TextSL: A Command-Based Virtual World Interface for the Visually Impaired

Eelke Folmer, Bei Yuan, Dave Carr, Manjari Sapre
Department of Computer Science and Engineering
University of Nevada, Reno
{efolmer, bei, carrd, msapre}@cse.unr.edu

ABSTRACT

The immersive graphics, large amount of user-generated content, and social interaction opportunities offered by popular virtual worlds, such as Second Life, could eventually make for a more interactive and informative World Wide Web. Unfortunately, virtual worlds are currently not accessible to users who are visually impaired. This paper presents the work on developing TextSL, a client for Second life that can be accessed with a screen reader. Users interact with TextSL using a command-based interface, which allows for performing a plethora of different actions on large numbers of objects and avatars; characterizing features of such virtual worlds. User studies confirm that a command-based interface is a feasible approach towards making virtual worlds accessible, as it allows screen reader users to explore Second Life, communicate with other avatars, and interact with objects as well as sighted users. Command-based exploration and object interaction is significantly slower, but communication can be performed with the same efficiency as in the Second Life viewer. We further identify that at least 31% of the objects in Second Life lack a descriptive name, which is a significant barrier towards making virtual worlds accessible to users who are visually impaired.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: Miscellaneous—*Voice I/O*

General Terms

Human Factors, Measurement, Design, Experimentation

1. INTRODUCTION

Second Life [5] and World of Warcraft [1] are two of the most popular virtual worlds that offer rich, three-dimensional environments for social interaction and that have experienced significant commercial success [6, 10]. These virtual worlds allow the user to control a digital puppet, called an avatar–with human capabilities, such as walking and gesturing–through a game-like, third person interaction mechanism. Users can explore vast virtual worlds and interact with other avatars and objects. Virtual worlds surpass the two-dimensional Internet by offering a much higher degree of interaction. Social interaction possibilities offered by virtual worlds significantly exceed Internet-based social interaction tools such as web forums, chat rooms, or messenger clients. Interaction with other avatars mimics real-life interaction, as users' avatars can perform a variety of activities such as dancing or playing games. Virtual worlds could someday replace the World Wide Web [23].

Two different types of virtual worlds can be distinguished: (1) game-like virtual worlds such as World of Warcraft or Star Wars Galaxies are modeled after role playing games with all the typical elements found in most games–such as: enemies to beat, levels to attain, a story line, goals to achieve, and the possibility for the avatar to die; (2) non-game-like virtual worlds, such as Second Life, There and Active Worlds are characterized by the lack of such elements. A main difference is between current game- and non-game-like virtual words is that non-game like virtual worlds are entirely created, owned, and maintained by their users. Though such virtual worlds are modeled after games in terms of their use of three-dimensional graphics and basic control features, they allow for a variety of different usages varying from playing games, visiting museums, taking classes or socializing with others in numerous communities. The application used to access these non-game-like virtual worlds is called a viewer as is very similar to a web browser, as the specific usage of non-game virtual worlds depends on the specific place visited in the virtual world. The viewer supports a set of basic functions such as navigating an avatar, interacting with objects, creating objects, and communication with other avatars. Studies reveal that social interaction is the most important aspect of virtual worlds [21].

Unfortunately, a significant number of users are excluded from accessing virtual worlds because of a disability [14, 18, 35]. Moving from a strictly two- dimensional, predominantly text-based environment, such as the Internet, to a three-dimensional, entirely visual environment has raised concerns about accessibility [14, 18]. Controlling your avatar in Second Life typically requires at least the use of a mouse. Users who are visually impaired face significant barriers to access. Though Second Life has some support for users with low vision through the use of scalable fonts, it lacks features that allow individuals who are severely visually impaired to access Second life using non-visual forms of feedback includ-

ASSETS'09, October 25-28, 2009, Pittsburgh, Pennsylvania, USA.

ing audio or tactile feedback. Individuals who are severely visually impaired often use screen readers to access word processors or browsers, but as Second Life lacks any textual representation[14], a screen reader cannot be used.

The US has an estimated 1.3 million individuals who are legally blind [9]. Second Life offers social communities for users with disabilities. The Virtual Ability organization [13] offers communities such as Cape Able Island or the Heron Sanctuary [30], which are meeting places for users with disabilities. Naughty Auties is a virtual resource center and meeting place for those with autism [12]. Individuals who are visually impaired are excluded from participating in such social communities, though it could benefit such users greatly as individuals with visual impairments often feel isolated and lonely [29]. Second Life has successfully drawn academic interest as a cyber learning environment due to it's high degree of customizability [20, 28, 2]. To make the use of virtual worlds comply with section 508 of the US Rehabilitation Act [32], it is important to investigate how to make them accessible.

This paper presents the work on developing TextSL; an interface for Second Life that can be accessed with a screen reader. The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 presents TextSL and discusses the mechanisms Section 4 discusses the results of a user study which compares TextSL with the Second life viewer. Section 5 discusses observations we made and problems we encountered as well as pointing out areas for future research and the paper is concluded in Section 6.

2. BACKGROUND AND RELATED WORK

An extensive list of video games accessible to users who are visually impaired can be found on `http://audiogames.net`. We survey the strategies used in the games that most closely resemble the interaction of virtual worlds.

- **Shades of Doom** [11] is a first-person-shooter (FPS) game modeled after the popular FPS Doom and uses audio cues such as footsteps or the sound of wind to help guide the way through levels. A navigation tool provides information about the player's surroundings using synthetic speech.
- **Audio Quake** [15] is a visually-impaired-accessible version of the FPS Quake. AudioQuake uses *"earcons"* which are brief, structured, sound patterns obeying musical conventions [17]. Earcons alert the player to an object or enemy. Stereo sound and volume are used to convey spatial information.
- **Terraformers** [34] is an FPS/adventure game. The game uses sound cues and forms of sonification, such as a sound radar, that can identify what is in front of the player and a sonar that uses pitch to indicate how far away objects are. Built-in speech synthesis provides information on items and objects.
- **Powerup** [31] is a multiplayer educational game developed by IBM that supports various accessibility features. Users who are visually impaired can play this game using built-in self-voicing and audio cues such as footsteps, which provide additional feedback. Users can issue a number of commands such as *"look left"* -activated by keyboard keys- to get information about their environment.

The main strategy used in these games is transforming visual feedback into a form of audio. To allow for techniques such as speech or audio cues, these games require some form of augmentation. Augmentation can include adding audio cues or names and descriptions to objects or locations. Levels in these games are designed such that there are at most a few objects or avatars of interest around the player, to avoid overwhelming the player with information when players query their environment. An additional benefits of such a design is that it is relative easy to implement interaction with objects and avatars, since there are relative few. Players typically automatically interact with the object or avatar that the player is supposed to interact with, as it is typically part of the story of the game.

The Second Life viewer was open sourced in 2007, allowing developers to enhance the viewer with new features. A modification of the viewer has been developed that allows users who are visually impaired to navigate their avatar using force feedback [25]. Different object types are distinguished through different vibration frequencies. However this approach is limited to a few object categories as the user has to memorize the vibration frequencies of these categories. Very recently, two alternative approaches closely related to TextSL were made available. The guide dog project [8] developed by the Virtual Ability Group [13] offers a virtual guide dog object that can be "worn" by a user's avatar. The guidedog provides a number of functions such as navigation and querying the environment through a chat-like interface. Feedback is provided using synthetic speech. Because it is integrated into the Second Life viewer, this solution allows for hearing in-game audio, but it may also require a sighted user to initially setup the guide dog object. Built-in speech synthesis typically does not allow for the same customizations as external screen readers. IBM's Human Ability and Accessibility Center developed a Web-based interface for Second Life [3] that can be accessed with a screen reader. This client provides basic navigation, communication, and perception functions using various hotkeys. A hotkey-based interface is limited in supporting a large number of different actions on large numbers of objects and avatars. A notable feature of this client is that it also provides a plug-in for the Second Life viewer, which allows users to add names to objects in Second Life, as such data is often missing. These names are stored in an external database and the IBM's client can retrieve such object descriptions when a user encounters an object with missing meta-information. A number of differences and similarities between TextSL and these clients can be identified. The guide dog object also uses a command-based interface and IBM's client also uses an external screen reader. Both clients offer limited functionality when compared with TextSL as they do not allow object interaction, collision free navigation or a mechanism that prevents overwhelming the user with audio feedback.

3. TEXTSL

TextSL allows users who are visually impaired to access Second Life with a screen reader. The enabling of screen reader access was motivated as follows. We first surveyed the strategies used in games accessible to players who are visually impaired with similar interaction mechanisms as virtual worlds. Haptic feedback was excluded from our strategies, as it had not been used in any of these games. Because virtual worlds and games are fundamentally different, we first

analyzed the differences to determine what type of feedback would be most feasible to explore. Combat is an essential element of many video games, and as a result the player is required to be able to respond quickly. Sound output mechanisms such as a sound radar [34], earcons [15], or even ambient spatial sounds [31, 11], all aim to allow for quick identification and localization of the enemy. Virtual worlds lack combat, but they do have a plethora of different objects and avatars. Where it is relative easy to use ambient sounds or earcons to distinguishing a limited number of enemies or objects, these techniques do not scale well when trying to discriminate large numbers of different objects and avatars. Many objects in Second Life, such as houses or furniture, are difficult to express with natural audio cues. A technical limitation of non-game virtual worlds are created and owned by different users and it not possible to augment objects directly with audio cues or text.

Objects in Second Life can be give names and descriptions, therefore we opted for exploring an approach that provides synthetic speech. There are two ways to provide synthetic speech e.g. by building it in our application using libraries provided by the operating system or using a screen reader. API's for self-voicing often have limited functionality when compared with the functionality offered by screen readers. Users may prefer a particular screen reader over others and often tailor the screen reader to their personal preferences. Because the Second Life viewer acts like a browser, we explored the use of a screen reader since individuals who are visually impaired typically use these for accessing web browsers.

Though it is technically possible to implement text extraction in the Second Life viewer, we opted for a solution as a stand alone client based on the libsecondlife [4] library. The libsecondlife library allows direct communication with the Second Life servers and offers a simple API for the most common functions used in Second Life. Libsecondlife is typically used to develop chat and spam bots. Developing TextSL as a standalone application has the additional benefit that it can serve as a virtual-world-agnostic research platform that is able to interface with other virtual worlds such as OpenSim and Active Worlds that have similar API's and that can be easily integrated in TextSL. As TextSL does not have to do any rendering it can run on a low-end machine or through a web interface. The TextSL executable can run on any platform as it is written in C# and it does not require any platform-specific libraries. The Second Life viewer requires frequent updates to be able to continue accessing Second Life but TextSL does not require any updates, avoiding tedious installation procedures that may be difficult to perform for individuals who are visually impaired.

Whereas virtual worlds provide feedback from different modalities and different sources simultaneously, a screen-reader produces synthetic speech from a single text source, sequentially, and consequently a form of iterative text extraction is required. Multi-user dungeon (MUD) games allow their players to explore dungeons, fight monsters, and interact with each other using a command-based interface that iteratively accepts commands from the user and that provides text output. Virtual worlds have their origins in MUD games [19] and their usages are very similar as they allow players to explore, interact and communicate. Because MUD games are entirely textual, visually impaired can play them with a screen reader without modifications.

A command-based interface was implemented for the following reasons: (1) Second Life supports a plethora of different avatar actions e.g. create an object, give an objects to another avatar, sit on an object, etc. By using scripts, users can create interactive objects that define their own actions e.g. a car door can be opened or a car can be driven. In the Second Life viewer, users right click on an object or avatar and then select an action from a list of available actions from a popup menu. Screen reader users may not be able to engage in such interactions. (2) All of the audio games listed in the related work section only support limited interaction through a generic *"use"* command, which is activated through a shortcut and automatically interacts with the object or avatar that is closest. Such a mechanism works for these games as they have been designed such to contain few objects or avatars of interest. A mechanism where actions are mapped to shortcuts is too restrictive to allow for numerous actions on large numbers of objects.

Instead, we implemented an interpreter that allows the user to issue commands using natural language. Alternatively to a generic *"give"* command, which may only work on the avatar closest to the user,- the user can specify *"give my flower to Jane"*, which allows for executing unlimited actions on large numbers of objects and avatars efficiently. TextSL supports a number of basic commands in the following three areas:

- **Exploration:** users can navigate their avatar using the *"move"* command, which requires specifying a direction and a distance. Users can also navigate their avatar using arrow keys mapped onto the *"move"* command. Other navigation commands are: *"fly", "follow"* and *"teleport"*. Users can query their environment using the *"describe"* command, which lists the names of objects and avatars found within a specified range around the user. To avoid having the user turning in different directions to find out what is around, all objects within 360 degrees are considered. The *"where"* command provides the distance and relative location of an object or avatar to the user. Whenever the user moves to a new location, information on their environment is provided automatically, minimizing the amount of input that the user is required to provide.

- **Communication:** What other avatars say is preceded by their name e.g.*"Jack says...."*. Users can talk in public using the *"say"* command or send private messages to other avatars using *"whisper"*. Noisy avatars can be muted using the *"mute"* command.

- **Interaction:** TextSL currently only supports two types of object interaction. Users can sit on or touch objects using the *"sit"* or *"touch"* command.

A tutorial teaches the user all the commands available and a help command *"help"* can be used to learn more about each command. To improve efficiency, each command can be accessed using their first letter. An interpreter allows the use of prepositions and adjectives such as, *"sit on the chair"*. Commands such as *"move"* can be issued with an arguments such as *"move north 10"* or *"move to the chair"*.

During the development of TextSL, two important observations were made: (1) objects in Second Life can be given a name and description, but as most content creators assume that users can see, they frequently leave these properties

Figure 1: Scene in Second Life

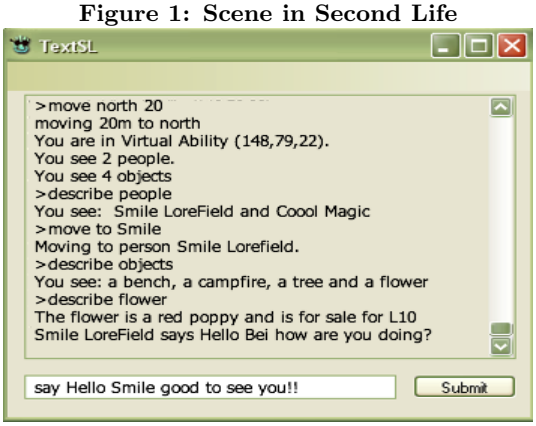

Figure 2: Same Scene in TextSL

with their default values (e.g. *"object"*). This is a problem when users query their environment as this may return the names of multiple objects called *"object"*, which are basically meaningless to TextSL users. To understand the magnitude of this problem, TextSL was modified to collect statistics on the names of objects. The Second Life world is partitioned into smaller chunks called regions. Currently 13,543 regions exist [7]. An avatar logged in with TextSL will teleport to a randomly selected region and will retrieve the names of all objects in that region and store them in a local database. 433 regions were sampled and we found that with 95% confidence, 31.3% $\pm$ 2.02% (SD=21.4%) of the objects in Second life are called *"object"*. This number varies significantly as in one region it was as high as 85%. Further analysis revealed that names such as *"shapeblock"* and *"BuiltUsingSkidzPrimz"* dominate the top hundred of most frequently used names. These names may be descriptive to content creators, but not to TextSL users. We estimate that about 40% of objects in Second Life lack a descriptive name, a barrier to making virtual worlds accessible to screen reader users who require a textual representation.

(2) Second Life is also densely populated with objects. Though regions may vary significantly in size and depending on whether the region is an island or part of the mainland. The 433 regions we sampled contained on average 860 objects (SD=543). Through periodically issuing describe commands while our avatar moves randomly around in the region we found that on average 13 (SD=7.2) objects (including objects called object) could be found within a 10 meter radius around the user. The object density may also vary within each region but this was not analyzed. This poses some challenging research issues with regard to implementing a usable *"describe"* command, as we want to be able to return a list of all objects within a range comparable with what a sighted user can see (up to 40 meters) while at the same time avoiding overwhelming the user with information. An additional problem with this object density is that it is fairly difficult to navigate without colliding into an object. Screen reader users who helped with testing early prototypes of TextSL, frequently found themselves stuck. Two features were added to TextSL to deal with these problems:

- **Summarizer:** a mechanism has been implemented that retrieves all objects and avatars within a specified range (default is 20 meters) around the user. The list of objects is prioritized as follows: objects are ranked according to their distance from the user and the length of their name -assuming that a longer name implies a more accurate description. Objects that lack a descriptive name are culled based on a dictionary of non-descriptive words. When issuing a *"describe"* command or when a user navigates to a new location, TextSL will return the number of objects and avatars found. Users can then iteratively query object or avatar sets or individual objects or avatars using the *"describe"* command with a parameter e.g. *"describe objects"* returns the names of objects found and *"describe the chair"* returns a specific description of the object when available. Names are stripped from any adjectives and partial name matching avoids players having to type in the name exactly. When an object or avatar name is known, users can issue other commands such as *"where is the chair?"*.

- **Collision free navigation:** Users can navigate their avatar using the arrow keys, but this requires the user to circumnavigate any obstacles found in their path. TextSL provides feedback when you cannot move in a particular direction. In audio games there may be good reasons for letting the user do the pathfinding themselves e.g. to detect enemies or to build a mental map of their environment. Second Life lacks any combat and building a mental map of the environment may be extremely difficult, especially since users can fly and are able to build structures in the sky. Though in Second Life there is no risk of injury when you run into an object, having the user do the pathfinding may unnecessary slow down the users' ability to explore Second Life, while obstacles could be easily avoided using path planning. When users specify *"move north 100"*, TextSL automatically plots a collision-free course to this destination using an A* algorithm [27], offloading the task of path finding from the user. Users are automatically teleported to the desired location when they don't arrive within the estimated time limit.

Figures 1 and 2 show a scene in the Second Life viewer and as it is represented in TextSL. A Beta version of TextSL was made available in November 2008 on the website `http://textsl.org` and currently has been downloaded more than 420 times. TextSL's source code has been released under the GNU Public License.

4. USER STUDY

Second Life supports a variety of usages such as playing games or taking classes. Evaluation as to whether Second Life is fun or educational is not an intrinsic property of the Second Life viewer but primarily a property of the place that is visited in Second Life, which offers these activities. The Second Life viewer acts like a browser, facilitating a broad number of usages through four basic functions namely 1) exploration 2) communication 3) interaction and 4) content creation. Our TextSL client does not support any content creation; therefore this functionality is excluded from our study. The "Beyond Accessibility to Efficiency" principle states that assistive technology should be more ambitious than simply providing access [22]. In addition to evaluating whether TextSL users are able to perform exploration, communication and interaction with the same success rates (accessibility) as sighted users can with the Second Life viewer, we verify whether screen reader users can do it with the same usability. The focus of our study is to evaluate whether the specific mechanisms used in TextSL (command-based interface and screen reader output) allow screen reader users to access virtual worlds. To evaluate the usability of TextSL we use the quality attributes proposed by Nielsen [24] e.g. learnability, efficiency, memorability, errors and satisfaction. Consequently the following two hypotheses were defined:

H0/H1: TextSL allows screen reader users to perform exploration, communication and interaction with the same success rate/usability as sighted users using the Second Life viewer.

Design: A user study was conducted with 8 screen reader users and 8 sighted users. 4 JAWS screen reader users (2 female/2 male) between the ages of 18 and 51 (M=37, SD=14) were recruited locally using the email list of the Nevada Federation of the Blind. 4 screen reader users (3 females/1 male) between the ages 16 and 34 (M=24, SD=7.6) were randomly selected from a number of individuals who expressed an interest in participating in remote user studies after a request for participants had been announced on our project website. A risk with remote evaluations is that user experience specialists often lack detailed understanding of how people with disabilities use their assistive technologies [26]. Though local screen reader users were closely involved in the development of TextSL, we ensured that half of our participants were tested locally to validate our assumptions on screen reader users interact with their computers. 8 sighted participants (3 female/5 male) (M=27 SD=3.9) were recruited to allow for quantitative analysis between TextSL and the Second Life viewer. All participants were briefly interviewed to identify their experience with computers, video games, and screen readers. None of the participants had used TextSL nor had accessed Second Life before, though 6 sighted users had played first person shooters. All screen reader users used JAWS. The user study was performed on a private island in Second Life for two reasons: (1) access to the island was restricted to other avatars to avoid any interference; and (2) to ensure that all objects on our island had descriptive names. Our Island was modeled after a typical island in Second Life.

Prior to the user test, users received a tutorial. Screen reader users were allowed to customize JAWS to their personal preferences. For TextSL, a built-in tutorial explained the commands required for navigation, exploration, communication and interaction as well as the *"help"* command. TextSL allows for specifying tutorials or tests in XML. An interpreter parses the XML file into a number of steps, where each step provides an instruction to the user. Each instruction may also include a pass condition for going to the next step (which can be the successful execution of a command or a specific word). As TextSL users receive instructions from within TextSL, we made the user tests for sighted users identical to those in TextSL. A modified version of TextSL can stream the output of a tutorial as instant messages to another user, as well as feed the input to the interpreter that runs the tutorials or tests. Sighted users receive a similar tutorial explaining how to navigate, interact and communicate. For local user studies, an observer is present to answer any questions. The observer manually triggers the bot to go to next step when a sighted user has successfully completed a step in the tutorial, as they don't use commands.

After the tutorial, the observer leaves and participants are teleported to a specific location where the user test starts. Observers continue observing the participants in Second Life with an avatar that has been made invisible to users. Both groups of users were required to perform the following four tasks: (1) navigate to a location and identify a number of objects (exploration); (2) interact with an object (interaction); (3) repeat of task 1 for a different location and objects; and (4) chat with an avatar (communication). We arranged objects and avatars on our island as such to accommodate performing these tasks. Users would not be able to see or query the other place from their current place. Sighted users are always able to see the avatars and objects around them, but in TextSL users only know what is around them when they have issued a *"describe"* command, therefore we added two exploration tasks prior to tasks related to interaction and communication, to allow screen reader users to perform these tasks with the same amount of information available to them as sighted users. Neither sighted nor TextSL users are able to see or query the objects or avatars at both locations from their prior location and the distance to be traveled for task 1&3 is exactly the same. Tasks were designed independently from each other in such a way that failing one task did not affect the successful completion of the next task. If the user fails to complete a task, the task will time out and the user is automatically teleported to the next location. Tasks 1&3 are provided as questions to which the user must find an answer e.g. *"travel 20 meters to the north and find out which objects are there"* upon which the user must provide the right names *"dog and flower"*, which the interpreter then validates before going to the next task. Sighted users receive and answer instructions through the private chat screen with the bot. Correct execution of tasks 2&4 is validated in TextSL through the successful execution of the desired command and an observer monitors sighted users and triggers the bot to provide the next task when they execute their tasks successfully. User tests with sighted users were video recorded using a feature of the Second Life viewer. All dialogue in TextSL was recorded in a text file. Users who were tested remotely were required to email us this logfile. After the user test, screen reader users were allowed to play with TextSL for up to 20 minutes. Screen reader users then participated in a qualitative survey that was conducted by the observer in person or over the phone. This survey analyzed the usability of the specific mechanisms implemented in TextSL. Users could also point out suggestions for improvement.

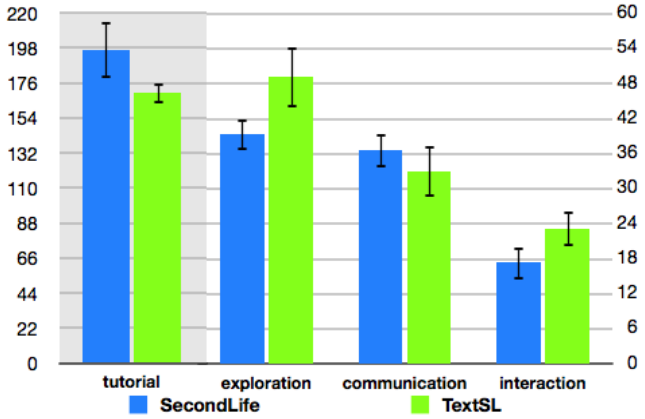

Figure 3: Average Tutorial & Task Performance in seconds and standard error

Results: All participants successfully completed each one of the tasks they were given. Using Fishers exact probability test (P=1.0, α=0.01) we could not reject H0. For validating H1 we compared the attributes learnability, efficiency, memorability and errors of both clients. Satisfaction was measured in isolation through a questionnaire and not used to validate H1. To analyze learnability and efficiency we analyzed the transcripts and videos that were recorded and measured how long it took for users to complete the tutorial and each one of the tasks. Figure 3 shows the average performance times and their standard error (the performance of tutorial is charted on a different scale). The tutorial for sighted users was slightly different as it did not include an explanation how to use the *"describe"* command, as that is not required in the Second Life viewer, but contained identical instructions for navigation, interaction and communication. The average execution time for the tutorial for TextSL was lower but not significantly different (T=0.79, P <0.01), most likely explained by that most TextSL users were simply following the instructions in the tutorial. We found three sighted users playing around with their avatar during the tutorial, most likely explained due to their familiarity with the type of interface the Second Life viewer provides. Interaction with TextSL was found to be significantly slower, specifically the tasks exploration and interaction (T=3.3/1.93, P>0.1). The difference in performance times is mainly explained due to the fact that TextSL is iterative and the Second Life viewer is not. Users can navigate their avatar while at the same time seeing what is there, whereas in TextSL users must first navigate to a location and then have to iteratively query their environment. As communication is iterative in nature, we found it also to be the only task where no significant difference in performance times could be detected. On average interacting with TextSL was found to be 23% ± 13% slower than the Second Life viewer -if all tasks are weighed equally. Whether this performance is acceptable or not is difficult to analyze, due to to inability to directly compare both clients. Screen reader users cannot use the regular viewer and sighted users may not be proficient with the use of a screen reader. Social interaction has identified to be the most common activity in virtual worlds [21] and although it may involve more than communication, the slower interaction of TextSL may not be a major problem -also because Second life does not require quick responses like video games.

To analyze errors and memorability, we found that screen reader user issued an average of 0.75 (SD=0.71) unrecognized commands. This included typos but also issuing a command without a parameter. Sighted users made an average of 0.38 mistakes (SD=0.17). Two of those were clicking on the wrong menu item in the popup to interact with an object and one user initially clicked on the wrong object. The number of times users required help was used as an indicator for memorability. Screen reader users issued the *"help"* command an average of 1.0 times (SD=0.93) during the user test to learn more about a specific command (most commonly the *"move"* and *"describe"* commands). None of the sighted users used any form of help to learn more about commands. The Second Life viewer has a help menu available that sighted users were made aware of during the tutorial. The number of errors (T=1.4,P<0.01) users made was not found to be significantly different between both clients, but TextSL performs significantly worse with regard to memorability (T=3.04, P>0.1). A possible explanation for this could be that six of our sighted users were familiar with the type of interface offered by the Second Life viewer through playing video games with similar interaction mechanisms. Based on these results, H1 was rejected as TextSL fails to be as efficient or support memorability to the same extent as the Second Life viewer.

Due to the inability to directly compare clients, the satisfaction attribute of TextSL was evaluated in isolation through a questionnaire. Screen reader users were asked to rate on a 5-point Likert scale (ranging from very satisfied to very dissatisfied) the ease of use of the mechanisms used in TextSL. Users were satisfied with the command-based interface (M=2.3, SD=0.46) and the screen reader output (M=1.6, SD=0.52). Users also found the interaction efficient (M=1.8, SD=0.83), though due to a lack of reference this result does not corroborate with the result of our quantitative analysis. Nevertheless it may indicate that TextSL's less efficient interaction is acceptable, even if not optimal. Our questionnaire failed to provide a conclusive answer as to whether the amount of feedback is adequate (M=2.8, SD=1.4). In the suggestions for improvement some users found the amount of feedback provided to be too overwhelming *"The help command should be shorter"*, whereas others found it was too little *"provide the names of objects without typing a command"*. Though our *"describe"* command was designed to be concise, the *"help"* command was not as it provides a list of commands including a short description. Future studies could determine the amount of digestible feedback or validate a mechanism that provides feedback up to a user specified word limit.

Many of the suggestions we received included suggestions to add shortcuts to frequently used commands. Most likely shortcuts could improve the efficiency of interaction and exploration, but shortcuts are limited in allowing interaction with multiple objects and avatars. If users would have to interact with multiple objects during our study, the limitations of shortcut-based mechanism would have been apparent to the user. Some users suggested using audio cues to indicate objects or avatars and to create a richer experience. Overall the results of the user studies were encouraging and the suggestions we received are taken into account in the future development of TextSL.

5. DISCUSSION AND FUTURE WORK

The development of TextSL allowed us to identify some of the unique characteristics of virtual worlds and provided valuable insights and experiences with making virtual worlds accessible to individuals who are visually impaired. We identified the following issues and areas for future research:

Meta-information: One of the major problems we identified is the apparent lack of meta-information for many objects in virtual worlds. This is a significant barrier to making virtual worlds accessible to individuals who are visually impaired as they require a textual description to be available. Based on the results of our study (see Section 3) we estimate that about 40% of objects in Second Life lack a descriptive name. We believe this problem is not unique to Second Life and virtual world developers need to be made aware of this as not to exclude individuals who are visually impaired from participating in virtual worlds. A simple but elegant solution would be for virtual world developers to implement a mechanism that prevents users from creating objects without a name. Such a solution does not work for virtual worlds that already exist, which have millions of objects, and a *post-hoc* approach to retroactively label objects must be explored. This problem is similar to web images lacking "alt" tags, -though it may be even more difficult to solve- as the descriptions of web images can sometimes be derived from text surrounding the image or from the filename [16], properties which objects in virtual worlds lack. Manually labeling millions of objects may be tedious, error prone, and susceptible to spam. Because virtual worlds employ a game-like interaction mechanism, we seek to explore using a game based labeling approach, similar to those used in labeling web images [33], to add metadata to objects.

Interface: whether a command-based interface as implemented in TextSL, is better than one based on shortcuts, as used in all related audio games [34, 15, 31, 11], is open for discussion. Our user study shows that a command mechanism is inherently slower, however it allows for efficient interaction with large numbers of different objects and avatars, which is difficult to support using short cuts. Another benefit of a command-based interface is that synonyms of actions are mapped onto the same action. This avoids users having to memorize a large number of commands or shortcuts and may allow for interacting using natural language. Though recently voice over IP was implemented in Second Life, the majority of Second Life users still interact through text chat. As users engage in social interaction most of the time [21] a shortcut based mechanism may require the user to shift the position of their hands when changing from chatting to navigating their avatar. In TextSL, users can keep their hands at the same position and the overhead of having to type in a command as opposed to pressing a shortcut may actually outweigh the overhead incurred in the changing of hand positions on the keyboard, which may be error prone. For these specific reasons we deem a command-based mechanism more applicable to make virtual worlds accessible to users who are visually impaired, though a comparative study of the effectiveness of both interfaces could provide a better insight which mechanism is better.

Feedback: Second Life is very densely populated with objects, and though many objects lack names, some form of aggregation of feedback is required to avoid overwhelming the user. TextSL offers a simple form of summarization but we seek to explore the following mechanisms to develop more concise, but richer, forms of feedback. A larger amount of feedback could be digested through a multimodal approach when a screen reader is used in conjunction with a tactile or Braille display. Alternatively audio such as audio cues could be used to indicate types of objects, if this does not interfere with the use of a screen reader. 3D sound can be used to convey spatial information. Another research direction could be to aggregate textual feedback through grouping and classification of objects using a taxonomy, which could allow for providing a more usable and descriptive form of feedback. For example, instead of *"you see a cat, a cat, and a dog."* TextSL could return *"you see 3 animals"*. This concise form of feedback could then be augmented with spatial or activity related information such as *"3 animals are sitting in front of you"* to create more descriptive forms of feedback.

Functionality: TextSL and other Second life solutions for individuals who are visually impaired miss two important features e.g. the ability to create objects and interact with interactive objects. Certain objects, such as a car, can be made interactive using the Linden Scripting Language. When a user activates an interactive object in TextSL for example by touching it no output is provided as these objects typically exhibit behavior which can only be perceived visually. Our current research efforts are focused on developing a "black box" observer that can describe textually what happens when a user starts interacting with an object e.g. *"the billboard started playing a video"*. TextSL does not support object creation as this is an entirely visual activity, however a command-based interface has the benefit over a short cut based interface that it can support a basic form of parametrized content creation very efficiently e.g. *"set color green"*, *"create cube greencube"* *"rotate greencube X 45"*.

6. CONCLUSIONS

This paper presents the results of developing and evaluating a command-based interface for virtual worlds called TextSL. Virtual worlds are different from games and require a different solution to make them accessible to users who are visually impaired. TextSL allows screen reader users to access the popular virtual world of Second Life using a command-based interface that is inspired by multi-user dungeon games and which allows efficient interaction with large numbers of objects and avatars. User studies with TextSL show that a command-based interface is a feasible approach, as TextSL allows users to explore Second Life, communicate with other avatars, and interact with objects with the same success rates as sighted users using the Second Life viewer. Command-based exploration and object interaction is significantly slower in TextSL, but users can communicate with the same efficiency as sighted users in the Second Life viewer, which is good, as social interaction has been identified as one of the most important aspects of virtual worlds. We identified two major problems: (1) objects in virtual worlds such as Second Life often lack metadata which makes it difficult to provide an accurate textual description; and (2) virtual worlds have large numbers of objects that may easily overwhelm a user when turned into audio feedback. Future research focuses on: adding commands for creating content, exploring how users can interact with scripted objects, exploring the use of a game for adding names to objects lacking metadata and exploring different mechanisms such as multimodal output using haptic feedback and audio cues that may allow users to digest larger amounts of feedback.

7. ACKNOWLEDGEMENTS

This work is supported by NSF Grant IIS-0738921.

8. REFERENCES

[1] Blizzard studios, world of warcraft, http://www.worldofwarcraft.com.

[2] Eductional uses of second life.http://sleducation.wikispaces.com/educationaluses.

[3] Ibm human ability and accessibility center, virtual worlds user interface for the blind, http://services.alphaworks.ibm.com/virtualworlds/.

[4] lib second life, http://libsecondlife.sourceforge.org.

[5] Linden research, secondlife, http://www.secondlife.com.

[6] Second life economic statistics, http://secondlife.com/whatis/economy_stats.php.

[7] Second life region statistics, http://stats.slbuzz.com/sims/browse/.

[8] Virtual guide dog project, http://virtualguidedog.org.

[9] American foundation for the blind (2002), statistics and sources for professionals, http://www.afb.org/info_document_view.asp?documentid=1367. 2008.

[10] Blizzard entertainment, world of warcraft hits 10 million subscribers,http://www.blizzard.com/us/press/080122.html, 2008.

[11] Gma games, shades of doom,http://www.gmagames.com/sod.html, 2008.

[12] Naughty auties' battle autism with virtual interaction, http://www.cnn.com/2008/HEALTH/conditions/03/28/sl.autism.irpt/index.html, 2008.

[13] Virtual ability, http://virtualability.org, 2008.

[14] P. Abrahams. Second life class action, http://www.it-analysis.com/blogs/Abrahams_Accessibility/2006/11/second_life_class_action.html. 2006.

[15] M. T. Atkinson, S. Gucukoglu, C. H. C. Machin, and A. E. Lawrence. Making the mainstream accessible: redefining the game. In *sandbox '06: Proceedings of the 2006 ACM SIGGRAPH symposium on Videogames*, pages 21–28, New York, NY, USA, 2006. ACM.

[16] J. P. Bigham. Increasing web accessibility by automatically judging alternative text quality. In *IUI '07: Proceedings of the 12th international conference on Intelligent user interfaces*, pages 349–352, New York, NY, USA, 2007. ACM.

[17] M. M. Blattner, D. A. Sumikawa, and R. M. Greenberg. Earcons and icons: Their structure and common design principles (abstract only). *SIGCHI Bull.*, 21(1):123–124, 1989.

[18] W. S. Carter and G. D. Corona. Exploring methods of accessing virtual worlds. *American Foundation for the Blind AccessWorld. 9(2)*, 9(2), 2008.

[19] P. Curtis and D. A. Nichols. MUDs grow up: Social virtual reality in the real world. In *COMPCON*, pages 193–200, 1994.

[20] M. Dickey. Three-dimensional virtual worlds and distance learning: two case studies of active worlds as a medium for distance education. *British Journal of Educational Technology*, 36(3):439–451.

[21] M. Griffiths, M. Davies, and D. Chappell. Breaking the stereotype: The case of online gaming. *Cyber Psychology and Behavior*, 6(1):81–91, 2003.

[22] T. Hedgpeth, J. John A. Black, and S. Panchanathan. A demonstration of the icare portable reader. In *Assets '06: Proc. of the 8th int. ACM SIGACCESS conference on Computers and accessibility*, pages 279–280, New York, NY, USA, 2006. ACM.

[23] D. Kirkpatrick. No, second life is not overhyped, http://money.cnn.com/2006/11/09/technology/fastforward/secondlife.fortune/index.htm, 2006.

[24] J. Nielsen. *Usability Engineering*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[25] M. Pascale, S. Mulatto, and D. Prattichizzo. Bringing haptics to second life for visually impaired people. In *EuroHaptics '08: Proceedings of the 6th international conference on Haptics*, pages 896–905, Berlin, Heidelberg, 2008. Springer-Verlag.

[26] H. Petrie, F. Hamilton, N. King, and P. Pavan. Remote usability evaluations with disabled people. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1133–1141, New York, NY, USA, 2006. ACM.

[27] D. Rina and J. Pearl. Generalized best-first search strategies and the optimality of a*. *Journal of the ACM*, 32(3):505–536, 1985.

[28] S. S. Robbins. Immersion and engagement in a virtual classroom: Using second life for higher education. In *EDUCAUSE Learning Initiative Spring 2007 Focus Session*, 2007.

[29] B. Robinson and L. J. Lieberman. Effects of visual impairment, gender, and age on self-determination. *Journal of Visual Impairment & Blindness*, 98:351–366, 2004.

[30] A. Talamasca. The heron sanctuary helps the disabled find a second, http://eurekadejavu.blogspot.com/2008/01/story-of-heron-sanctuary.html, 2008.

[31] S. Trewin, V. Hanson, M. Laff, and A. Cavender. Powerup: An accessible virtual world. In *Assets '08: Proc. of the 9th international ACM SIGACCESS conference on Computers and accessibility*, pages 171–178, New York, NY, 2008. ACM.

[32] U.S.Government. 1998 amendment to section 508 of the rehabilitation act. *SEC. 508. Electronic and information Technology*, 1998.

[33] L. von Ahn, S. Ginosar, M. Kedia, R. Liu, and M. Blum. Improving accessibility of the web with a computer game. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 79–82, New York, NY, USA, 2006.

[34] T. Westin. Game accessibility case study: Terraformers - a realtime 3d graphic game. In *Proc. of the The Fifth Int. Conference on Disability, Virtual Reality and Associated Technologies*, 2004.

[35] G. R. White, G. Fitzpatrick, and G. McAllister. Toward accessible 3d virtual environments for the blind and visually impaired. In *DIMEA '08: Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts*, pages 134–141, New York, NY, USA, 2008. ACM.