 presents a character facing in the north direction, with a herbivore vision type and a vision depth of two tile units. A character's vision type does not necessarily depict the ecological behaviour of that character, but merely the way the character see an iTiles world.

Character vision takes into account both the vision type and the distance a character can see. However when calculating what tiles a character can see it needs to be determined whether a tile is located either diagonally, horizontally or vertically in relation to the position of the character, as the distance covered from the character to the tile may not entirely satisfy the vision depth specified. This is because the distance from a tile's edge to the same parallel edge is shorter than the distance covered between two opposite corners. The iTiles approach uses a Pythagoras calculation to check whether a character can see a tile that is diagonally located in relation to a character's position. This is illustrated in Figure 21, where the tiles that are diagonally located in relation to the character have been marked as not visible by that character.

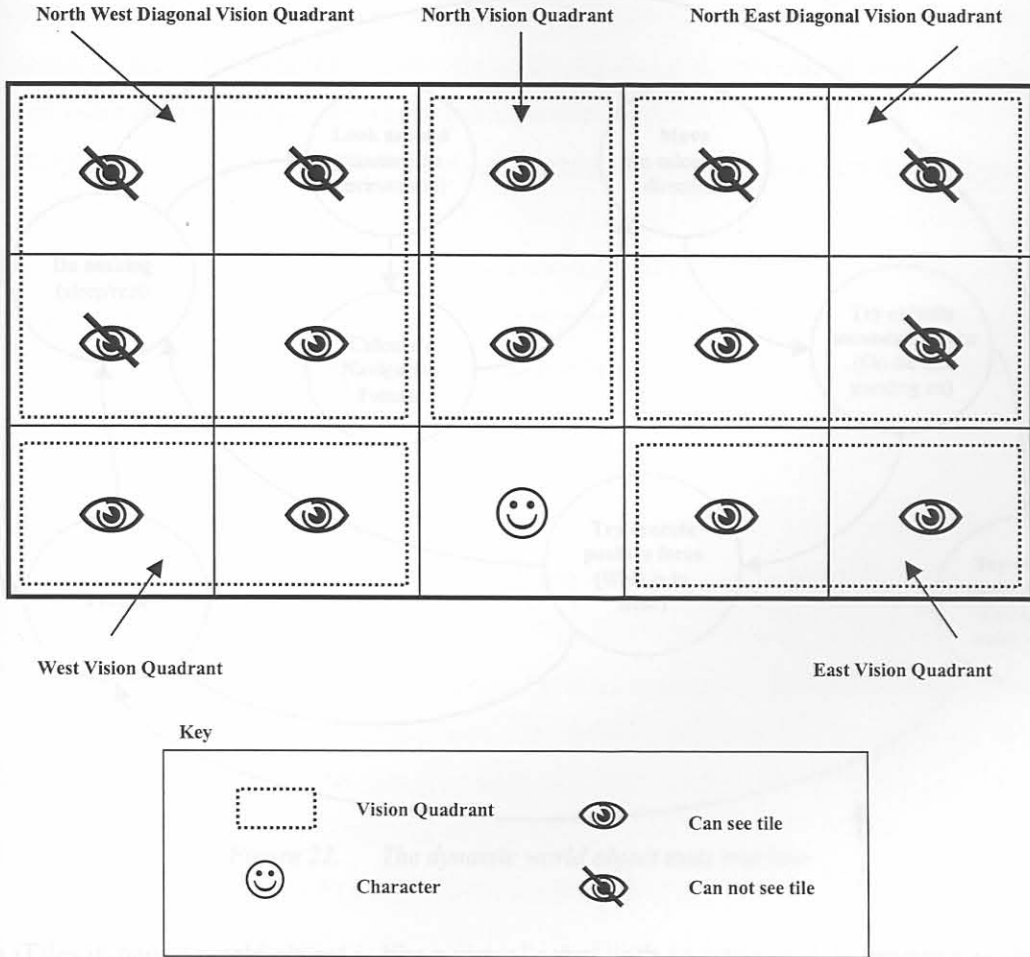


Figure 21. A character facing north, with herbivore vision type with vision depth 2

3.6.2 The dynamic world object state machine

A state machine is described as any device that stores the status of something at a given time and can operate on input to change the status and/or cause an action or output to take place for any given change. In summary a state machine has an initial state or record of something stored, a set of possible input events, a set of new states that may result from the input and lastly a set of possible actions or output events that result from a new state. [3, 49]

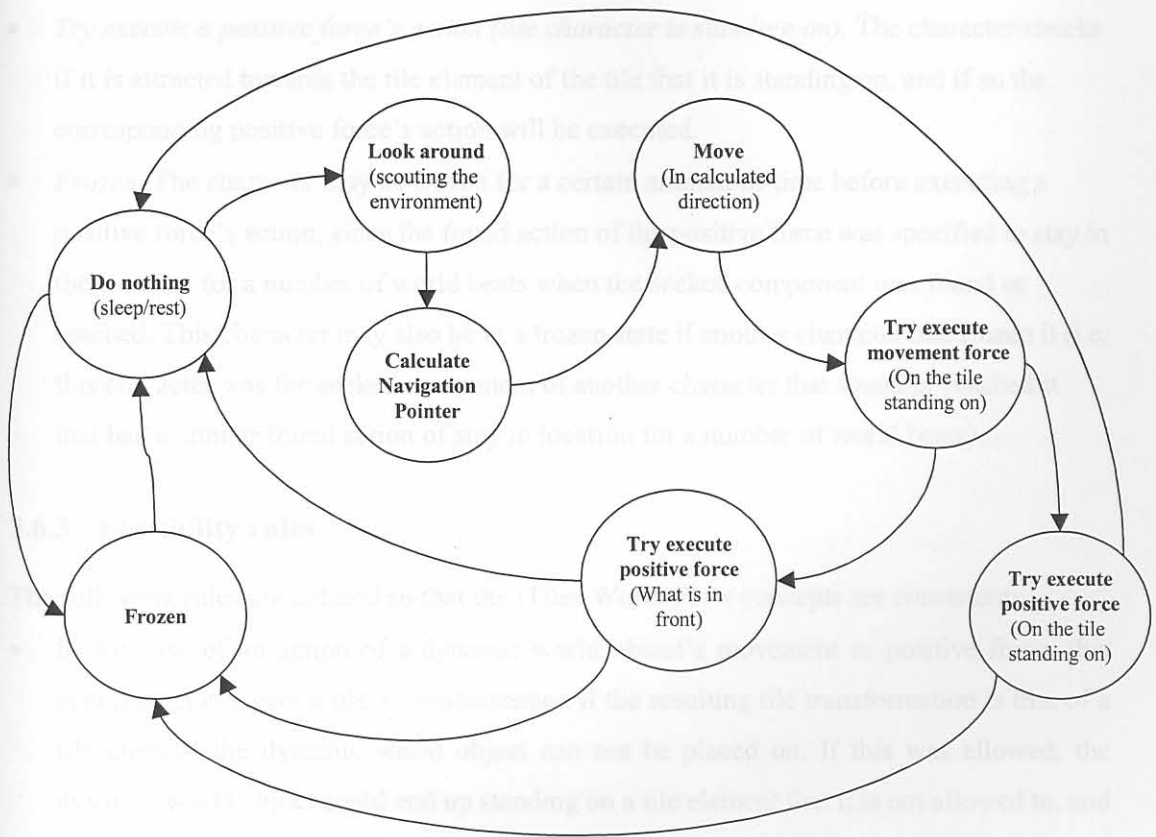


Figure 22. The dynamic world object state machine

An iTiles dynamic world object is like a virtual robot with an internal state machine as shown in Figure 22. In the simulation of an iTiles world, a dynamic world object can be in either of the following states:

- *Do nothing.* The character is paused for the remainder of the world beat.
- *Look around.* The character scouts the environment and determines what it can see in the environment in terms of the character's vision type and depth.
- *Calculate navigation pointer.* The character determines its relationship with the components it can see and determines the most favourable direction to move for the navigation pointer.
- *Move.* The character moves from tile to tile, or changes the direction it is currently facing as indicated by the navigation pointer.
- *Try execute a movement force's action.* The character checks if it has a movement force for the tile element type of tile it is standing on, and if so, the movement force's action will be executed.
- *Try execute a positive force's action (tile in front of character).* The character checks if it is attracted towards a component in front of it (either the tile or the world object that is standing on that tile), and if so, the corresponding positive force's action will be executed.

- *Try execute a positive force's action (tile character is standing on).* The character checks if it is attracted towards the tile element of the tile that it is standing on, and if so the corresponding positive force's action will be executed.
- *Frozen.* The character may be frozen for a certain amount of time before executing a positive force's action, since the found action of the positive force was specified to stay in the location for a number of world beats when the sought component was found or reached. This character may also be in a frozen state if another character had frozen it (i.e. this character was the sought component of another character that found or reached it that had a similar found action of stay in location for a number of world beats).

3.6.3 Feasibility rules

The following rules are defined so that the iTiles World Flow concepts are consistent:

- In the case of an action of a dynamic world object's movement or positive force, this action cannot trigger a tile's transformation if the resulting tile transformation is that of a tile element the dynamic world object can not be placed on. If this was allowed, the dynamic world object could end up standing on a tile element that it is not allowed to, and this would invalidate the rules of an iTiles world.
- A dynamic world object cannot freeze another dynamic world object that is already frozen (i.e. if the sought component of a dynamic world object's positive force is a dynamic world object, the sought dynamic world object will only be considered found for the first dynamic world object that finds it). If this were allowed situations of deadlock could arise, since a component may be sought by more than one character.
- A dynamic world object cannot have a negative force towards a tile element appearing in its tile attribute list.
- A dynamic world object can either be attracted to a specific component or repelled by it. In other words either a positive force or a negative force can be specified for a certain component.
- Movement forces for dynamic world objects can only be specified for tile elements in the dynamic world object's attribute list.

3.7 Learning with an iTiles virtual laboratory

The simulation of an iTiles world is presented to young learners in computer-aided education. The simulation is the fun part of an iTiles virtual laboratory, experienced by young learners to enjoy. The iTiles framework provides several ways of experiencing the simulation, ranging from "seeing through a character's eyes" to taking control of a character. In this section advantages of using the iTiles World Flow concepts are discussed, presenting examples of the

use of such concepts for specifying ecological relationships. Educational advantages of the iTiles framework are also presented.

3.7.1 Learning through interaction

Interactivity during the simulation forms the basis of the pedagogy and the learning approach of the iTiles framework. A user can learn by doing and learn by experiencing the simulation of an iTiles world. The iTiles framework allows the user (learner) to select different world objects inhabiting an iTiles world. When a world object is selected the camera is focussed on that selected world object. If the selected world object is a dynamic world object, selection of that dynamic world object results in camera tracking and the learner can observe the character's behaviour and "follow" the character as it moves through the world. An added advantage of selecting a dynamic world object is that a learner may take control of that dynamic world object, and is given the ability to control the movement of that character in the simulation.

The 3D virtual environment can be rotated, zoomed in and out and the learner can experience the same ecosystem from different viewpoints in terms of perspective. Virtual reality techniques can also be used to present a first or third person view from and of the selected character. With a third person view, learners will be able to follow the movements of a character from a distance, whether they are currently observing passively or taking control of the character. In first person view, a learner can experience the world from the world object's point of view, as if looking through the character's eyes, viewing the world in the direction the world object is facing. The iTiles approach includes a mode in which only the tiles visible by the selected character can be displayed in the simulation. This approach is called **inner vision** and takes into account the vision type and depth of the selected character. When this inner vision mode is activated only the tiles and world objects visible to the selected character are displayed, as illustrated in Figure 21.

The novel perspective of oneself experience and shaping a natural phenomenon, instead of acting as a passive observer, is intrinsically motivating [33]. By taking control of a character and experiencing the different perspectives of an iTiles world, this is realised. A third person view of a world object can be considered a perspective of the virtual world from an exocentric frame of reference, and the first person view a perspective of the virtual world from an egocentric frame of reference. By enabling learners to explore the world from different perspectives, the bicentric frame of reference [34] adopted here helps learners increase knowledge and gain more information of the virtual world (See Section 2.4 –Virtual Reality).

In [35] an approach called **virtual identity** is presented. A virtual identity is defined by knowledge about itself, its perception of the environment, its virtual embodiment and the physical simulation and kinematics of the virtual embodiment. Virtual identities and their virtual existence relate to their interaction with the environment, their reaction to the environment, their cognition of the environment and their emotional reactions. Taking on the role of, or selecting a world object in an iTiles simulation, can be expressed as a learner taking on a virtual identity. The first person view and character vision concepts further exemplify the concept of taking on a virtual identity.

3.7.2 Learning with world transformations

World transformations are visual or auditory changes that can occur in the simulation of an iTiles world, and are associated with the tile and world object components. World transformations of the gradual type can represent changes that require more time or exposure for the change to occur. This can often happen in nature, since for example, the growth of animal takes a considerable amount of time.

The concept of tile transformations presents a method to change to the terrain of an environment during a simulation. Depending on the tile element type, altering this tile element type with a tile transformation can represent environmental occurrences. For example, transforming grass tiles to sand tiles can depict erosion and overgrazing, and transforming water tiles to sand tiles can depict evaporation and depletion of water in times of drought. Other concepts related to animal growth, death and the depletion of resources (e.g. plants), can be denoted by the different types of world object transformations:

- A world objects growth can be represented by scaling a world object up.
- The depletion of a resource can be represented by scaling a world object down. For examples, plants could get smaller as they are eaten.
- A world object's death can be represented by a world object disappearing.

3.7.3 Learning with world forces

World forces represent the relationships of characters to one another and to other components of the environment. As in real life, forces of attraction and repulsion are present, which are similarly depicted by the iTiles World Flow concept of world forces. This enables for natural ecosystem relationships and behaviour to be portrayed from the relationships defined within world forces.

Different types of ecosystem relationships and behaviour can be described with the specified sought component of a character's positive force:

- *Static world object.* A herbivore character can have a positive force towards a static world object such as a plant. When the plant is found by that character, the character's positive force's action can be set to trigger the plant's world object transformation to scale the plant down. The positive force's action can also be set to trigger the character's world object transformation to scale the character up. Triggering these world object transformations to occur can depict the characters growth and the plant's depletion.
- *Tile element.* Herbivore characters such as sheep or cows can, for example, have a positive force towards a grass tile element. When the sheep or cow characters find grass tile elements their behaviour can be depicted as grazing. Thirst can be represented by a character's positive force towards a water tile element. When the character finds the water tile element, the behaviour can be depicted as drinking.
- *Dynamic world object.* A predator-prey relationship or a mutual relationship can be defined by a character's positive force towards another character. The transformation triggers that are specified in the character's positive force's action can depict the type of relationship defined. For example, a mutual relationship can be depicted if no transformation triggers are specified for the action, which could illustrate, for example, the characters' need to converse with each other. However, if world transformations of those characters were triggered in the character's positive force's action (consider for example that the sought character's world object transformation to disappear was triggered), this could illustrate one character's desire to eat the other character if it were a carnivore.

3.7.4 Educational advantages of the Tiles framework

A character's positive force need not represent an ecological relationship. For example, a character may have an attraction towards a static world object such as a tree, however the relationship is neutral in that no transformations occur when the character finds the tree. In this case a bird chirping sound clip is specified as the character's positive force's found sound. This sound clip is played when the character finds the tree, which makes for a more enjoyable simulation.

History produced using the Tiles framework provides teachers a useful aid in teaching. It provides a teacher with the following:

Negative forces, which represent repulsion of characters with other components of the environment, can also help learners in understanding ecosystem relationships. For, example, by being afraid of a certain character, that scared character could represent the prey in a predator-prey ecosystem relationship. Also, a character's inability to swim can be depicted by being afraid of a tile element such as water.

The sound clips specified for positive forces and negative forces can also help establish ecosystem relationships and behaviour. For example, consider a leopard and baboon character in a predator-prey relationship. The leopard character has a positive force towards a baboon character, and the baboon character has a negative force towards the leopard character. The leopard's positive force's sighted sound is specified as a growl sound clip. The baboon's negative force's sighted sound is specified as a danger/warning cry sound clip. Therefore when the leopard sees the baboon it make a growling sound, and when the baboon sees the leopard it will alert the baboon troop of danger by making a warning cry sound.

Movement forces represent the relationship of the movement of characters on tiles. Environmental effects such as erosion can be represented by a character's movement force's action triggering transformations of the tile elements the character is moving on. With a gradual tile transformation, a certain character may influence the tile transformation's meter more than another. This can depict the weight of characters, as heavier characters would cause more erosion (i.e. weight could be directly proportional to the meter alteration percentage specified).

Many more ecosystem relationships are possible to be created using the iTiles World Flow concepts of world transformations and world forces. It is the task and imagination of the author of an iTiles world to provide these relationships. Education advantages of the iTiles framework are presented next.

3.7.4 Educational advantages of the iTiles framework

As exemplified in the previous sections an iTiles virtual laboratory can be used in teaching ecological biological processes involved in the field of earth science. In this section teaching with an iTiles virtual laboratory and the pedagogies of an iTiles virtual laboratory are discussed.

An iTiles virtual laboratory produced using the iTiles framework provides teachers a tool to aid in teaching. It provides a teacher with the following:

- Ability to present lessons on various topics in the field of earth science.
- Control of lesson contents through authoring an iTiles world.
 - Creativity in an iTiles world creation/lesson contents.

- In specifying an iTiles World Flow for an iTiles world, the iTiles World Flow concepts can be used to teach ecological relationships and behaviour in the simulation of that iTiles world.

Interactive simulations offer a fun and effective way to enable students to learn by doing and learn by experiencing. By using computer-based simulations, we can broaden the range of things students can learn. The iTiles framework has addressed what young learners can do with, and what they learn from a simulation of an iTiles world. Young learners can improve and learn fundamental cognitive skills such as mathematics and communication by:

- *Observation.* Learners can deduce the rules of an environment through observation of the simulation. They experience and observe the behaviour exhibited by characters through these characters' actions, movements, transformations and sounds they make.
- *Understanding.* Children learn to understand the interactions and relationships of characters within a world.
- *Exploration.* The simulation presents an opportunity for learners to explore. Children of different abilities can explore an iTiles world through the interaction mechanisms provided.
- *Communication.* A similar goal is likely to stimulate interaction between learners. This interaction improves communication skills as children can learn about an ecosystem together.
- *Authoring a virtual environment.* More advanced students could author their own worlds.

Pedagogy can also be embedded when presenting an iTiles world simulation to students. Elementary school science is about asking questions, collecting data that bear on those questions, and building support for answers [30].

Students can learn the following early skills in mathematics:

- *Matching.* Understanding the inhabitants of a world, and relating them to world objects of a similar type.
- *Counting.* Students could be given data collection tasks. For example, they could count the number of world inhabitants of a certain type.
- *Classification.* Students could be given the task of classifying different types of world objects according to their ecological relationships.
- *Ordering.*
- *Making patterns.*

3.8 Summary

In this chapter the Intelligent Tiles framework was introduced and the aims in terms of addressing the major problems of existing virtual laboratories were presented. The composition of a virtual environment produced with the iTiles framework was then discussed, followed by how one could simulate behaviour in such a world. How students can learn through the simulation of an iTiles world was presented, followed by the educational advantages of the iTiles framework. The next chapter presents an implementation of the iTiles framework using the concepts described in this chapter.

"The worlds into which our steps through virtual reality are limited are by the imagination of the programmer"
D. Powers, M. Durrant [3]

Implementation

In this chapter the implementation of the Intelligent Tiles Virtual Laboratory framework is presented. Firstly an overview of the implementation is highlighted, covering subjects such as the programming languages used for implementation and user interface design. Next system implementation design concepts are presented. Lastly an overview of the implemented system is presented with screenshots.

4.1 Overview of implementation

The iTiles Ecosystem Virtual Laboratory is a virtual reality and critical experiment tool using the iTiles framework and concepts presented in Chapter 3. The iTiles system is composed of three connected and dependent tools: the iTiles Workbench tool which is used for the authoring of an iTiles world, the iTiles World Flow tool which is used for specifying the behaviour of an iTiles world, and the iTiles Virtual World tool that presents a simulation of the authored iTiles world. These tools share system specific iTiles concepts and are used as independent tools in the process of authoring and simulation. Each iTiles world developed using the iTiles Ecosystem Virtual Laboratory can be considered an independent virtual laboratory.

The implementation of the iTiles framework has been accomplished using open source libraries. Open source initiatives support portability and applications developed using open source libraries are most likely to be able to be ported to other operating system environments.

Chapter 4

“The worlds into which one steps through virtual reality are limited only by the imagination of the programmer”

- D. Powers, M. Darrow [5]

Implementation

In this chapter the implementation of the Intelligent Tiles Virtual Laboratory framework is presented. Firstly an overview of the implementation is highlighted, covering subjects such as the programming languages used for implementation and user interface design. Next system implementation design concepts are presented. Lastly an overview of the implemented system is presented with screenshots.

4.1 Overview of implementation

The **iTiles Ecosystem Virtual Laboratory** is a virtual reality application implemented using the iTiles framework and concepts presented in Chapter 3. This iTiles system is composed of three connected and dependent tools: the **iTiles Workbench** tool which is used for the authoring of an iTiles world, the **iTiles World Flow** tool which is used for specifying the behaviour of an iTiles world, and the **iTiles Virtual World** tool that presents a simulation of the authored iTiles world. These tools share system specific iTiles concepts and are used as independent tools in the process of authoring and simulation. Each iTiles world developed using the iTiles Ecosystem Virtual Laboratory can be considered an independent virtual laboratory.

The implementation of the iTiles framework has been accomplished using open source libraries. Open source initiatives support portability and applications developed using open source libraries are most likely to be able to be ported to other operating system environments

(e.g. Macintosh, Linux). The iTiles Ecosystem Virtual Laboratory application has been implemented on a Microsoft Windows environment. The object oriented programming language used in implementation is C++. The standard C++ header libraries are used in conjunction with the Standard Template Library (STL) in implementation. Section 4.1.3 describes the programming approach used in more detail.

4.1.1 Application programming interfaces (APIs)

The Open Graphics Library (OpenGL) and Open Audio Library (OpenAL) APIs are the open source libraries used in implementation of the iTiles framework. The key concepts of these libraries are presented in this section.

OpenGL is a widely accepted API for interactive 3D graphics rendering and 2D imaging. It provides device-independent support for common low-level 3D graphics drawing operations such as polygon specification, basic lighting control, transformation specification, and framebuffer operations like blending and depth-buffering. It also provides mechanisms for sending and retrieving 2D images to and from the framebuffer, and integrates 3D graphics with 2D imaging through texture mapping. While other low-level graphics APIs have provided similar functionality to that of OpenGL, OpenGL takes a novel approach in the presentation of many of its features. In addition to providing a simple model for combining 3D graphics with 2D imaging, OpenGL makes a clear separation between high- and low-level functionality, stresses fine-grained control and feature orthogonality, and is designed for excellent performance from basic PCs to high-end graphics workstations. OpenGL also specifies a fixed rendering pipeline that both provides a model for implementations and a clear foundation for adding new functionality. [42]

The OpenGL Utility Toolkit (GLUT) is a window-system-independent toolkit, written by Mark Kilgard, to hide the complexities of different windowing systems [41]. The GLUT API simplifies the implementation of programs using OpenGL rendering and requires very few routines to display a graphics scene. The GLUT API, like the OpenGL API, is stateful, and the initial state is reasonable for simple programs. [43]

OpenAL is a joint effort to create an open, vendor-neutral, cross-platform API for interactive, primarily spatialized audio [45]. The Open Audio Library provides a cross-platform API that is as useful and powerful for 3D audio as OpenGL is for 3D graphics [44]. OpenAL supports spatial audio where sounds can be played in a way as to reflect the position of objects in the

virtual environment. OpenAL, similarly to OpenGL, has an abstracted interface through a utility toolkit called ALUT.

4.1.2 Hardware and software used in implementation

The iTiles Ecosystem Virtual Laboratory application is aimed for use on a desktop VR solution. The application has been implemented on a Microsoft Windows 2000 environment on a Pentium III 733Mhz processor with a 3D accelerator, the ASUS GeForce 256 GPU with an nVidia chipset.

Various software tools have been used in implementation:

- *Graphic application.* For texture creation of the tile set and the application's 2D interface (Corel Photopaint 9.0).
- *Integrated Development Environment (IDE).* A programming interface to program and compile C++ code (Microsoft Visual Studio 6.0).
- *3D modelling.* A shareware tool has been used for the modelling and converting of 3D models for use as world objects (Milkshape 3D, available from chUmbaLum sOfT at <http://www.swissquake.ch/chumbalum-soft/>).
- *Sound editing.* For sound clips (Creative WaveStudio).

4.1.3 Programming approach

An object-oriented approach has been used in the implementation of the iTiles framework concepts discussed in Chapter 3. Objects are used to represent these concepts, by taking advantage of inheritance and polymorphism object-oriented concepts. The class diagram of the iTiles framework concepts is presented in Appendix A.

A singleton class is an object-oriented design pattern. An implemented class exists as one and only one instance in memory at run time. Using such a global object ensures that the instance is easily accessible and provides a global point of access [50]. In the implementation of the iTiles system the use of singleton objects has provided simplicity in the GUI application and system functions.

The vector and map standard template libraries (STLs) have been used in implementation, as these STL collections can be defined to store a certain type of object, and can grow and shrink in size. The collections STLs were used to handle some of the difficult memory management problems that arise when implementing in C++. The STL string class was very useful in storing collections of characters.

4.1.4 User Interface Design

Computing exists now as part of many peoples' cognitive strategies, helping create and develop new ideas. We test, we model, we try out, and we interactively simulate our conceptions through use of computers [10]. The art world has discovered the potential of virtual reality for visual innovation [5]. Virtual worlds have become more and more visually elaborate and emotive. Some even aim at an environment nearly indistinguishable from the real world. Transferring natural interaction and communication principles from the real world to cyberspace in a seamless fashion is a very challenging task. Here, an attempt has been made to implement a fun and friendly user interface by adhering to user-centred design principles. Research on human factors in human-computer interaction shows that users of a new interface demonstrate a consistent desire to "get started right away" [46].

Easy-to-use products do not just happen [47]. To create productive and enjoyable user experiences an approach such as user-centred design must be adopted. In [46], basic fundamentals of such user-centred design principles include:

- *Affinity*. Bringing objects to life through good visual design
- *Assistance*. Assist the user in performing a variety of tasks, through hints or system help
- *Availability*. Allow the user to use all objects within a view in any sequence at any time
- *Encouragement*. Make actions predictable by ensuring that an action produces the expected results
- *Familiarity*: Build on the user's prior knowledge of the system. A user-friendly system enables the user to learn new concepts and techniques from accomplishing one task and apply them to a broad spectrum of tasks
- *Obviousness*. Make objects and controls visible and intuitive. Use visual or textual cues to help users understand functions.
- *Safety*. Protecting the user from making errors, where the burden of keeping the user out of trouble rests upon the designer. Report results of actions immediately
- *Satisfaction*. Create a feeling of progress and achievement. Allow the user to make uninterrupted progress and enjoy a sense of accomplishment. Report the results of actions immediately. Communicate to the user in the event that a function cannot be performed
- *Simplicity*. Do not compromise usability for functionality. Keep the interface simple and straightforward. Minimise the number of objects and actions in an interface
- *Support*. Place the user in control.
- *Versatility*. Support alternative interaction techniques. Allow the user to switch between methods to accomplish a single interaction

The above fundamentals provide guidelines for implementing systems for different users with different abilities. They also play a vital role in user-centered design for young learners as users. However added simplicity and pictorial representations form a superior supporting user-interface.

4.2 Implementing the iTiles Ecosystem Virtual Laboratory

In this section major implementation specifics of the iTiles Ecosystem Virtual Laboratory are highlighted. Some of the implementation classes are mentioned in this section: see Appendix A for better understanding their relationships with other classes.

4.2.1 iTiles system management

The *iTilesSystemManager* singleton class is responsible for all system specific functions for specifying which iTiles system tool is currently active, whether a new or existing iTiles world is being authored, and for loading the iTiles system interfaces.

4.2.2 Abstracting GLUT function calls

The classes implementing the different tools of the iTiles system are:

- A *display class* extending the interface of the *iTilesDisplay* class representing the visual display of the tool (e.g. *iTilesWorkBenchDisplay*); and
- A *logic class* for the ‘behind the scenes’ logic algorithms and the functionality of the tool (e.g: *iTilesWorkBench*)

The interface of the *iTilesDisplay* class is:

- virtual void keyboard(unsigned char key, int x, int y);
- virtual void special(int key, int x, int y);
- virtual void display2D();
- virtual void display3D();
- virtual void reshape(int newWidth, int newHeight);
- virtual void motion(int x, int y);
- virtual void mouse(int button, int state, int x, int y);
- virtual void idleFunction();
- virtual void updateFluidMotionValues();
- virtual void updateCameraFrozenPosition();

To simplify the functionality of the display classes of the iTiles system tools, the singleton class *iTilesDisplayManager* is implemented. It abstracts GLUT function calls and GLUT and OpenGL initialisation procedures. The *iTilesDisplayManager* class stores information on which display class of which iTiles system tool is currently active and passes on all GLUT call back functions to this currently active display class through the *iTilesDisplay* interface (as presented above).

The GLUT 'display' call-back function that the *iTilesDisplayManager* class implements extends GLUT functionality by firstly calling the `display3D()` function followed by the `display2D()` function for overlaying 2D on screen. The *iTilesDisplayManager* class is also responsible for abstracting some OpenGL drawing functions such as displaying text on a specific part of the screen and drawing textured rectangles.

4.2.3 2D GUI

Controls are the software elements, usually shown on a display, that you use to set preferences and make choices. These elements are necessary for interaction between people and computers, and are much like such hardware controls as knobs and dials, as they can be used to control many different things [48]. The iTiles system has a unique look and feel in terms of the way it captures user input and control. A custom widget library is written for this purpose. The library has primitive widget components such as buttons, radio buttons and list boxes. These widgets are integrated and used to make input screens.

The OpenGL co-ordinate system starts at the bottom right corner of a window, and the widget components are modified so that co-ordinates are given as if the co-ordinate system started at the top left. Appropriate reshape callback functions are implemented for correct resizing of screens, and information is passed on to the widget components affected by the resizing.

The widget components are divided into:

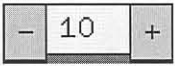
- *Standard components.* E.g. button, radio button, radio button group, list box, image button
- *Custom components.* The standard widget components are re-used and combined to create system specific components (e.g. plus minus control and world sound chooser)

The widget components are currently dependent on (closely linked to) the iTiles system. However some portions of code could be combined to form an independent library of widget components. The widgets are presented in their visual form in Figure 23.

4.2.4 Textures

The `TextureManagerSingleton` class is implemented in `TextureManager`. It manages the 2D and 3D parts of the display system. The windows BMP file format is used and the `ImpFont` class represents such a texture object.

plusMinusControl



imageButton



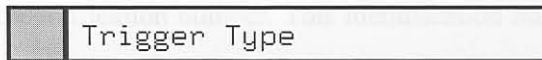
radioButton



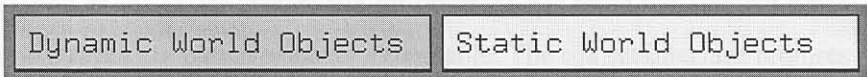
listDisplay



blockHeader



tabButton



worldSoundChooser

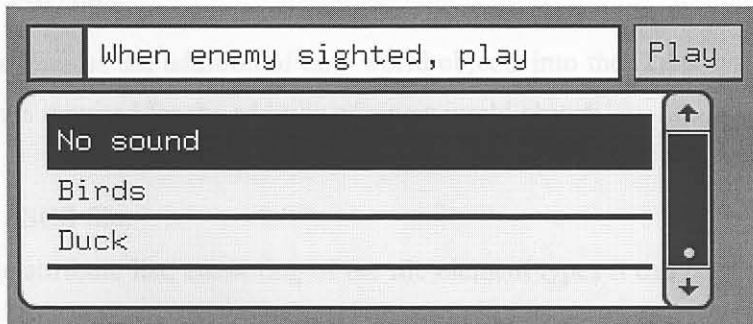


Figure 23. *iTiles widgets classes visual representation*

Overlaying a 2D user interface over parts of a 3D interface combines both interfaces to form a unique desktop VR experience. The 2D user interface consists of widget components that are used to interface with the virtual reality application. This combined interface is similar to an aeroplane cockpit interface, which is used in many flight simulators. This 2D overlay mechanism has been implemented in the *iTiles* Workbench and *iTiles* Virtual World tools presented in Section 4.3.

4.2.4 Textures

The *iTilesTextureManager* singleton class is implemented for texture management. Textures are loaded on demand and cached for future use. These textures are used in both the 2D and 3D parts of the display system. The windows BMP file format is used and the *bmpTexture* class represents such a texture object.

4.2.5 iTiles system interfaces

Interfaces for managing both the world objects and tile elements are implemented for the addition of such components to an iTiles system. The **world object interface** manages world objects, and the **tile set interface** is responsible for the tile elements types. For the various sound clips that may be specified for world forces or world transformations, a **world sound interface** is implemented. The world sound interface manages sound clips for an iTiles system. Each entity (component or sound clip) added to the iTiles system is uniquely identified with a system generated identification number. This identification number is used for internal referencing.

4.2.5.1 Tile set interface

The current implementation of the iTiles system tile set interface has three tile element types in the tile set. Tile element types are however easily extendable. The tile set interface requires for each tile element: a tile element name and a texture file association.

4.2.5.2 World object interface

The world object interface enables the addition of new world objects into the iTiles system. The following information is required for the addition of a new world object:

- World object name.
- MilkShape 3D model ASCII file.
- The world object's tile attribute list, consisting of the tile element types it can be placed on.
- Whether the world object is a static or a dynamic world object.
- Minimum and maximum scale of the object so that it falls in the visual boundary of a tile element.
- If the world object is a dynamic world object, the vision type (eg: herbivore, carnivore) and vision depth (number of tiles the dynamic world object can see).

Milkshape 3D is a shareware application, and has support for popular 3D file formats such as those models used in popular games such as Quake and Half-life. The MilkShape 3D

application is used for converting models in a 3D Studio format to the Milkshape 3D ASCII file format to be imported into the iTiles system. A MilkShape 3D model ASCII file loader is used for displaying a 3D model in an OpenGL 3D scene.

4.2.5.3 World sound interface

The world sound interface requires a sound clip name and sound file (WAV file) to add a world sound to the iTiles System.

The *worldSoundsManager* class, implemented using the OpenAL API and ALUT, has been tasked to load and cache sound clips. Furthermore it registers a sound source with a sound clip, and this sound source is programmed to play the sound clip spatially, at a certain point in 3D space.

4.2.6 Collision detection

A collision detection model is implemented in order for the dynamic world objects moving in an iTiles world simulation not to occupy or overlap the same 3D space as other world objects, or to occupy tiles with tile elements not in their tile attribute list. The collision detection model is simplistic in nature as a world object exists on top of a tile, and a tile may either be empty or a world object may be on top of it. If a dynamic world object wishes to move to another tile, it needs to enquire if the tile it wishes to move to is empty. The collision model also restricts dynamic world objects to move to tile elements in their tile attribute list. Another important objective of the collision detection model is to keep characters in the world, and restrict them to movement only within the world so that a character cannot move off the edge of an iTiles world.

4.2.7 Tile merging

The base terrain of an authored iTiles world is not visually appealing since the tile edges stand out and are rectilinear. This is overcome by adopting a tile merging technique from [40]. Using this technique the tiles that were specified in the base terrain of an authored iTiles world can be merged for the simulation of this iTiles world, to present a visually attractive environment. Figure 24 illustrates the tile merging process. Figure 24a represents an example of a base terrain of an authored iTiles world, consisting of four tiles of different tile element types. In the first step of the process, each tile in the authored iTiles world is divided into four quadrants, as illustrated in Figure 24b. In the following step, two extra rows and two extra columns are added to the iTiles world, as illustrated in Figure 24c. These newly created tile quadrants are then assigned the tile element types from their adjacent dominant elements, as

illustrated in Figure 24d. In the final step, nine new ‘mega tiles’ are assigned, each consisting of a group of four tiles, as illustrated in Figure 24e. The tile elements of these mega tiles are merged. The dotted lines in Figure 24e represent the merging of adjacent tiles in a mega tile. The resultant tiled world, as illustrated in Figure 24e, is somewhat larger than the original (containing 36 tiles overall), however it is more visually appealing since tiles are merged.

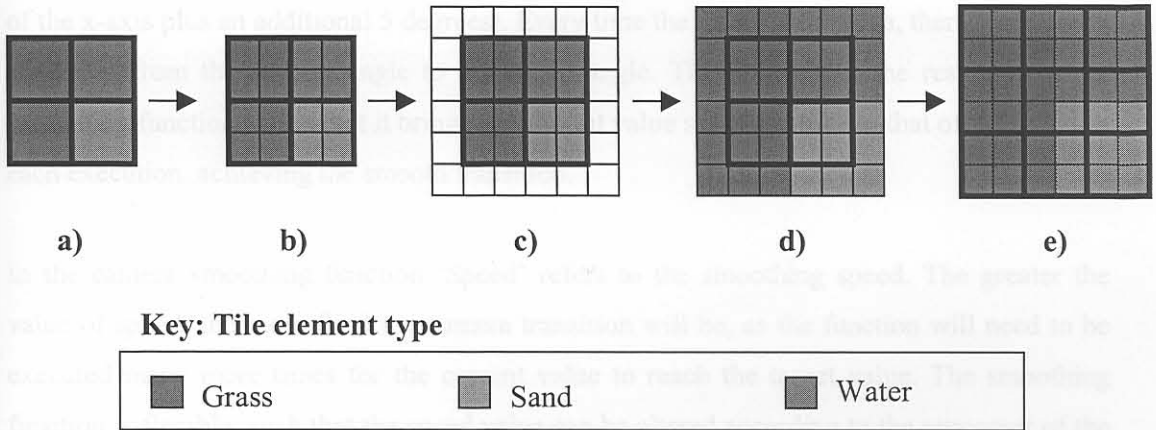


Figure 24. The tile merging process

4.2.8 Camera

The iTiles system provides multiple viewing points of a scene of an iTiles world. Functionality is also provided for zooming in and out of a scene, and rotating or shifting the scene on different axes. Additionally, the iTiles system provides a smooth transition between the different viewing points, including the zooming, rotation or shifting processes. This is achieved by implementing a camera smoothing function that makes use of OpenGL camera techniques. In general, the smoothing function is useful for any angle or position in 3D space, such as an x, y or z co-ordinate or angle.

For each angle or position in 3D space, two storage variables are needed for using the smoothing function:

- *Current*. The current value of the angle or position in space.
- *Target*. The target angle or position in space to which the current angle or position must adjust.

In each GLUT idle call-back function the following function is executed:

$$\text{Current} = \text{Current} + ((\text{Target} - \text{Current})/\text{Speed})$$

Consider for example, that every time a frame of an OpenGL scene is drawn, the scene must firstly rotate to a certain x-axis angle. Now consider a change of view in the scene, where the x-axis is rotated by positive 5 degrees. When using the smoothing function, when such a change of view occurs in a scene the camera does not move directly the new x-axis angle. Instead the camera rotation always references the current x-axis angle variable. When the change of view occurs, the target angle variable for the x-axis is calculated (the current angle of the x-axis plus an additional 5 degrees). Every time the frame is redrawn, there is a smooth transition from the current angle to the target angle. This is because the resultant of the smoothing function call is that it brings the current value slightly closer to that of the target in each execution, achieving the smooth transition.

In the camera smoothing function 'Speed' refers to the smoothing speed. The greater the value of speed, the more fluid the camera transition will be, as the function will need to be executed many more times for the current value to reach the target value. The smoothing function is flexible, such that the speed value can be altered according to the processor of the underlying PC hardware (i.e. a smaller value for a slower processor and larger value for a faster processor).

4.3 Overview of the iTiles Ecosystem Virtual Laboratory

The iTiles Ecosystem Virtual Laboratory is a virtual reality application that is used to author ecosystem type virtual laboratories using the iTiles framework. In this section an overview of the application is presented. The authoring process is firstly presented, starting from how the iTiles system interfaces are populated with tile elements and world objects, followed by how an iTiles world is authored using the iTiles Workbench. The iTiles Workbench tool, for specifying behavioural properties for an iTiles world is presented next. Lastly the iTiles Virtual World tool that presents a simulation of the authored iTiles world is presented. These iTiles system tools are discussed according to their functionality and implementation. Screenshots are also presented.

4.3.1 Populating iTiles system interfaces

How a user populates the iTiles system interfaces will greatly influence the use of the iTiles Ecosystem Virtual Laboratory. Since we are aiming to simulate an ecosystem, animals from the animal kingdom and plants from the plant kingdom are used. To create an ecosystem world, we should be able to represent grass, sand and water areas on the ground, and various trees and animals that would inhabit the world. Using the iTiles concept of world components, the grass, sand and water elements are represented as tile element types, and the

trees and animals are represented as world objects. These tile element types and world objects are added to the iTiles system using the tile set interface and world object interface respectively.

Figure 25 shows the textures used for the creation of the grass, sand and water tile element types. The tile set interface has been populated with these tile element types.

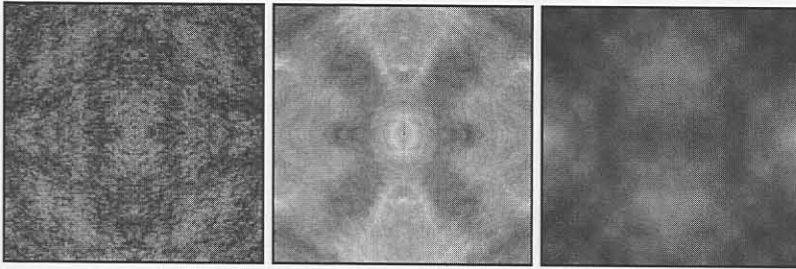


Figure 25. The grass, sand and water tile element textures

A rich variety of trees are modelled and imported as world objects, as shown in Figures 26, 27, 28 and 29. These trees are static world objects, but animals are dynamic world objects. Several animals are modelled and included in the ecosystem, such as a duck, elephant, pig, mouse, turtle and dog (see Figures 30-35). In Table 1 and Table 2 the attributes specified for these world objects (for the world object interface) are presented.

In Table 3, the sound clips that have been added to the iTiles system using the world sound interface are presented. These sounds clips are intended as sounds the animals of the ecosystem may make.

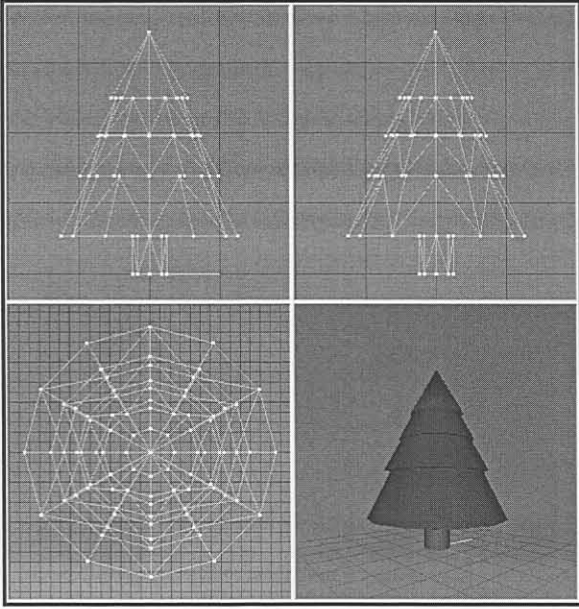


Figure 26. Fir tree

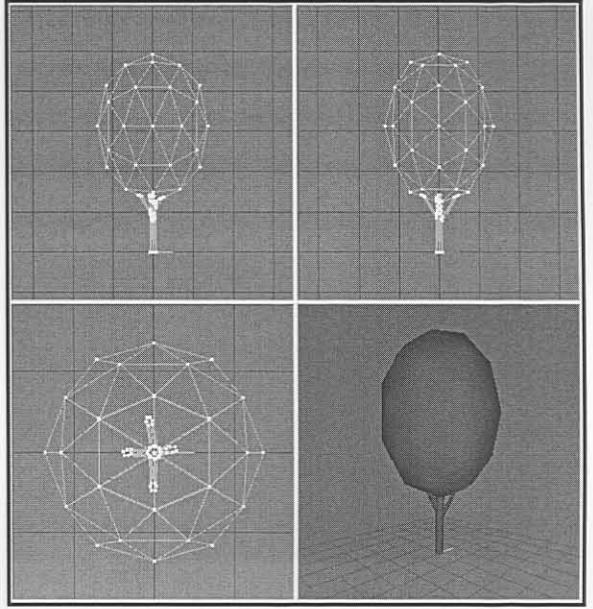


Figure 27. Broad tree

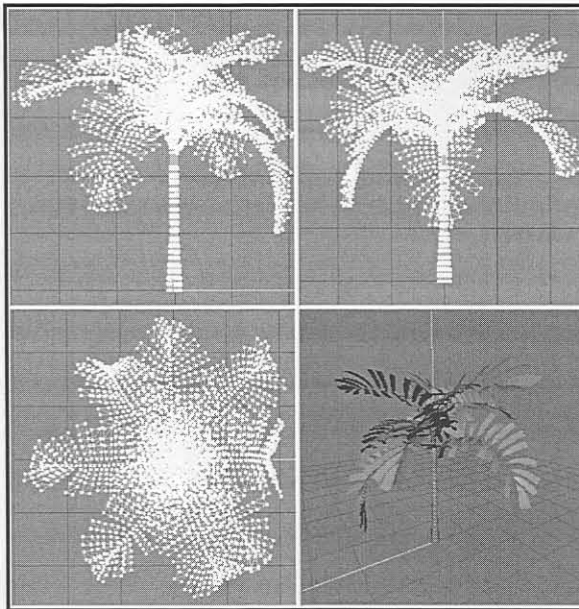


Figure 28. Palm tree

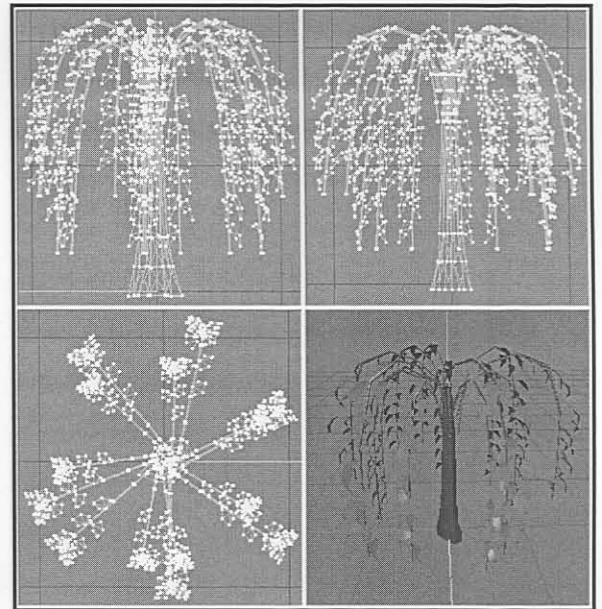


Figure 29. Willow tree

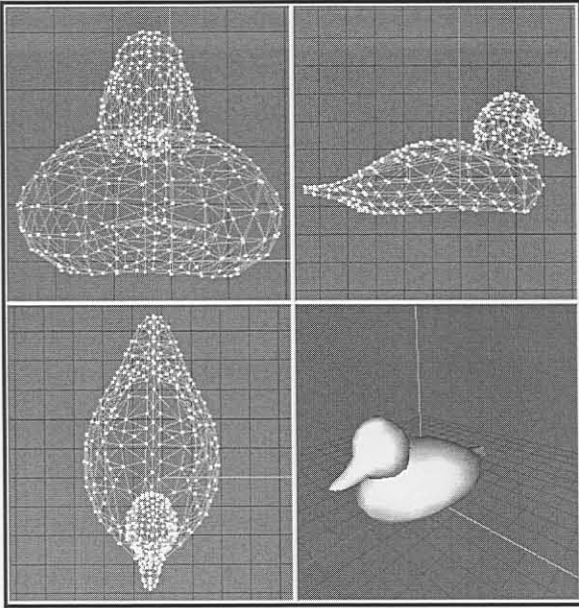


Figure 30. Duck

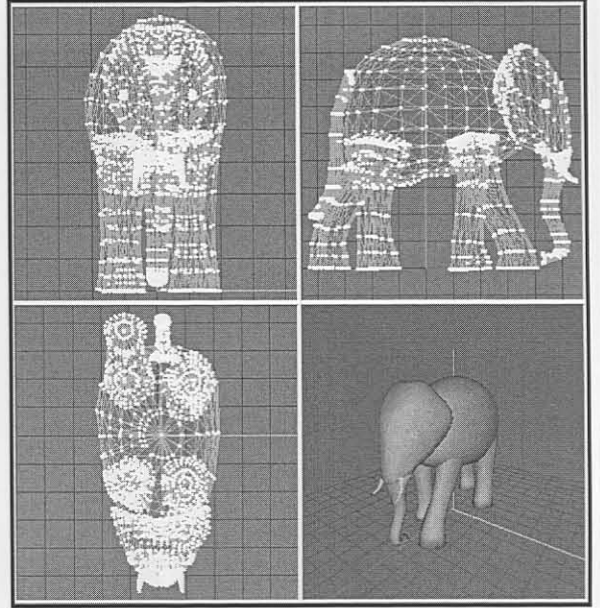


Figure 31. Elephant

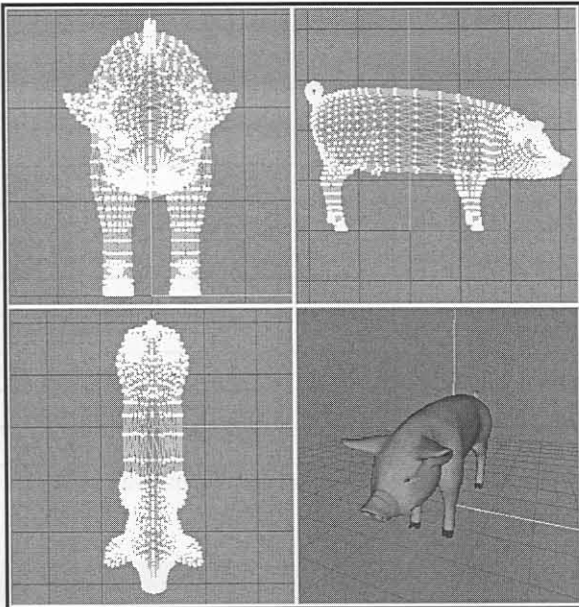


Figure 32. Pig

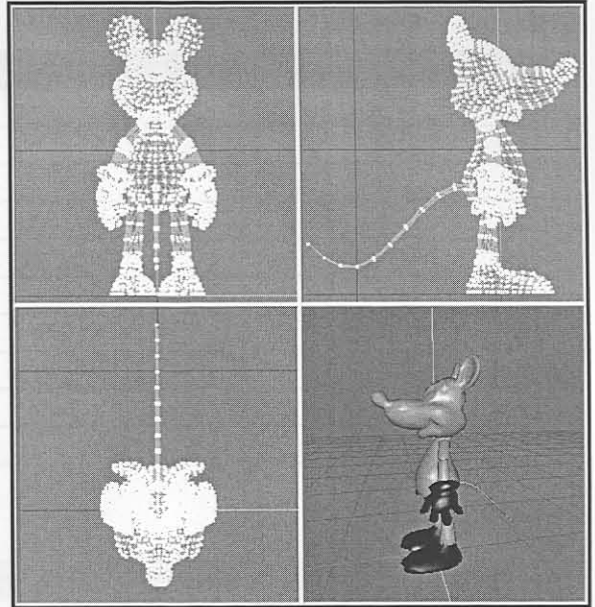


Figure 33. Mouse

Elephant	Herbivore	1
Pig	Herbivore	1
Mouse	Carnivore	1
Turtle	Herbivore	2
Dog	Carnivore	4

Table 2: Dynamic social object robot properties

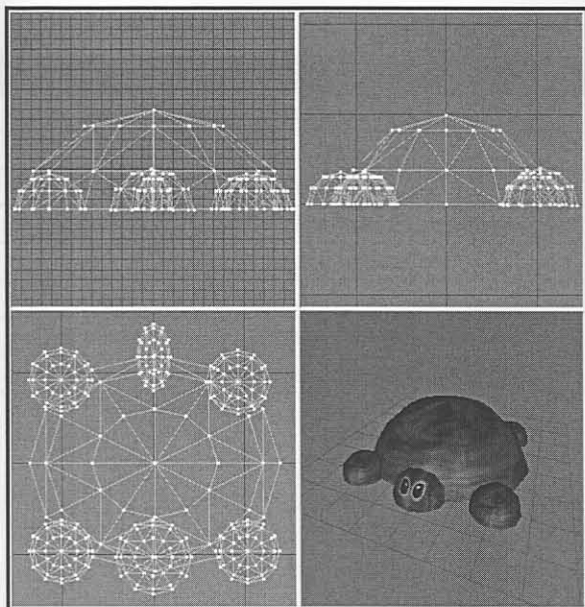


Figure 34. Turtle

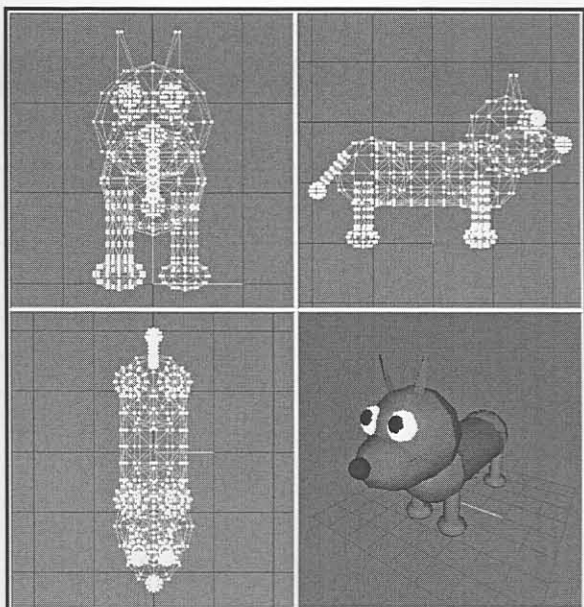


Figure 35. Dog

World Object	Grass	Sand	Water
Duck	×	×	✓
Elephant	✓	✓	×
Pig	✓	✓	×
Mouse	✓	✓	×
Turtle	×	✓	✓
Dog	✓	✓	×
Fir tree	✓	✓	×
Broad tree	✓	✓	×
Palm tree	✓	✓	×
Willow tree	✓	✓	×

Table 1: Tile attribute list of world objects

Dynamic World Object	Vision type	Vision depth
Duck	Herbivore	2
Elephant	Herbivore	4
Pig	Herbivore	3
Mouse	Carnivore	3
Turtle	Herbivore	2
Dog	Carnivore	4

Table 2: Dynamic world object vision properties

to go back to the introductory screen, by selecting the menu option 'choose another world'.

Filename	Description of sound
birds.wav	Birds chirping
dogbark.wav	A dog barking
duck.wav	A duck quacking
elephant.wav	An elephant trumpeting
pig.wav	A pig snorting
water.wav	Splashing of water
magic.wav	A magic sound effect
music.wav	A magical melody

Table 3: World sounds

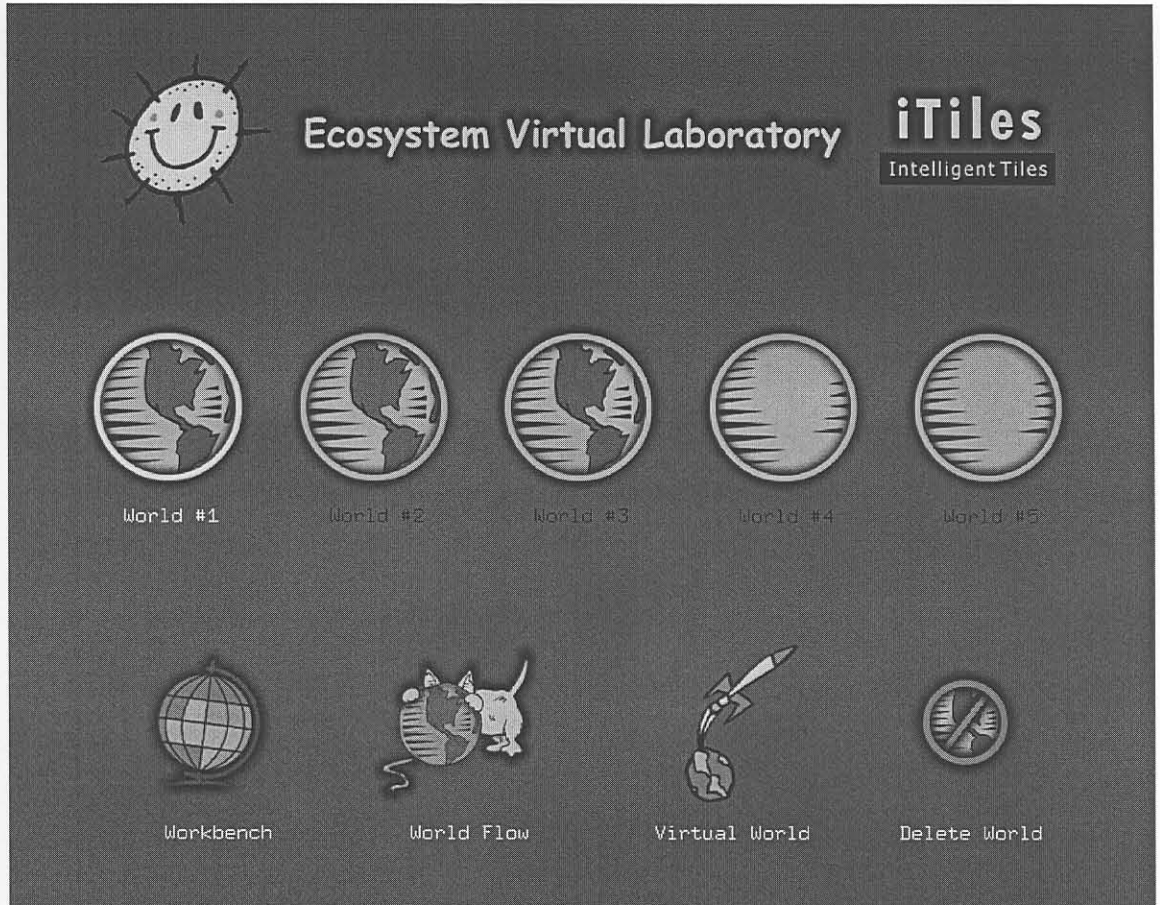
4.3.2 The introductory screen and main menu

The introductory screen of the iTiles Ecosystem Virtual Laboratory is the screen that is presented to the user when the application is launched. Figure 36 presents a screenshot of the introductory screen. The application provides space for five iTiles worlds to be created. A sphere of the earth represents an already existing iTiles world, and an empty sphere indicates a space to include a new world. Existing iTiles worlds are easily accessed by a mouse click on a sphere. When a sphere is selected, it is highlighted with a yellow circle, and the iTiles world is selected. A user may remove an existing iTiles world making space to include a new world, by selecting an existing world and clicking on the 'delete world' image button.

The application provides a mechanism for controlling the order in which the iTiles system tools may be launched for a specific iTiles world. This mechanism translates to the three steps of the iTiles framework, as described in Section 3.1.1. An iTiles world firstly needs to be authored using the iTiles Workbench tool before specifying the behaviour of that iTiles world using the iTiles Work Flow tool. Only once an iTiles world has been authored and the behaviour specified, may the iTiles Virtual World simulation tool be launched. The icons underneath the spheres are image buttons used to launch the iTiles system tools. These image buttons may either be enabled or disabled according to the state of the selected iTiles world (i.e. the iTiles world may not have an iTiles World Flow defined and therefore the iTiles Virtual World tool will not be able to be launched). To launch a tool that is enabled, a user can simply click the image button for the corresponding tool.

Figure 37 presents a screenshot of the iTiles main menu. The main menu is accessible at any time during program execution by pressing the ESC key on the keyboard. With this menu a user can launch an iTiles system tool for the selected iTiles world. The menu will also not display any iTiles system tool that cannot be activated for the selected iTiles world. If one of the iTiles system tools is currently active (not the introduction screen), the menu allows a user

to go back to the introductory screen, by selecting the menu option ‘choose another world’. Functionality is also provided to activate a full screen mode if the application is running in a window, activate a window mode if the application is running full screen, and to exit the iTiles system. The menu also allows a user to alter the camera smoothing speed (as discussed in Section 4.2.8).



tiles of an *Figure 36. The iTiles Ecosystem Virtual Laboratory introductory screen* as described in Section 4.2.8. The image buttons appearing in the top right of the screen are used as the three camera modes (see Figure 38 or 39). Consider these image buttons be tested mouse, button, and button, or appearing from left to right, top to bottom. Button, is used for activating a camera rotation mode, where the iTiles world can be rotated in different axis. Button, is used for activating a camera mode where a user can walk in and out of an iTiles world, and shift the world on each of the axes. Button, is used for locking and unlocking the camera. When the camera is locked the current view of the iTiles world will remain unchanged. However, when the camera is unlocked, the selected tile will always appear in the middle of the screen, and the view of the iTiles world will constantly shift in relation to the selected tile.

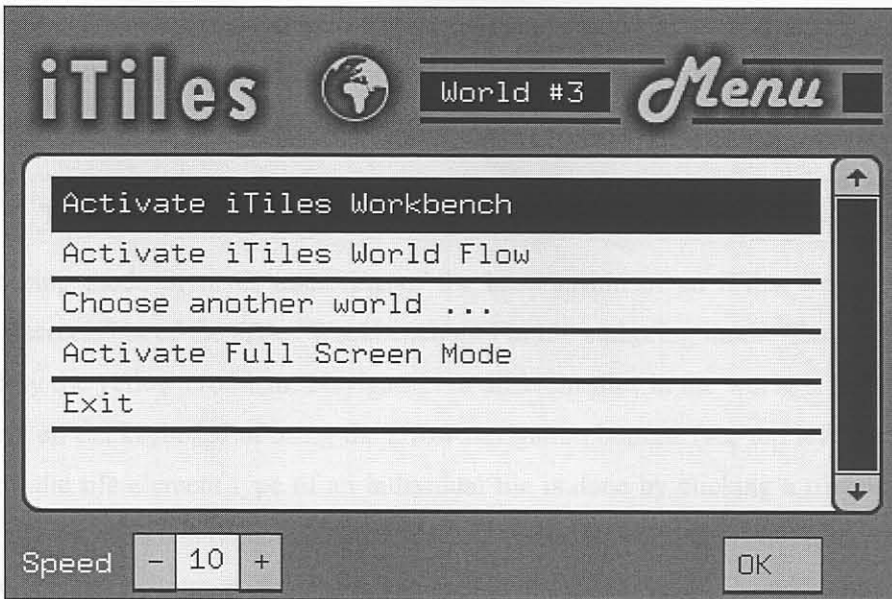


Figure 37. The iTiles main menu

4.3.3 The iTiles Workbench tool

The iTiles Workbench is the authoring tool for an iTiles world and allows teachers to author an iTiles world for use in the presentation of a lesson. With the iTiles Workbench tool a new or an existing iTiles world can be authored. Authoring of an iTiles world is a visual process that involves the procedures of authoring the base terrain of tile elements, and specifying which world objects will inhabit the world. These authoring procedures have been divided in the iTiles Workbench tool as two different modes: *tile authoring mode* and *world object authoring mode*.

In both modes of the iTiles Workbench, functionality is provided for navigating the individual tiles of an iTiles world and for camera rotation, shifting and zooming functions as described in Section 4.2.8. The image buttons appearing in the top right of the screen are used to for these camera modes (see Figure 38 or 39). Consider these image buttons be called $button_1$, $button_2$ and $button_3$, as appearing from left to right, top to bottom. $button_1$ is used for activating a camera rotation mode, where the iTiles world can be rotated on different axes. $button_2$ is used for activating a camera mode where a user can zoom in and out of an iTiles world, and shift the world on each of the axes. $button_3$ is used for locking and unlocking the camera. When the camera is locked the current view of the iTiles world will remain unchanged. However, when the camera is unlocked, the selected tile will always appear in the middle of the screen, and the view of the iTiles world will continually shift in relation to this selected tile.

The iTiles Workbench tool is generic authoring process since it allows a user to choose tile element types of the base terrain and world objects when authoring an iTiles world. Therefore an infinite number of possible worlds can be created.

4.3.3.1 Tile authoring mode

Tile authoring mode involves authoring of the base terrain of an iTiles world. Figure 38 presents a screenshot of the iTiles Workbench tool in tile authoring mode. The selected tile is indicated by the yellow crosshair. Navigation to different tiles in the world is done using the arrow keys on the keyboard or using the arrow navigation buttons (see top left in Figure 38). Specifying the tile element type of an individual tile is done by clicking a tile element type from the tile set, which is depicted by the horizontal ‘traffic light’ (see bottom right in Figure 38). Functionality to grow or shrink the base terrain is also provided. To add more tiles in all directions to the base terrain (to the perimeter), a user can click the plus button (see left, middle in Figure 38). To add tiles in a specific direction a user can click an arrow button surrounding the plus button, in the corresponding direction. Removing tiles from the base terrain is similarly done by clicking the minus button, and arrow buttons surrounding the minus button.

The larger the base terrain of an iTiles world, the more computational dependencies arise. In this implementation of the iTiles system, an iTiles world has been restricted to a base terrain of the maximum dimensions of 10x10. This also restricts the number of world objects that can be placed in the world. The more dynamic world objects present, the more the simulation engine will need to accommodate in terms of processing time and storage for the movement and behaviour of characters.

4.3.3.2 World object authoring mode

Figure 39 presents a screenshot of the iTiles Workbench tool in world object authoring mode. In the Pokémon™ craze in South Africa (January 2001 - June 2002) the South African potato chip manufacturer Simba (Pty) Ltd, included circular plastic discs of the pokémon characters called tazos in packets of potato chips. These tazos indicated the skills and abilities of a pokémon character, and were intended for children to collect, and used in a game children could play. The circular shape at the bottom right of the screen uses this tazo metaphor for iTiles world objects. When a world object is selected from the list of available world objects (see left part of Figure 39), magnification of the world object is displayed in the tazos. The tazos indicate whether the world object is a dynamic or static world object, and displays the world object’s tile attribute list, for indicating what tile element types the world object may be

¹ Tazos is a trademark and copyright of Nintendo.

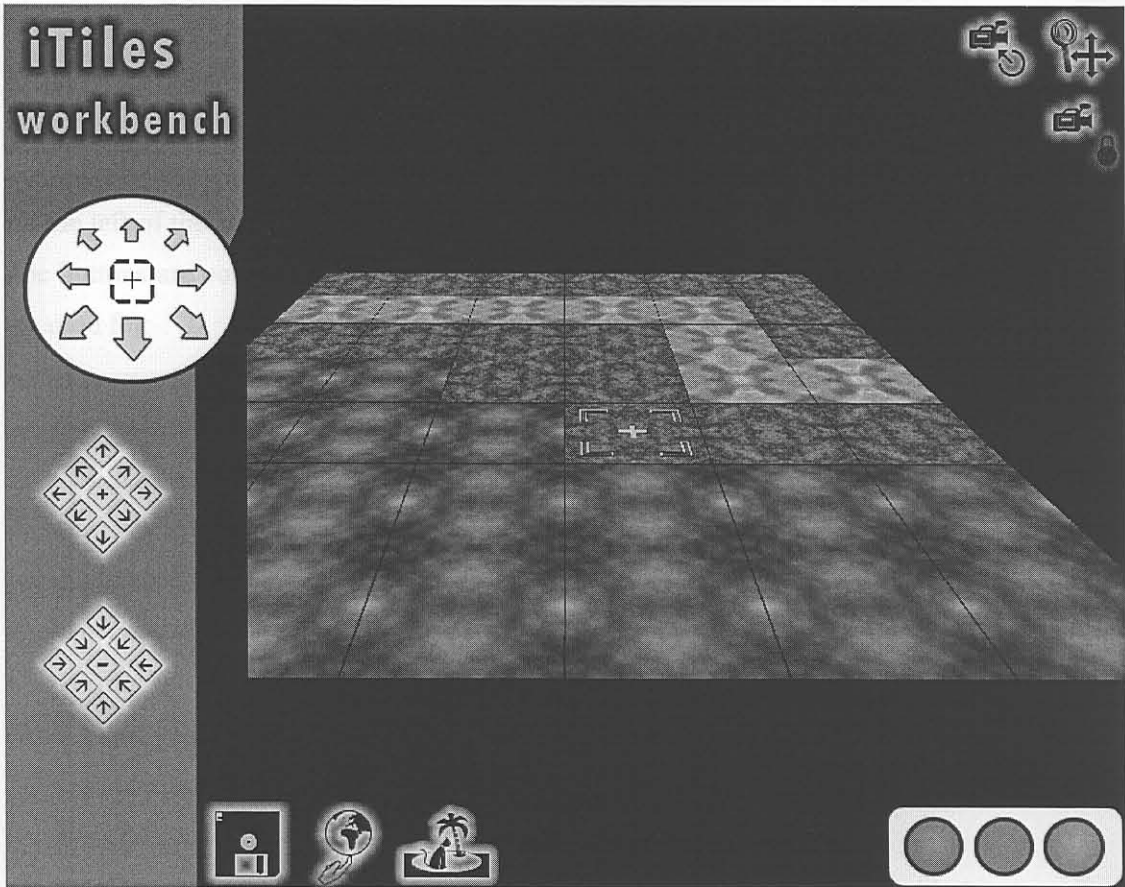


Figure 38. A screenshot of the iTiles Workbench tool in tile authoring mode

Clicking the image button with a ‘globe spinning on a finger’ is used for activating the tile authoring mode. Clicking the ‘cat on an island’ image button is used for activating the world object authoring mode, which is described next.

4.3.3.2 World object authoring mode

Figure 39 presents a screenshot of the iTiles Workbench tool in world object authoring mode. In the Pokémon™ craze in South Africa (January 2001 - June 2002), the South African potato chip manufacturer Simba (Pty) Ltd, included circular plastic discs of the pokémon characters called tazos in packets of potato chips. These tazos¹ indicated the skills and abilities of a pokémon character, and were intended for children to collect, and used in a game children could play. The circular shape at the bottom right of the screen uses this tazo metaphor for iTiles world objects. When a world object is selected from the list of available world objects (see left part of Figure 39), information of the world object is displayed in the tazo. The tazo indicates whether the world object is a dynamic or static world object, and displays the world object’s tile attribute list, for indicating what tile element types the world object may be

¹ Tazo is a trademark and copyright of Nintendo.

placed on. A wire frame cube is also used to indicate which world object is selected. In the screenshot in Figure 39, the dog is selected. It is a dynamic world object, and may only be placed on grass or sand tile element types.

To the left of the *tazo*, are buttons providing functionality for the authoring of world objects. The buttons are coloured green when enabled, and red when disabled. These image buttons are used for:

- *Placement.* Placement of world objects involves the addition (clicking the plus image button) and removal (clicking the minus image button) of a world object on a tile in the base terrain. World objects can only be placed on tiles in the world according to their tile attribute list. The plus image button will be coloured red if a user cannot add a world object to the selected tile, or if the tile is currently occupied by another world object. This is because only one world object may be placed on an individual tile.
- *Orientation.* A world objects can be rotated so that it faces a certain direction.
- *Scaling.* A world object can be scaled (up or down) to a certain percentage between 0% and 100%. These percentages are linked to the world object's minimum scale and maximum scale, so that the world object retains visual conformation.
- *Movement.* A movement mode can be activated where the selected world object can be moved to another tile in the world that it may be placed on.
- *Search.* When a certain world object is selected, a user can search if such a world object exists in the world by clicking on the eye image button. If found, the world object will be selected. If more that one object of that type exists, clicking the button again, will navigate to the next found world object.



Figure 42. Approximate view for an iTiles Workbook file

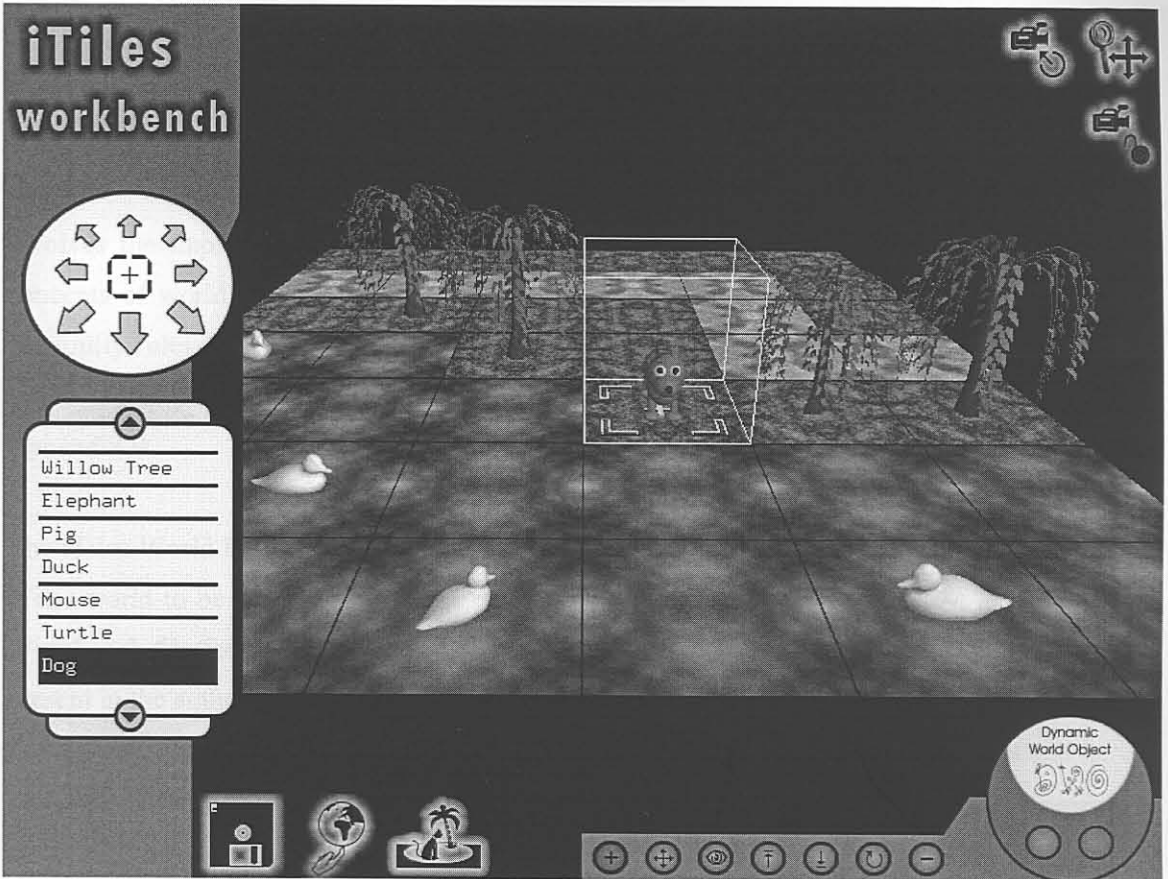


Figure 39. A screenshot of the iTiles Workbench tool in world object authoring mode

A user can save the authored iTiles world at any time by clicking the diskette image button (see Figure 38 or 39). Figure 40 summarises the information stored in a file for an iTiles world authored using the iTiles Workbench tool. This file contains information on the width and height of the base terrain, the tile element type of each tile in the base terrain, and the world objects that were placed in the world. The position, direction and scale properties specified for these world objects are their initial properties in the simulation of the iTiles world. Once an iTiles world is saved the behaviour of the iTiles world can be specified using the iTiles World Flow tool, which is described next.

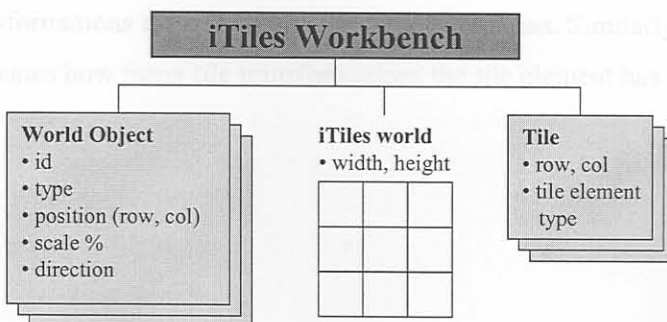


Figure 40. Information stored for an iTiles Workbench file

4.3.4 The iTiles World Flow tool

The iTiles World Flow tool is used for specifying rules that govern the simulation of an authored iTiles world. It gives a teacher the freedom to control the behaviour of the simulation of an authored iTiles world. The implementation of the iTiles World Flow tool involves the implementation of algorithms and screens to capture the iTiles World Flow concepts of world transformations and world forces, and adhere to the iTiles World Flow feasibility rules as described in Section 3.6.3. Many screens have been implemented for the iTiles World Flow concepts. A collection of screenshots of these screens is presented in Appendix B.

The iTiles World Flow tool provides for either a new or an existing iTiles World Flow of an iTiles world to be specified. When starting up, the tool uses information stored in an iTiles Workbench file for initialisation. From this file the tool determines what world objects were present in the authored world and constructs a list containing one kind (a specimen) of each of those world objects. This is because all world components (world object or tile) of a similar kind will share the iTiles World Flow properties as specified for that specific kind of world component in the iTiles World Flow tool. Therefore all world components (tile or world object) of a similar kind will exhibit the same behaviour in the simulation.

Figure 41 presents a screenshot of the main screen of the iTiles World Flow tool. The tab buttons at the top of the screen enable a user to navigate to a list of dynamic world objects, static world objects and tile elements. The lists of static and dynamic world objects contain those world objects from the constructed list. The list of tile elements contains a tile of each tile element type from tile set. In Figure 41, the dynamic world object's tab button is selected, and the duck dynamic world object is selected from the list. To the right of the list is the 'World Flow summary'. For a dynamic world object, this summary indicates how many positive forces, negative forces, world transformations, or movement forces are specified the selected dynamic world object. For a static world object the summary indicates how many world object transformations the selected static world object has. Similarly, for a tile element, the summary indicates how many tile transformations the tile element has.

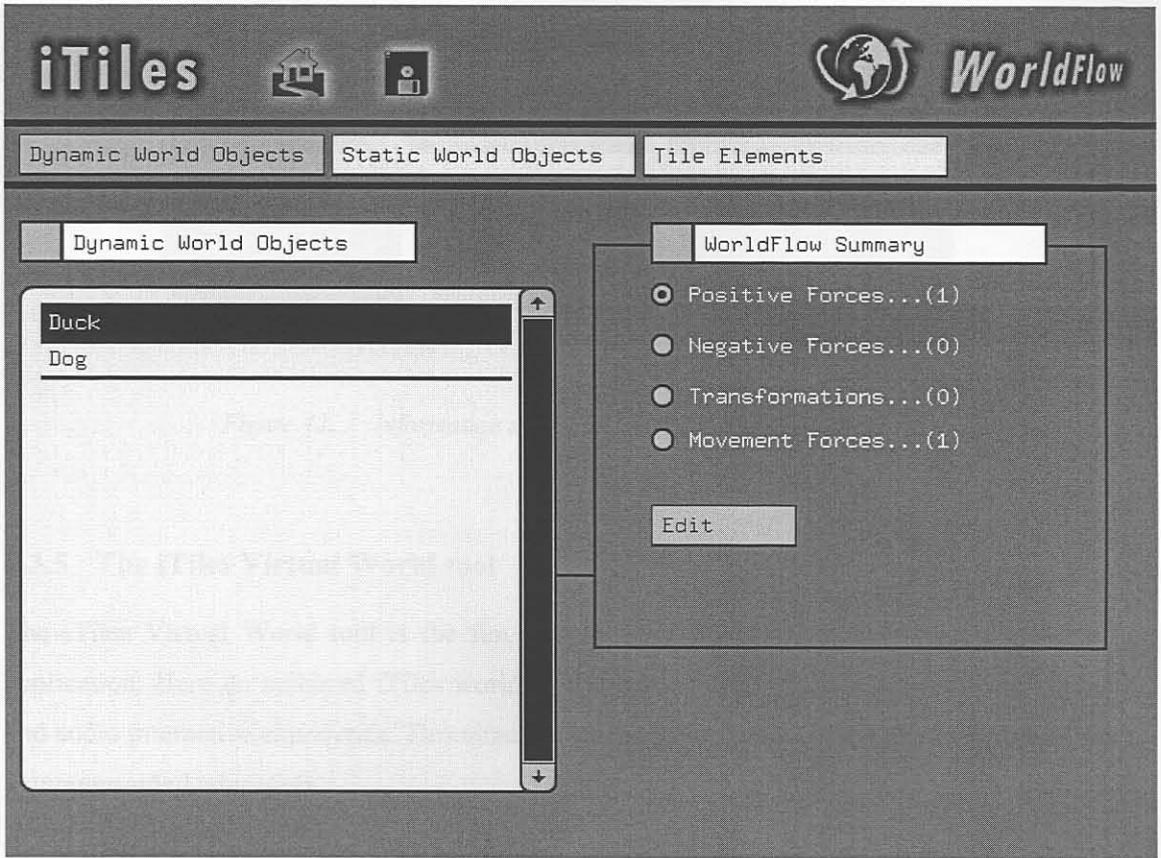


Figure 41. The main screen of the iTiles World Flow tool

To navigate to the positive forces of a selected dynamic world object, a user should select the 'positive forces' radio button in the World Flow summary, and click the edit button. A screen will be presented with a list of that dynamic world object's positive forces. A user can then add new positive force, edit an existing specified positive force or remove a positive force for the dynamic world object. Similarly, a user can add, edit or remove:

- Tile transformations of a specific tile element type.
- World object transformations of a static world object.
- Negative forces, movement forces or world object transformations of a dynamic world object.

The iTiles World Flow can be saved at any time clicking the diskette image button. Figure 42 summarises the information stored in a file for the iTiles World Flow tool. Once saved, the iTiles world is ready for simulation, and iTiles Virtual World simulation tool can be launched. This tool is discussed next.

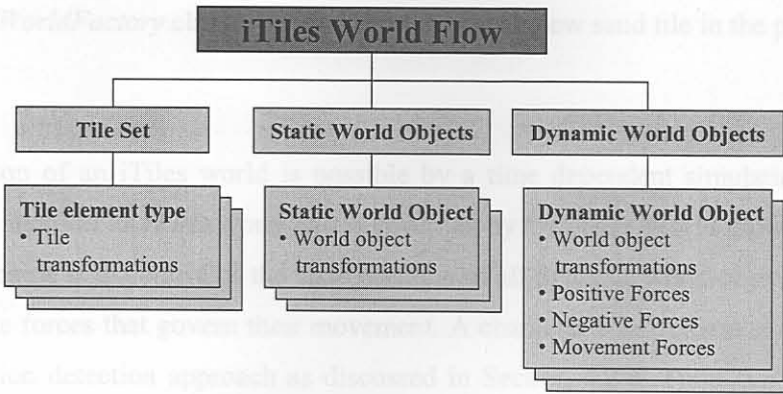


Figure 42. Information stored for an iTiles World Flow file

4.3.5 The iTiles Virtual World tool

The iTiles Virtual World tool is the fun part of the iTiles Ecosystem Virtual Laboratory application. Here an authored iTiles world is “brought to life” through a multimedia visual and audio interactive experience. This simulation is experienced by young learners to enjoy as computer-aided education.

The iTiles Virtual World tool combines information saved from the iTiles Workbench tool and the iTiles World Flow tool to present a simulation of an authored iTiles world. The *iTilesVirtualWorldFactory* class is responsible this initialisation procedure. This class uses the factory method, an object oriented programming design pattern, which is used when a specific class wants to delegate responsibility to one of several helper subclasses, which can instantiate objects with certain characteristics [50]. This concept is analogous to an industrial factory. The *iTilesVirtualWorldFactory* class is used to ‘manufacture’ world objects and tiles during the initialisation procedure of the iTiles Virtual World simulation tool. It integrates the information of world objects and tiles captured in both the Workbench tool and the World Flow tools, and combines that information as input into the Virtual World simulation tool. In this ‘manufacturing process’ each world object that has been specified in the iTiles Workbench tool is given the behavioural properties (world transformations and world forces), as defined for it’s kind in the iTiles World Flow tool. Similarly, tile transformations specified for the different tile element types are also attached individually to each tile of the base terrain of the iTiles Virtual World. This ‘manufacturing process’ is needed because world transformation and world forces have numeric attributes that will change during the course of simulation. The *iTilesVirtualWorldFactory* is also used to ‘manufacture’ new tile objects during the course simulation. This will occur when a certain tile transforms to a different tile element type. For example, if a grass tile transforms to a sand tile, the

iTilesVirtualWorldFactory class will be able to produce a new sand tile in the place of the old grass tile.

The simulation of the authored iTiles world as shown in Figure 39. The following functionality is provided by the iTiles Virtual World simulation tool:

The simulation of an iTiles world is possible by a time dependent simulation engine that monitors the status of an iTiles world and is governed by the iTiles World Flow concepts. The simulation engine is in control of the state machine of all dynamic world objects (see Section 3.6.2) and the forces that govern their movement. A character's movement is also monitored by the collision detection approach as discussed in Section 4.2.6. Time dependent updates such as the number of world beats that a dynamic world objects should remain frozen for, and increasing the force strength of incremental seeking positive forces of characters are implemented. The visual and auditory changes in the simulation in terms of iTiles World Flow concept of world transformations are implemented. The simulation engine is also responsible for the tile merging process as described in Section 4.2.7. Since the merged world is larger, the world objects positions are shifted accordingly when placed on tiles in the iTiles world during the initialisation procedure.

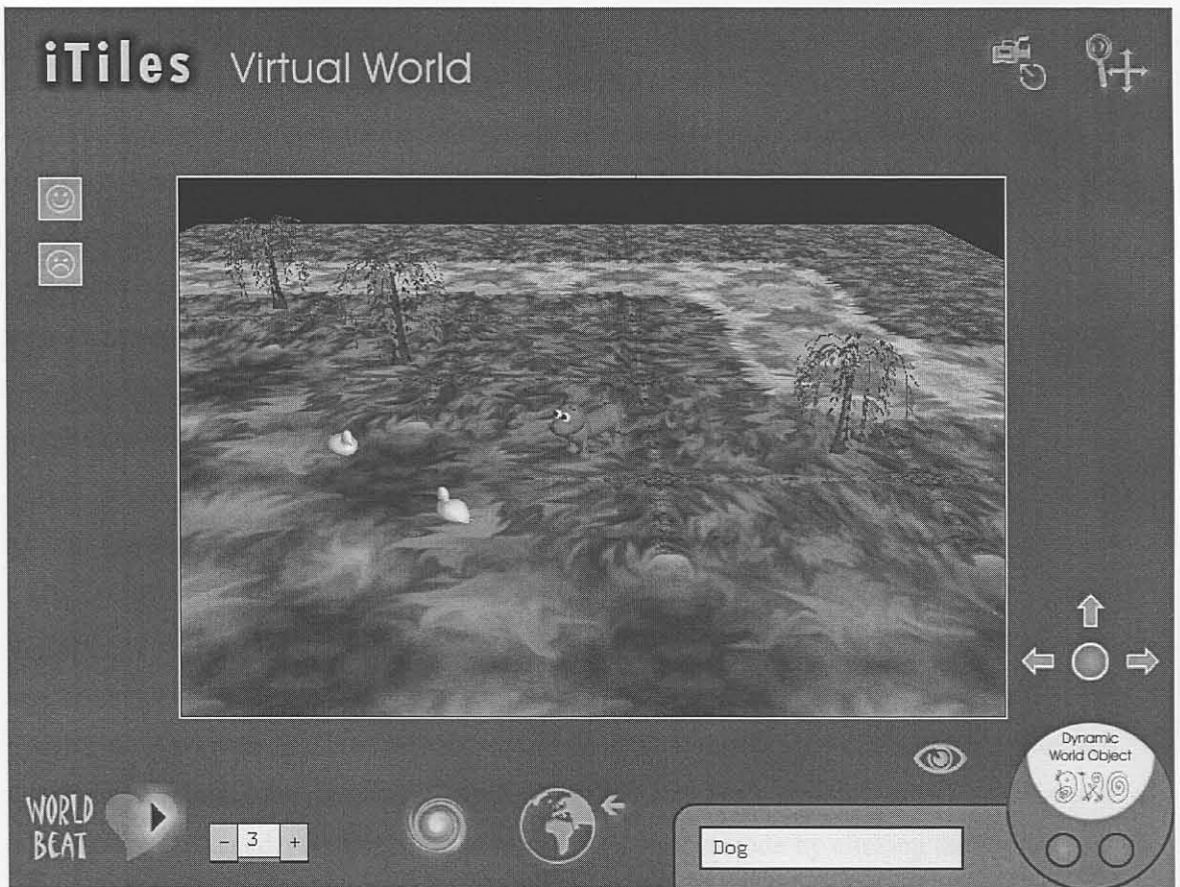


Figure 43. A screenshot of the iTiles Virtual World tool

Figure 43 presents a screenshot of the iTiles Virtual World tool. The iTiles world shown here is the simulation of the authored iTiles world as shown in Figure 39. The following functionality is provided by the iTiles Virtual World simulation tool:

- *Managing simulation time.* The simulation of an iTiles world can be started or stopped by clicking on the ‘heart’ image button in the bottom left of the screen. The simulation time can also be sped up or slowed down by specifying how many seconds each iTiles world beat consists, using the plus minus widget control provided.
- *World object selection and taking control of characters.* Navigation and interaction is implemented as discussed in Section 3.7.1. A user can navigate through the world objects of an iTiles world by clicking on ‘swirl/portal’ image button (see bottom part of Figure 43). This portal metaphor is used to symbolise an idea that a user is magically transported to the world object. The selected world object will always appear in the middle of the screen. Taking control of a character (i.e. taking on a virtual identity) and being able to move it through an iTiles world is also available. The ‘circle’ image button above the tazoo (see bottom right part of Figure 43) is used to take control of a selected character, and the ‘arrow’ image buttons are used to control the character’s movement in the world.
- *Identification.* Information of the selected world object is presented similarly to the iTiles Workbench tool. The world object’s name is displayed in addition to the tazoo, indicating the tile attribute list of the world object. Additional information of what world components (world objects or tiles) the character likes and does not like in terms of its positive and negative forces can also be displayed as a list. These lists are displayed when a user clicks on the ‘smiley’ and ‘sad smiley’ image buttons (see top left part of Figure 43). If the selected world object is a static world objects, these ‘like’ and ‘don’t like’ lists are respectively populated with characters that like and do not like the static world object.
- *View of an iTiles world.* Seeing an iTiles world through different perspectives and points of view is available. The camera rotation, shifting and zooming modes as discussed for the iTiles Workbench tool are also available (see top right part of Figure 43). However the camera mode that enables a user to lock or unlock the camera is not available. The camera remains unlocked in the iTiles Virtual World tool. The view types discussed in Section 3.7.1 are implemented. Figure 44 shows a third person view point of a dog, while Figure 45 shows the first person view point from the dog’s eyes. A user can alternate between these view types by clicking the ‘globe’ image button. The inner vision mode is also implemented. A user can activate the inner vision mode by clicking the ‘eye’ image button. For example, Figure 46 shows the third person view of a duck character with the inner vision mode activated. This mode is also available in first person view.

- *Spatial sound.* The sounds that occur in the simulation are played spatially in relation to position of the selected world object. The user is considered at the position of the world object. For example in Figure 44, a user would hear a duck quacking on his left speaker or earphone.



Figure 44. *Third person view*



Figure 45. *Virtual identity view*

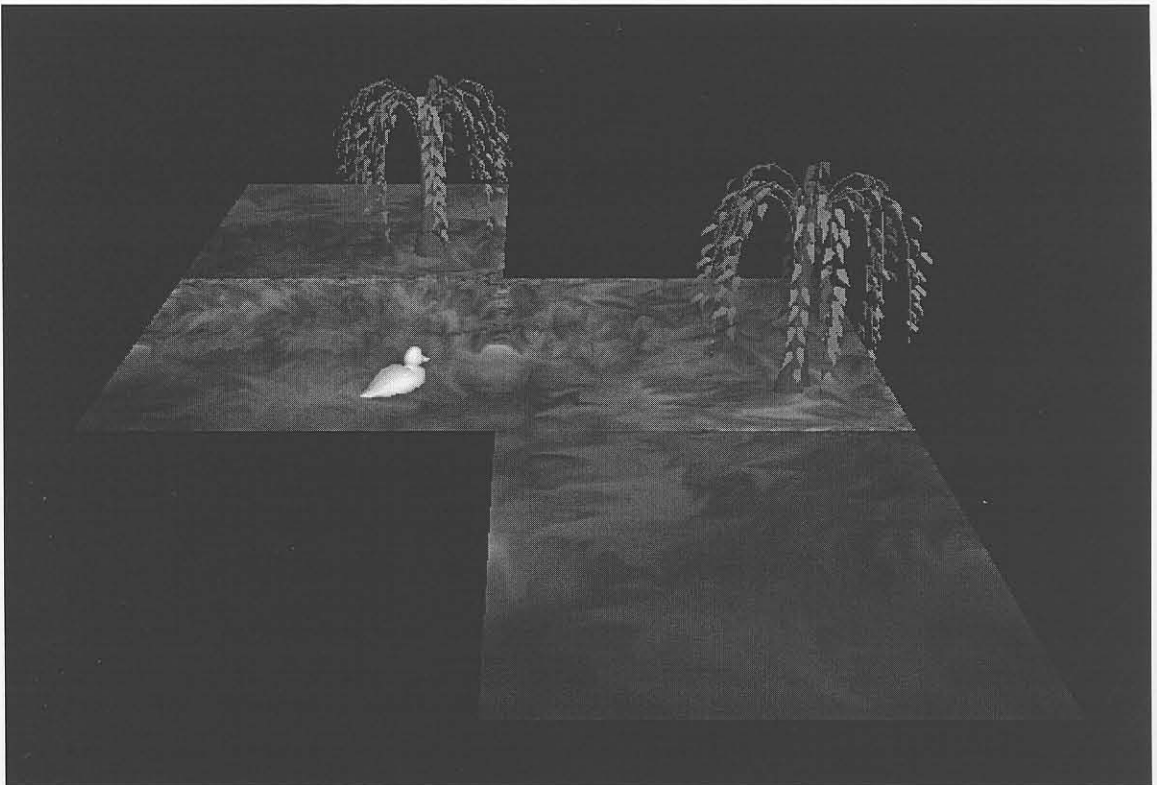


Figure 46. *Inner vision view*

4.4 Summary

This chapter presented the implementation of the Intelligent Tiles Virtual Laboratory framework. Firstly an overview of the implementation was given, covering subjects such as the programming languages used for implementation and user interface design. Next implementation design concepts were presented. Lastly an overview of the implemented system was presented with screenshots. In the next chapter an iTiles world produced with the iTiles Virtual Laboratory application is presented.

"Virtual worlds aren't pictures, they're places. You don't observe them, you experience them"
-D. Steward 1991 (3)

Drought in Africa

In this chapter the design, authoring and simulation of a virtual laboratory developed with the iTiles Intelligent Virtual Laboratory application is presented. This virtual laboratory is titled "Drought in Africa", and is aimed at teaching young learners the ecological occurrence of drought. Firstly the lesson objects of the virtual laboratory are highlighted, followed by a description of the authoring of the iTiles world. Next the behaviour specified for the iTiles world is discussed, followed by the simulation of the iTiles world.

5.1 Identifying lesson objectives

The Drought in Africa virtual laboratory aims at teaching young learners the environmental effects of drought in an African savannah. By using this virtual laboratory, young learners should understand the following ecological consequences:

- Scarce food and water supply in times of drought.
- Thirst of animals.
- Fragile terrain.
- Depletion of resources such as a water source and plant life.
- Plant life affected by overgrazing and over-browsing.
- The desertification of a geographical region as a result of drought.

5.2 Authoring

After taking into account the requirements and lesson objectives of the virtual laboratory, the iTiles Workbench tool is used for authoring of the iTiles world. The focus here is the layout of the base terrain and world objects that are added to the authored iTiles world.

Chapter 5

The base terrain of the iTiles world is authored to look like an arid African savannah during a harsh drought. See Figure 47 below, where the majority of the base terrain is sand, and contains areas with a sparse supply of water with patches of grass near the water.

"Virtual worlds aren't pictures, they're places. You don't observe them, you experience them"

-D. Steward 1991 [5]

Drought in Africa

In this chapter the design, authoring and simulation of a virtual laboratory developed with the iTiles Ecosystem Virtual Laboratory application is presented. This virtual laboratory is titled 'Drought in Africa', and is aimed at teaching young learners the ecological occurrence of drought. Firstly the lesson objects of the virtual laboratory are highlighted, followed by a description of the authoring of the iTiles world. Next the behaviour specified for the iTiles world is discussed, followed by the simulation of the iTiles world.

5.1 Identifying lesson objectives

The *Drought in Africa* virtual laboratory aims at teaching young learners the environmental effects of drought in an African savannah. By using this virtual laboratory, young learners should understand the following ecological occurrences:

- Scarce food and water supply in times of drought.
- Thirst of animals.
- Fragile terrain.
- Depletion of resources such as a water source and plant life.
- Plant life affected by overgrazing and over-browsing.
- The desertification of a geographical region as a resultant of drought.

5.2 Authoring

After taking into account the requirements and lesson objectives of the virtual laboratory, the iTiles Workbench tool is used for authoring of the iTiles world. The focus here is the layout of the base terrain and world objects that are added to the authored iTiles world.

In tile authoring mode, the base terrain of the iTiles world is authored to look like an arid African savannah during a harsh drought. See Figure 47 below, where the majority of the base terrain is sand, and contains areas with a scarce supply of water, with patches of grass near the water.



Figure 47. Authoring the base terrain to look like an arid African savannah

In world object authoring mode, the base terrain is thereafter populated with world objects (see Figure 48). The world objects added are:

- A herd of elephants (classified as dynamic world objects), which have been scaled to different sizes to depict ages such as young calves, adult cows and bulls.
- A few broad trees (classified as static world objects), to serve as food for the elephants.



Figure 48. Populating the African savannah with elephants and trees

5.3 Specifying behaviour

Using the iTiles World Flow tool, the behaviour of the world objects and environmental changes that should occur in the simulation of the authored iTiles world are specified. Adding world transformations and world forces to the iTiles world helps establish the lesson objectives. A description of the world transformations and world forces added is presented next, followed by a summary at the end of this section for a clearer understanding thereof.

World transformations

The following tile transformations are added:

- *Grass to sand.* A grass tile element should transform to a sand tile element gradually. The ‘magic’ sound clip should be played when the transformation occurs (transformation sound).
- *Water to sand.* A water tile element should transform to a sand tile element gradually. The ‘water splashing’ sound clip should be played when the transformation meter is triggered (meter alteration sound). The ‘magic’ sound clip should be played when the transformation occurs (transformation sound).

The following world object transformations are added:

- *Broad tree scale smaller immediately.* A broad tree should transform in scale, and should scale smaller by 10% of its current scale immediately. The ‘magic’ sound clip should be played when the transformation occurs (transformation sound).

World forces

The following movement forces are added:

- *Elephant grass movement force.* When an elephant moves on the grass tile element, this movement force’s action should trigger the ‘grass to sand’ transformation, and increase the gradual transformation meter by 10% with each trigger.

The following positive forces are added, indicating to which world components an elephant is attracted:

- *Another elephant.* This is a constant positive force seeking type with a force strength of 50%. When the other elephant is found, the elephant will do nothing (found action). The ‘elephant trumpet’ sound clip should be played when this elephant finds another elephant (found sound).
- *A broad tree.* This is a constant positive force seeking type with a force strength of 50%. Once the elephant finds the broad tree, it should stay in the location for one world beat (found action). This positive force’s action should trigger the broad tree’s transformation of scaling smaller by 10% immediately to occur.
- *Water tile edge.* This is an incremental positive force seeking type, with an initial force strength of 10% which increases by 10% every 2 world beats, and resets to a force strength of 40% once the water tile edge is found. Once the water tile edge is found, the elephant will remain in the location for two world beats (found action). This positive force’s action should trigger the water tile element’s ‘water to sand’ transformation to occur and increase this gradual transformation meter by 20%.

Summary

The world transformations and world forces described above can be summarised as follows: The elephants drink from the scarce supply of water, and it dries out to sand, indicating the drought as the water source gets depleted. The elephants also walk on the fragile grass and it erodes to sand, indicating the fragile terrain. In times of drought elephants eat the leaves and bark of a tree to absorb the moisture. With the broad tree getting smaller, this depicts that this resource is being depleted. Also, when elephants come in close contact they trumpet, indicating communication.

5.4 Simulation

The iTiles Virtual World tool presents a simulation of the authored iTiles world. Figure 49 presents the simulation of the iTiles world at the start of the simulation. With simulation time as a factor, the environment is slowly transformed in time. Scenes at various time intervals in the simulation of the Drought in Africa virtual laboratory are presented in Figure 50. In time the iTiles world is eventually transformed to sand and the food supply gets scarce, indicating the desertification process. Figure 51 shows thirsty elephants drinking the last bit of water.

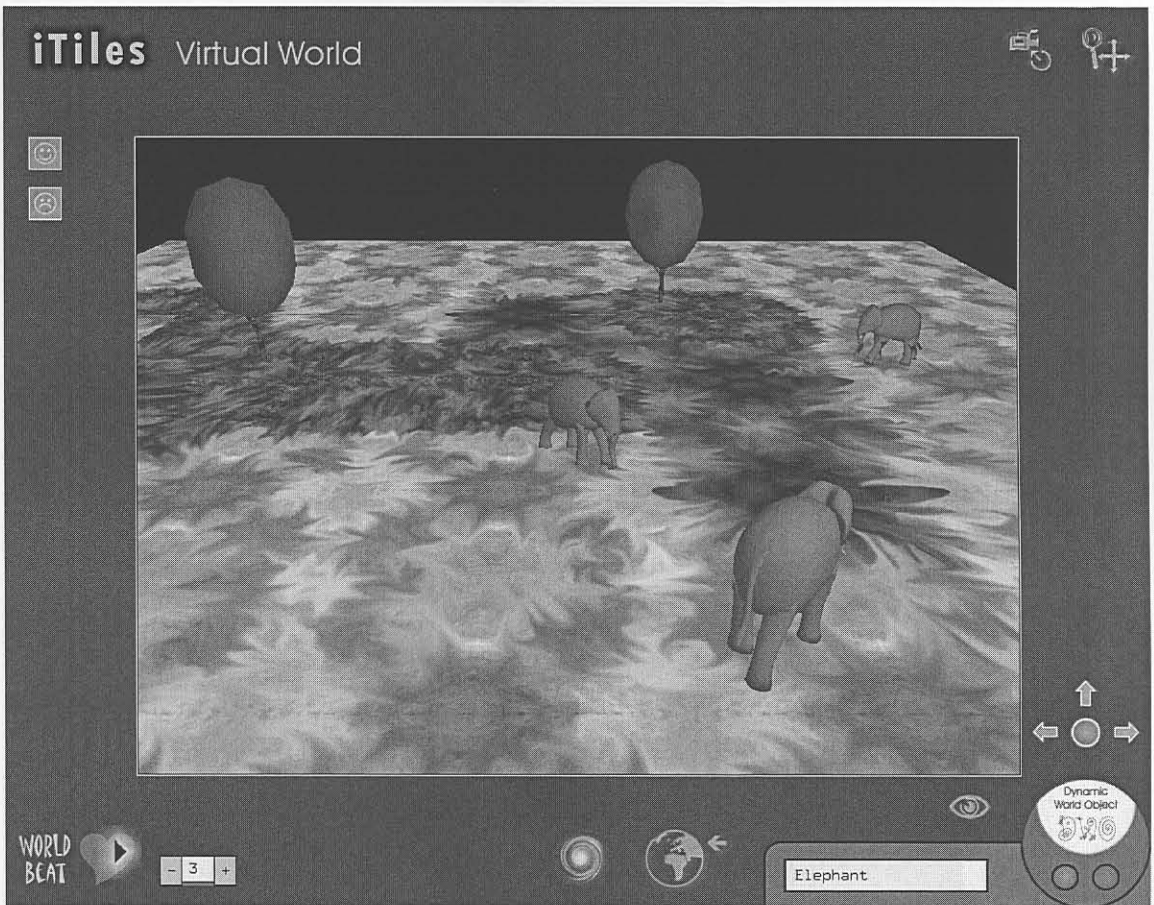


Figure 49. The iTiles world at the start of the simulation

Figure 51. Thirsty elephants drinking the last bit of water

The following pedagogies can be addressed when teaching young learners with the Drought in Africa virtual laboratory:

- *Observation.* By observing the simulation, learners will understand the processes of drought. The lesson objectives in terms of the pedagogical occurrences that young learners

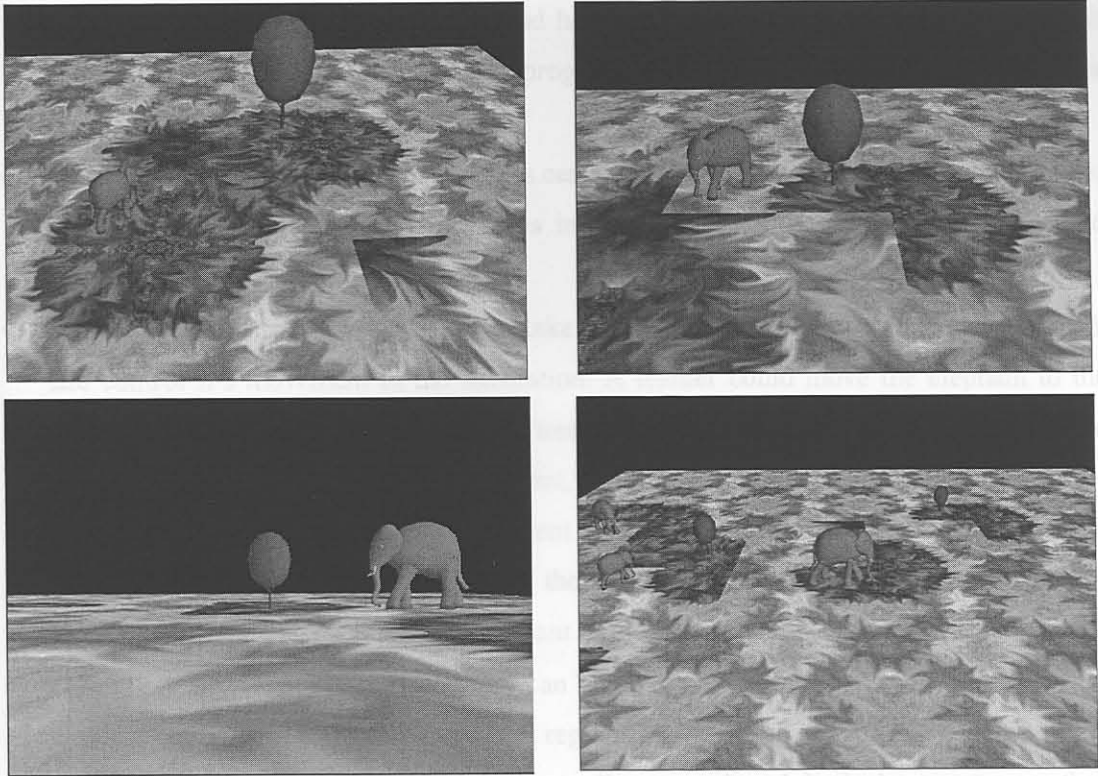


Figure 50. Scenes from the simulation of the Drought in Africa world

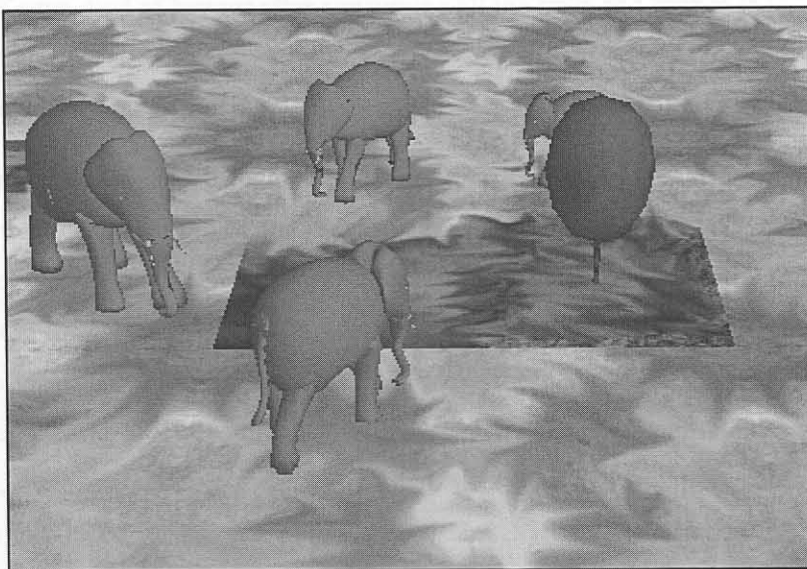


Figure 51. Thirsty elephants drinking the last bit of water

The following pedagogies can be addressed when teaching young learners with the Drought in Africa virtual laboratory:

- *Observation.* By observing the simulation, learners will understand the harshness of drought. The lesson objectives in terms of the ecological occurrences that young learners

should understand are achieved through observation of the simulation. The behaviour of the elephants will depict their thirst, and how the environment is transformed by their actions. Learners will also experience progression of time and the desertification of a region.

- *Mathematics and classification.* Learners can be tasked to count the number of elephants. They too can understand the differences in sizes of elephants, depicting their age and gender.
- *Interaction.* A learner can be tasked to take on the role of an elephant (virtual identity), and control it's movement in the simulation. A learner could move the elephant to the waters edge to cause it to drink, or to a tree to browse leaves and bark. A learner could also move that elephant to another elephant, and both elephants would trumpet. A learner could also be tasked to experience different views of the world. When taking on the role of an elephant, a learner can experience the world as an observer from the third person view, or from the perspective of an elephant in the first person view.
- *Offline activity.* This virtual laboratory can be linked to another curricula activity where children learn about elephants and desert regions. An example of such a curricula activity could include learning facts about elephants. For example, adult elephants eat more than 200kg per day of vegetation. Elephant conservation efforts can also be addressed. For example, during a severe drought in the 1970's, 7000 elephants of Tsavo East National Park in Kenya died of hunger after consuming all the trees and bushes [51].

5.5 Summary

This chapter presented the Drought in Africa virtual laboratory as an example of a virtual laboratory produced with the iTiles Ecosystem Virtual Laboratory. This virtual laboratory aims at teaching young learners about the environmental effects of drought. The chapter described how the iTiles Workbench was used for authoring the iTiles world and how the behaviour of that iTiles world was specified using the iTiles World Flow tool. Lastly the iTiles Virtual World tool illustrated how this virtual laboratory can address pedagogies in observation, mathematics and classification and offline activities. Other examples of virtual laboratories produced using the iTiles Ecosystem Virtual Laboratory are presented in Appendix C.

this application teaches children about the desertification of an African savannah during a harsh drought.

This thesis proves that it is possible to apply a cheaper virtual reality platform such as Desktop VR, and achieve a user-friendly (teacher and learner friendly) product of high quality. It presents a framework for virtual laboratories attempting to address many of the

Chapter 6

virtual laboratories. It provides an application that can be used by a non-technical user (i.e. teacher) and is re-configurable (i.e. generic and adaptable). The

framework developed in this study also provides an educational model and content.

"Virtual education will enable us to ask and answer many fundamental questions about the nature of our existence"

- J. Canton 1999 [52]

One of the key advantages of the Intelligent Tiles approach of this thesis, is that it is user friendly. The Intelligent Tiles Framework encapsulates both

Conclusions and Future Work

the development of virtual laboratories. It also considers the development of virtual laboratories, which are easy to grasp. Assembling a virtual world from simple components such as tiles and world objects does not

6.1 Conclusions

This thesis focuses on applying virtual laboratories in an educational setting. Although virtual laboratories have been used in an educational setting with great results, the development of many of these applications is costly, and often expensive when immersive and projection based virtual reality is used. In addition these virtual laboratories are usually of fixed implementations that require programming and a technical background.

The Intelligent Tiles (iTiles) framework developed in this study attempts to overcome many of the above-mentioned limitations of virtual laboratories. It is a generic and adaptable framework that supports teacher driven development of virtual laboratories. By using this framework, virtual laboratories can easily be authored and be used in the education of young learners in teaching topics in earth science, such as ecosystems. The iTiles Ecosystem Virtual Laboratory is a virtual reality application implementing the iTiles framework. This application has the advantage of being affordable, since it is developed for a desktop virtual reality solution, considered the least expensive and most widely used of all virtual reality platforms. With simulations of virtual laboratories using the iTiles framework, young learners can explore, understand and gain mathematical skills such as counting, sorting and classification as well as learning ecological concepts and relationships between different animals and plant life. For example, the 'Drought in Africa' virtual laboratory produced with

this application teaches children about the desertification of an African savannah during a harsh drought.

This thesis proves that it is possible to apply a cheaper virtual reality platform such as Desktop VR, and achieve a user-friendly (teacher and learner friendly) product of high quality. It presents a framework for virtual laboratories attempting to address many of the major problems related to virtual laboratories. It provides an application that can be used by a non-technical user (i.e. teacher) and is re-configurable (i.e. generic and adaptable). The framework developed in this study also puts emphasis on educational needs and content, resulting in an educationally viable product.

One of the key advantages of the application developed using the Intelligent Tiles approach of this thesis, is that it is user friendly. The Intelligent Tiles Framework encapsulates both teacher and learner roles for teaching and learning with virtual laboratories. It also considers the elements of a virtual environment, with simple abstract concepts, which are easy to grasp. Authoring a virtual world from simple components such as tiles and world objects does not add much complexity to the world. The qualities and attributes of world objects bring the simulation of the world to life. The generic and adaptable approach of the framework is aimed at ease of use and extensions.

This study also indicates that by using a well-designed framework it is possible to translate an educational concept into useful applications. The iTiles Ecosystem Virtual Laboratory is implemented in C++ using the open source libraries OpenGL and OpenAL. This application proved that one could implement the Intelligent Tiles framework and concepts in a manner in which the teacher driven development of further virtual laboratories is possible without the need for special technological skills. The simulation of a virtual laboratory authored by a teacher can be used in the education of young learners, where learners can learn by experiencing and learn by doing. The 'Drought in Africa' virtual laboratory presents an example of a virtual laboratory produced using this application. Lesson objectives and educational content were realised with this virtual laboratory.

The iTiles framework and VR applications show that complex ecological concepts regarding eco-conservation could be thought to young learners by employing simple simulations. In this way, learning a complex concept becomes 'fun' rather than unreachable abstract.

The iTiles Ecosystem Virtual Laboratory application has been successfully tested on windows operating systems and a user manual has been written for ease of use (available in Appendix C).

Based on the work carried out in this thesis a publication was presented in a recent international conference on computer graphics [39].

6.2 Future Work

The Intelligent Tiles framework and the iTiles Ecosystem Virtual Laboratory can be extended and improved. In this section such extensions and improvements are suggested.

The Intelligent Tiles Framework currently accommodates a single layer of tiles. By adding another layer of tiles the system can be improved. For example, adding an additional layer of ‘invisible’ tiles above the original layer, can represent the sky. World objects such as clouds and a sun can be added to this new layer. These world objects can trigger world transformations to occur on the world objects and tiles in the layer below. For example, a cloud could rain occasionally causing trees below to grow.

The Intelligent Tiles Framework has been applied in building virtual worlds with an ecological nature, in the field of earth science. This type of ecological world is dependent on the tiles and world objects that are added to the system. One can consider using the iTiles framework in other fields. For example, using the iTiles concepts in a field such as agriculture and forecasting, tractors and sheep can be added as world objects.

The iTiles Ecosystem Virtual Laboratory currently has a non-user friendly mechanism for extending the iTiles system interfaces, for adding new world objects and world sounds. To make it easier for a user to add additional world objects and sounds to the system, an additional tool for the visual representation of these iTiles interfaces can be implemented. This will help in determining the best scaling for a world object on a tile, and easy maintaining of the interface library. To improve the educational experience and content the world object interface can also be extended to include more information about world objects. For animals represented by dynamic world objects, textual information or multimedia videos on the ecological habits and habitats of these animals would be beneficial to students.

In improving the iTiles World Flow tool for specifying the behaviour of an iTiles world, one can modify the input screens to be pictorial or in words, so that young learners too could set

the behavioural properties of an iTiles world. Eliminating or abstracting the numeric attributes of the iTiles World Flow can also be considered, as indicated by the fuzzy logic representation in Chapter 3.

The iTiles Ecosystem Virtual Laboratory application has been implemented on a Windows platform. Portability to other operating systems such as Linux can be considered since the implementation has adhered to an ANSI C standard and uses the open source libraries OpenGL and OpenAL.

The iTiles framework is currently focussed on standalone a VR system. An investigation into extending this into a collaborative virtual environment can be considered. Collaborative authoring of an iTiles world, and collaborative experiencing and participation of the simulation of an iTiles world would be a great enhancement. Students at different locations around the world could “learn together” about a certain world even though they are separated geographically by a vast distance. Such a collaborative system would involve sharing the iTiles system interfaces. A central database can be considered, from which world objects and tiles could be downloaded, as to retain unique identification in each independent iTiles system.

- owners, D., Dunrow, M. *Special Education and Virtual Reality: Challenges and Possibilities*. *Journal of Research on Computing in Education*. 1994, 27 (1), 111
- [6] Abbott, J., Paris, S. Integrating Technology into Preservice Literacy Instruction: A survey of elementary education students' attitudes towards computers. *Journal of Research on Computing in Education*. 2000, 33 (2), 149.
- [7]. Leiniche, R. The successive contributions of computers. *European Journal of Engineering Education*. 1993, 23 (3), 297.
- [8] Dede, C., Selman, M., Loftin, B., Ash, K. Virtual Reality Supports Science Education. *Curriculum Administrator*. 1999, 35 (3), 31.
- [9] Auld, L., Pantelidis, V. Virtual Reality: The Virtual Reality and Education Laboratory at East Carolina University. *T.H.E. Journal*. 1999, 27 (4), 45.
- [10] Holzman, B., Hooper, S. Computers as cognitive media: examining the potential of computers in education. *Computers in Human Behaviour*. 2009, 16, 537-552.
- [11] Open Learning Technology Cooperation Ltd (OLTC), May 1996, *Learning with Software: Pedagogies and Practice* [Online], Australian Department of Employment, Education Training and Youth Affairs (DEETYA). Available: <http://www.education21.edu.au/archives/OLTCdefault.htm> [Accessed November 2002].

- [12] Feldman, D. 2 October 2000, *Technology and Early Literacy: A Recipe for Success*. [Online] Children and Computers website. Available: <http://www.childrenandcomputers.com/> [Accessed: 4 October 2001]
- [13] Furness, T., Wisp, W., Yu, B. The Impact of Three Dimensional Immersive Virtual Environments on Modern Pedagogy: Global Change, PBL and Learning Workflows. Report of workshop held in Seattle, Washington and at the University of Maryland, May-June 1997, 30 January 1998

Bibliography

- [14] Jelsa, C., Savitz, K. (No date), *Computers and Kids: The Best Kind of On-screen Fun*. [Online] Sesame Workshop. Available: <http://www.sesame.com/kids/> [Accessed: 22 May 2002]
- [1] Woolf, H. *Webster's New Collegiate Dictionary*. Springfield: G. & C. Merriam Company, 1979, p. 636.
- [2] internet.com and INT Media Group Incorporated (No date) *Lycos Tech Glossary* [Online] Available: <http://lycos.webopedia.com> [Accessed: 4 April 2002]
- [3] TechTarget Network (No date) *whatis.com IT-specific encyclopedia* [Online] Available: <http://www.whatis.com> [Accessed: 22 May 2002]
- [4] Andolsek, D. Virtual Reality in Education and Training. *International Journal of Instructional Media*. 1995. **22** (2), 145.
- [5] Powers, D., Darrow, M. Special Education and Virtual Reality: Challenges and Possibilities. *Journal of Research on Computing in Education*. 1994, **27** (1), 111.
- [6] Abbott, J., Faris, S. Integrating Technology into Preservice Literacy Instruction: A survey of elementary education students' attitudes towards computers. *Journal of Research on Computing in Education*. 2000, **33** (2), 149.
- [7] Lelouche, R. The successive contributions of computers. *European Journal of Engineering Education*. 1998, **23** (3), 297.
- [8] Dede, C., Salzman, M., Loftin, B., Ash, K. Virtual Reality Supports Science Education. *Curriculum Administrator*. 1999, **35** (3), 31.
- [9] Auld, L., Pantelidis, V. Virtual Reality: The Virtual Reality and Education Laboratory at East Carolina University. *T.H.E Journal*. 1999, **27** (4), 48.
- [10] Hokanson, B., Hooper, S. Computers as cognitive media: examining the potential of computers in education. *Computers in Human Behaviour*. 2000, **16**, 537-552.
- [11] Open Learning Technology Corporation Ltd (OLTC), May 1996, *Learning with Software: Pedagogies and Practice* [Online], Australian Department of Employment, Education Training and Youth Affairs (DEETYA). Available: <http://www.educationau.edu.au/archives/CP/default.htm> [Accessed: November 2001]

- [12] Feldman, D. 2 October 2000, *Technology and Early Literacy: A Recipe for Success* [Online]. Children and Computers website. Available: <http://www.childrenandcomputers.com/> [Accessed: 4 October 2001]
- [13] Furness, T., Winn, W., Yu, R. The Impact of Three Dimensional Immersive Virtual Environments on Modern Pedagogy. *Global Change, VR and Learning Workshops*. Report of workshops held in: Seattle, Washington and at the University of Loughborough, England, May-June 1997. 30 January 1998
- [14] Jabs, C., Savetz, K. (No date), *Computers and Kids: The Best Kind of On-screen Play* [Online] Sesame Workshop, Available: <http://www.sesameworkshop.org/parents/advice/article/0,4125,74280,00.html> [Accessed: 28 September 2001]
- [15] Technology and Young Children website. (No date), *National Association for the Education of Young Children Technology and Young Children Interest Forum* [Online] Available: <http://www.techandyoungchildren.org/> [Accessed: 14 September 2001]
- [16] Schacter, J. Milken Family Foundation, April 1999, *The Impact of Education Technology on Student Achievement: What the Most Current Research Has to Say* [Online], Milken Exchange on Education Technology, Available: <http://www.mff.org/publications/publications.taf?page=161> [Accessed: 4 October 2001]
- [17] Clements, D, Swaminathan, S. Technology and school change: New lamps for old?. *Childhood Education*. 1995, **71**, 275-281
- [18] Isdale, J. September 1998, *What is Virtual Reality?* [Online] Available: <http://www.isdale.com/jerry/VR/WhatIsVR.html> [Accessed: 22 May 2002]
- [19] Coffin, T. 5 January 1999, *CAVERNUS Image Gallery* [Online] Available: <http://archive.ncsa.uiuc.edu/VR/cavernus/GALLERY/hardware.html> [Accessed: 21 May 2002]
- [20] Institute for Applied Knowledge Processing VR-Lab SAVE, (No date), *Hardware used: Desktop VR* [Online]. Available: <http://www.faw.uni-linz.ac.at/save/>. [Accessed: 21 May 2002]
- [21] Fitzmaurice, G., Buxton, B. Interaction in 3D Graphics: Compatibility and Interaction Style in Computer Graphics. *ACM SIGGRAPH Computer Graphics Newsletter*. 1998, 32 (4)
- [22] 5DT Fifth Dimension Technologies Website. (No date) *5DT Products* [Online] Available: <http://www.5dt.com> [Accessed: 5 July 2002]

- [23] Czernuszenko, M., Pape, D., Sandin, D. DeFanti, T., Dawe, G.L., Brown, M.D. The ImmersaDesk and Infinity Wall Projection-Based Virtual Reality Displays. *Computer Graphics*. 1997, **31** (2), 46-49
- [24] Cruz-Neira, C., Sandin, D.J., DeFanti, T.A., Kenyon, R.V., Hart, J.C. The CAVE: Audio Visual Experience Automatic Virtual Environment. *Communications of the ACM*. 1992, **35** (6), 65-72
- [25] Pape, D., Cruz-Neira, C., Czernuszenko, M. 11 May 1997, *CAVE User's Guide* [Online], Electronic Visualization Laboratory, University of Illinois at Chicago, Available: <http://evlweb.eecs.uic.edu/pape/CAVE/prog/CAVEGuide.html> [Accessed: 22 May 2002]
- [26] Youngblut, C. *Educational Uses of Virtual Reality Technology*. IDA Document D-2128, Alexandria, VA: Institute for Defense Analyses, 1998.
- [27] Argus VR International. (No date) *Educational Profiles, Pond-Eco-System simulator* [Online] <http://www.argusvr.com/focus/profiles/pond.htm> [Accessed: 16 July 2002]
- [28] Struzka, P. 3D Editor for Traffic Playground. *CESCG '99*. Proceedings of the 3rd Central European Seminar on Computer Graphics for students. April 26-27, 1999
- [29] Johnson, A., Roussos, M., Leigh, J., Vasilakis, C., Barnes C., Moher, T. The NICE Project: Learning Together in a Virtual World. *VRAIS '98*. Proceedings of 1998 Virtual Reality Annual International Symposium, Atlanta, Georgia, Mar 14-18, 1998, pp.176-183.
- [30] Moher, T., Johnson, A., Cho, Y. First-Person Science Inquiry in Virtual Ambient Environments. *CHI 2001*. Proceedings of CHI 2001, extended abstracts, Seattle, WA., Mar 31 - Apr 5, 2001, pp. 261-262.
- [31] Johnson, A. Moher, T., Leigh, J., Lin, Ya-Ju. QuickWorlds: Teacher driven VR worlds in an Elementary School Curriculum. *SIGGRAPH 2000 Educators Program*, New Orleans, LA, July 23-28, 2000, Conference Abstracts and Applications pp. 60-63.
- [32] Johnson, A., Moher, T., Ohlsson, S., Gillingham, M., The Round Earth Project: Collaborative VR for Conceptual Learning. *IEEE Computer Graphics and Applications*. **19** (6), 1999, 60-69.
- [33] Salzman, M. C., Dede, C., & Loftin, B. ScienceSpace: Virtual realities for learning complex and abstract scientific concepts. *IEEE Virtual Reality Annual International Symposium (VRAIS '96)*, New York: IEEE Press., 1996, pp. 246-253.
- [34] Salzman, M., Dede, C., Loftin, B. Virtual reality's frames of reference: A visualization technique for mastering abstract information spaces. *CHI '99*. Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems, Pittsburgh, PA, May 15-20, 1999, pp. 489-495.

- [35] Jackson C., Lalioti, V. Virtual cultural identities. *CHI-SA*. Human Computer Interaction in South Africa, in co-operation with ACM SIGCHI, University of Pretoria Conference Centre, Pretoria, South Africa, May 8-10, 2000. [Accessed: 13 June 2001]
- [36] Greef, M., Lalioti, V. Interactive Storytelling with Virtual Identities. *IPT/EGVE 2001*. 5th International Projection Technology Workshop and 7th Eurographics Workshop on Virtual Environments, Stuttgart, Germany, May 16-18, 2001.
- [37] de Bono, E. *Serious Creativity: Using the power of lateral thinking to create new ideas*. Glasgow: Harper Collins Publishers, 1993. p. 137. [Accessed: 13 June 2001]
- [38] Reynolds, C.W. Steering Behaviours For Autonomous Characters. *GDC 1999*. In 1999 Game Developers Conference, San Jose, CA, 1999
- [39] Kfir, R. Virtual Laboratories in Education. *AFRIGRAPH 2001*. Proceedings of the 1st International Conference on Computer Graphics, Virtual Reality and Visualisation in Africa, Camps Bay, Cape Town, South Africa, Nov 5-7, 2001, pp. 27-31
- [40] RTS Game Programming. 1999. *Implementing Rectangular Tiles* [Online]. NAND Industries. Available: <http://www.antimeta.com/projects/rts/tile.html> [Accessed: 13 December 2001]
- [41] Woo, M., Neider, J., Davis, T., Shreiner, D. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 1.2*. 3rd ed. Massachusetts: Addison Wesley, 1999, p. 15.
- [42] Segal, M., Akeley, K. 15 July 1997. *The Design of the OpenGL Graphics Interface* [Online]. Silicon Graphics Computer Systems. Available: http://www.opengl.org/developers/documentation/white_papers/opengl/index.html [Accessed: 10 May 2002]
- [43] Kilgard, J. 13 November 1996, *The OpenGL Utility Toolkit (GLUT) Programming Interface* [Online]. Silicon Graphics, Inc. Available: <http://www.xmission.com/%7Enate/glut.html> [Accessed: 10 May 2002]
- [44] Linux Games. 8 March 2000. *Introduction to OpenAL: Linux Enters the World of 3D Audio* [Online]. Available: http://www.linuxgames.com/articles/openal_intro/ [Accessed: 10 May 2002]
- [45] Loki Entertainment Software. (No date). *About OpenAL* [Online]. Available: <http://www.openal.org/about/> [Accessed: 10 May 2002]
- [46] Nielsen, J. *IBM User Interface Architecture*. Second Edition. International Business Machines Corporation, 2001.
- [47] IBM User Centered Design Home Page. (No date). *User-Centered Design* [Online]. International Business Machines Corporation. Available: http://www-3.ibm.com/ibm/easy/eou_ext.nsf/Publish/570 [Accessed: 13 June 2001]

- [48] IBM Design Concepts Home Page. (No date) *Design Concepts* [Online]. International Business Machines Corporation. Available: http://www-3.ibm.com/ibm/easy/eou_ext.nsf/Publish/567 [Accessed: 13 June 2001]
- [49] Magnee, J., Kramer, J. *Concurrency: State Models and Java Programs*. Wiley, 1999
- [50] Gamma, E., Helm, R., Johnson, R., Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley, 1994
- [51] Wildlife Conservation in Africa. *Elephant Conservation and Management* [Online]. International Foundation for the Conservation of Wildlife. Available: <http://www.wildlife-conservation.org/elephconserv.html> [Accessed: 26 September 2002]
- [52] Canton, J. *Technofutures: How leading-edge technology will transform business in the 21st century*. Carlsbud: Hay House, Inc., 1999, p. 153.

Appendix A

iTiles Class Library

This section presents the class diagram, and description of classes used in the implementation of the iTiles system.



Figure 12 iTiles Class Diagram

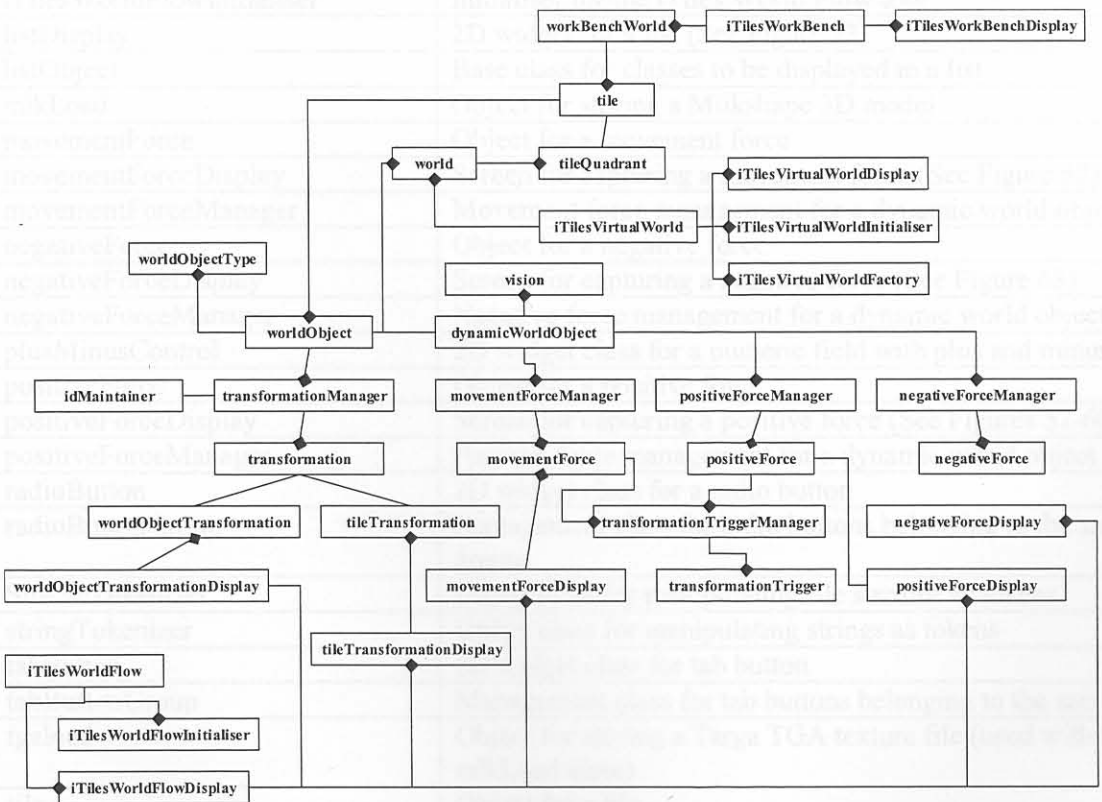
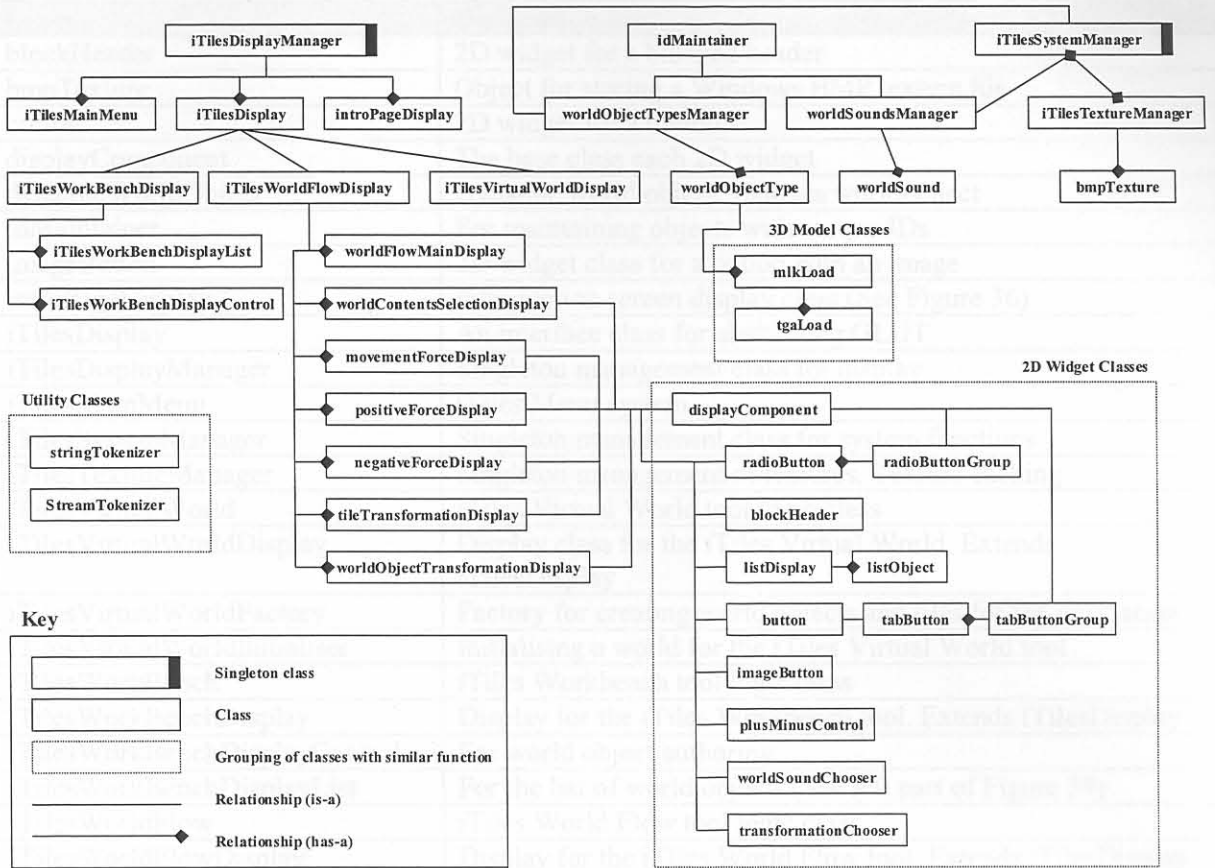


Figure 52. iTiles Class Diagram

Class	Description
blockHeader	2D widget for a blocked header
bmpTexture	Object for storing a Windows BMP texture file
button	2D widget for a button
displayComponent	The base class each 2D widget
dynamicWorldObject	Dynamic world object. Extends worldObject
idMaintainer	For maintaining objects with unique IDs
imageButton	2D widget class for a button with an image
introPageDisplay	Introduction screen display class (See Figure 36)
iTilesDisplay	An interface class for abstracting GLUT
iTilesDisplayManager	Singleton management class for display
iTilesMainMenu	iTiles Menu system
iTilesSystemManager	Singleton management class for system functions
iTilesTextureManager	Singleton management of textures. Texture caching
iTilesVirtualWorld	iTiles Virtual World tool logic class
iTilesVirtualWorldDisplay	Display class for the iTiles Virtual World. Extends iTilesDisplay
iTilesVirtualWorldFactory	Factory for creating world objects and tiles for the simulation
iTilesVirtualWorldInitialiser	Initialising a world for the iTiles Virtual World tool
iTilesWorkBench	iTiles Workbench tool logic class
iTilesWorkBenchDisplay	Display for the iTiles Workbench tool. Extends iTilesDisplay
iTilesWorkBenchDisplayControl	For world object authoring
iTilesWorkBenchDisplayList	For the list of world objects (See left part of Figure 39)
iTilesWorldFlow	iTiles World Flow tool logic class
iTilesWorldFlowDisplay	Display for the iTiles World Flow tool. Extends iTilesDisplay
iTilesWorldFlowInitialiser	Initialiser for the iTiles World Flow tool
listDisplay	2D widget for a list (See Figure 23)
listObject	Base class for classes to be displayed in a list
mlkLoad	Object for storing a Milkshape 3D model
movementForce	Object for a movement force
movementForceDisplay	Screen for capturing a movement force (See Figure 62)
movementForceManager	Movement force management for a dynamic world object
negativeForce	Object for a negative force
negativeForceDisplay	Screen for capturing a negative force (See Figure 68)
negativeForceManager	Negative force management for a dynamic world object
plusMinusControl	2D widget class for a numeric field with plus and minus
positiveForce	Object for a positive force
positiveForceDisplay	Screen for capturing a positive force (See Figures 57-60)
positiveForceManager	Positive force management for a dynamic world object
radioButton	2D widget class for a radio button
radioButtonGroup	Management class for radio buttons belonging to the same group
StreamTokenizer	Utility class for manipulating file streams as tokens
stringTokenizer	Utility class for manipulating strings as tokens
tabButton	2D widget class for tab button
tabButtonGroup	Management class for tab buttons belonging to the same group
tgload	Object for storing a Targa TGA texture file (used with the mlkLoad class)
tile	Object for a tile
tileQuadrant	A tileQuadrant extends tile, for use in the iTiles Virtual World tool
tileTransformation	Object for a tile transformation. Extends transformation

tileTransformationDisplay	Screen for capturing a tile transformation (See Figure 66)
transformation	Object for a transformation
transformationManager	Transformation management for world objects and tiles
transformationsChooser	A custom 2D widget for choosing transformations
transformationTrigger	Object for a transformation trigger
transformationTriggerManager	Management of transformation triggers
vision	Vision of a dynamic world object
workBenchWorld	A world for the iTiles Workbench tool
world	A world for the iTiles Virtual World tool
worldContentsSelectionDisplay	iTiles World Flow utility screen (See Figure 56, 61, 63, 65, or 67)
worldFlowMainDisplay	iTiles World Flow screen for main selection (See Figure 53, 54 or 55)
worldObject	Object for an iTiles world object
worldObjectTransformation	Transformation for a world object. Extends transformation
worldObjectTransformationDisplay	Screen for capturing a world object transformation (See Figure 64)
worldObjectType	World object type class
worldObjectTypesManager	Management of world object types in an iTiles system (interface for world objects)
worldSound	Representing a WAV file
worldSoundChooser	Custom 2D widget for selecting world sounds. (See Figure 23)
worldSoundsManager	Management of world sounds in an iTiles system (interface for world sounds)

Table 4: iTiles classes description

Appendix B

iTiles World Flow tool screens

This section presents screenshots of screens of the iTiles World Flow tool. The screenshots presented in Figure 53 – 66 have been captured from the iTiles World Flow tool for the iTiles World Flow of the Drought in Africa virtual laboratory presented in Chapter 5.

Figure 23. Dynamic world objects list (iTiles World Flow tool screen)



Figure 24. Static world objects list (iTiles World Flow tool screen)

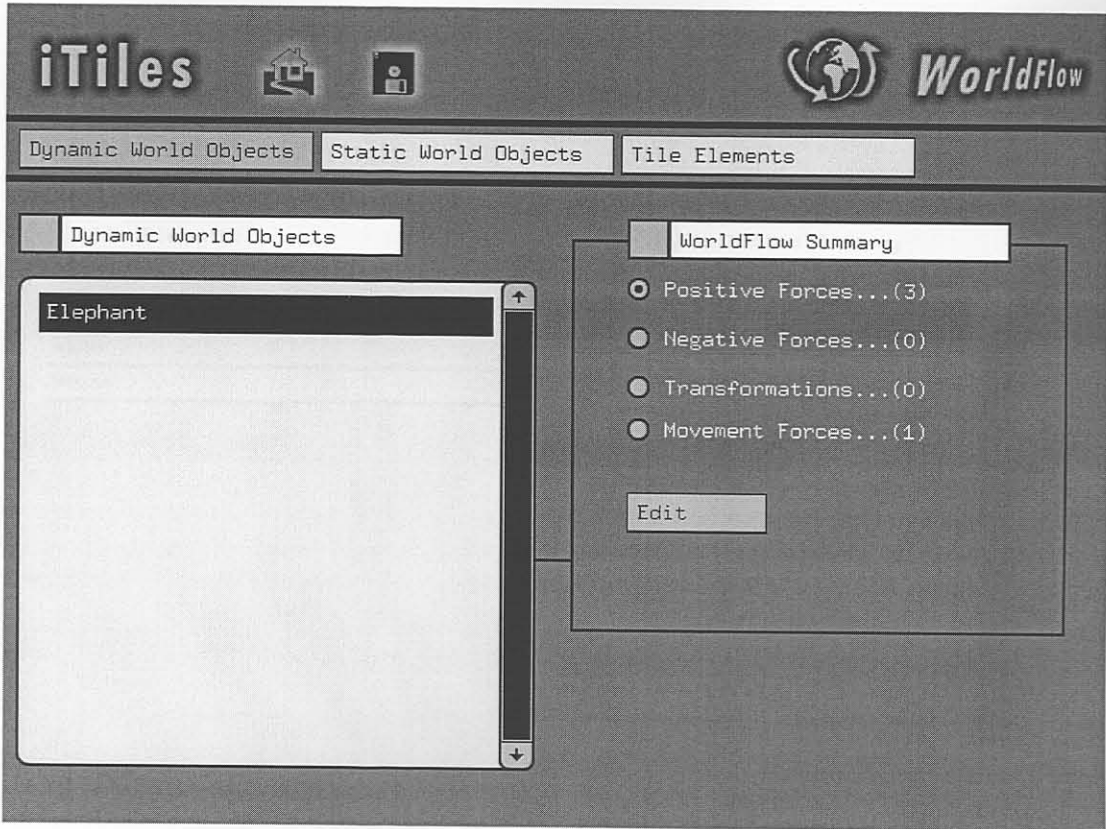


Figure 53. Dynamic world objects list (iTiles World Flow tool screen)

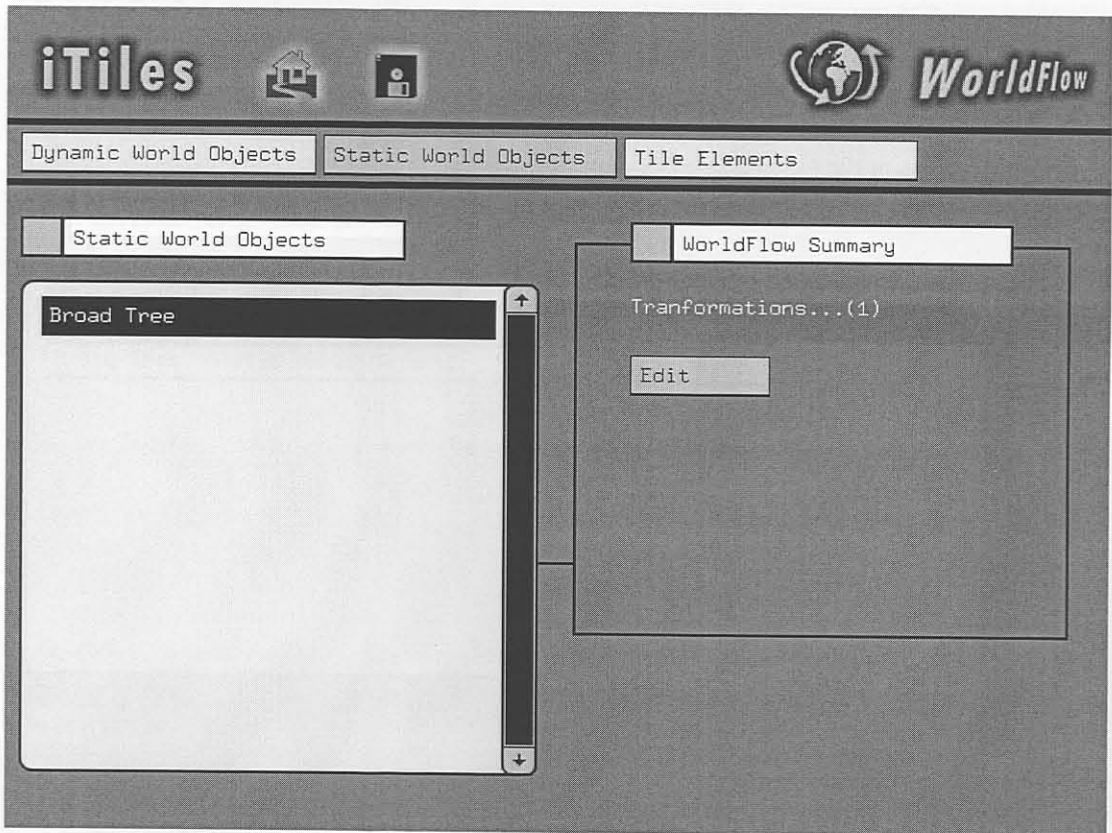


Figure 54. Static world objects list (iTiles World Flow tool screen)

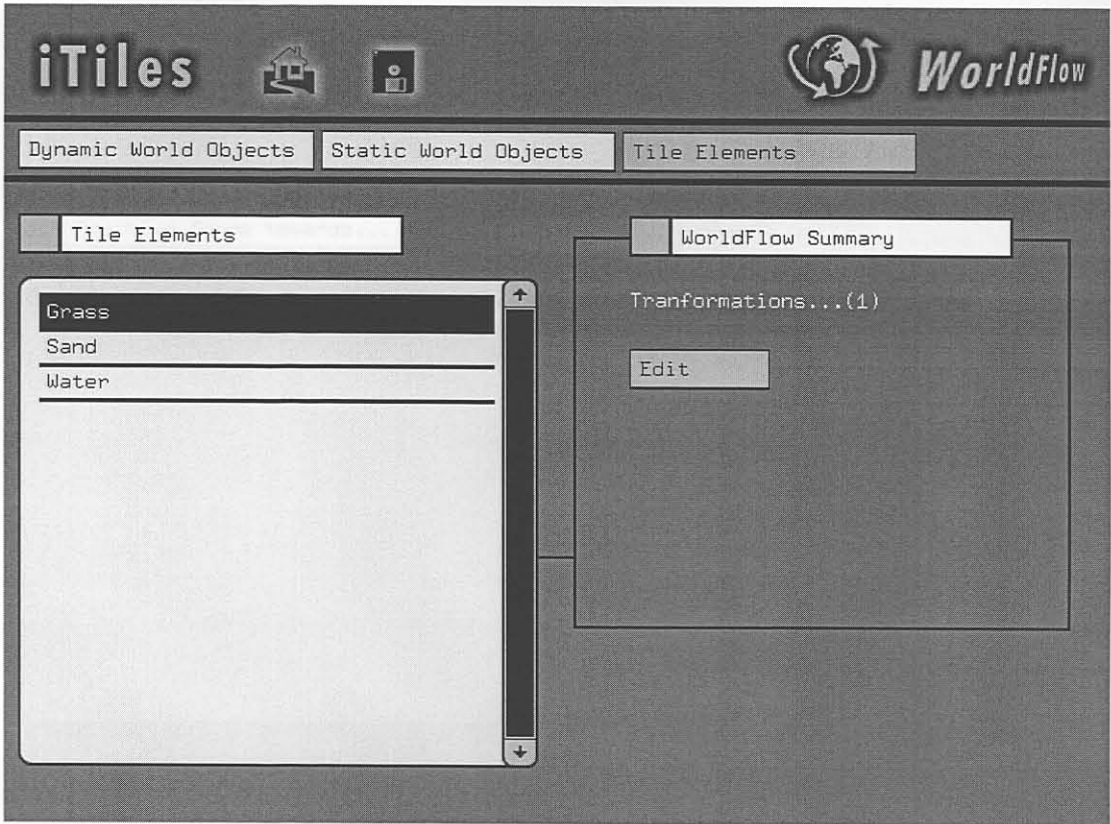


Figure 55. Tile elements list (iTiles World Flow tool screen)

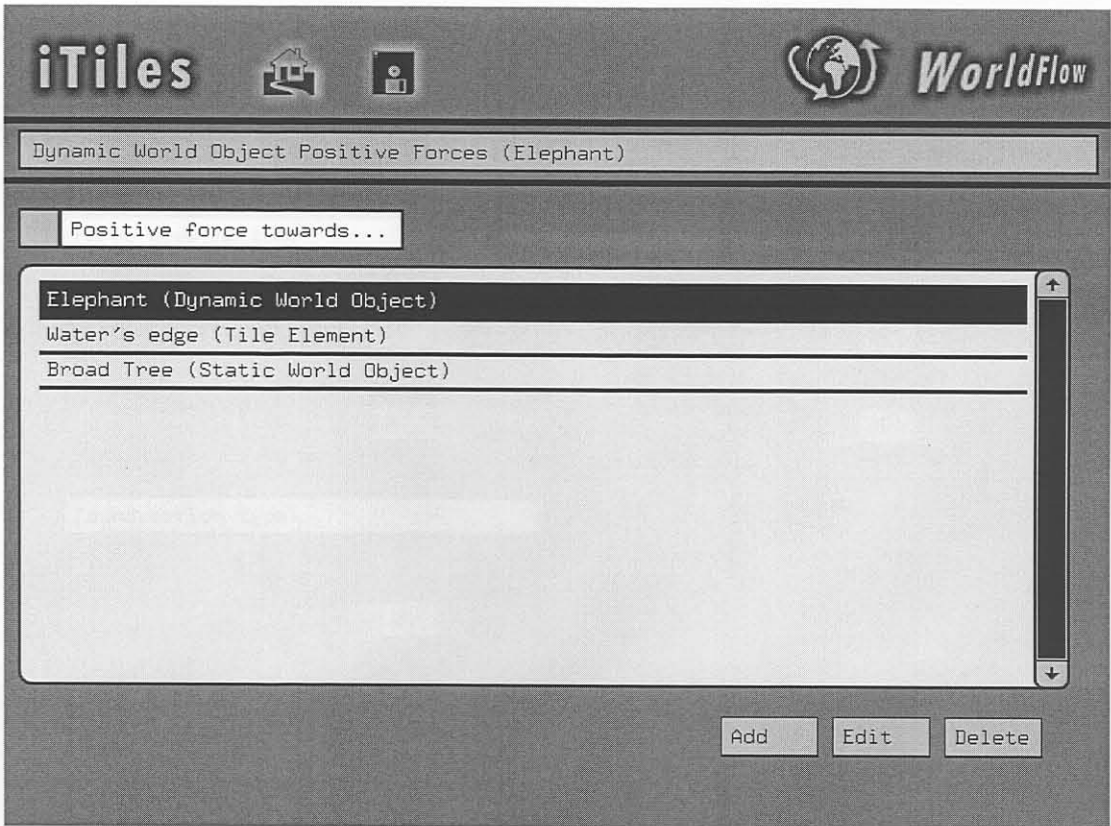


Figure 56. Positive forces list of a dynamic world object (iTiles World Flow tool screen)

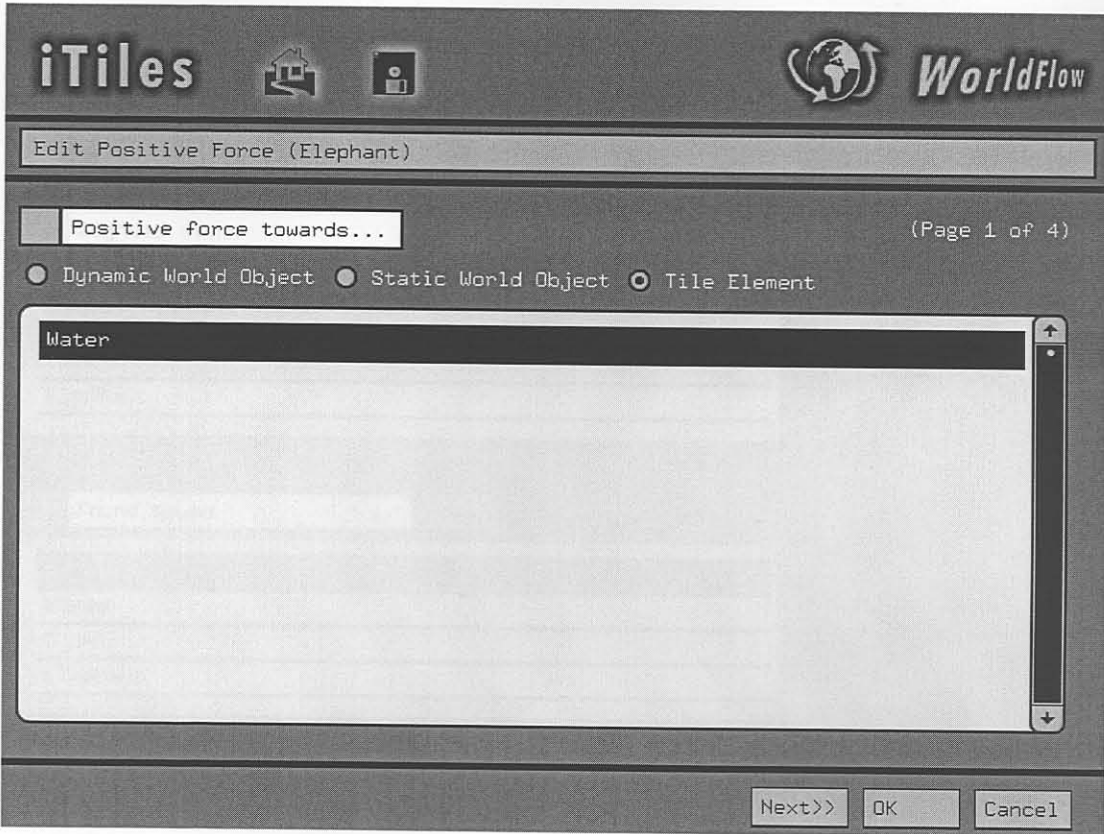


Figure 57. Positive force of a dynamic world object - Page 1 (iTiles World Flow tool screen)

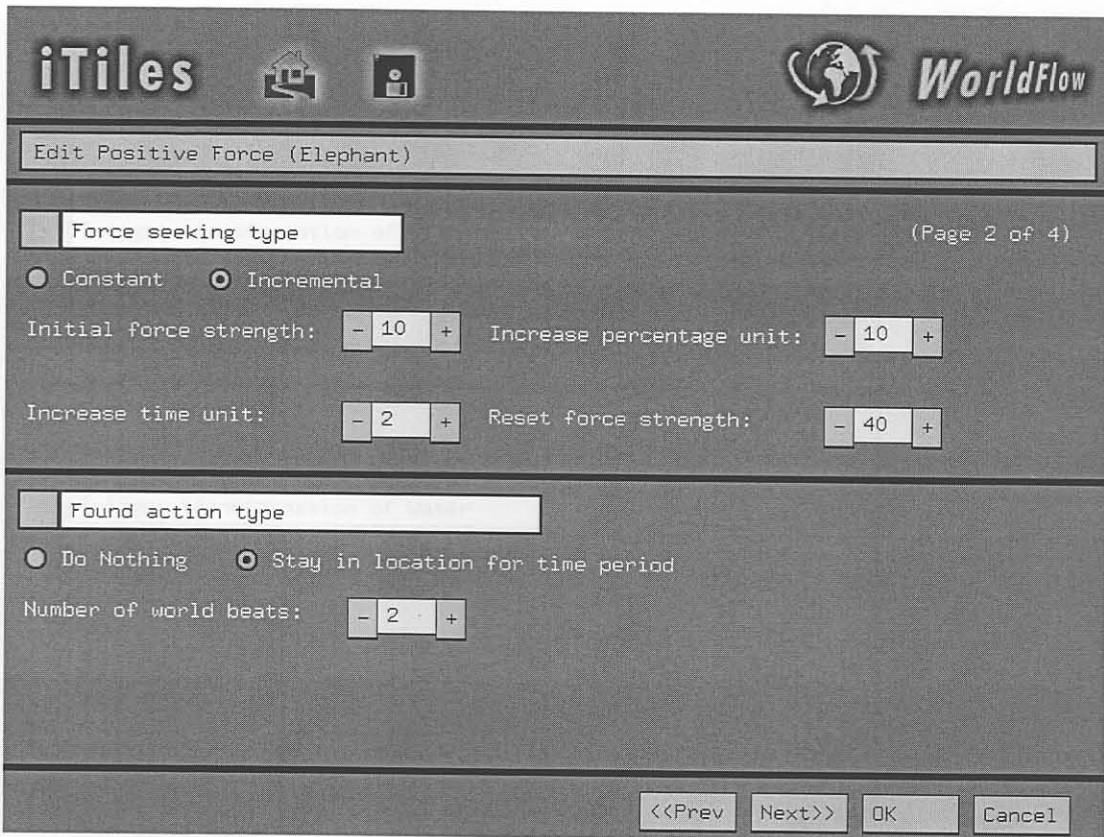


Figure 58. Positive force of a dynamic world object – Page 2 (iTiles World Flow tool screen)

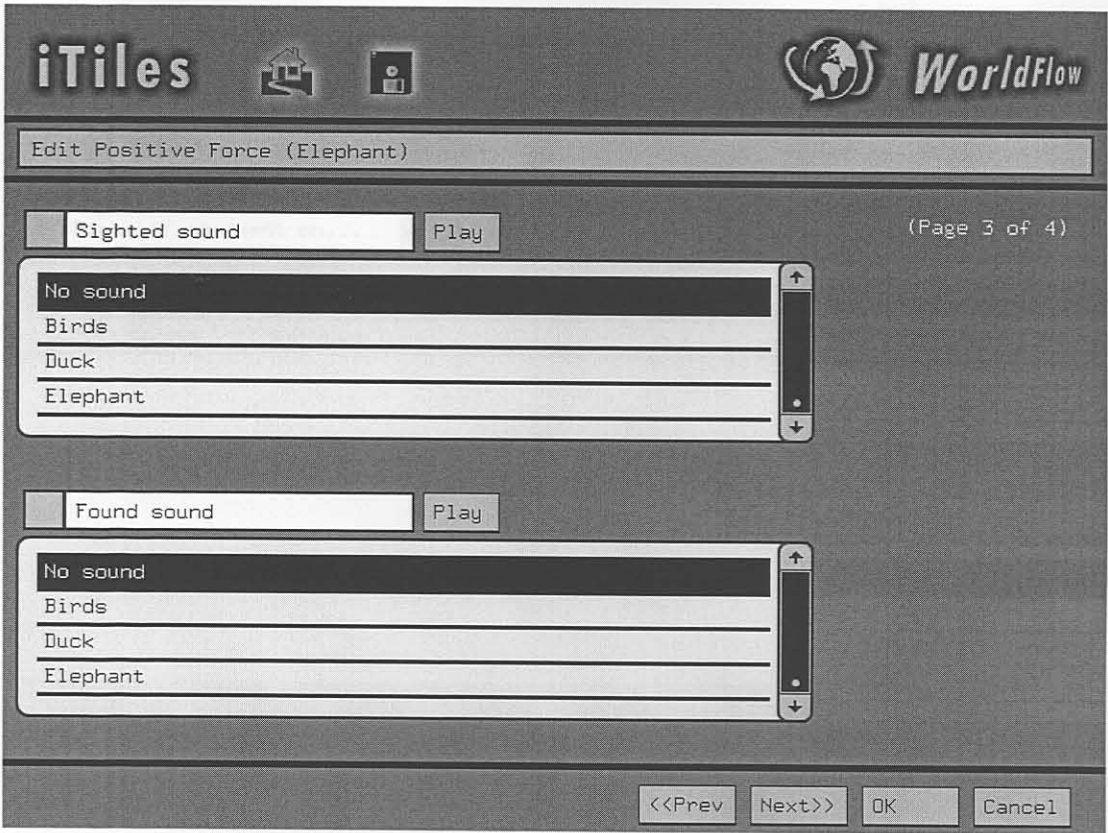


Figure 59. Positive force of a dynamic world object – Page 3 (iTiles World Flow tool screen)

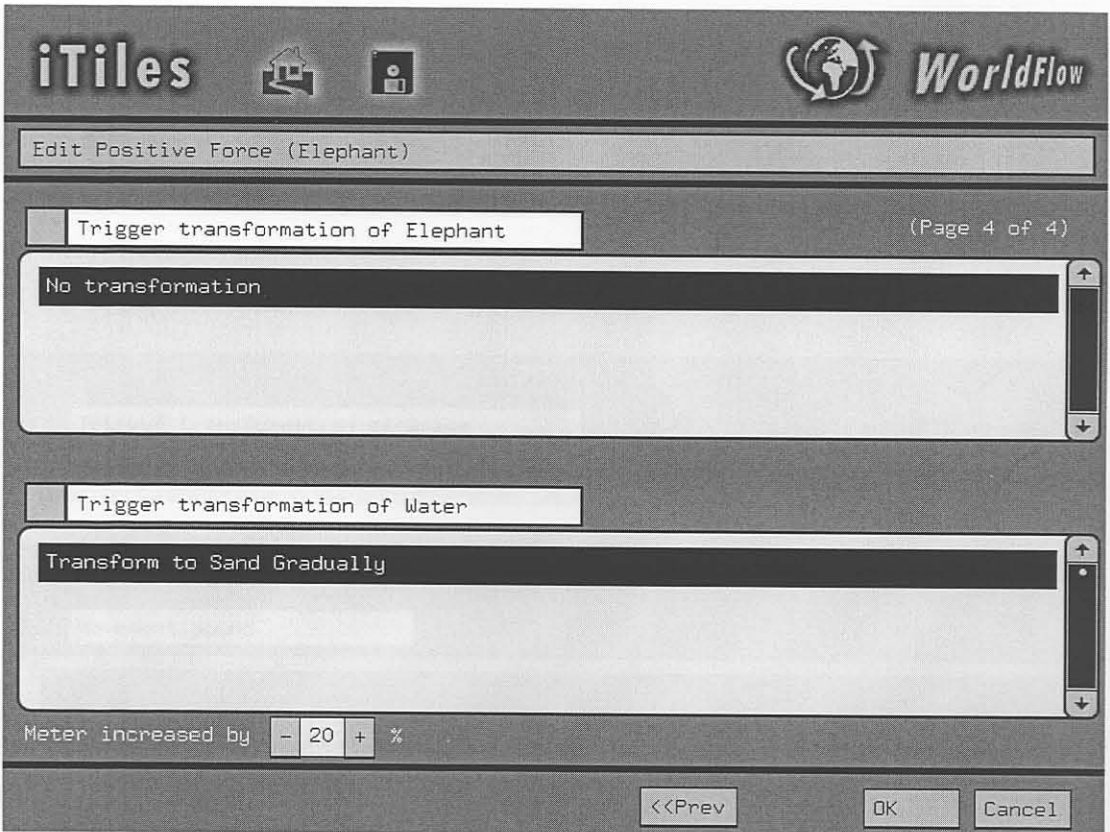


Figure 60. Positive force of a dynamic world object – Page 4 (iTiles World Flow tool screen)

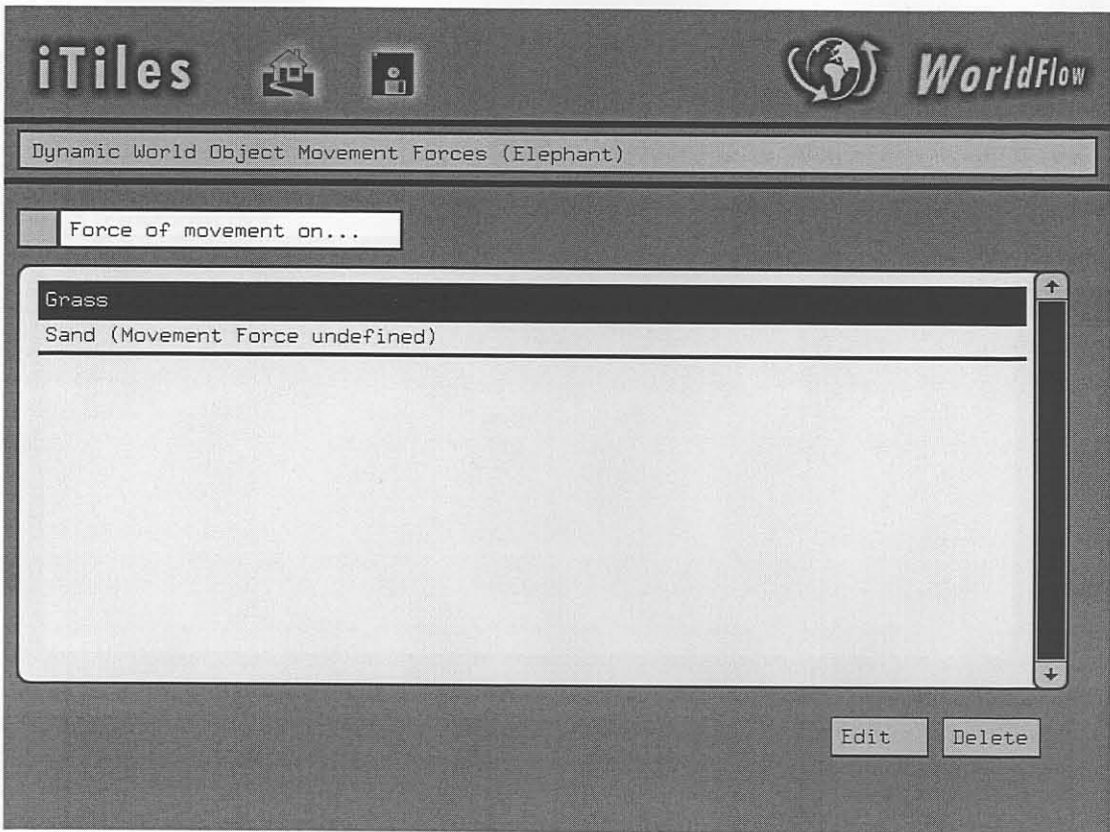


Figure 61. Movement forces list of a dynamic world object (iTiles World Flow tool screen)

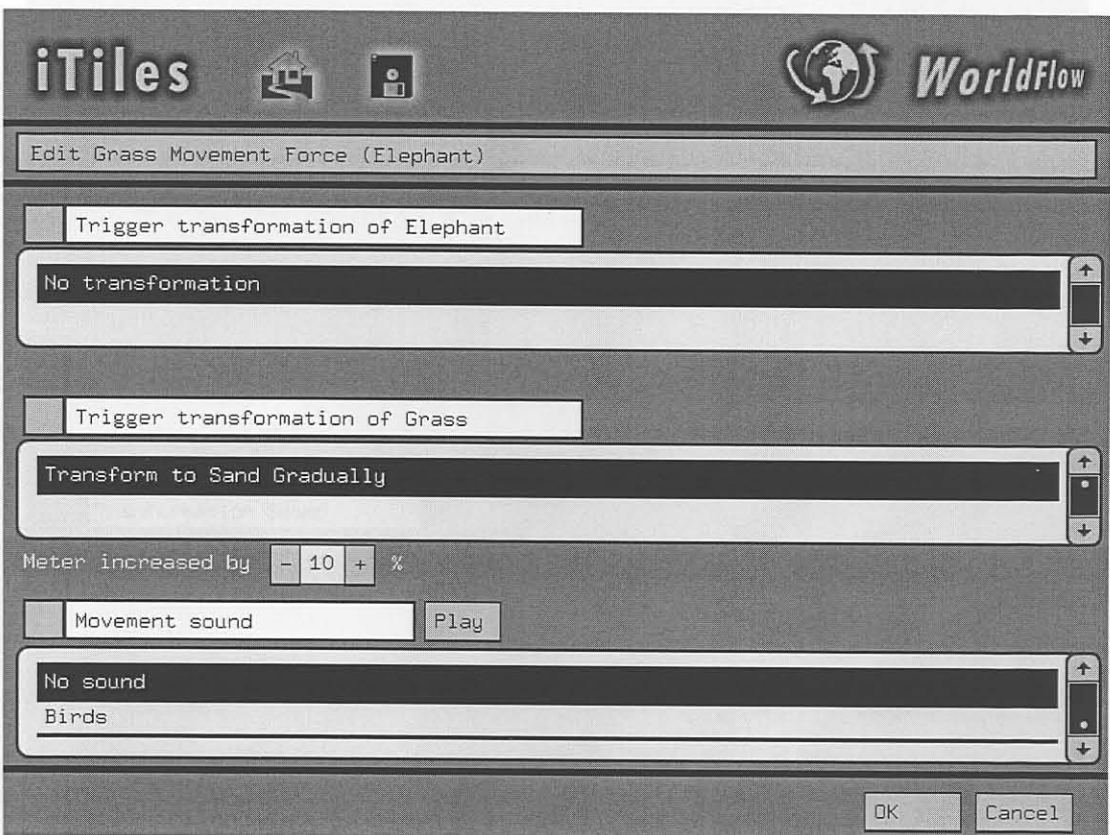


Figure 62. Movement force of a dynamic world object (iTiles World Flow tool screen)

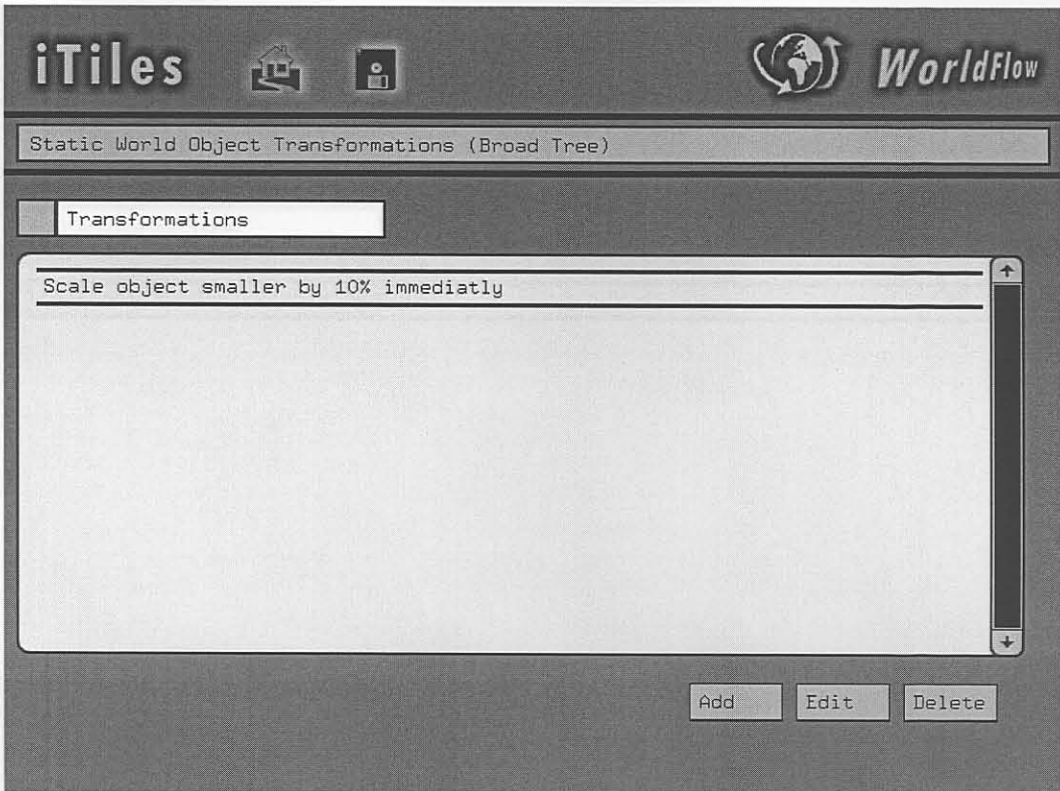


Figure 63. World object transformations list of a static world object (iTiles World Flow tool screen)

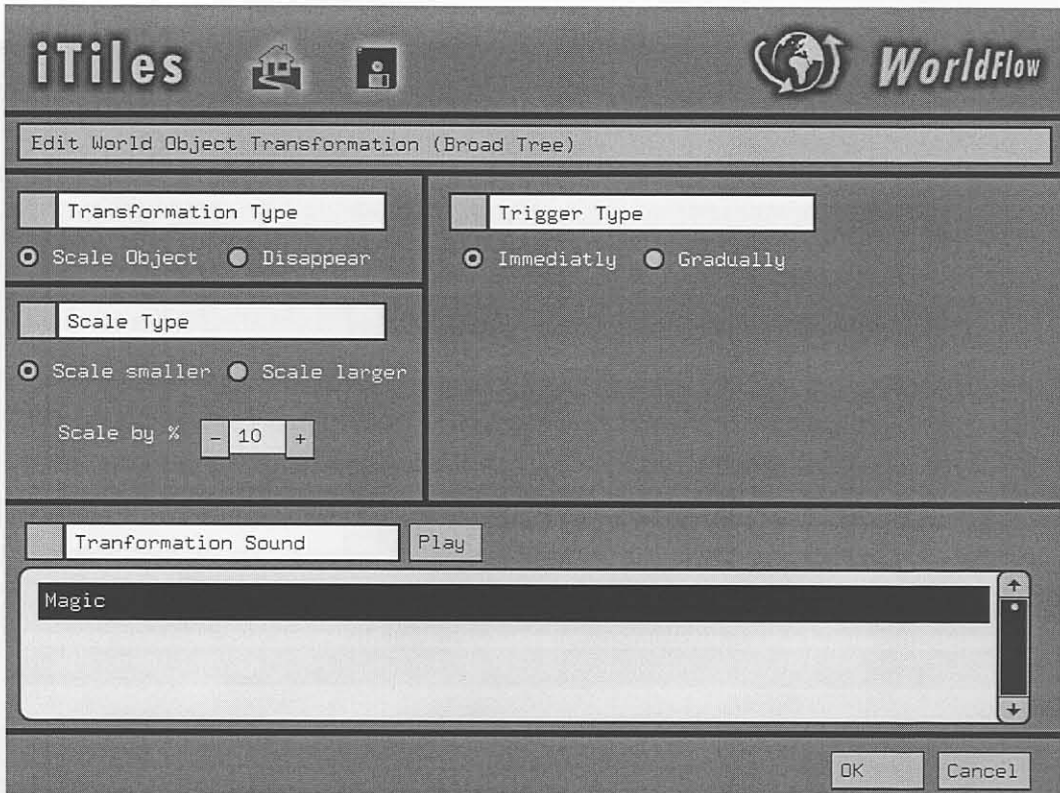


Figure 64. World object transformation of a static world object (iTiles World Flow tool screen)

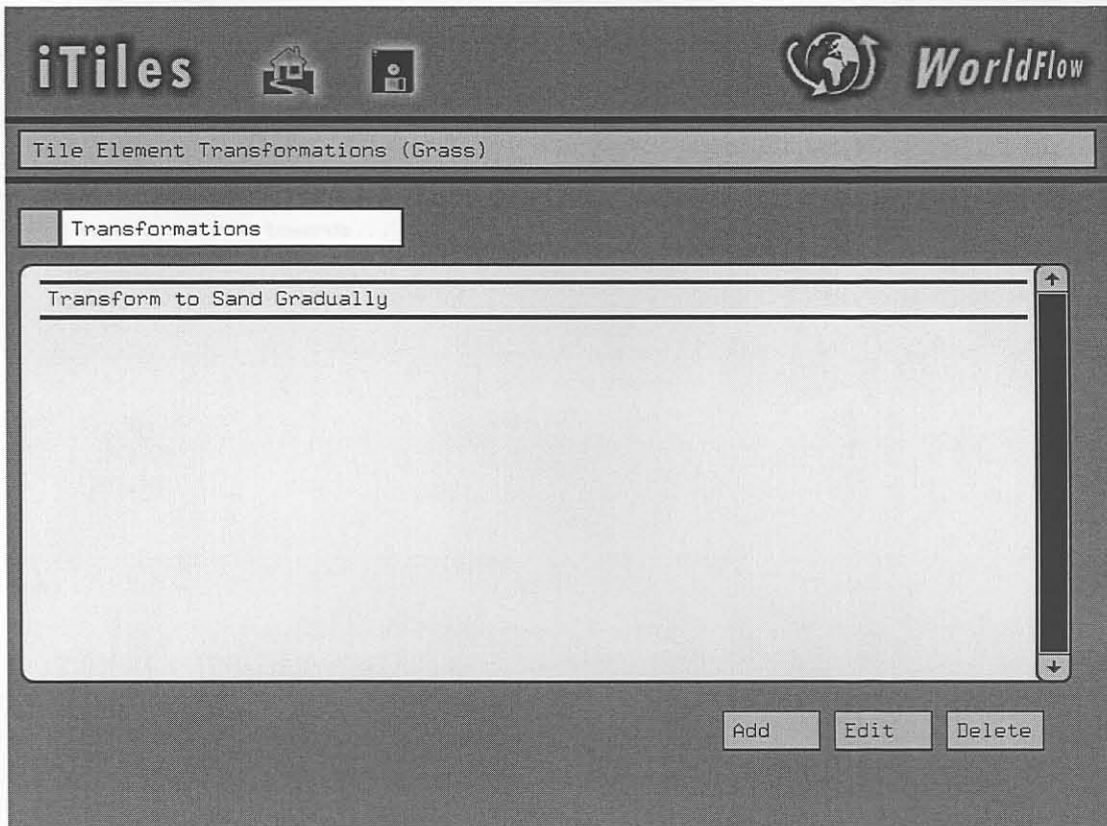


Figure 65. Tile element transformations list of a tile element (iTiles World Flow tool screen)

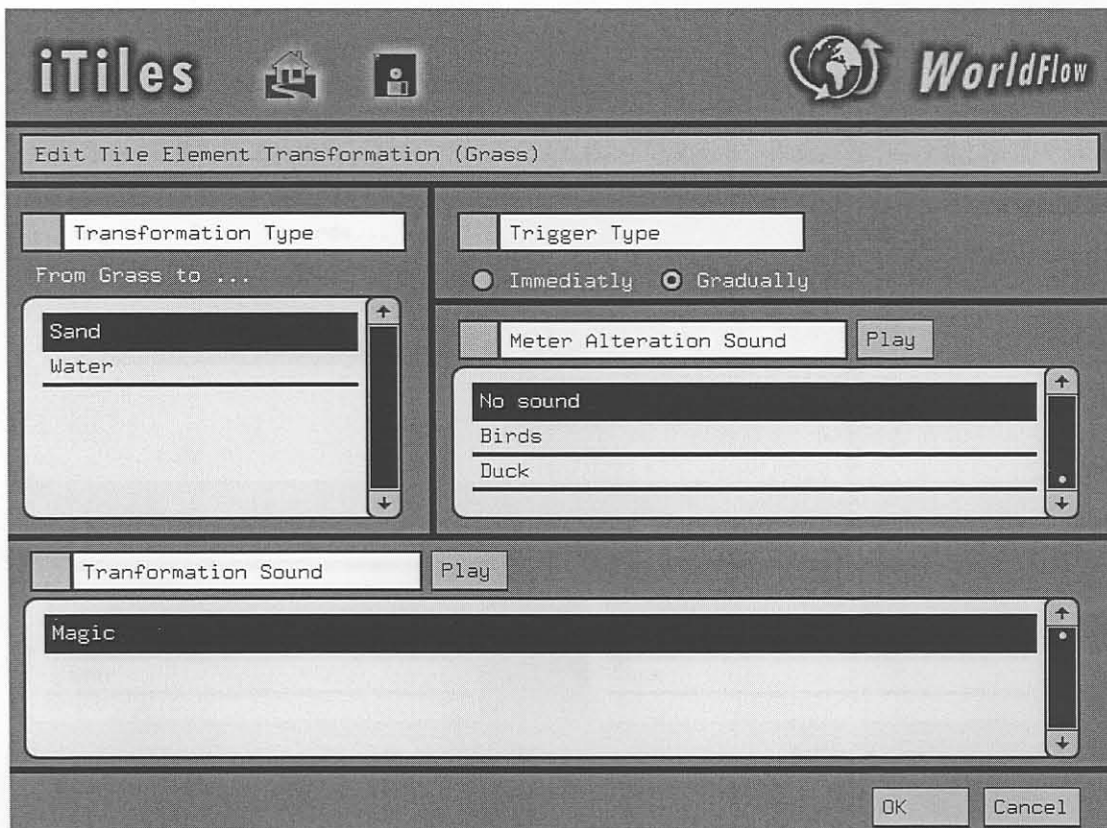


Figure 66. Tile element transformation of a tile element (iTiles World Flow tool screen)

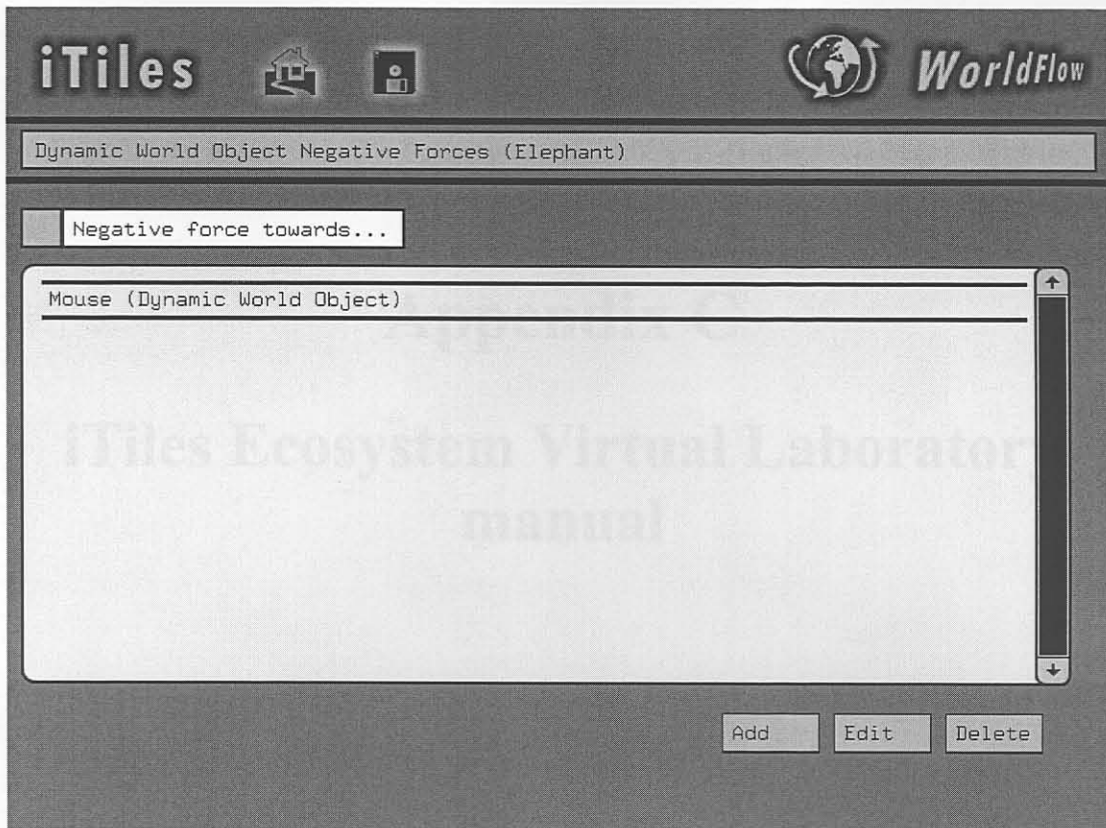


Figure 67. Negative forces list of a dynamic world object (iTiles World Flow tool screen)

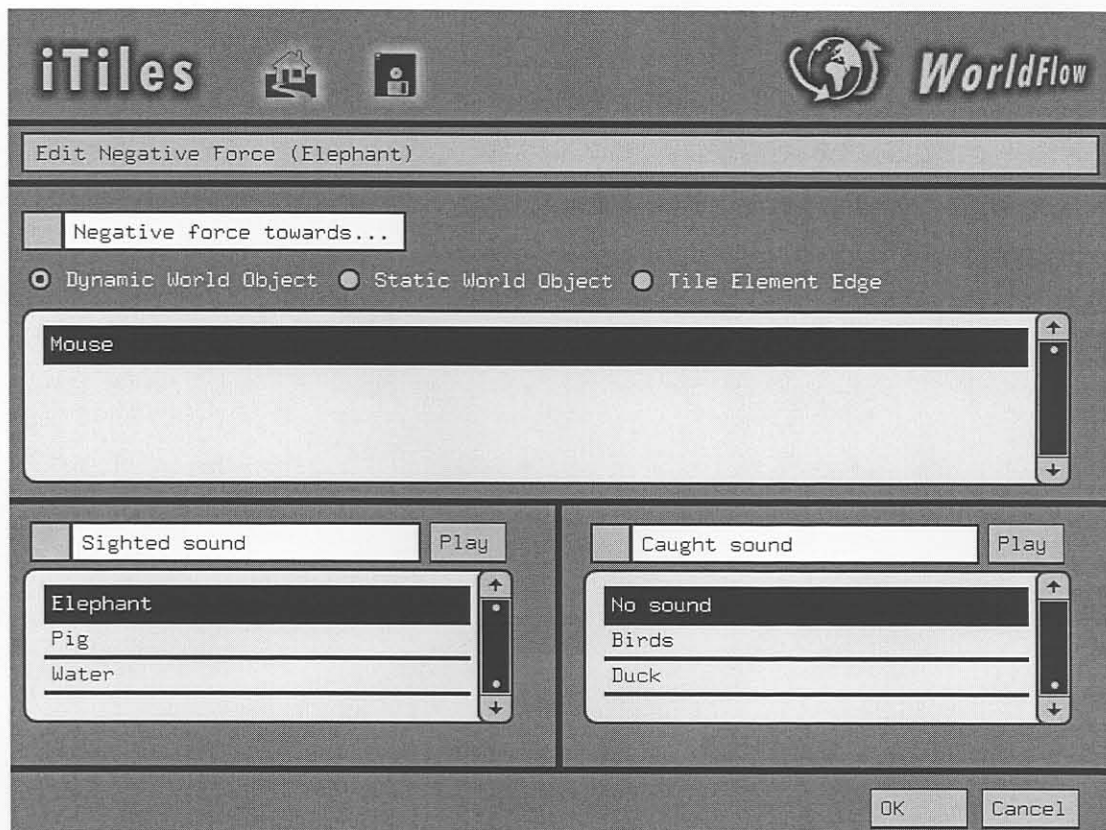


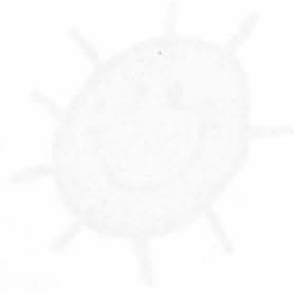
Figure 68. Negative force of a dynamic world object (iTiles World Flow tool screen)

Appendix C

iTiles Ecosystem Virtual Laboratory manual

A guide to using the iTiles
Ecosystem Virtual Laboratory

This section presents the manual for the iTiles Ecosystem Virtual Laboratory application.

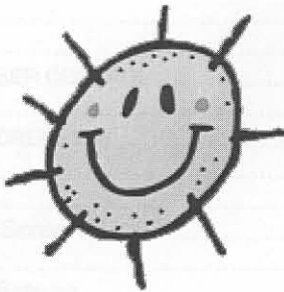


INSTALLATION INSTRUCTIONS	3
STARTING THE PROGRAM	3
USING THE PROGRAM	3
THE INTRODUCTION SCREEN	3
ACTIVATING THE MENU	4
Animation speed	5
USING THE ITILES WORKBENCH	5
NAVIGATION IN THE ENVIRONMENT	5
Selection indication	5
Navigation	5
CAMERA NAVIGATION	6
TILE EDITING MODE	7
Flipping tile elements	8
Growing and erasing the world	8
World Object Actions	9
Selecting a world object	10
The Tap	10
World object auto-rotation	11
SAVING THE ITILES WORKBENCH	11
USING THE ITILES WORLD FLOW TOOL	12
MAIN NAVIGATION RESPONSIBILITIES	12
SAVING THE WORLD FLOW	13
SELECTION SCREEN	14
LOADING THE WORLD FLOW CHOICES	15
LUTS AND FS	15
EXPLANATION OF WORLD FLOW SCREENS	15
Positive Force Screen	16
Negative Force Screen	17
World Object Transformation Screen	17
Movement Force Screen	17
Tile Element Transformation Screen	18
A NOTE ABOUT TRANSFORMATION TRIGGERS	16
USING THE ITILES VIRTUAL WORLD TOOL	19
CAMERA NAVIGATION	19
NAVIGATING WORLD OBJECTS	19
FIRST AND THIRD PERSON VIEWS	20
STARTING AND STOPPING THE SIMULATION	20
SPEEDING UP AND SLOWING DOWN TIME	20
LIVES AND DRUPEFS	21
CHARACTER VIEWFN	21
CONTROLLING A CHARACTER	21
ITILES SETUP AND CONFIGURATION FILES	22
EXTENDING THE ITILES SYSTEM	22
Adding ADDITIONAL WORLD OBJECTS	22
Adding static world objects	23
Adding dynamic world objects	23
Adding ADDITIONAL SOUNDS	23
FILES AND DIRECTORIES FOUND IN THE INSTALLATION DIRECTORY	23
SAMPLE WORLDS	24
ITILES WORLD 1: DUCKS IN THE PARK	24
ITILES WORLD 2: DROUGHT IN AFRICA	24
ITILES WORLD 3: PARADISE ISLAND	26

iTiles

Intelligent Tiles

A guide to using the iTiles Ecosystem Virtual Laboratory



INSTALLATION INSTRUCTIONS	3
STARTING THE PROGRAM	3
USING THE PROGRAM	3
THE INTRODUCTION SCREEN.....	3
ACTIVATING THE MENU.....	4
Animation speed.....	5
USING THE ITILES WORKBENCH TOOL	5
NAVIGATION IN THE ENVIRONMENT.....	5
Selection indication.....	5
Navigation.....	5
CAMERA NAVIGATION.....	6
TILE EDITING MODE.....	7
Editing tile elements.....	8
Growing and shrinking the world.....	8
WORLD OBJECT AUTHORING MODE.....	9
Selecting a world object.....	10
The Tazo.....	10
World object authoring.....	11
SAVING THE ITILES WORKBENCH.....	11
USING THE ITILES WORLD FLOW TOOL	12
MAIN NAVIGATION TECHNIQUES.....	12
SAVING THE WORLD FLOW.....	13
SELECTION SCREEN.....	14
USING THE WORLD SOUND CHOOSER CONTROL.....	15
LISTS IN ITILES.....	15
EXPLANATION OF WORLD FLOW SCREENS.....	15
Positive Force Screen.....	15
Negative Force Screen.....	17
World Object Transformation Screen.....	17
Movement Force Screen.....	17
Tile Element Transformation Screen.....	18
A NOTE ABOUT TRANSFORMATION TRIGGERS.....	18
USING THE ITILES VIRTUAL WORLD TOOL	19
CAMERA NAVIGATION.....	19
NAVIGATING WORLD OBJECTS.....	19
FIRST AND THIRD PERSON VIEWS.....	20
STARTING AND STOPPING THE SIMULATION.....	20
SPEEDING UP AND SLOWING DOWN TIME.....	20
LIKES AND DISLIKES.....	21
CHARACTER VISION.....	21
CONTROLLING A CHARACTER.....	21
ITILES SETUP AND CONFIGURATION FILES	22
EXTENDING THE ITILES SYSTEM.....	22
ADDING ADDITIONAL WORLD OBJECTS.....	22
Adding static world objects.....	22
Adding dynamic world objects.....	23
ADDING ADDITIONAL SOUNDS.....	23
FILES AND DIRECTORIES FOUND IN THE INSTALLATION DIRECTORY.....	23
SAMPLE WORLDS	24
ITILES WORLD 1: DUCKS IN THE PARK.....	24
ITILES WORLD 2: DROUGHT IN AFRICA.....	25
ITILES WORLD 3: PARADISE ISLAND.....	26

Installation instructions

The iTiles Ecosystem Virtual Laboratory application requires about 30Mb free hard drive space. On the CD please execute the installation program <install.exe> which can be found in the <Installation> directory. This will launch the WinZip Self-Extractor application that will prompt you to which folder you wish to install iTiles (the default is c:\iTiles).

Starting the program

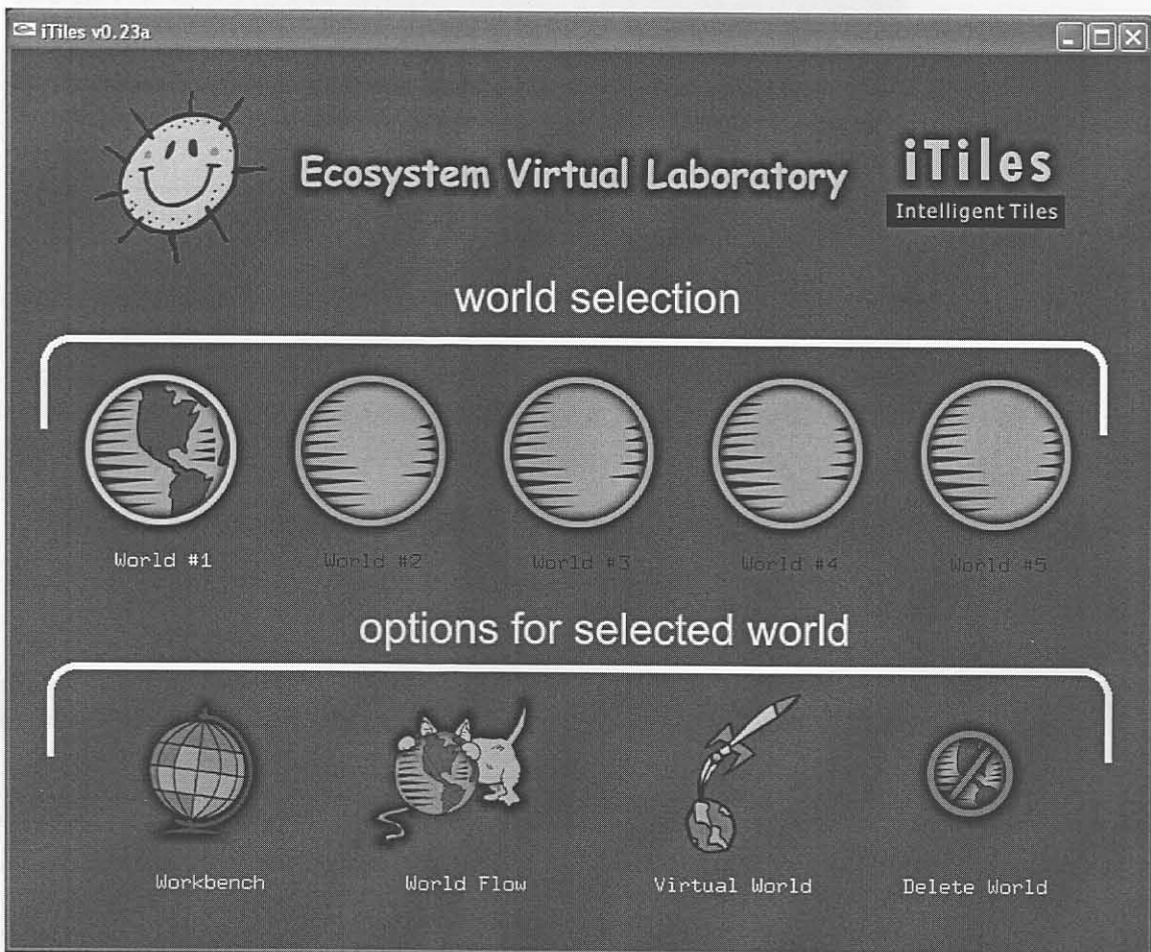
To start the program run the executable <iTiles.exe> found in the directory of installation.

All DLLs required for the execution of iTiles can be found on the installation CD in the directory named <Installation/DLL>. If you are experiencing problems starting the program, please copy the required DLL from this directory to the directory where you installed iTiles.

Using the program

The introduction screen

When the application is launched the introduction screen is shown, allowing you to choose which iTiles world you wish to work with during your iTiles session. The selected world will be highlighted with a yellow circle (as shown in the image below with World #1 being selected).



There are five world slots available in the iTiles system. If a world slot is empty, it is indicated with an empty world (in the image above, World #2-5 are empty). To clear a world slot, you can press the delete world button in the options section for the selected world.

The function buttons will be coloured red and appear as negatives if they are not available.

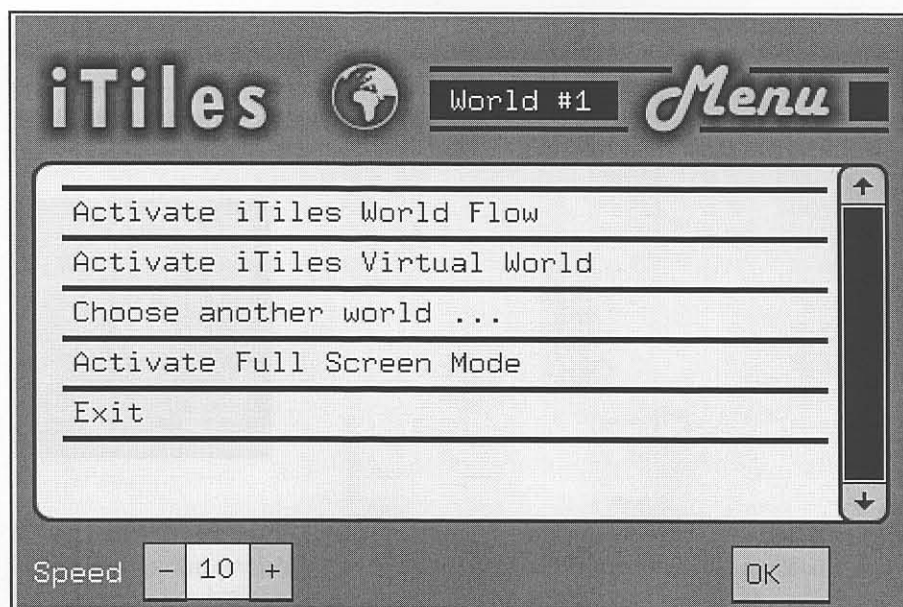
An iTiles world requires that the iTiles world first be authored using the iTiles Workbench tool before proceeding with the iTiles World Flow tool. Once the World Flow has been saved for that iTiles world, you activate the iTiles Virtual World simulation.

The text below the Workbench and World Flow functions will be coloured yellow if the world already has these attributes, and white if it doesn't and the function is available to proceed with (e.g.: If a world already has an authored world defined in a workbench, the text will be yellow)

Activating the menu

The iTiles menu is available at any time during the program.

Press the <Esc> key on the keyboard to activate the menu. Press <Esc> again to make it disappear if you do not wish to use the menu. For instructions on how to change the animation speed value please see below.



The menu also displays the currently selected world slot (in the image above, world slot #1 is selected).

The following functions are available from the menu. Please note that some options may not be available, since they are dependent on the program's current state.

Function	Description
Activate iTiles Workbench	Launches the iTiles Workbench tool
Activate iTiles World Flow	Launches the iTiles World Flow tool
Activate Virtual World	Launches the iTiles Virtual World simulation tool
Choose another world ...	Takes you back to the introduction screen where you can choose another iTiles world
Activate Full Screen Mode	Make the application fill the whole screen
Activate Windowed Mode	Application is enclosed within a window
Exit	Exit the iTiles Ecosystem Virtual Laboratory

Animation speed

The animation speed value is used in a camera smoothing function for the animation of smooth transitions in the 3D virtual environment. The smoothing function is dependent on how fast a computer's processor is, and whether or not the PC has a 3D accelerator card. The larger the animation speed value, the smoother the camera animation will be. For a slow computer without a 3D accelerator, a low animation speed value is recommended.

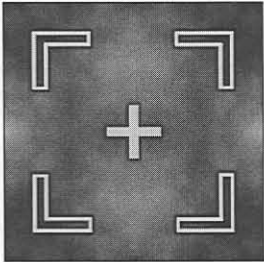
To change the animation speed, press <ESC> to display the menu, change the speed value and press <ESC> again to make the menu disappear (If you don't do this and press the OK button, an option in the list may have been selected and you'll turn up where you don't want to be).

Using the iTiles Workbench tool

Navigation in the environment

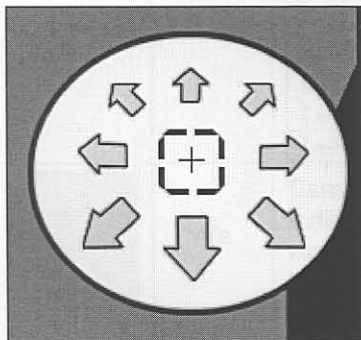
Selection indication

The selected tile in the environment is indicated by a crosshair. In camera tracking mode (see below in the camera navigation section), the selected tile will always appear in the centre of the screen. When world object authoring mode is activated a white wire frame cube will be drawn around the selected world object.



Navigation







Navigation in the environment is done by clicking the arrows indicated in the image below, or by using the equivalent arrow key on the keyboard.



Camera navigation

The view of the world can be changed. The world's axes can be changed and functionality for zooming in and out is available.

The following table explains the different camera modes available. The cursor will change when certain camera modes are activated.

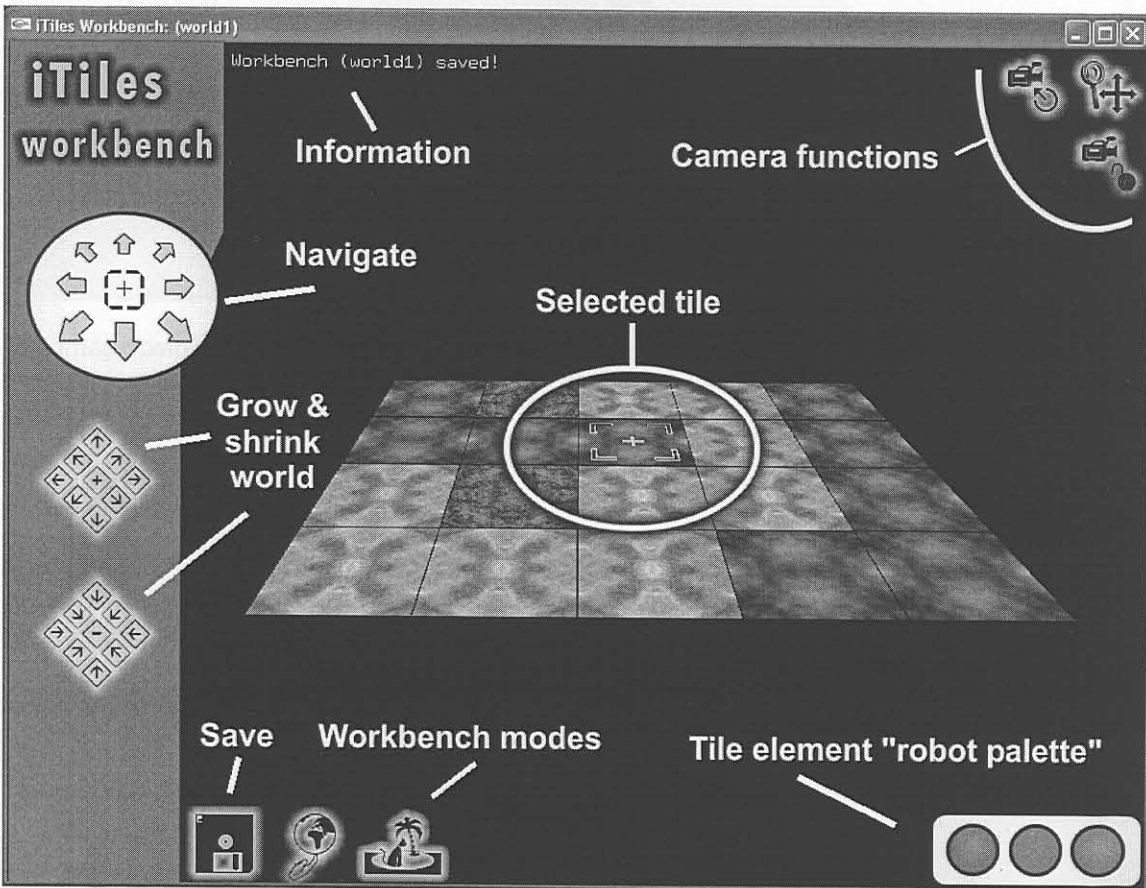
Button	Cursor	Description																					
		<p>Camera Rotation Click this button to enable Camera Rotation mode. When this mode is activated the image will be blue and the mouse cursor will change as indicated. To deactivate the mode, re-click the button.</p> <p>The following mouse movements influence a transformation in the camera's position and are achieved by keeping the mouse button pressed and moving the mouse in a particular direction (drag the mouse gently and slowly in the desired direction).</p> <table border="1"> <thead> <tr> <th>Mouse button</th> <th>Mouse movement</th> <th>Result</th> </tr> </thead> <tbody> <tr> <td>Right</td> <td>Left</td> <td>World rotates left</td> </tr> <tr> <td>Right</td> <td>Right</td> <td>World rotates right</td> </tr> <tr> <td>Left</td> <td>Up</td> <td>World tilts up</td> </tr> <tr> <td>Left</td> <td>Down</td> <td>World tilts down</td> </tr> <tr> <td>Left</td> <td>Left</td> <td>World moves left</td> </tr> <tr> <td>Left</td> <td>Right</td> <td>World moves right</td> </tr> </tbody> </table>	Mouse button	Mouse movement	Result	Right	Left	World rotates left	Right	Right	World rotates right	Left	Up	World tilts up	Left	Down	World tilts down	Left	Left	World moves left	Left	Right	World moves right
Mouse button	Mouse movement	Result																					
Right	Left	World rotates left																					
Right	Right	World rotates right																					
Left	Up	World tilts up																					
Left	Down	World tilts down																					
Left	Left	World moves left																					
Left	Right	World moves right																					
		<p>Zoom and panning This is for zooming and panning the world. When activated, the button colouring will be blue and the mouse cursor will change as indicated. To deactivate the mode, re-click the button.</p> <p>The following mouse movements influence a transformation in the camera's position and are achieved by keeping the mouse button pressed and moving the mouse in a particular direction (drag the mouse gently and slowly in the desired direction).</p> <table border="1"> <thead> <tr> <th>Mouse button</th> <th>Mouse movement</th> <th>Result</th> </tr> </thead> <tbody> <tr> <td>Right</td> <td>Up</td> <td>Zoom in</td> </tr> <tr> <td>Right</td> <td>Down</td> <td>Zoom out</td> </tr> <tr> <td>Left</td> <td>Up</td> <td>World moves up</td> </tr> <tr> <td>Left</td> <td>Down</td> <td>World moves down</td> </tr> <tr> <td>Left</td> <td>Left</td> <td>World moves left</td> </tr> <tr> <td>Left</td> <td>Right</td> <td>World moves right</td> </tr> </tbody> </table>	Mouse button	Mouse movement	Result	Right	Up	Zoom in	Right	Down	Zoom out	Left	Up	World moves up	Left	Down	World moves down	Left	Left	World moves left	Left	Right	World moves right
Mouse button	Mouse movement	Result																					
Right	Up	Zoom in																					
Right	Down	Zoom out																					
Left	Up	World moves up																					
Left	Down	World moves down																					
Left	Left	World moves left																					
Left	Right	World moves right																					
	n/a	<p>Camera tracking Indicates camera-tracking mode is activated. The selected tile or world object will always be drawn at the centre of the screen. Click on this button to lock the camera in its current position.</p>																					
	n/a	<p>Camera locking The camera will be locked in its current position. Click on this button to change back to camera tracking. Please note, when resizing the world in tile editing mode, the camera mode will revert back to camera tracking mode if it is locked.</p>																					

The following keyboard equivalents are available for most of the above:

Key	Description
r	Reset the camera to its original location. This may be needed as sometimes the camera is completely disoriented.
x	Tilt world down
X	Tilt world up
z	Rotate world left
Z	Rotate world right
+	Zoom in
-	Zoom out
y	Tilt world left
Y	Tilt world right

Growing and shrinking the world

Tile editing mode

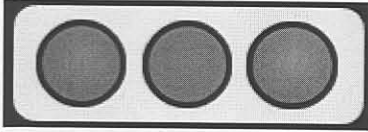


The iTiles Workbench tool automatically starts in tile editing mode. If the workbench is in world object editing mode, you can enter tile-editing mode by clicking the following button:



Editing tile elements

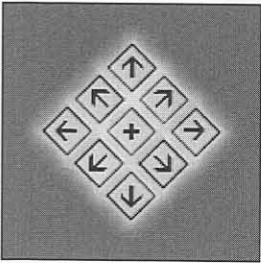
In tile editing mode, you can set the element of the selected tile by clicking on desired element type on the “traffic light palette”. The numeric keys 1,2 or 3 can also be used.



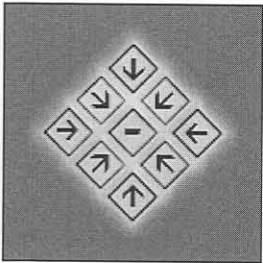
Please note: If a tile that contains a world object element is changed to an element that the world object can't be placed on, then the world object will be automatically removed from the tile.

Growing and shrinking the world

The image below contains buttons that are used for growing the world. By clicking the plus, the world will grow in all directions. Using the appropriate arrow, the world will grow in the specified direction if it hasn't reached its maximum size.

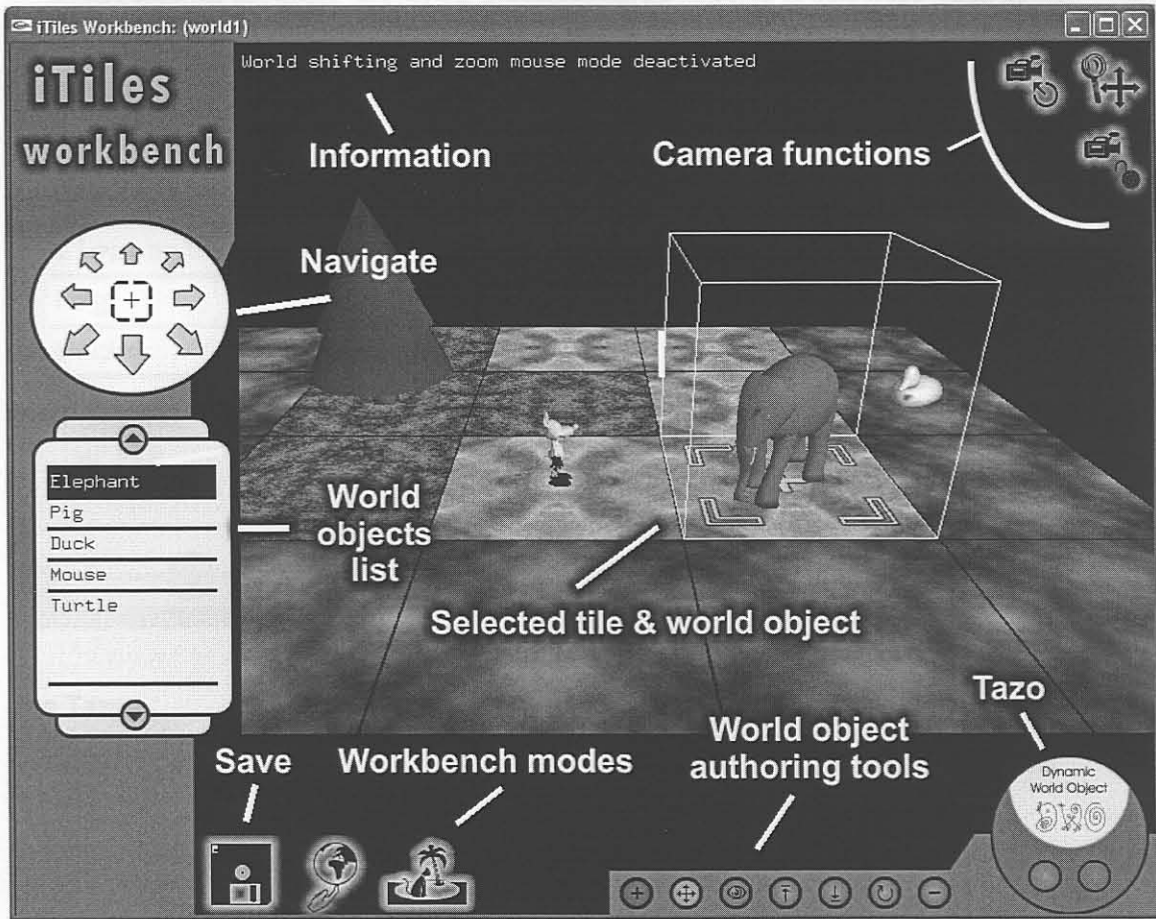


The image below contains buttons that are used for shrinking the world. By clicking the minus, the world will shrink in all directions. The appropriate arrow will shrink the world in the specified direction if it hasn't reached its minimum size.



Please note: An iTiles world has a maximum size of 10x10 and a minimum size of 2x2.

World Object authoring mode

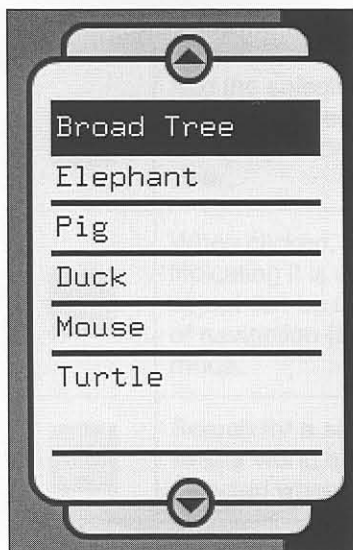


Clicking the following button will activate world object authoring mode:



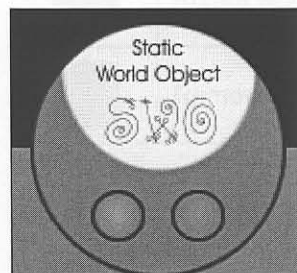
Selecting a world object

The selected world object will be indicated in the world object list. Please note that an empty tile first needs to be selected in order to change the selected world object in the list. When navigating through the world, the list will automatically be updated to select the world object that is currently being selected.










The Tazo

The Tazo indicates the tile properties of a world object that is selected and what tile elements the world object may be placed on. It also indicates whether the world object is a static or dynamic world object.



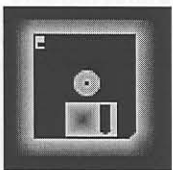
World object authoring

The following buttons are available for authoring the world objects in a world. The buttons will be coloured red if they are disabled, and green when enabled. With some buttons, a mode can be activated, and in this case the button will be coloured orange to indicate the mode is active.

Button	Description
	Add the selected world object from the list to the world on the selected empty tile. If this is red, this means that according to the world object's properties, the world object can't be placed on that particular tile type or the tile may not be clear.
	When clicked, it will activate world object movement mode (Will become orange, indicating it is activated). As long as this mode is active, the selected world object can be moved to the closest free tile it may be placed on in the direction of navigation (by navigating tiles as usual). Re-click the button to deactivate the mode.
	Search for a similar world object type in the world. This will be red if there are no similar world objects in the world. The first similar world object type will be selected when found.
	Scale world object up. When red, the world object is at its maximum scale
	Scale world object down. When red, the world object is at its minimum scale
	Rotate world object 45 degrees clockwise
	Remove the selected world object from the world

Saving the iTiles Workbench

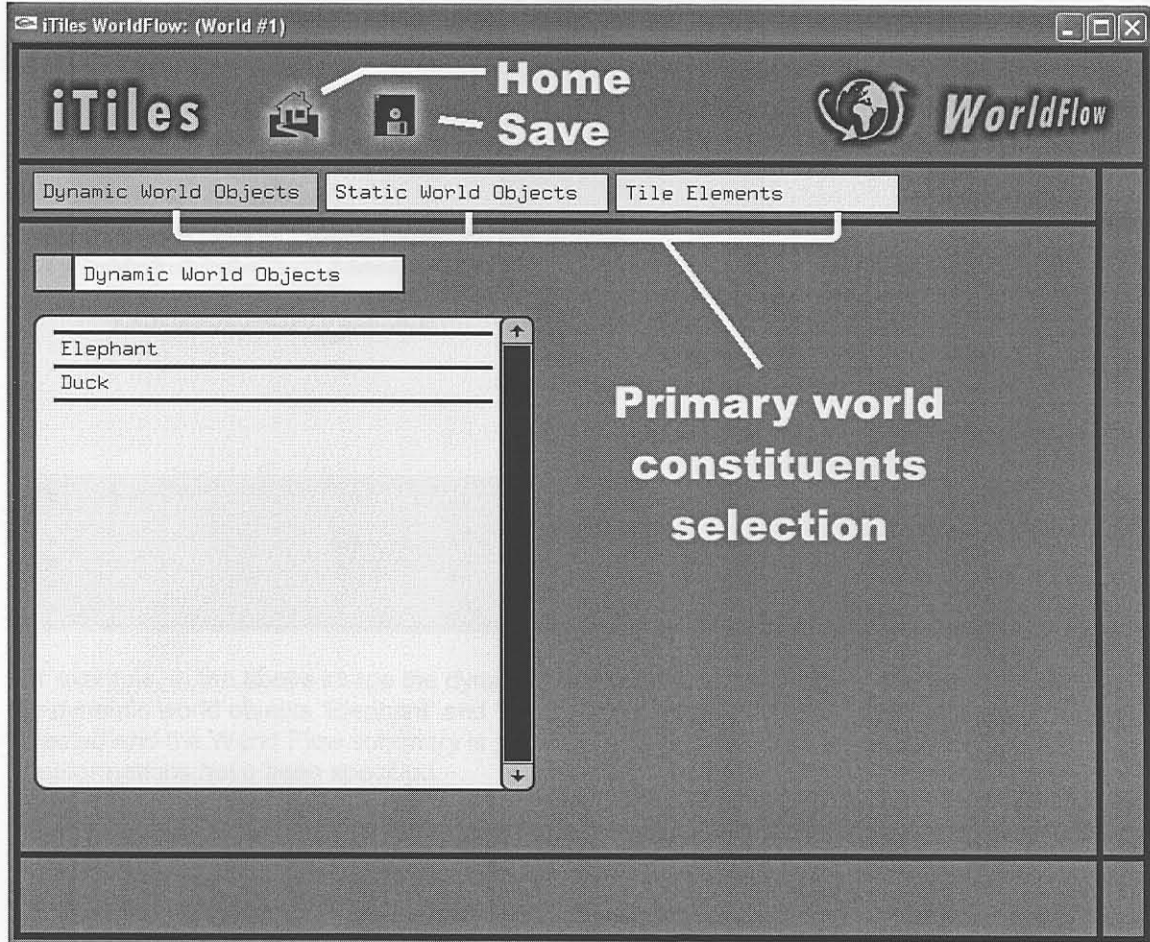
When you are happy with the world you have created, please click the following button:



Using the iTiles World Flow tool

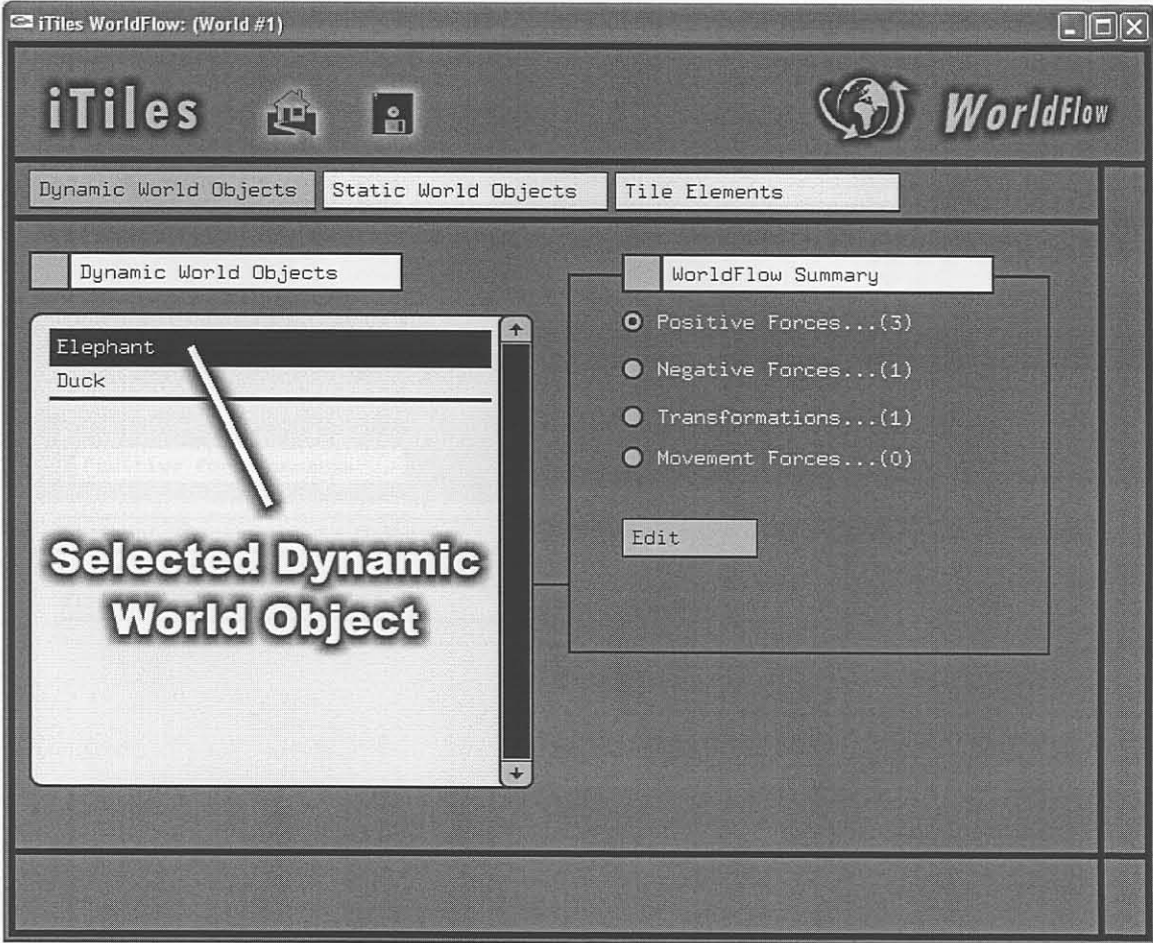
Main Navigation techniques

The iTiles World Flow main screen is presented in the image below. The home button will transport you back to this screen from any other screen in the World Flow, and the screen will remain in the same state in which you last left it.



The tab buttons at the top indicate the primary world components. When clicking on a tab button the corresponding world components will be displayed in the list. Please note that for static and dynamic world objects, the list will only contain those world object types specified when authoring the iTiles world in the iTiles Workbench tool. If the list is empty, this means that the authored world contained no such world objects types.

When something is chosen from the list, a World Flow summary is presented indicating how many forces (dynamic world objects only) and transformations exist for that selected world component. The editing and addition of these forces or transformations is done by selecting the corresponding radio button, and pressing the edit button. Pressing the edit button will take you to the selection screen which is described later in this section.



For example, in the above image the dynamic world object tab is selected, the list contains the dynamic world objects 'Elephant' and 'Duck'. The Elephant dynamic world object is selected and the World Flow summary is shown indicating how many forces and transformations have been specified.

Saving the World Flow

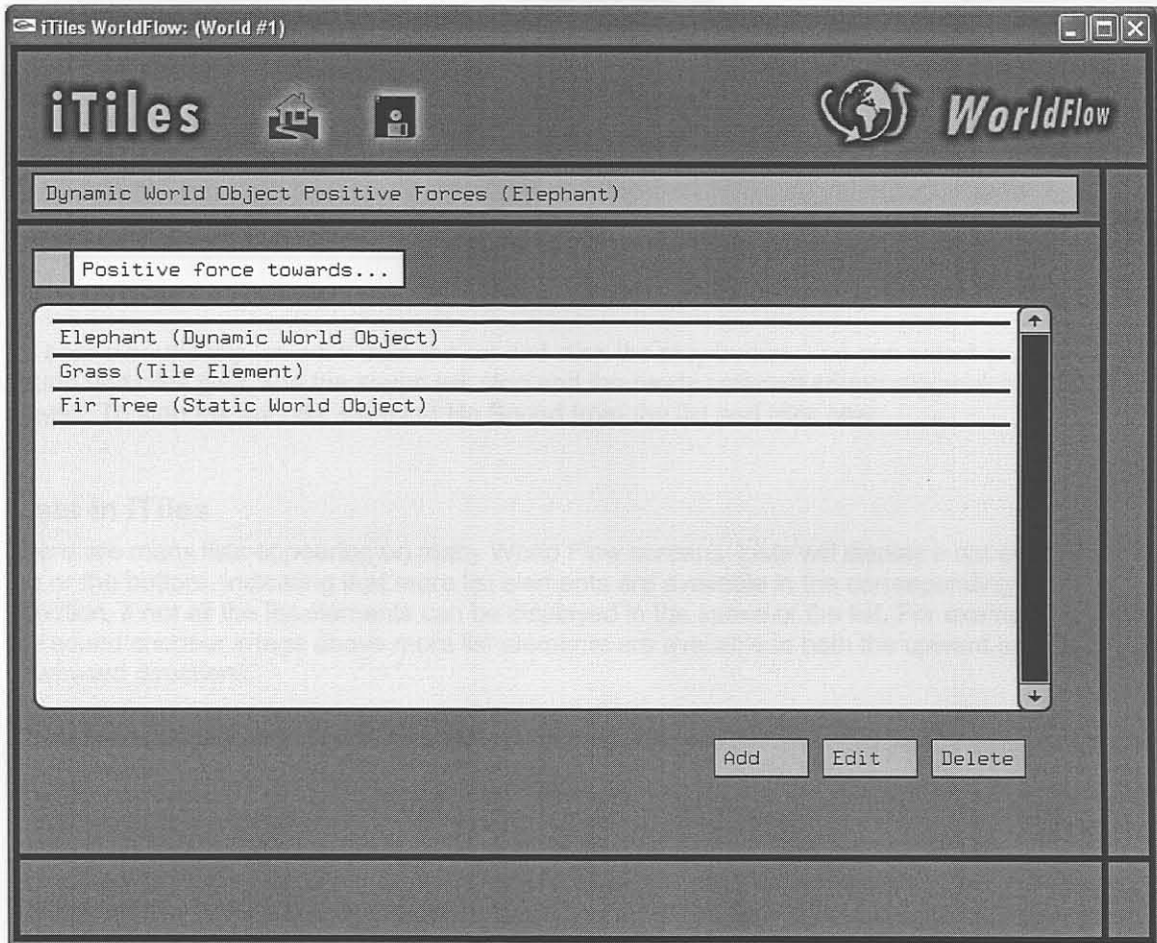
Press the save button at any time when you are happy with the World Flow you have specified.

Button	Description
Add	A new force or transformation can be added by pressing the add button.
Edit	To edit an existing force or transformation, select it from the list and press the edit button.
Delete	To delete an existing force or transformation, select it from the list and press the delete button.

Please note: In the case of movement forces, these can only be added or deleted. Also, if a transformation is deleted, all those forces that have been specified to trigger the transformation will be reset, whereby no transformation will be specified for those where the transformation that was deleted was selected.

Selection screen Sound Chooser control

When pressing the edit button on the World Flow main screen summary for transformations or forces you will be taken to a selection screen for those transformations or forces for the dynamic world object, static world object or tile element you have chosen.



The above image indicates the selection screen for the 'Elephant' dynamic world object's positive forces.

The following functions are available on a selection screen:

Button	Description
Add	A new force or transformation can be added by pressing the add button.
Edit	To edit an existing force or transformation, select it from the list and press the edit button
Delete	To delete an existing force or transformation, select it from the list and press the delete button

Please note: In the case of movement forces, these can only be edited or deleted. Also, if a transformation is deleted, all those forces that have been specified to trigger the transformation will be reset, whereby no transformation will be specified for them where the transformation that was deleted was selected.

Using the World Sound Chooser control

Many screens in the iTiles World Flow contain the world sound chooser control and you may have to select sounds for various forces and transformations.

Please note not to select too many sounds in the iTiles World Flow tool, as in the simulation of the iTiles world there may be much confusion with too sounds playing.



To hear a sound clip, select it from the list and click the play button. You can select another sound and click play, and the sound will stop and the newly selected sound clip shall be played. To stop the sounds, selected No Sound from the list and click play.

Page 5 from Tool

Lists in iTiles

There are many lists appearing on many World Flow screens. Lists will display a dot either on top or the bottom, indicating that more list elements are available in the corresponding direction, if not all the list elements can be displayed in the space of the list. For example, in the sound chooser image above more list elements are available in both the upward and downward directions.

Explanation of World Flow screens

For a dynamic world object, positive forces, negative forces, transformations or movement forces can be specified. For static world objects, world object transformations can be specified and for tile elements, tile transformations can be specified. The following is an explanation of the screens used to capture these World Flow concepts.

Each screen can be in editing mode or adding mode (depending on whether you have chosen to edit or add the specific force or transformation), and will behave differently in these modes, whereby some components may not be editable (read only) in edit mode.

In most cases, when the input captured is sufficient to satisfy a World Flow concept, an OK button will appear in the bottom left of a World Flow screen.

Positive Force Screen

The positive force information is captured on a total of four pages. Next and previous buttons will appear in the bottom left corner of the screen when they are available. Use these buttons to navigate through the pages.

Page 1:

On the first page you have the option to select to what the world object is attracted to. This list will only contain those world components that have not been specified yet. In other words the lists will only contain those world components that have not already been defined as positive or negative forces. A dynamic world object can't be attracted to or afraid of the same thing, so if a negative force is already defined for that world component, it won't appear in the list.

Once a world component is selected, you may go to the next page (click on the next button).

Page 2:

On the second page the force seeking type and found action type are specified.

Force Seeking Type

The force seeking type can be specified as either constant or incremental. With a constant force seeking type, a constant force strength (a value between 50 and 100) is specified. Thus the dynamic world object will constantly have an attraction for this world component.

An incremental force seeking type is a force that gets stronger over time. The following parameters are needed:

Parameter	Description
Initial force strength	The strength of the force at the start of the simulation
Increase percentage unit	The amount the force increases by when the increase time has elapsed
Increase time unit(s)	The amount of world beats that elapse for a force increase to occur
Reset force strength	The value the force is reset to when the positive force is acted upon (i.e. what the dynamic world object was seeking was found)

Found Action Type

When the world component is eventually found, the dynamic world object can either do nothing or stay in the location for a specific time period (in world beats).

Page 3:

On the third page the sighted and found sounds are specified.

Sighted sound

When the dynamic world object sees what it likes, it will output the sighted sound. Please note that when a dynamic world object looks around, it will only play this sound for the first thing that it likes is spotted (otherwise there will be too many sounds in the simulation).

Found sound

This sound will be played when the dynamic world object acts on this positive force and finds what it likes.

Page 4:

On the fourth page, the transformations that are triggered are specified. The top transformation list displayed is always that of the dynamic world object and the bottom is of the world component specified (the world component that dynamic world object likes).

If the transformation chosen has a trigger type that is a gradual transformation, there will be an option available to specify by how many units trigger this transformation's meter is increased.

Movement Force Screen

Movement forces can be specified for the elements that a dynamic world object can move on.

When a dynamic world object moves on a particular tile element, transformations of the tile and itself can be triggered. A movement sound can also be specified, and will be played when the dynamic world object moves on that particular tile element.

Negative Force Screen

Here you have the option to select what the world object is afraid of. This list will only contain those world components that have not been specified yet. In other words the lists will only contain those world components that have not already been defined as positive or negative forces. A dynamic world object can't be attracted to or afraid of the same thing, so if a positive force is already defined for that world component, it won't appear on the list.

Please note, a dynamic world object can't be afraid of tile elements that it can be placed on or move on, and thus these tile elements wouldn't appear in the list. Thus a dynamic world object can only be afraid of the edge of a tile element it can't be placed on (eg: if it's afraid of water it'll try and avoid the water's tile edge).

Sighted sound

A sighted sound can be specified, and in the simulation when the dynamic world object sees what it is afraid of, it will output this sighted sound. Please note that when a dynamic world object looks around, it will only play this sound for the first thing that it dislikes is spotted (otherwise there will be too many sounds in the simulation).

Caught sound

If the world component specified is a dynamic world object (ie: the dynamic world object is afraid of another dynamic world object) then a caught sound can be specified. This sound will be played with the other dynamic world object "finds it" if it's attracted to it (it being the dynamic world object to which this negative force belongs to).

World Object Transformation Screen

On this screen the world object transformation is specified.

Transformation Type

Firstly you must specify the transformation type whereby a world object can either specified to be scaled (larger or smaller) or made to disappear. If the transformation type is chosen as scale object, a percentage unit must be specified by how much the world object will be scaled larger or smaller when the transformation occurs.

Trigger Type

The trigger type of a transformation can be specified to happen immediately or gradually. If the trigger type is chosen as gradual, the transformation has a meter that can be influenced by a transformation trigger by a certain unit. When the meter reaches 100% (it starts at zero) the transformation will occur.

Meter Alteration Sound

If the trigger type is chosen as gradual, a meter alteration sound can be specified that will be played when the transformation is triggered.

Transformation Sound

This sound will be played when the transformation occurs.

Movement Force Screen

Movement forces can be specified for tile elements that a dynamic world object can move on.

When a dynamic world object moves on a particular tile element, transformations of the tile and itself can be triggered. A movement sound can also be specified, and will be played when the dynamic world object moves on that particular tile element.

Tile Element Transformation Screen

A tile element transformation is very similar to a world object's transformation, except that the tile element transformation is the change of one tile element to another tile element. Please note that once a tile transforms to another tile element in the simulation, it gets all tile transformations as specified in the original transformation listing of the tile it is transforming to.

A note about transformation triggers

For dynamic world object's movement and positive forces, please note the following: Tile transformations cannot be triggered that will alter to a tile element that the selected dynamic world object won't be able to walk on once the transformation occurs. This is only the case if the tile element selected is that which the dynamic world object can move on or be placed on. These rules are enforced in the world flow screens, and you need not worry about selecting the wrong options (it's just if you wanted to do something and were wandering why it isn't displayed, or why you cannot select such options).

Navigating world objects

When the Virtual World starts, no world object is selected by default. By clicking the button below (which signifies a magic portal that transfers you to a particular part of the world), a different world object will be selected. Firstly all dynamic world objects will be composed through, then all static world objects. If any world object is to disappear in the world, the navigation order will be reset.



On the keyboard, the <back> key is the equivalent of pressing the button.

The selected world object's name will appear in the box as indicated in the image below (Duck is selected). The Tazoo will also be displayed (please see the explanation of the Tazoo in the Warbler section).

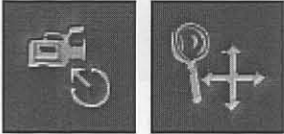


Using the iTiles Virtual World tool

First and third person views are available for a selected world object. In first person view you see the world from the director's point of view (3rd person) and will take on the identity of the object.

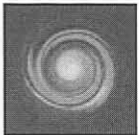
Camera navigation

Camera navigation is the same as in the iTiles Workbench. The camera functions are only available when no world object is selected, or when a world object is selected in 3rd person view (see below).



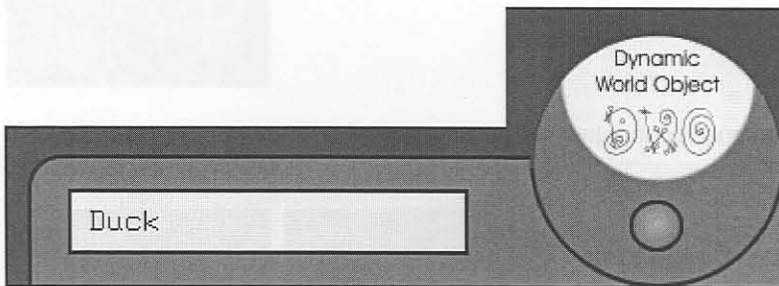
Navigating world objects

When the Virtual World starts, no world object is selected by default. By clicking the button below (which signifies a magic portal that transfers you to a particular part of the world), a different world object will be selected. Firstly all dynamic world objects will be navigated through, then all static world objects. If any world object is to disappear in the world, the navigation order will be reset.



On the keyboard, the <space bar> key is the equivalent of pressing the button.

The selected world object's name will appear in the box as indicated in the image below (Duck is selected). The Tazo will also be displayed (please see the explanation of the Tazo in the Workbench section).



Speeding up and slowing down time

The slider next to the world list signifies the amount of time of each world beat. E.g. in the Tazo world, each world beat occurs every three seconds. By changing the value the simulation can be slowed down or sped up.

Likes and dislikes

First and third person views

First and third person views are available for a selected world object. In first person view you see the world from the character's point of view (inner vision) and will take on the identity of that world object (virtual identity).

To enter the world, and go into first person view, press the following button:



To go back to third person view, press the following button.



On the keyboard, the <enter> key is the equivalent of pressing both buttons.

Please note: When the virtual world is animated, sometimes the camera-smoothing algorithm may be a bit jerky. This is due to the calculations being performed by each dynamic world object in the world to determine the most favourable direction to move.

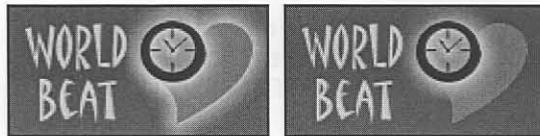
Starting and stopping the simulation

When the Virtual World is started up, time is initially paused.

To start the animation of the simulation, press the play button below:



The following button images indicate that the simulation is running. Press these buttons to pause the simulation.



Speeding up and slowing down time

The number next to the world beat signifies the amount of time of each world beat. E.g.: In the image below, each world beat occurs every three seconds. By changing this value the simulation can be slowed down or sped up.



Likes and dislikes

You can dig into a character's brain and see what it likes and doesn't like. When clicking on the buttons below (the happy face signifying what the character likes, and the unhappy face signifying what the character dislikes), a list is shown with the relevant information. Click on the button again to make the list disappear.



If a static world object is selected, all the dynamic world objects that like it will appear on the likes list, and vice versa for the don't like list.

Character vision

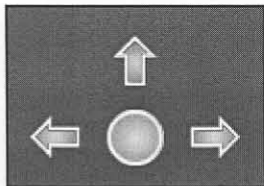
To see what a character sees when making its decision to move, press the eye button. When activated it will be yellow. Deactivate character vision mode by pressing the button again.

In this mode you can see how the characters look around and see what they like and don't like and steer themselves in a favourable direction.



Controlling a character

You have the ability to control a dynamic world object. See the button explanation below for details.



Buttons	Description
	The control character button can be the following colours, with the following meanings:
	Red: The character can't be controlled
	Blue: The character can be controlled by pressing this button
	Yellow: The character is being controlled. Press this to disable control
	The equivalent key on the keyboard for this button is the insert key.
	The arrows control the movement of the character. The arrow buttons are only functional when yellow. The up arrow moves the character straight, the left arrow right turns the character left and the right arrow turns the character right.
	When the character is being controlled, the buttons may become red indicating the character is busy performing an action for a period of time (frozen), and control will only be available as time passes (character finishes action)
	The equivalent arrow keys on the keyboard can also be used for navigation.

iTiles Setup and Configuration Files

The following is an explanation of the configuration files on the iTiles system.

Extending the iTiles system

Currently the system supports extension, however it needs to be done by a manual process, by editing system configuration files. A front-end user interface to make extending iTiles in a much more user-friendly way will be made available in future versions of iTiles.

When adding any new thing to a configuration file, please use the next number in the sequence for the unique identifier (last number plus 1).

Adding additional world objects

The 3D model file format used in iTiles is the Milkshape 3D ASCII file format. Textures specified for these models can only be in TGA file format. For best results the 3D model should be placed centrally on the main axis facing the user. Determining the minimum and maximum scale of a world object is sometimes a cumbersome process, since the world object must fit perfectly on a tile in the system. This is done by experimentation and by changing the values and testing the effects. Hopefully this process will be simplified in future versions of iTiles.

Adding static world objects

The static world object's configuration file <staticobjects.ini> will be found in the installation directory. Please edit this configuration file to add additional static world objects.

The file is composed of lines of comma-separated values. An example of a line is: "1,Fir Tree,1,1,0,0.1,0.5,models/static/tree/firtree.txt".

These fields and values are described in the table below:

Field	Name	Description (including possible values)
1	ID	Unique identifier
2	Name	Name of the static world object
3	Grass Property	0 (Can't be placed on element) 1 (Can be placed on element)
4	Sand Property	"
5	Water Property	"
6	Minimum Scale Value	Floating point number
7	Maximum Scale Value	Floating point number
8	3D Model Filename	A filename relative to the iTiles installation directory

Adding dynamic world objects

The dynamic world object's configuration file <dynamicobjects.ini> will be found in the installation directory. Please edit this configuration file to add additional dynamic world objects.

The file is composed of lines of comma-separated values. An example of a line is:
"3,Duck,0,0,1,0.3,0.8,models/dynamic/duck/whiteduck.txt,0,2".

The contents of fields 1-8 are identical to the static world objects configuration file. Fields 3,4,5 also indicate the elements that the character can move on. The additional fields are:

Field	Name	Description (including possible values)
9	Vision type	The vision type affects how the character sees the world. 0 – Herbivore vision type 1 – Carnivore vision type
10	Vision depth	The vision depth indicates how many tiles a character can see (how far the character can see).

Adding additional sounds

The listing of sounds used in the iTiles system is stored in the configuration file <worldsounds.ini>, that can be found in the installation directory.

The file is composed of lines of comma-separated values. An example of a line is:
"5,Water,wav/water.wav".

These fields and values are described in the table below:

Field	Name	Description (including possible values)
1	ID	Unique identifier
2	WAV Filename	A filename relative to the iTiles installation directory

Please note that the sound files used in the iTiles System can only be of WAV PCM format (8 bits per sample).

Files and directories found in the installation directory

The iTiles worlds are stored in the installation directory. The files <world*> are the saved Workbench files and the <world*_wf> are the saved World Flow files (where * can be a number from 1-5).

The following directories are present in the iTiles directory, and have the following functions:

Directory Name	Description
models	All 3D models are stored in this directory
wav	All WAV sounds are stored in this directory
img	All images needed for iTiles are stored in this directory

Sample Worlds

Some sample iTiles worlds (virtual laboratories) are presented to display the functionality of the iTiles Ecosystem Virtual Laboratory application. They illustrate the vast amount of worlds that can be authored, explain some use of the iTiles World Flow concepts and some ideas of how the iTiles Virtual World simulation can be used.

Please note: These iTiles worlds are also available in the iTiles Ecosystem Virtual Laboratory application.

iTiles World 1: Ducks in the park

iTiles Workbench

The world is authored to look like a park. It has a dam, and there is dirt path for walking along the dam. Ducks swim in the dam, and the park has a local resident dog that likes to walk around. Willow trees are placed along the water's edge.

iTiles World Flow

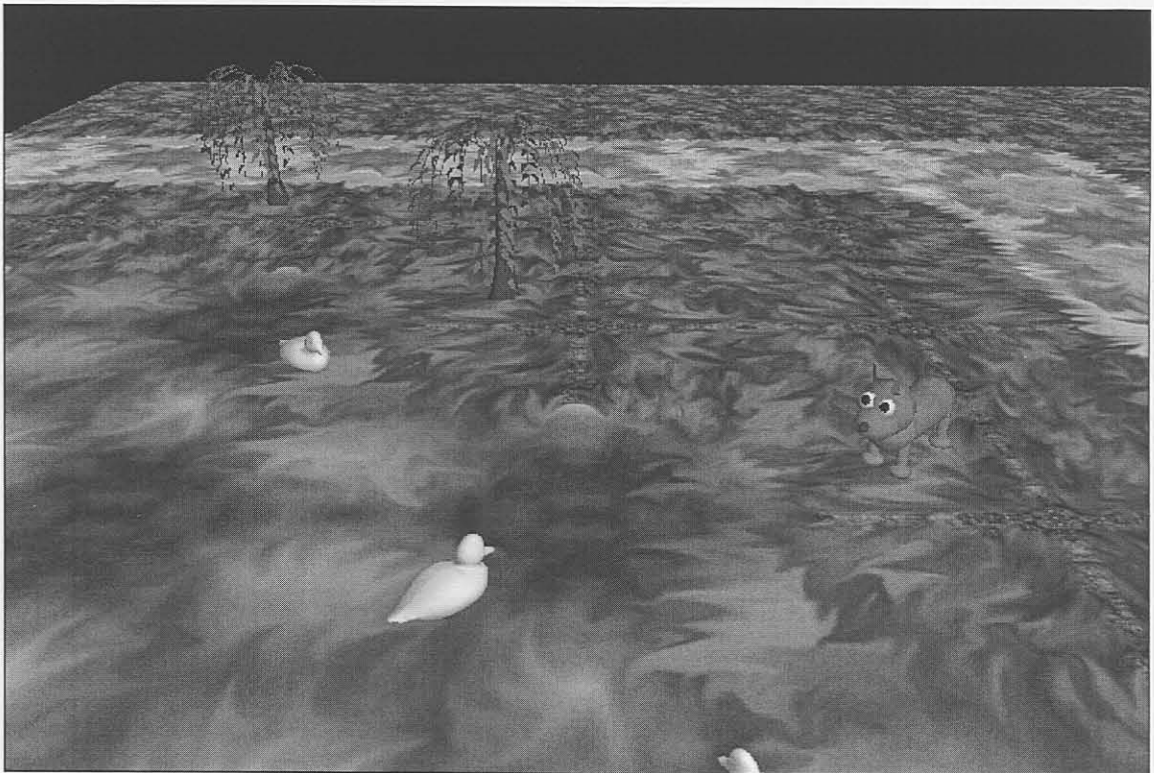
When the dog walks near trees, we can hear the birds singing. He doesn't like ducks much, so when he sees a duck he barks at it. The dog sometimes gets thirsty and comes to drink at the water's edge.

Ducks like each other, and when they see each other they quack. If they are close to each other, they stop for quick chat.

iTiles Virtual World

Things kids can do: Have fun exploring the park!

They can hear the ducks quack and the dog barking at the ducks. Hear the birds sing when a dog approaches a willow tree.



iTiles World 2: Drought in Africa

iTiles Workbench

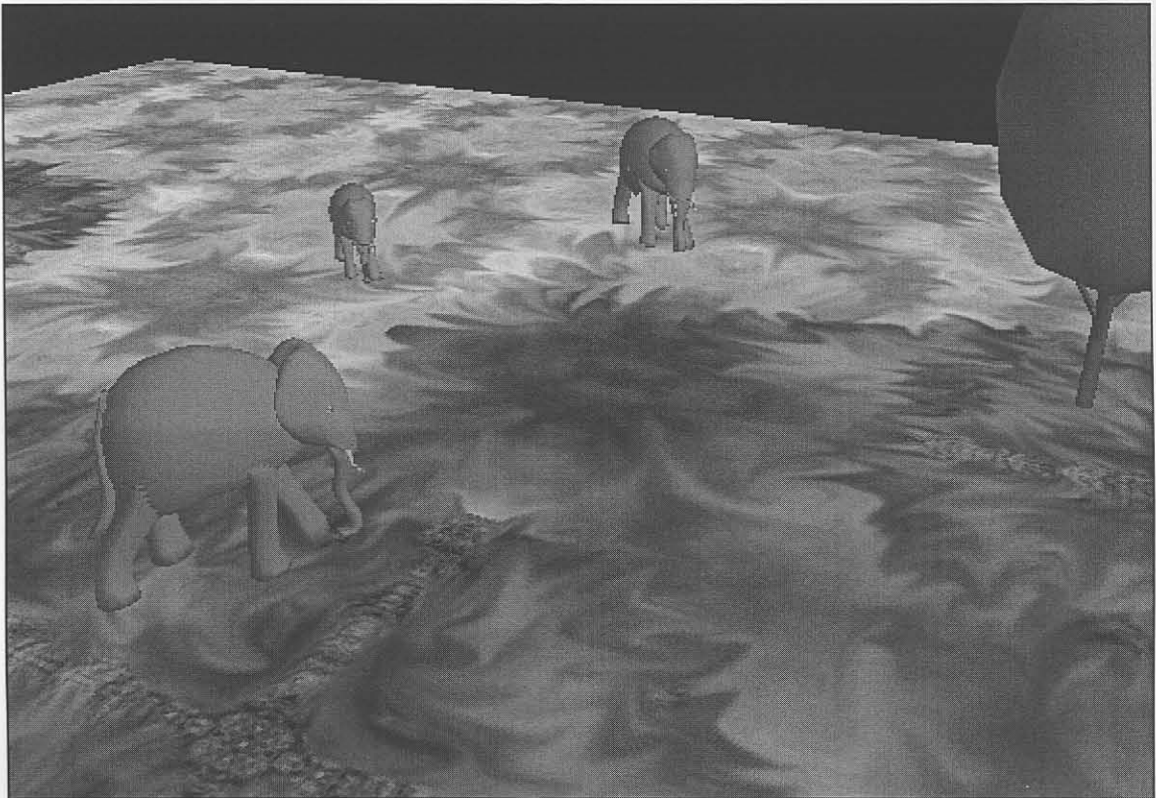
The world is authored to look like an arid African savannah during a harsh drought. There are areas with a bit of water, from which the elephants can drink, with patches of grass around the water.

iTiles World Flow

This iTiles world presents how the environment can change. The elephants drink from the scarce supply of water, and it becomes sand indicating the drought. The elephants also walk on the fragile grass and it erodes to sand. Elephants eat the leaves from the trees and the trees become smaller.

iTiles Virtual World

The main lesson of this world is to show how tough it can be in times of drought. Kids can see how thirsty the elephants are, and how the environment is transformed by their actions. The environment is eventually transformed to sand, and food supply (the trees) gets scarce.



iTiles World 3: Paradise Island

iTiles Workbench

The world is authored to look like seashore on a paradise island. There are palm trees and some wild pigs that roam free. In the sea there are some turtles.

iTiles World Flow

It's the season when the turtles lay their eggs, so they come on shore now and again and lay some eggs. The wild pigs roam the island and when they see each other they snort to say hello. Turtles like the water and you can hear it splashing when they do, but when it's time to lay eggs they have a mission: find land (you won't hear any splashing then).

iTiles Virtual World

This world illustrates how turtles lay their eggs on the seashore. The wild pigs can be observed as they travel around the island.

