University of the Philippines Manila
College of Arts and Sciences
Department of Physical Sciences and Mathematics

acquilaVR: Teaching Activities of Daily Living To School-aged Children With Autism Using Virtual Reality Therapy

A special problem in partial fulfillment
of the requirements for the degree of
Bachelor of Science in Computer Science

Submitted by:

Rai Earl Don S.J. Quito

June 2018

ACCEPTANCE SHEET

The Special Problem entitled "*acquilaVR*: Teaching Activities of Daily Living To School-aged Children With Autism Using Virtual Reality Therapy" prepared and submitted by Rai Earl Don S.J. Quito in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science has been examined and is recommended for acceptance.

Marvin John C. Ignacio, M.S. (*cand.*)
Adviser

EXAMINERS:

	Approved	Disapproved
1. Gregorio B. Baes, Ph.D. (*cand.*)		
2. Avegail D. Carpio, M.S.		
3. Richard Bryann L. Chua, Ph.D. (*cand.*)		
4. Perlita E. Gasmen, M.S. (*cand.*)		
5. Marvin John C. Ignacio, M.S. (*cand.*)		
6. Vincent Peter C. Magboo, M.D., M.S.		
7. Geoffrey A. Solano, Ph.D. (*cand.*)		

Accepted and approved as partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science.

Ma. Sheila A. Magboo, M.Sc.
Unit Head
Mathematical and Computing Sciences Unit
Department of Physical Sciences
and Mathematics

Marcelina B. Lirazan, Ph.D.
Chair
Department of Physical Sciences
and Mathematics

Leonardo R. Estacio Jr., Ph.D.
Dean
College of Arts and Sciences

Abstract

Autism Spectrum Disorder is a developmental disability characterized by social ability impairments and atypical patterns of behaviors. Teaching daily living skills to individuals with this disorder can be difficult in the real world as uncontrolled variables may cause unexpected scenarios to occur. A virtual reality application can be used to simulate real-world scenarios in a safe and controlled environment and can allow the individual to safely explore and interact with the environment, cultivating better skills-acquisition and learning.

Keywords: Autism Spectrum Disorder, Adaptive Behavior, Community-based Skills, Socialization, Virtual Reality Therapy

Contents

List of Figures

I. Introduction

A. Background of the Study

Autism Spectrum Disorder (ASD) is the name for a group of developmental disorders that include wide range, or a ”spectrum”, of symptoms, skills, and levels of disability [1]. People with ASD often exhibit characteristics in three core areas: social-interaction difficulties, communication challenges and a tendency to engage in repetitive behaviors [2]. Children with ASD may exhibit characteristics such as appearing withdrawn, appearing indifferent to other people, preferring to play alone and be difficult to comfort. They may also accept contact if initiated by others, approach other children in an unusual way, and use overly formal-language and be ’rule-bound’ [3]. However, these characteristics of autism vary widely in severity from one person to another. Some people with ASD are mildly impaired by their symptoms, while others are severely disabled [1]. The basic symptoms of autism are often accompanied by other medical conditions and challenges, which vary widely in severity as well.

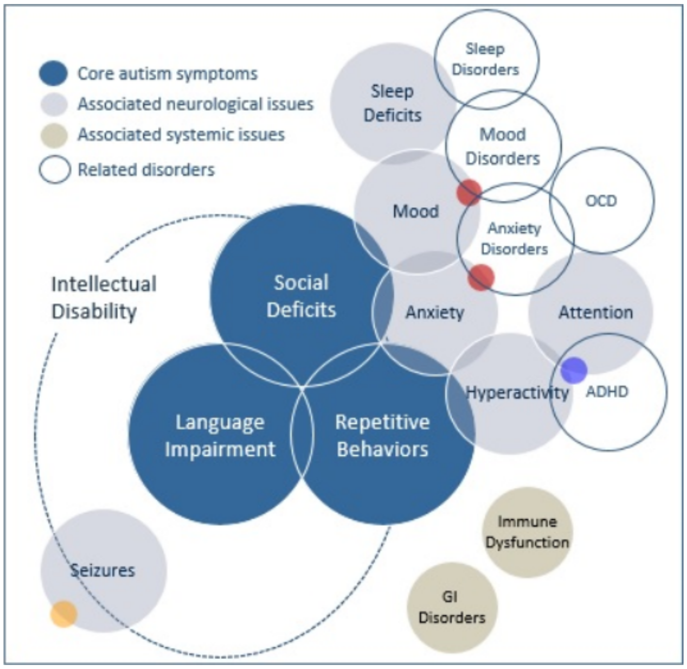
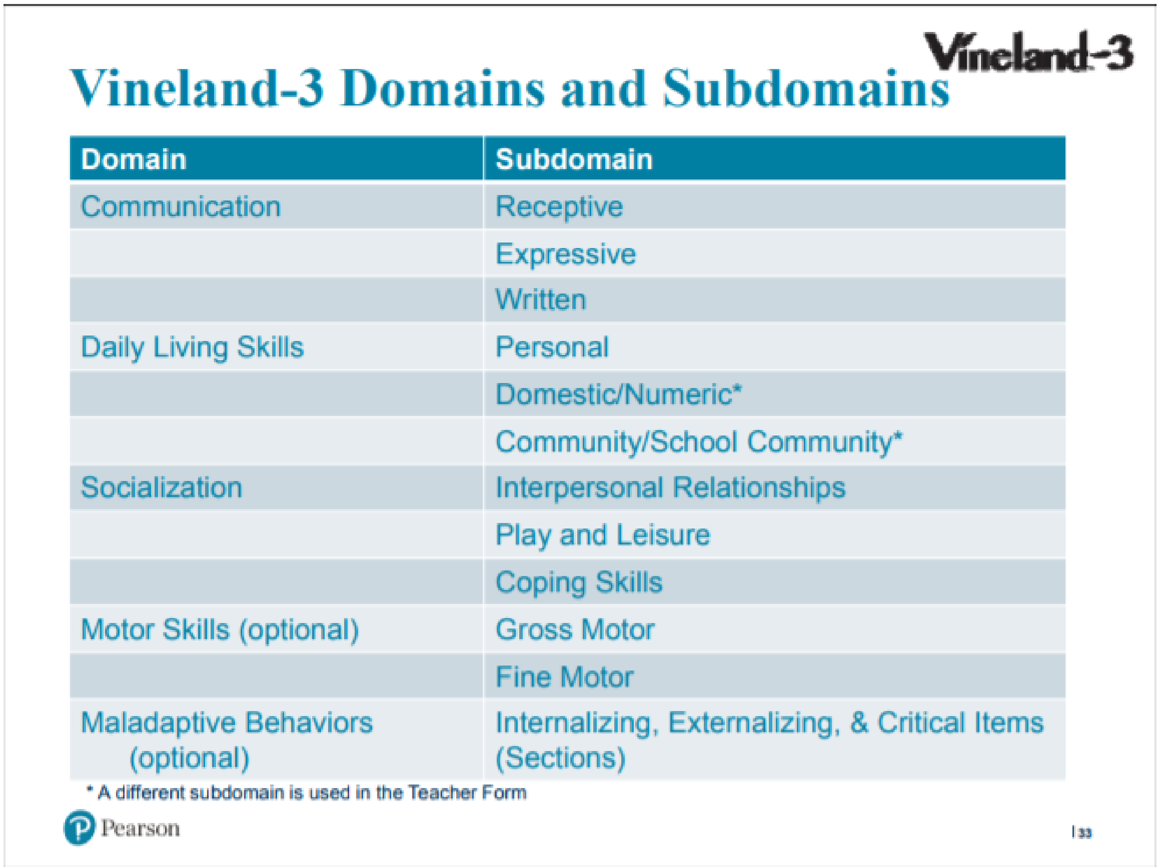


Figure 1: Symptoms of Autism and Other Medical Conditions and Challenges

According to the Centers for Disease Control and Prevention (CDC), the leading national public health institute of the United States, an estimated 1 in 68 children have been identified with ASD as of the year 2014. They also found that studies in Asia, Europe, and North America showed an average prevalence of 1% to 2% for individuals with ASD [4]. In that same year, Erlinda Borromeo, the president of the Autism Speaks Foundation, stated that the number of cases of autism in the Philippines has risen to an estimated one million individuals. However, this estimate could be even higher, as she also mentions that many parts of the Philippines were not covered by the estimate [5]. Diagnosing ASD can be difficult, as there is no medical test, such as a blood test, that can diagnose ASD. Doctors instead review a child's behavioral development to reach a conclusion. At times, ASD can be detected at 18 months or younger, and by age 2 a diagnosis from an experienced professional is considered very reliable. There are two steps to diagnosing an ASD: developmental screening, which involves regular short tests to tell if children have learned the basic skills they should at their current age, and comprehensive diagnostic evaluation, which is a thorough review of a child's behavior and development through interviewing the child's parents and medical testing [6]. For patient evaluation, psychologists and psychiatrists use the diagnostic criteria for ASD as it appears in the fifth edition of the Diagnostic and Statistical Manual of Mental Disorders (DSM-5)[7]. The product of more than 10 years of effort by hundreds of experts in the field of mental health, the DSM-5 is the authoritative manual that defines and classifies mental disorders for the purpose of improving diagnoses, treatment, and research [8]. Beyond these diagnoses, people with ASD are often labeled as being either "high functioning" or "low functioning." These labels are used to describe people with autism based on the degree of their similarity to a person not on the autism spectrum. For example, a person with ASD who uses spoken language to communicate would most likely be termed as "high functioning". Meanwhile, a person with ASD that has limited or no spoken language and would rather use technology or picture boards to communicate would be labeled as ”low functioning.” These descriptions can be misleading, however, as they do not necessarily describe a person’s intelligence, talents, anxiety, or sensory abilities and are instead being defined by their similarity to ”normal” people [9].

One critical aspect of a person’s behavior is known as their adaptive behavior. Grossman defines adaptive behavior as ”the effectiveness or degree with which an individual meets the standards of personal independence and social responsibility expected for age and cultural group. [10]” These include simpler activities such as wearing glasses and hand washing, to more complex and community-based skills such as pedestrian safety, purchasing skills, and commuting from place to place. A primary consideration for selecting an individual’s adaptive living goals should be the age-appropriate skills needed to promote independence in the community. An example would be children that haven’t been toilet trained being less likely to have access to classrooms with peers that are normally developing. To assess the adaptive behavior of an individual, adaptive behavior scales have been developed. Behavior scales are used to support diagnosis of intellectual and developmental disabilities, determine eligibility of an individual for special education, plan rehabilitation or intervention services, and track and report the progress of a patient. One such behavior scale is the Vineland Adaptive Behavior Scales (VABS). VABS was first developed by Sparrow, Balla, and Cicchetti in 1984, and has undergone 3 revisions since then. The latest revision, Vineland-3, was developed in 2016 and featured 5 adaptive domains to score patients on: the communication domain, the daily living skills domain, the socialization domain, the optional motor skills domain, and the optional maladaptive behaviors domain. These domains were also separated into several subdomains under each main domain [11]. Using these adaptive behavior scales is particularly important for children with autism as it provides a measure of a child’s ability in generalizing teaching across different settings, helps therapists target which areas are in need of skills acquisition, can contribute to a diagnostic evaluation, and is essential in developing interventions and finding if an individual is eligible for some services and treatments [10][11].



Vineland-3

Vineland-3 Domains and Subdomains

Domain	Subdomain
Communication	Receptive
	Expressive
	Written
Daily Living Skills	Personal
	Domestic/Numeric*
	Community/School Community*
Socialization	Interpersonal Relationships
	Play and Leisure
	Coping Skills
Motor Skills (optional)	Gross Motor
	Fine Motor
Maladaptive Behaviors (optional)	Internalizing, Externalizing, & Critical Items (Sections)

* A different subdomain is used in the Teacher Form

Pearson

| 33

Figure 2: Vineland-3 Domains and Subdomains

Many treatment and intervention methods have been developed to teach adaptive behavior skills and other critical life skills to individuals with ASD, as well as to help reduce the symptoms a person with ASD might exhibit. The CDC breaks down the different types of treatments into 4 categories. The first category is the Behavior and Communication Approaches. These approaches provide structure, direction, and organization in addition to family participation. One notable treatment approach that falls under this category is Applied Behavior Analysis (ABA), which encourages positive behaviors and discourages negative behaviors in order to improve a variety of skills in the patient. Other treatments in this category include occupational therapy, speech therapy, the DIR or developmental, individual differences, relationship-based approach, and sensory integration therapy. The second category is Dietary Approaches. Approaches in this category are based on the idea that ASD symptoms can be caused by food allergies or a lack of vitamin and minerals. The third category of approach is through Medication. Medications can help people with symptoms that are related to ASD, such as managing high energy levels, inability to focus, depression, and seizure. Finally we have the fourth category, Complementary and Alternative Medicine (CAM). These are the treatments that are not typically recommended by a pediatrician [12]. Despite being somewhat controversial, a study in 2011 found that between 30 to 95% of children with ASD have been provided and treated with CAM [13]. However, Dr. Robert Hendren, D.O., a leading researcher in the field of CAM treatments for autism and the director of child and adolescent psychiatry at the University of California, San Francisco, emphasizes that CAM therapies should be used to complement proven behavioral treatments, not replace them. He also says that ”...because autism encompasses a complex group of disorders, some treatments may work for one person but not another” [2]. Out of all the aforementioned treatments, ABA is shown to be the most effective. However, factors such as the lack of enough trained therapists to handle the increasing number of children diagnosed, as well as poor access to effective treatments make it difficult for the majority of the ASD population in the Philippines to undergo these intensive behavioral interventions [14].

In this regard, innovative technology promises alternative and assistive therapeutic paradigms in increasing intervention accessibility. One of these new forms of therapy is based upon virtual reality [15]. Virtual reality (VR) is the creation an artificial environment that is presented to the user in such a way that the user suspends belief and accepts it as a real environment [16]. Implemented using computer technology, a range of systems have been developed to be used for this purpose. Headsets, omnidirectional treadmills, and special haptic gloves are used to actually stimulate our senses to give an illusion of reality [17]. In ASD therapy, virtual reality is a new addition that is used to help autistic adults and children develop social and lifestyle skills necessary for independence. Virtual environments are ideal for educating autistic individuals as it allows them to learn these skills in a safe and controlled environment [18]. A good example of a VR system used for treatment of ASD was developed in 2008 and led by researchers Prof. Naomi Josman and Prof. Tamar Weiss from the University of Haifa in Israel [19]. The researchers developed a desktop street-crossing virtual environment to determine whether the skills learned from the simulation would transfer into real-life. The system ran on a Pentium 3 desktop computer, was displayed on a 17-inch LCD monitor, and featured 9 stages that were graded on difficulty. Participants were told to use three keyboard keys that corresponded to changing the user's view from left to right and initiating street-crossing. Using a scored pedestrian safety checklist, they assessed the street crossing skills of children with ASD aged between seven and twelve on a protected real sidewalk before and after VR intervention. Significant differences were found in the performance between their experimental and control groups, and moreover the experimental group showed substantial improvement after the VR intervention, proving that ASD children are capable of learning real-life skills in virtual environments [20]. A more recent VR system used for ASD intervention is Floreo VR. This system was developed for mobile devices making it an accessible and affordable therapy supplement, has supervision functionalities allowing parents and therapists to monitor and guide the progress of the user, and has automated data collection and analysis [21].

B. Statement of the Problem

While it is true that some daily living skills can be taught at home or at a clinic, interacting with the real world can still prove to be too intimidating to children with autism. Some children also have problems with tactile sensations when introduced to new objects and first have to undergo systematic desensitization before they can interact with an object.

Existing VR systems to teach these skills such as Floreo VR are available in other countries. However there are no such systems available in the local setting of the Philippines. A similar VR system to teach skills would be beneficial as there are few facilities for teaching daily living-based therapy in the Philippines. Also, in Floreo VR, the child is unable to walk around the virtual environment. Not being able to explore their environment may inhibit a child's growth when presented with some activities of daily living that rely on their gross motor skills.

C. Objectives of the Study

This research has created a Virtual Reality application with the following functionalities:

(1) Allows the patient to:

- (I) Learn how to navigate the virtual environment and interact with objects through a tutorial.
- (II) Navigate a virtual reality environment modeled after a common household home with the following areas:
  - (A) a garden,
  - (B) a kitchen,
  - (C) and a play area.

(III) Accomplish the following goal in each area of the virtual environment.

   (A) Water flowers in the Garden.

      (i) Open a Faucet to fill a Watering Can with water.

      (ii) Pick up the Watering Can.

      (iii) Water the plants in the first Flower Patch.

      (iv) Water the plants in the second Flower Patch.

      (v) Water the plants in the third Flower Patch.

      (vi) Return the Watering Can to the Parent.

   (B) Make a sandwich in the Kitchen.

      (i) Get a clean plate from the Kitchen.

      (ii) Make a Sandwich by following a sequence of ingredients:

         (a) Bread,

         (b) Ham,

         (c) Cheese,

         (d) Tomato,

         (e) Lettuce, and

         (f) Bread

      (iii) Give the Sandwich to the parent so they can eat it.

      (iv) Pick up the Dirty Plate.

      (v) Put the Dirty Plate in the Sink.

   (C) Pick up toys in the Play Area.

      (i) Pick up Bear plush toy.

      (ii) Put the Bear toy in the Toy Chest.

      (iii) Repeat the previous two items with the other toys in the environment:

         (a) Penguin,

(b) Rabbit,

(c) Sheep and

(d) Monkey.

(IV) Interact with objects and individuals in the virtual environment.

(A) Have interactive dialogue with the virtual ”Dad” avatar.

(i) Ask ”Dad” avatar for tips in accomplishing current goal.

(ii) Follow instructions from ”Dad” avatar.

(iii) Choose from a Dialogue Box which activity they want to do.

(B) Pick up certain objects.

(C) Store viable objects in an inventory.

(V) Follow an arrow to guide them towards their current goal.

(VI) See the names of certain objects in the virtual environment through a Label interface.

D. Significance of the Project

Using a virtual reality application, we can simulate a real-world situation for the user, but without the actual dangers involved. Since the users will be in a controlled environment, teaching a child with ASD daily living skills will be much easier as no unknown variables will be present during the session. This is more difficult to accomplish in a real world setting. The application can also be a form of desensitization for children who are uncomfortable in unfamiliar environments and activities. This can be a venue for them to generalize the skills they've learned until they are ready for the real activity. The patient will be also be able to hone their critical thinking skills and learn to be more independent by being given a choice in which objective they will pursue and by exploring the environment on their own. Also, though the project focuses on teaching skills that fall under the daily living skills domain of Vineland-3, it also helps teach receptive and expressive skills from the communication domain and gross motor skills from the motor skills domain.

E. Scope and Limitations

(1) This project mainly involves teaching Activities of Daily Living (ADLs).

(2) This project simulates a realistic home setting with only 3 areas:

  (I) a garden,

  (II) a kitchen,

  (III) and a play area.

(3) This project is designed for school-aged patients of 9+ years old.

(4) The project is developed for smartphones with Android version 5 (Lollipop) or higher, there are no plans to develop for IOS or Windows phones as of yet.

(5) The project is built for mobile VR devices such as Google Cardboard, rather than the powerful Desktop-based VR devices such as Oculus Rift or Vive, as such the application's visual quality will be limited due to the lack of processing power.

(6) This project is not suitable for children who are uncomfortable or become stressed due to tactile sensations when wearing headgear.

(7) This project does not record the patient's progress throughout the session.

F. Assumptions

(1) The patient can turn off the application if the application causes too much stress to the user.

(2) The patient should have a level of expressive and receptive language enough to relay information and ask for assistance from the therapist during the session.

(3) The patient should be familiar enough with common household items such as those used in this application.

(4) The patient should fall under the Mild to Moderate levels of the Autism Spectrum as they have a relatively high level of cognitive ability.

(5) The patient should be able to follow simple instructions and navigate in a familiar environment.

(6) The therapist should be present to monitor and guide the player during the session.

(7) The therapist will be given information on what scenarios and experiences can occur in the simulation beforehand to better guide the player forward.

II. Review of Related Literature

In recent years, virtual reality applications have found much use for ASD intervention. According to a study by Shoaib et al. [22], virtual reality applications are being considered as the best method to monitor ASD intervention. Achieved through everyday VR-based human computer tasks, the journal states that VR technology possesses various qualities like flexibility, controllability, dependability, modifiable tactile incitement, the capacity to individualize intervention methodologies and support systems. Children as well as adults with autism also find communication with computers to be safer.

Many VR applications for ASD intervention focus on training the social skills of individuals with ASD.

In 2016, one study from the University of Texas at Dallas [23] investigated the impact of VR social cognition training to enhance the social skills of children with ASD. They subjected 30 participants with ASD or a combination of ASD and Attention Deficit Hyperactivity Disorder (ADHD) of ages between 7 and 16 to a variety of social scenarios in a virtual environment in designed to emphasize a social learning objective. These objectives included meeting new people, bonding with friends, dealing with a bully, and other social dilemmas constructed to represent common social situations that children face. A novel aspect for this study was that each session had 2 age-matched children with High-Functioning Autism (HFA) which were monitored by a trained clinician that would offer feedback throughout the interaction. The VR environment was designed in the game Second Life and was presented on a 24-inch monitor. They concluded that a VR-SCT offers a feasible social cognitive intervention alternative for children ages 7 years and older as they found that the participants showed significant increases on affect recognition, intentionality, and analogical reasoning tests. They also concluded that with a peer as well as a trained clinician (as compared to an artificial intelligence) an individual can practice a wide variety of enhanced social encounters with unpredictable consequences, which made for more realistic, individualized, and meaningful interactions.

Mourning and Tang from Rowan University in Glassboro, New Jersey [24] developed a similar social training application for adolescents with HFA in October of the same year. Their game immersed individuals in a virtual mall environment wherein they could interact with various characters and accomplish social tasks such as completing dialogue with the in-game characters. The application uses an adaptive system that logs player actions and can provide effective instructional responses to aid the player if they are performing poorly or are struggling to understand the information they receive through the virtual social interactions. This adaptive system was comprised of 3 components: a neural network that was trained to recognize behavioral patterns that characterize player performance, an instruction effectiveness prediction component that is used to predict how well the player will perform after instructional intervention based on current performance and the state of the virtual environment at the time of the intervention, and a instruction selection component to determine which instruction, if any, should be displayed to the player. Their results found that an adaptive training system has the potential to become an effective and efficient solution to teaching problems, with more sophisticated systems required for providing a more appropriate instruction to some individuals.

Also in 2016, focusing on social adaptation training for school-aged children diagnosed with ASD in their transition from kindergarten to primary schools, Ip et al. [25] presented a VR-enabled system with six unique VR scenarios. Four of those scenarios covered various social encounters in typical school life, the fifth scenario focused on helping children generalize what they've learned in those four previous scenarios, and the last scenario was designed to relax and get participants used to the VR environment. The study used an immersive four-sided computer-assisted virtual environment (CAVE) with head tracking, and during training sessions a trainer would be present to control the learning content as the child would interact with the environment via body gestures and communication with the trainer. Significant differences were found in emotion recognition and social reciprocity assessment tests they ran before and after training, showing the potential of using VR and its enabling technologies in for ASD training.

Volioti et al. conducted a study in 2014 [26] that focused on applying educational research on individuals with ASD using a virtual learning environment (VLE) based on the use of Social Stories (SS) for the purpose of teaching children diagnosed with ASD how to solve social problems. The VLE used in this study included a realistic representation of school premises in Greece and a typical classroom which featured an interactive whiteboard. They also used a virtual female avatar to represent the teacher for the VLE. Four SS that were deemed familiar scenarios occurring at home or at school were made: 1) breaking a vase at home which caused parents to become upset and accusatory towards the child, 2) a teacher commenting that a classmate's efforts were not enough for their math test which angered the classmate, 3) choosing between following what their parents asked him to do (i.e. tidy up their room) or doing what they desire to do (i.e. ride on their bike), and 4) dealing with a classmate that wanted revenge on another child for breaking their favorite object. Each problem required the participant to accomplish three steps: a) to stay calm, b) to explain what they wanted to do/correct the other person on what is the right thing to do, and c) to carry out the solution. A rewarding auditory and visual message appears on completion of the appropriate step, otherwise a "help" message would appear. After interviewing 43 special education (SPED) teachers and gathering their evaluations of the application, they concluded that the application they designed and developed was usable for SPED teachers and students with ASD. The SPED teachers also regarded the chosen SS as suitable, useful, having logical sequence, and providing of appropriate feedback for the participants, however they would prefer if there was a rich SS database from which they could select the appropriate SS according to a child's needs.

One critical social skill an individual must learn is emotional and facial recognition. In 2013, a study conducted by Bekele et al. from Vanderbilt University in Nashville, TN [27] developed an innovative VR-based facial emotional recognition system to help in understanding the fundamental mechanism of emotion recognition in individuals with ASD. The study consisted of two phases, an online phase which consisted of eye tracking, physiology monitoring, and stimuli presentation and an offline phase which consisted of offline data processing and analysis. The online phase was composed of 3 major applications running separately and communicating via a network: an eye tracking application, the physiological monitoring application, and the VR task presentation engine developed using Unity. Presented on a 24" flat LCD panel monitor, 28 tasks which corresponded to 7 emotional expressions with 4 intensity levels each were presented to 10 high functioning subjects with ASD and a typically-developing control group from the local community. Results did not suggest powerful differences between ASD and the typically-developing (TD) group, but some interesting differences were found in how facial expressions were processed and decoded between the two groups. They found that adolescents with ASD payed significant attention to context irrelevant areas such as the forehead as compared to the TD group which focused more on the context relevant regions of interest (ROI) such as the mouth, and that adolescents with ASD spent more time examining faces presented to them before responding and were often less confident in their answers.

3 years later in 2016, Bekele et al. [28] produced another study that focused on tracking and collecting physiological signals and data for the purpose of providing further insight into the emotion recognition process of children with ASD. They presented the design, development, and a usability study based evaluation of a system that in the presence of a conversational social context can present controlled facial emotional

expressions, the Multimodal Adaptive Social Interaction in VR or MASI-VR. The system tracked eye gaze for use in generating patterns for their online gaze-sensitive adaptation as well as collected electrophysiological signals including peripheral physiological signals and electroencephalography (EEG) data which are related to emotion recognition. Similar to their previous study, they used a 24" flat LCD panel monitor to present 28 trials to two groups, a control group without ASD and a group of high functioning subjects with ASD. The virtual environment they employed for this study was a typical school cafeteria filled with dining student avatars. However, their system differed from existing systems at the time due to employing the online gaze-based adaptation mechanic that occluded the face and would slowly reveal parts of the face as the subject would look upon context relevant ROI, as well as use of a question-answer conversational dialog which helped teach proper emotion processing in social interactions with the virtual avatars. Furthermore, they randomly assigned half of their sample size to the gaze-sensitive part of the system with the remaining half used as controls without the online gaze feedback mechanism. Analysis of their usability study results from pre-test and post-test indicate that the gaze-sensitive group was able to lower the performance gap between them and the control group by more than 3% after the training while maintaining similar engagement to the control group, proving the usefulness of the application for observing and possibly training children with ASD in their core deficit areas for social functioning.

Another application, named eMot-iCan, was developed by Sturm, Peppe, and Ploog of the College of State Island in 2016 [29] which focused on emotion recognition in individuals with ASD as well. Though the application did not employ VR, many of the functionalities created in this application could be reused in a virtual environment and could be developed further in a VR application. They developed a serious game designed for the purpose of assessing and remediating social accessibility in these individuals. The application implemented a matching-to-sample (MTS) paradigm, where a sample stimulus is presented followed by similar stimuli to compare and choose from. In this game, the stimuli chosen are photos of faces divided into top (T) and bottom (B) portions which could be positive (+) or negative (-) e.g. happy eyes and smiling mouth = T+ and negative eyes and mouth = B-. Players then underwent trials in which they had to match the correct stimuli, and selection of the correct match results in a reward in game points, tokens, an entertaining video clip, or a fun brief computer game or puzzle. The system also had 3 types of custom script building that allowed researchers to design trials for the specific needs of their players. This enabled hypothesis testing to be as objective as possible.

VR applications for teaching daily living skills such as public transportation and shopping have also been developed and researched on.

In 2015, Bernardes et al. [30] addressed the need to prepare individuals with ASD for the real-life scenario of using public transportation to get around. They developed a serious head mounted display (HMD) VR game in which participants must navigate a three-dimensional city and accomplish tasks which involve taking the bus to specific destinations for the purpose of facilitating independence and training players to become more comfortable with the process of commuting via bus. Participants for the study comprised of an experimental group which had five individuals with ASD of an average age of 32 years and a control group of individuals without a history of ASD. The game also monitored and recorded the user's progress as well as their time to complete each task. From their results they found that participants with ASD took an average of 1.89 times longer to complete a task than the control group, with an average time of 8.77 minutes for participants with ASD as compared to an average time of 4.62 for control participants. This difference in performance reveals that ASD travel training is a viable target for a serious game intervention.

Lamash et al. in 2017 [31] examined the effectiveness of a meta-cognitive intervention program that uses virtual reality training to improve an adolescent with ASD's performance in carrying out a shopping task. They used the Virtual Action Planning Supermarket (VAP-S) as the environment for the study, which simulates a fully textured, medium-sized supermarket environment with a diverse array of products and four cashiers. The VAP-S can also record user positions and provide an overview of a user's shopping trajectory and stops. Here, the user would perform similar processes and tasks to a normal real-life supermarket. Actions such as finding objects from a grocery list in the store as well as paying and exiting the establishment are available. 56 adolescent participants with ASD were chosen for the study, with ages between 11-19 years old, 33 were picked for the intervention group with the remaining 23 used as the control group. 3 assessment tests were performed on the participants pre and post-training, WebNeuro software, the Behavior Rating Inventory of Executive Function - Self Reported version (BRIEF-SR), and the Test of Grocery Shopping Skills (TOGSS). Analysis of the results of the assessment exams indicate that the intervention program created was significantly effective, with the individuals in the intervention group showing improvement in both the WebNeuro components as well as the TOGSS examination indices as compared to the control group. However, no significant differences were observed for the BRIEF-SR questionnaire after the intervention, though this was attributed to the dependence of the evaluation on the awareness capability of the adolescents and the time needed for essential changes in the self-perception of an individual. Despite this, the study is still meaningful for the development of future intervention and assessment programs.

III. Theoretical Framework

A. Autism Spectrum Disorder

Autism Spectrum Disorder (ASD) is a developmental disability that affects how a person would see the world and how they would interact with others [32]. A person with ASD is characterized by exhibiting atypical behavioral patterns and possessing impaired social skills, and children with ASD may have unusual characteristics such as being 'rule-bound', speaking in an overly formal manner, or appearing withdrawn or indifferent to other people [3]. In 2014, it was estimated that 1 in 68 children worldwide and more than 1,000,000 people in the Philippines have been diagnosed with ASD [4]. Being a spectrum condition, autistic people share the same certain difficulties but how much and in what way these difficulties would affect a person is different from individual to individual. In some cases, learning disabilities, mental health issues or other conditions and ailments could also be present in the individual with ASD. All people on the spectrum can develop and learn, but each person will need different and appropriate methods and levels of support to improve their daily living [33].

B. Adaptive Behavior

Adaptive Behavior skills are defined as the practical, everyday skills needed by an individual person to meet the demands of one's environment, to independently take care of oneself, and to interact with other people [10]. Although the range of adaptive behaviors can be defined broadly, almost all categorizations of these behaviors include a focus on self-care skills such as eating and sleeping. Other categorizations pertain to home and community living skills such as grooming and safety, and school-related skills such as academic behaviors and play skills [11].

B.1 Vineland Adaptive Behavior Scales

The Vineland Adaptive Behavior Scales (VABS) are one of the most widely used instruments to assess adaptive behavior and is used to assess adaptive behavior of people aged 0-90. The latest revision of VABS, the Vineland-3, focuses on assessing skills that fall within 5 domains, and 11 subdomains. The first domain, Communication, is composed of the Receptive, Expressive, and Written subdomains. The second domain, Daily Living Skills, is composed of the Personal, Domestic, and Community subdomains. The third domain, Socialization, is composed of the Interpersonal, Play/Leisure, and Coping Skills subdomains. The optional fourth domain, Motor Skills, is composed of Gross Motor and Fine Motor subdomains. Lastly, the optional fifth domain, Maladaptive Behavior, is composed of three areas of maladaptive behavior: internalizing, externalizing, and critical items.

C. Applied Behavior Analysis

Applied behavior analysis (ABA) is the use of these techniques and principles to bring about meaningful and positive change in behavior. It is widely recognized as a safe and effective treatment for autism, with a number of a completed studies demonstrating that ABA techniques can produce improvements in communication, social relationships, play, self care, school and employment [34]. Below are a few concepts and techniques commonly used in ABA.

C.1 Task Analysis

Task analysis involves breaking complex skills into smaller, teachable units, the products of which is a series of sequentially ordered steps or task [35]. Pictures and words are made for each step of the task, and the child is prompted through each step of the entire task to help them learn [36].

C.2 Chaining

”Chaining” is how task analyses are created. This involves breaking down a complex skill down into step by step components. Each step is repeated until the child has mastered the step and afterwards are chained into a larger whole [35].

C.2.1 Forward Chaining

In forward chaining, the first step in a complex task is taught and reinforced after the task is completed. The following steps are added one at a time until the entire chain has been trained and the child has mastered the skill [36].

C.2.2 Example of Forward Chaining

Objective: Use soap and water to wash his/her hands.

For Forward Chaining, you would say/prompt:

(1) Go to the sink

(2) Turn on the water

(3) Put two squirts of soap on your hands

(4) Rub your hands together under the water

(5) Make sure you get the soap rinsed off

(6) Turn the water off

(7) Go to the towel

(8) Rub your hands dry on the towel

C.2.3 Backward Chaining

Backward chaining involves the therapist/teacher working through each step of the task with the child from start to finish. Then, training is begun by focusing on the last step first. Once the child starts completing the last step consistently and independently, we add the second to the last task to the chain and continue training. This is repeated until the entire chain has been trained and mastered [36].

C.3 Systematic Desensitization

Developed by Wople during the 1950s, systemic desensitization is a type of behavioral therapy based on the principle of classical conditioning. This therapy aims to remove the fear response of a phobia, and substitute a relaxation response to the conditional stimulus by counter conditioning the patient over time. There are three phases in the treatment: First, the patient is taught relaxation techniques and exercises to relieve tension. Second, a fear hierarchy is created (either by the patient themselves or by the therapist) starting with the stimuli that create the least amount of anxiety and fear to the most fear provoking images. Third, the patient works their way up the created fear hierarchy, moving up only when they feel comfortable enough at their current level. If they become upset they can return to an earlier stage and regain their relaxed state.

The client repeatedly imagines (or is confronted by) this situation until they experience no more anxiety. This indicates that the therapy has been successful. This process is repeated with each level of the anxiety hierarchy until no more anxiety is evoked from the patient [37].

C.4 Picture Exchange Communication System

The Picture Exchange Communication System or PECS approach is a modified ABA program designed for the training of early nonverbal symbolic communication. It is an effective program used to assist individuals with ASD in becoming more effective communicators [38]. People using PECS are taught to approach another person and give them a picture of a desired item in exchange for that item, thus, initiating communication [39].

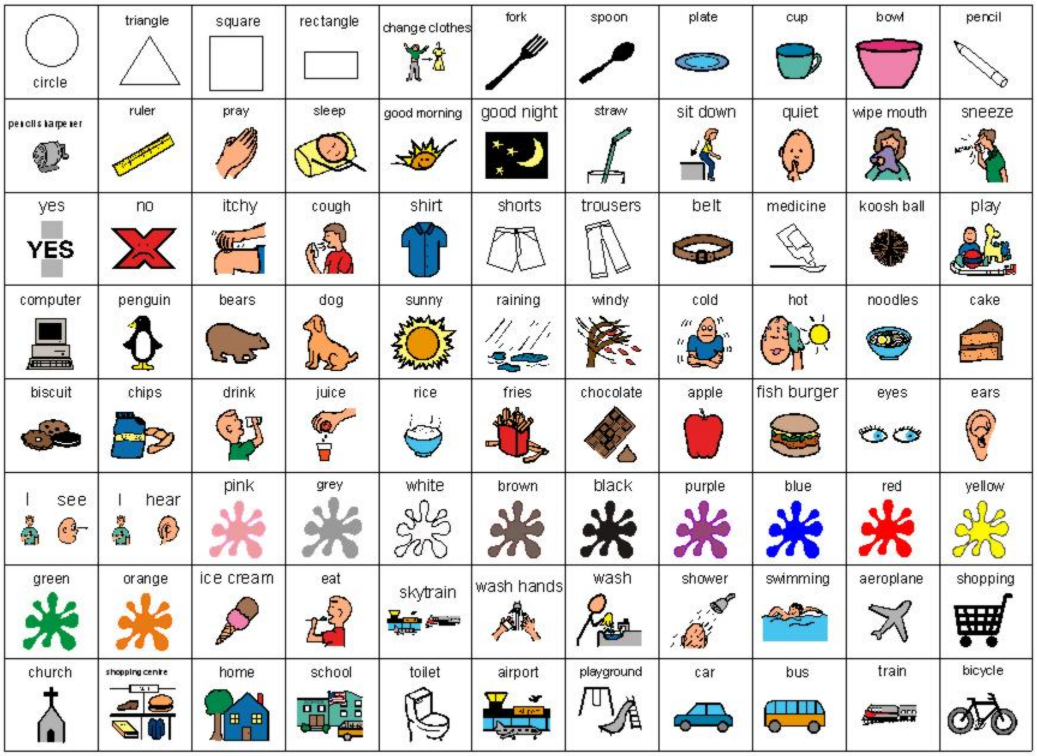


Figure 3: An example PECS.

D. Virtual Reality

Virtual Reality (VR) refers to the use of computer modeling and simulation to enable an individual to interact with an artificial three-dimensional environment. Interactive devices such as goggles, headsets, and haptic gloves and suits provide feedback to the user, immersing them and giving an illusion of ”being there”, also known as telepresence. VR has a wide range of applications which include training or education in fields such as architecture, sports, and medicine or even for recreational activities in the form of video games or artistic applications [40].

D.1 Head-mounted Display

A Head-mounted Display (HMD) is a computer monitor/display device that is worn on the head as goggles or as part of a helmet [41]. Associated with VR, they are meant to be worn for total immersion of the user as typically any movement made by the user would reflect in the virtual experience they are viewing [41][42]. Besides use in VR gaming, they've also seen use for training in military, medical, and engineering contexts [42].

D.1.1 Google Cardboard

Google Cardboard was developed as a complete VR platform that primarily uses foldable and low-cost viewers. The viewers are designed with cardboard lenses and a magnetic or capacitive-taped lever to operate the screen. The user runs a Cardboard-compatible app on their phone inserts their device into the viewer. There the content can be viewed through the lenses in VR [43].

D.2 Google Virtual Reality Software Development Kit

The Google Virtual Reality Software Development Kit (Google VR SDK) is used to build apps that can display 3D VR environments. Users make use of stereoscopic rendering, spatial audio, and head-tracking and movement effects in building VR-

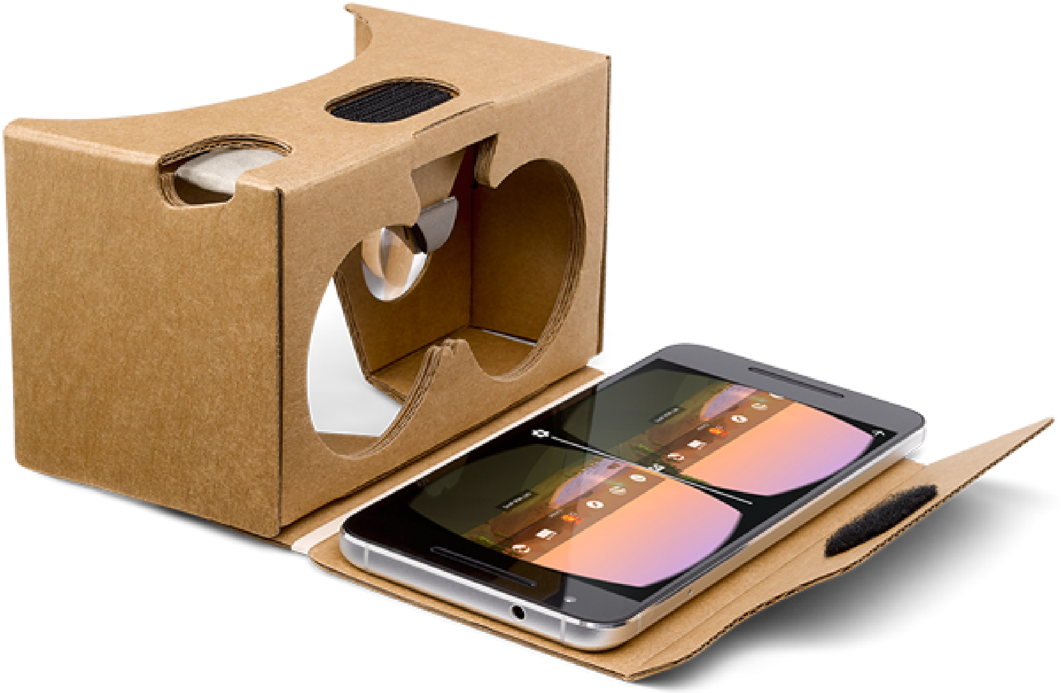


Figure 4: The Google Cardboard with a smartphone displaying a VR application.

enabled applications. SDKs are available for Android, Unity, Unreal Engine, iOS, and for embedded 360 degree VR View media for websites and other applications [44].

E. Mobile Application

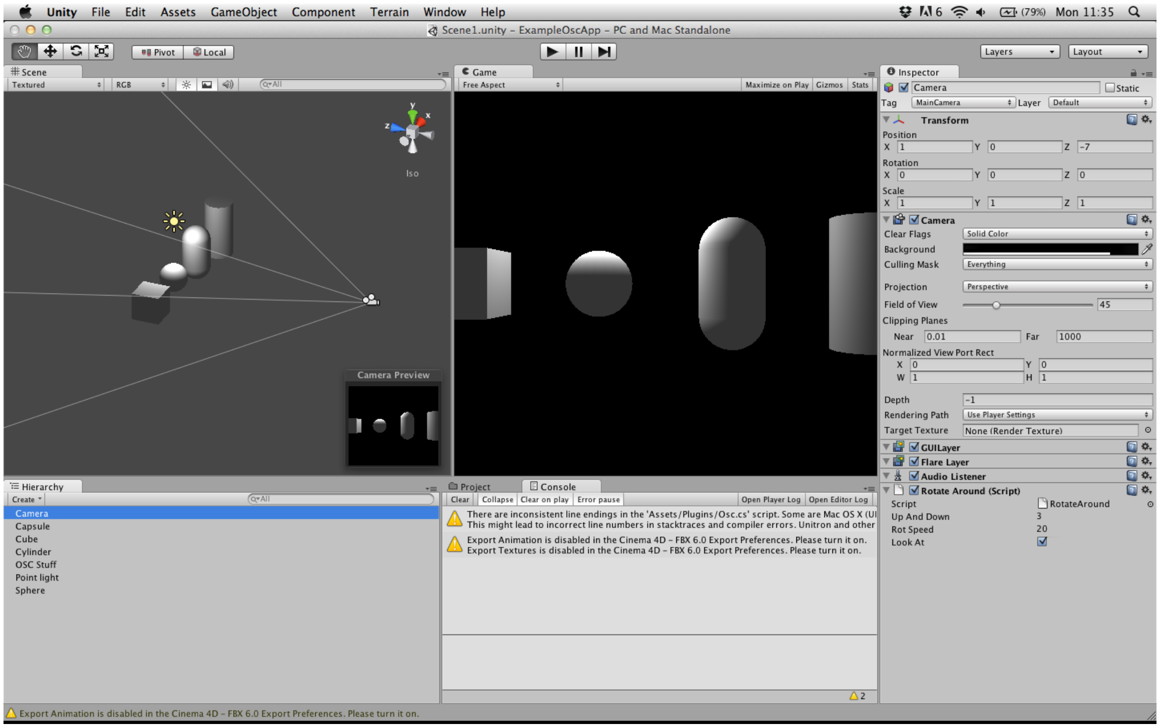
Application software designed and developed for Android-based mobile devices.

E.1 Android

Android is a popular Linux-based operating system (OS) owned by Google. Designed primarily for mobile devices, the Android OS has been deployed in everyday devices such as smartphones, watches, and tablets, but it has also been used in home appliances like refrigerators and televisions [45].

F. Unity

Unity is one of the leading game development engines in the world, with the latest version being Unity 2017. It features an all-in-one-editor for both the Windows and Mac OS that can support 2D as well as 3D game development across many different platforms including mobile, desktop, VR, console, and web [46].



Camera ☑ □ Static
Tag MainCamera Layer Default

Transform
Position X 1 Y 0 Z -7
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

☑ Camera
Clear Flags: Solid Color
Culling Mask: Everything
Projection: Perspective
Field of View: 45
Clipping Planes Near 0.01 Far 1000
Normalized View Port Rect X 0 Y 0 W 1 H 1
Depth: -1
Rendering Path: Use Player Settings
Target Texture: None (Render Texture)

☑ GUILayer
☑ Flare Layer
☑ Audio Listener
☑ Rotate Around (Script)
Script: RotateAround
Up And Down: 3
Rot Speed: 20
Look At: ☑

```
There are inconsistent line endings in the 'Assets/Plugins/Osc.cs' script. Some are Mac OS X (UI
This might lead to incorrect line numbers in stacktraces and compiler errors. Unitron and other
Export Animation is disabled in the Cinema 4D – FBX 6.0 Export Preferences. Please turn it on.
Export Textures is disabled in the Cinema 4D – FBX 6.0 Export Preferences. Please turn it on.
Export Animation is disabled in the Cinema 4D – FBX 6.0 Export Preferences. Please turn it on.
```

Figure 5: The Unity workspace.

G. Blender

Blender is a free and open source software suite used for creating 3-dimensional assets for both movie and video game use. The entire 3D pipeline from modeling, rigging, animation, simulation, rendering, and all the other steps of the and rigging to compositing and motion tracking is supported in Blender [47].

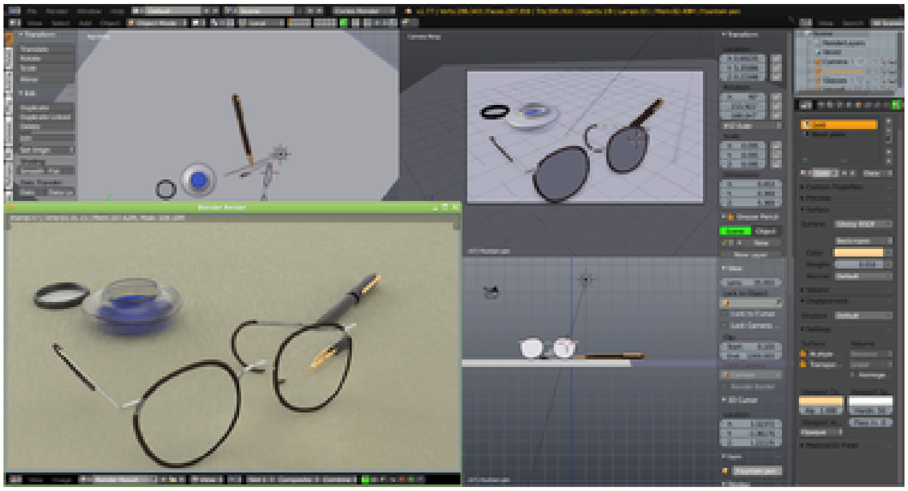


Figure 6: A Blender project.

IV. Design and Implementation

A. Use Case Diagram

The application's main user is the patient.

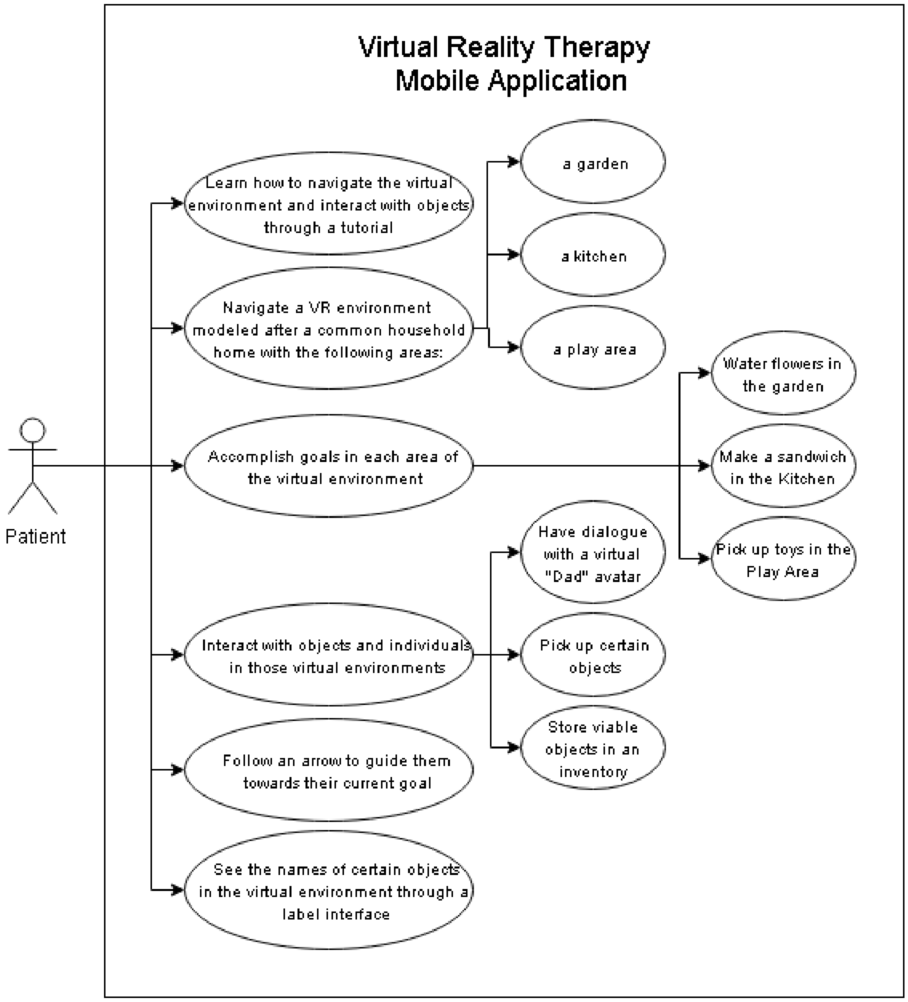

Virtual Reality Therapy
Mobile Application
Patient
Learn how to navigate the virtual environment and interact with objects through a tutorial
Navigate a VR environment modeled after a common household home with the following areas:
a garden
a kitchen
a play area
Accomplish goals in each area of the virtual environment
Water flowers in the garden
Make a sandwich in the Kitchen
Pick up toys in the Play Area
Interact with objects and individuals in those virtual environments
Have dialogue with a virtual "Dad" avatar
Pick up certain objects
Store viable objects in an inventory
Follow an arrow to guide them towards their current goal
See the names of certain objects in the virtual environment through a label interface

Figure 7: Use Case Diagram - Patient

After being taught the controls with help from the therapist, the patient can explore the environment, interact with objects found therein, and accomplish goals. They can also engage in simple dialogue with the ”Dad” avatar and ask for a tip to help them accomplish their current objective. They can also shut down the application if it causes them distress at any time during the session.

B. Activity Diagram

Patient

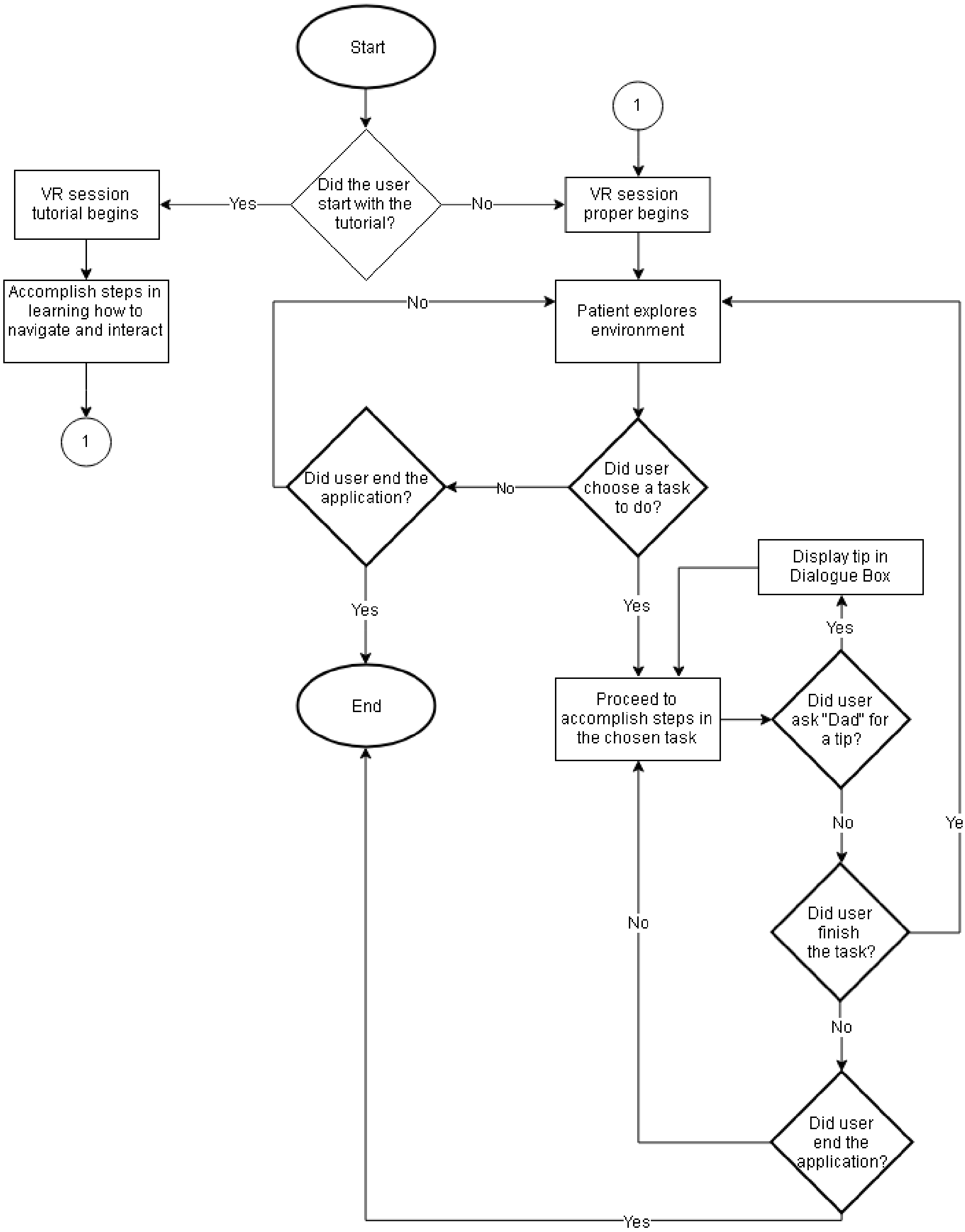

Start
Did the user start with the tutorial?
Yes
No
VR session tutorial begins
VR session proper begins
1
Accomplish steps in learning how to navigate and interact
1
No
Patient explores environment
Did user end the application?
No
Did user choose a task to do?
Yes
Yes
End
Display tip in Dialogue Box
Yes
Proceed to accomplish steps in the chosen task
Did user ask "Dad" for a tip?
No
Yes
No
Did user finish the task?
No
Did user end the application?
Yes

Figure 8: Activity Diagram of Patient

The user can choose to begin with the tutorial or with the session proper. If they choose the tutorial first, a quick series of tasks introducing and explaining of the controls will begin. Afterwards, the session proper will start. The patient will be able to navigate the VR environment and accomplish the tasks in each of the areas in the home. They can repeat all of these tasks as many times as they want. In the event that the patient does not wish to continue, they can simply turn their machine off.

C. Technical Architecture

The following smartphone specifications are required to run the mobile application:

1. Android OS Version 4.4 ‘Kitkat’ or higher

2. 2 GB of RAM or higher

3. Accelerometer

4. Gyroscope

5. Magnetometer

V. Results

acquilaVR is a mobile VR application that simulates a household setting to help teach ADLs to children with autism.

A. Main Menu

The application begins with the main menu which has a Start button and a Quit button. On pressing the Start button, the user is asked whether the session will begin with the tutorial or not. After selecting an option, an animation will play while the virtual environment is loading. A message will also appear user prompting the user to put on their VR Headset.

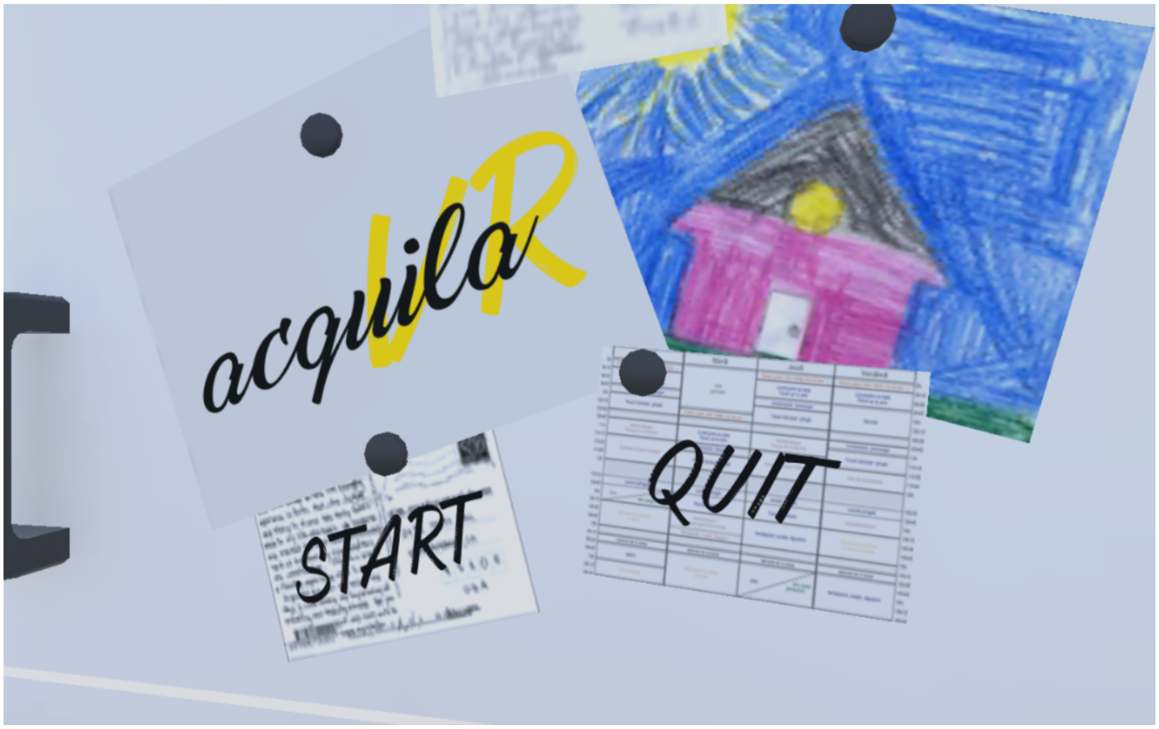


Figure 9: Acquila Main Menu

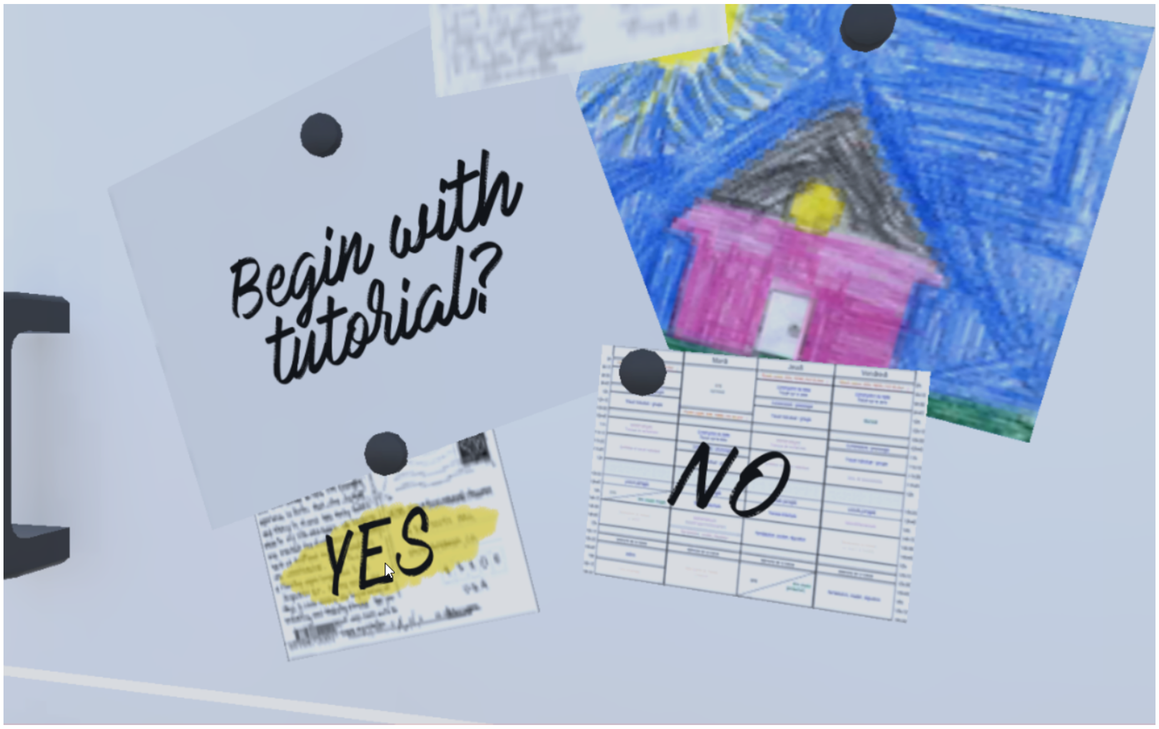
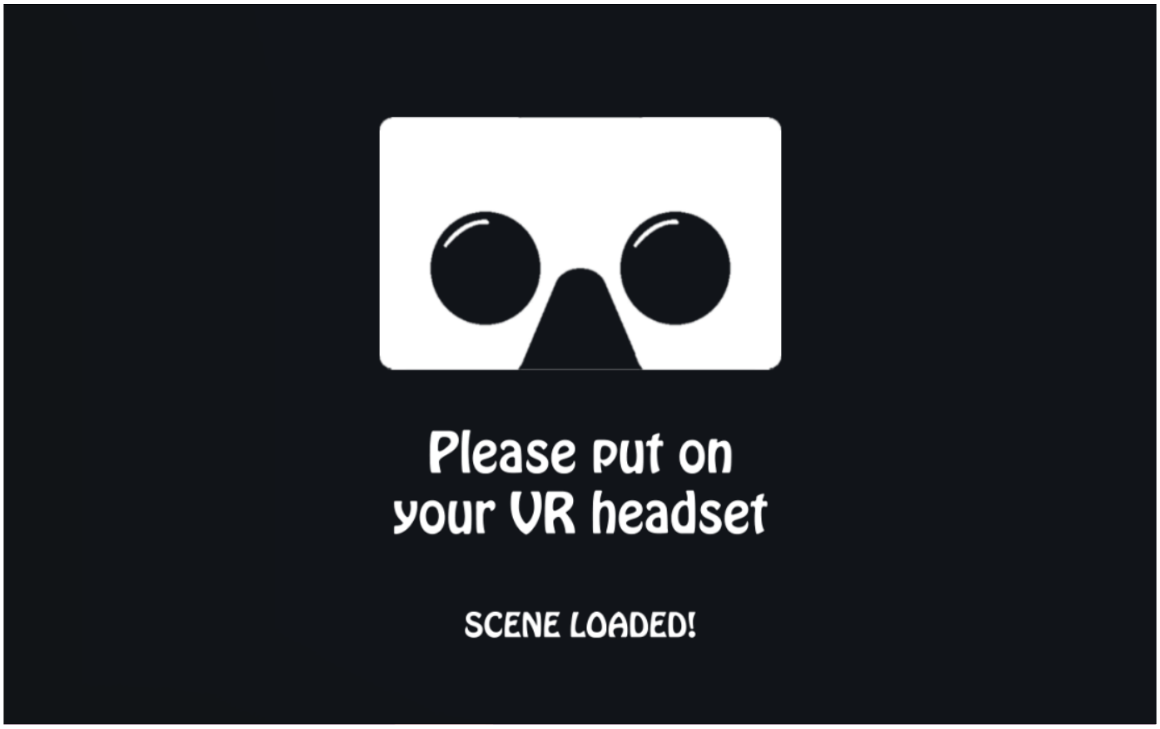


Figure 10: Tutorial Option Menu


Please put on your VR headset

SCENE LOADED!

Figure 11: Acquila VR Headset Prompt

B. Virtual Environment

The virtual environment is modeled after a generic home setting with 3 areas: a garden, a play area, and a kitchen.

The patient's's starting position in the virtual environment.

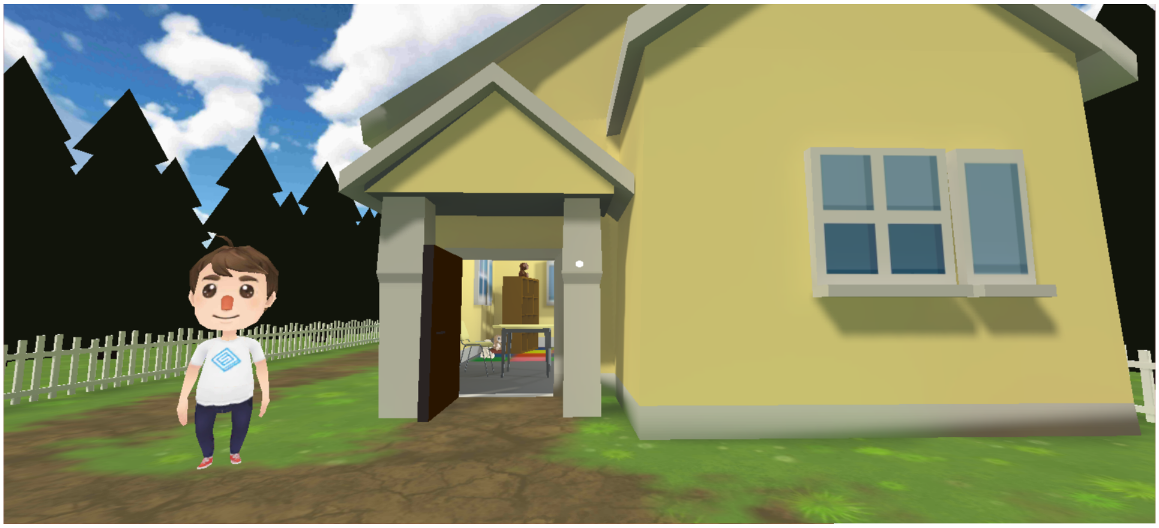


Figure 12: Front of house

B..1 Garden

The garden area of the virtual environment. This area has a smattering of flower patches to water, and a large tree in the back.

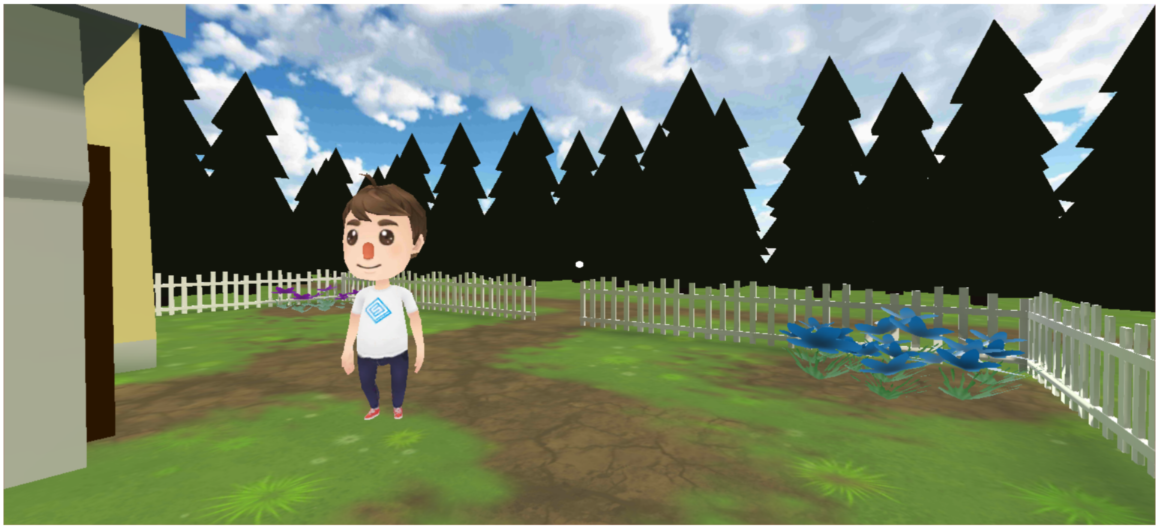


Figure 13: Garden Front

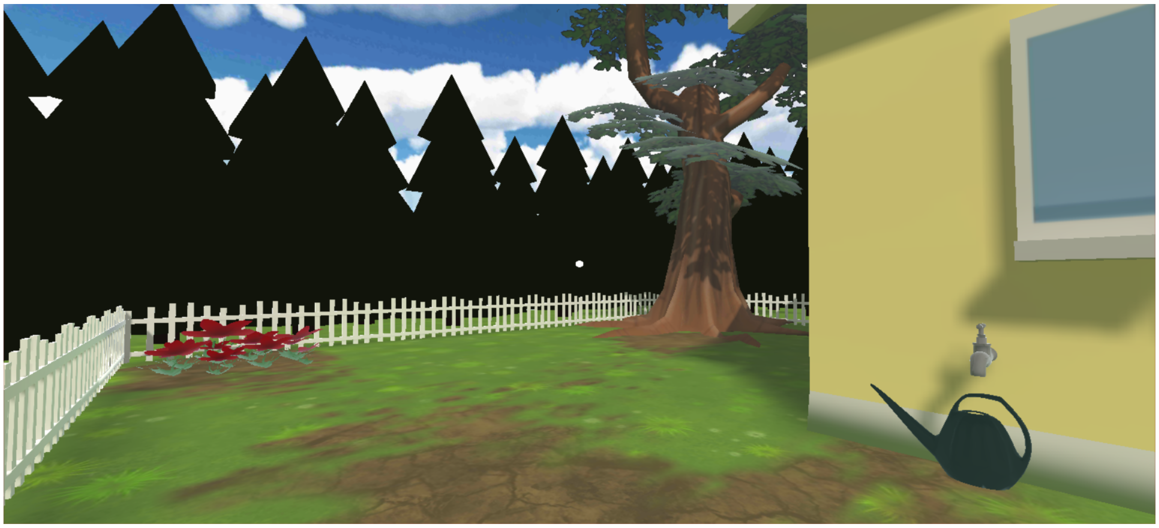


Figure 14: Garden Back

B..2 Play Area

The playroom area in the virtual environment. Plush toys are littered around the room for picking up and returning to the toy chest in the corner.

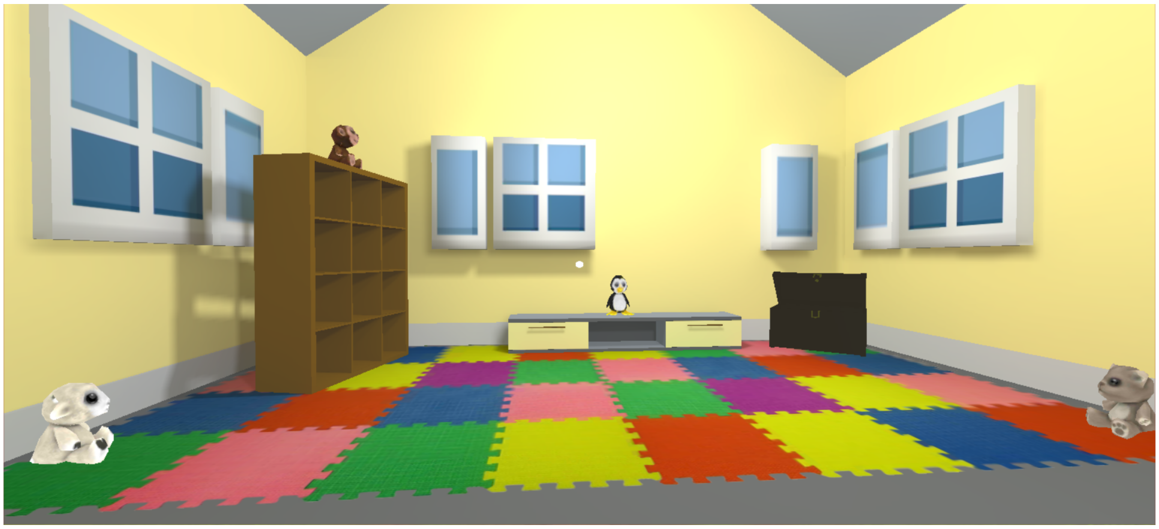


Figure 15: Play Area

B..3 Kitchen

The kitchen in the virtual environment. A few clean plates are next to the sink in this area, and the sandwich ingredients are prepared on the kitchen table.

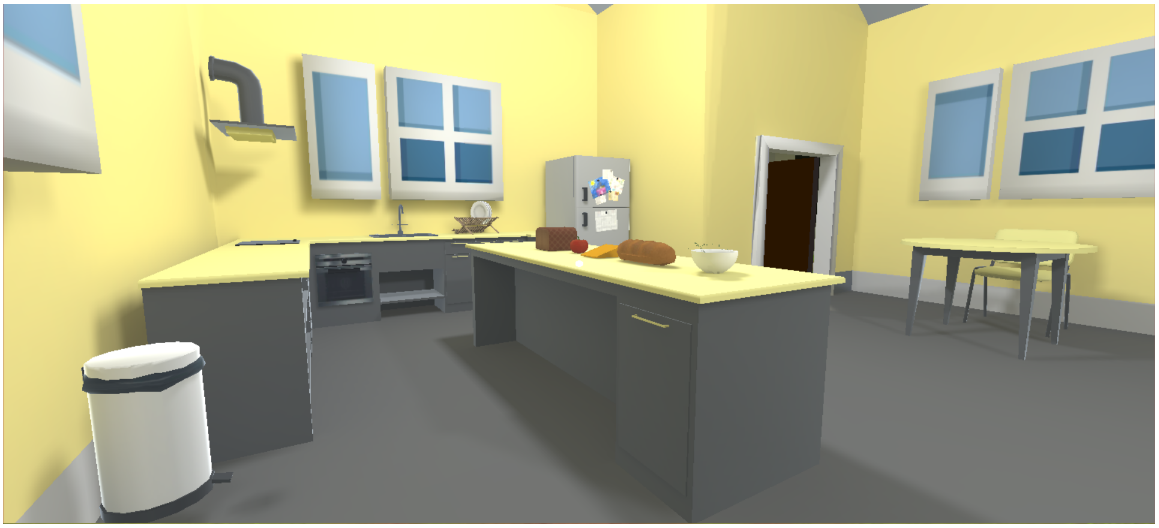


Figure 16: Kitchen

VI. Discussions

acquilaVR is an application designed as a supplement in helping teach and generalize daily living skills to children with autism. The program simulates a household setting with three different areas, each with a different goal. Each goal can be accomplished by following a series of steps, where the next step will only be displayed after the player accomplishes the previous step.

Three applications were used in the development of this project: Blender, Adobe Photoshop and Unity. Blender is a free and open source software suite used for creating 3-dimensional assets. It was used to create new models, edit existing ones, and make animations for use in the application. The house for example used to be taller and empty inside, but using Blender it was shortened in height and populated with walls and windows. The eating animation was also made using Blender by editing a Drinking animation found in the online animation library, Mixamo. Photoshop is a raster graphics editor developed and published by Adobe Systems. It was used to create the logos of the applications and clean or edit existing object textures. Unity is a game development engine that can support 2D as well as 3D game development across many different platforms. All models and textures were then imported into Unity to be placed in the virtual environment. Scripting was done in Unity's supplied MonoDevelop integrated development environment (IDE). MonoDevelop is an open source IDE for Linux, macOS, and Windows.

A prominent feature of this application is the QuestManager system. Designed after traditional linear role-playing game (RPG) quest systems, this script contains a list of Quest objects. These Quest objects contain an integer quest ID, the ID of the next quest in the quest chain, a reference to a source object, a reference to the target object, and a list of strings that describe events such as which objects to activate, what dialogue to trigger, where the arrow should point to, and what animations to play. After a user completes the current quest by activating the target object, the next quest in the chain is called using the next quest ID and the events described in the new Quest object are read by the QuestManager script. The strings are parsed and a corresponding function for each string is called, activating all the events described in the Quest object.

A few things changed from the initial proposal of this SP. Firstly, the original proposal described 2 virtual environments, a mall and a neighborhood. After consulting with a domain expert, we decided that the activities associated with these virtual environments are not as important as those that can be found at home. Thus, we changed our virtual environment from a neighborhood and a mall to a household where more pivotal daily living skills can be developed. Secondly, the video recording objective has been removed. This is due to Unity closing down the Everyplay Service, a free mobile game replay platform. Currently, their SDK no longer works for new Unity projects. However, the developers of the service plan to release a final SDK by June 2018 where it can be used to store replay files locally. This objective has been added to the Recommendations section in the hopes that it can be implemented when the final Everyplay SDK is released in the future.

VII. Conclusions

acquilaVR is a mobile VR application that simulates a household setting to help teach ADLs to children with autism. It offers a realistic home environment with three areas: a garden, a kitchen, and a play area. In each area, an objective based on an ADL can be accomplished. The application also has an tutorial to teach the user how to navigate in the environment and interact with objects.

The application was developed using three programs: Blender, Adobe Photoshop, and Unity. Blender and Photoshop were used to create assets to be used in Unity, though a number of free assets were provided by the Unity asset store. The environment was designed in Unity, and scripting was done using Unity's supplied IDE, MonoDevelop.

acquilaVR was developed to be easy to use and deploy, at the office or at the comfort of the patient's home. It is not meant to replace ABA, but rather to act as a supplement in the teaching/generalizing of skills in ABA.

VIII. Recommendations

The application can be improved upon by iterating upon the virtual environment and adding other typical household rooms, such as a bathroom or a bedroom. In this way, they can learn a wider variety of activities of daily living. It can also be improved by increasing the number of objectives available in a single area of the environment instead of having one objective per area, as only having one objective per area could wrongly indicate to the patient that no other activities can be done in this area.

One other possible improvement to be made is a change to the linearity of the QuestManager system. Instead of quests following a single chain, some quests could give the user a choice at a certain point in the chain so they can have more freedom in how to go about accomplishing a certain objective. In the sandwich making quest chain for example, after getting the initial slice of bread the patient could be given the option to choose other ingredients to go on the sandwich first rather than following the order described by the quest chain. Or later on in that same quest they may also opt to eat the sandwich they made themselves rather than give it to the Dad Avatar. Another possibility that improves upon the linearity of the quests is to have some of the quests in a chain randomize which order they should be undergone, as long as the quest order still makes sense logically. In the pickup toys quest chain for example, the quest doesn't necessarily have to begin with the Bear plushtoy first, the Penguin plushtoy second, and etc. If the user repeats this task enough times, they might form the incorrect generalization in the skill that they always need to start with the Bear plushtoy first even in real life.

Customization of the Parent avatar could also be a possible improvement for the system. The patient might be more comfortable in the environment if the avatar resembled their actual parent more closely. Another improvement in customization would be an option to change the avatar's gender for children who are more comfortable with a mother figure as their guardian.

IX. Bibliography

[1] "Autism spectrum disorder." https://www.nimh.nih.gov/health/topics/autism-spectrum-disorders-asd/index.shtml. Accessed: 13- Dec- 2017.

[2] "Autism spectrum disorders - symptoms." https://www.autismspeaks.org/what-autism/symptoms. Accessed: 13- Dec- 2017.

[3] "Social interaction for children - nas." http://www.autism.org.uk/about/communication/social-skills/young-children.aspx, 2017. Accessed: 01- Nov- 2017.

[4] "Cdc — data and statistics — autism spectrum disorder (asd) — ncbddd." https://www.cdc.gov/ncbddd/autism/data.html, 2015. Accessed: 01- Nov- 2017.

[5] E. Jaymalin, "Number of people with autism increasing." http://www.philstar.com/headlines/2014/04/10/1310840/number-people-autism-increasing, 2017. Accessed: 01- Nov- 2017.

[6] "Cdc — screening and diagnosis — autism spectrum disorder (asd) — ncbddd." https://www.cdc.gov/ncbddd/autism/screening.html, 2015. Accessed: 01- Nov- 2017.

[7] "How is autism diagnosed?." https://www.autismspeaks.org/what-autism/diagnosis, 2017. Accessed: 01- Nov- 2017.

[8] "Dsm-5." https://www.psychiatry.org/psychiatrists/practice/dsm, 2017. Accessed: 03- Nov- 2017.

[9] "Is there really such a thing as high and low functioning autism?." https://www.verywell.com/high-and-low-functioning-autism-260599, 2017. Accessed: 04- Nov- 2017.

[10] “Introducing the vineland3: The new revision of the vineland adaptive behavior scales.” https://www.mspaonline.net/wp-content/uploads/2017/01/Kimbell-Vineland-3-MSPA-handout.pdf, 2017. Accessed: 04- Nov- 2017.

[11] C. Lord and J. McGee, *Educating Children with Autism*. Washington, D.C.: National academy Press, 2001.

[12] “Cdc — treatment — autism spectrum disorder (asd) — ncbddd.” https://www.cdc.gov/ncbddd/autism/treatment.html, 2017. Accessed: 06- Nov- 2017.

[13] E. Huffman, “Management of symptoms in children with autism spectrum disorders: a comprehensive review of pharmacologic and complementary-alternative medicine t... - pubmed - ncbi.” https://www.ncbi.nlm.nih.gov/pubmed/21160435, 2017. Accessed: 06- Nov- 2017.

[14] “Recovery and treatment.” http://autismpartnershipph.com/en/about-autism/recovery-and-treatment/, 2017. Accessed: 07- Nov- 2017.

[15] “Virtual reality treatment for autism - virtual reality society.” https://www.vrs.org.uk/virtual-reality-healthcare/autism-treatment.html. Accessed: 09- Nov- 2017.

[16] M. House, “What is virtual reality? - definition from whatis.com.” http://whatis.techtarget.com/definition/virtual-reality, 2015. Accessed: 10- Nov- 2017.

[17] “What is virtual reality? - virtual reality society.” https://www.vrs.org.uk/virtual-reality/what-is-virtual-reality.html. Accessed: 11- Nov- 2017.

[18] “Virtual reality treatment for autism - virtual reality society.” https://www.vrs.org.uk/virtual-reality-healthcare/autism-treatment.html. Accessed: 12- Nov- 2017.

[19] "Virtual reality helps autistic kids cross the road." https://www.israel21c.org/virtual-reality-helps-autistic-kids-cross-the-road/, 2008. Accessed: 14- Nov- 2017.

[20] N. Josman, H. Ben-Chaim, S. Friedrich, and P. Weiss, "Effectiveness of virtual reality for teaching street-crossing skills to children and adolescents with autism," *International Journal on Disability and Human Development*, vol. 7, 2008.

[21] "Floreo — social and communication therapy that is immersive and realistic." http://floreotech.com/#product, 2017. Accessed: 15- Nov- 2017.

[22] M. Shoaib, I. Hussain, H. Mirza, and M. Tayyab, "The role of information and innovative technology for rehabilitation of children with autism: A systematic literature review," *2017 17th International Conference on Computational Science and Its Applications (ICCSA)*, 2017.

[23] N. Didehbani, T. Allen, M. K. D. Krawczyk, and S. Chapman, "Virtual reality social cognition training for children with high functioning autism," *Computers in Human Behavior*, vol. 62, pp. 703–711, 2016.

[24] R. Mourning and Y. Tang, "Virtual reality social training for adolescents with high-functioning autism," *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2016.

[25] H. Ip, S. Wong, D. Chan, J. Byrne, C. Li, V. Yuan, K. Lau, and J. Wong, "Virtual reality enabled training for social adaptation in inclusive education settings for school-aged children with autism spectrum disorder (asd)," *Lecture Notes in Computer Science*, pp. 94–102, 2016.

[26] C. Volioti, T. Tsiatsos, S. Mavropoulou, and C. Karagiannidis, "Vlss – virtual learning and social stories for children with autism," *2014 IEEE 14th International Conference on Advanced Learning Technologies*, 2014.

[27] E. Bekele, Z. Zheng, A. Swanson, J. Crittendon, Z. Warren, and N. Sarkar, "Understanding how adolescents with autism respond to facial expressions in virtual reality environments," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, pp. 711–720, 2013.

[28] E. Bekele, J. Wade, D. Bian, J. Fan, A. Swanson, Z. Warren, and N. Sarkar, "Multimodal adaptive social interaction in virtual environment (masi-vr) for children with autism spectrum disorders (asd)," *2016 IEEE Virtual Reality (VR)*, 2016.

[29] D. Sturm, E. Peppe, and B. Ploog, "emot-ican: Design of an assessment game for emotion recognition in players with autism," *2016 IEEE International Conference on Serious Games and Applications for Health (SeGAH)*, 2016.

[30] M. Bernardes, F. Barros, M. Simoes, and M. Castelo-Branco, "A serious game with virtual reality for travel training with autism spectrum disorder," *2015 International Conference on Virtual Rehabilitation (ICVR)*, 2015.

[31] L. Lamash, E. Klinger, and N. Josman, "Using a virtual supermarket to promote independent functioning among adolescents with autism spectrum disorder," *2017 International Conference on Virtual Rehabilitation (ICVR)*, 2017.

[32] "What is autism? - nas." http://www.autism.org.uk/about/what-is.aspx, 2017. Accessed: 21- Nov- 2017.

[33] "Autism - nas." http://www.autism.org.uk/about/what-is/asd.aspx, 2016. Accessed: 20- Nov- 2017.

[34] "Applied behavior analysis (aba)." https://www.autismspeaks.org/what-autism/treatment/applied-behavior-analysis-aba, 2017. Accessed: 3- May- 2018.

[35] “Chaining and task analysis.” https://www.relias.com/resource/chaining-and-task-analysis, 2018. Accessed: 4- May- 2018.

[36] “Teaching children with autism daily living skills.” https://www.tbh.com/autism-parenting/teaching-children-with-autism-daily-living-skills-hand-washing/, 2017. Accessed: 3- May- 2018.

[37] “Systematic desensitization.” https://www.simplypsychology.org/Systematic-Desensitisation.html, 2015. Accessed: 4- May- 2018.

[38] “What is the picture exchange communication system or pecs?.” https://www.iidc.indiana.edu/pages/What-is-the-Picture-Exchange-Communication-System-or-PECS, 2002. Accessed: 5- May- 2018.

[39] “The picture exchange communication system (pecs).” https://www.nationalautismresources.com/the-picture-exchange-communication-system-pecs/, 2018. Accessed: 5- May- 2018.

[40] G. Blatt, “virtual reality (vr) — computer science.” https://www.britannica.com/technology/virtual-reality, 2017. Accessed: 21- Nov- 2017.

[41] “What is a head-mounted display (hmd)? - definition from techopedia.” https://www.techopedia.com/definition/2342/head-mounted-display-hmd. Accessed: 21- Nov- 2017.

[42] “Head-mounted displays (hmds) - virtual reality society.” https://www.vrs.org.uk/virtual-reality-gear/head-mounted-displays/. Accessed: 21- Nov- 2017.

[43] J. Hildenbrand, “What is google cardboard?.” https://www.androidcentral.com/what-google-cardboard, 2015. Accessed: 21- Nov- 2017.

[44] “Developer overview — google vr — google developers.” https://developers.google.com/vr/cardboard/overview#sdks, 2017. Accessed: 21- Nov- 2017.

[45] M. Karch, “Everything you need to know about the android os.” https://www.lifewire.com/what-is-google-android-1616887, 2015. Accessed: 21- Nov- 2017.

[46] “Unity - products.” https://www.lifewire.com/what-is-google-android-1616887, 2017. Accessed: 22- Nov- 2017.

[47] “blender.org - home of the blender project - free and open 3d creation software.” https://www.blender.org/, 2017. Accessed: 22- Nov- 2017.

X. Appendix

ArrowController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ArrowController : MonoBehaviour {

        //public bool isActive;
        private Transform currentTarget;

        [Range(0, 180)]
        public float Offset;
        public float TargetAdjust = 90;
        // Update is called once per frame

        void Update () {
                if (gameObject.activeSelf) {
                        gameObject.transform.LookAt (currentTarget);
                        gameObject.transform.Rotate (new Vector3 (TargetAdjust, 0, 0));
                        if (Mathf.Abs (gameObject.transform.rotation.eulerAngles.x) >= 45) {
                                Vector3 newRot = (gameObject.transform.rotation.eulerAngles.x > 0)
                                      ?
                                        new Vector3 (-Offset, gameObject.transform.rotation.
                                             eulerAngles.y, gameObject.transform.rotation.
                                             eulerAngles.z) :
                                        new Vector3 (Offset, gameObject.transform.rotation.
                                             eulerAngles.y, gameObject.transform.rotation.
                                             eulerAngles.z);
                                gameObject.transform.rotation = Quaternion.Euler (newRot);
                        }
                }
        }

        public void TargetArrow(Transform target) {
                currentTarget = target;
                //Debug.Log ("Arrow targeting " + target.gameObject.name);
        }

        public void ShowArrow() {
                gameObject.SetActive (true);
        }

        public void HideArrow() {
                gameObject.SetActive (false);
        }
}
```

DialogueController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;

public class DialogueController : MonoBehaviour {

        public TextMeshPro dialogueText;
        //public TextMeshPro[] choicesList;
        public GameObject[] pecsPanels;
        public PlayerController playerReference;
        public Vector3 panelPos1;
        public Vector3 panelPos2;
        public Vector3 panelPos3;
        [Range(1,5)]
        public float moveDistance;
        [Range(1,30)]
        public float maxDistance;
        public float yPos;

        // Use this for initialization
        void Start () {
                yPos = gameObject.transform.position.y;
        }

        void Update () {
                Vector3 target = playerReference.transform.position - transform.position;
                //Vector3 target = playerReference.transform.position - transform.parent.transform
                     .position;
                float dist = target.magnitude;
                if (dist > maxDistance) {
                        HideDialogue ();
                }
        }

        /* Update Dialogs:
         * 1. Text Only
         * 2. Text with PECS
         * 3. PECS Only
         */
        public void UpdateDialogue (string dialog) { // Update Dialogue #1
                dialogueText.SetText (dialog);

                foreach (GameObject panel in pecsPanels) {
                        panel.gameObject.SetActive (false);
                }
                /*
                foreach (TextMeshPro text in choicesList) {
                        text.gameObject.SetActive (false);
                }
                */
        }

        public void UpdateDialogue (string dialog, PECS[] pecsList) { // Update Dialogue #2
```

```
            ShowDialogue();
            dialogueText.SetText (dialog);
            UpdateDialogue (pecsList);
    }

    public void UpdateDialogue (PECS[] pecsList) { // Update Dialogue #3
            ShowDialogue();
            for (int i = 0; i < pecsList.Length; i++) {
                    pecsPanels [i].SetActive (true);
                    pecsPanels [i].transform.GetChild (0).gameObject.GetComponent<Renderer> ()
                        .material.mainTexture = pecsList[i].PECSImg;
                    pecsPanels [i].transform.GetChild (1).gameObject.GetComponent<TextMeshPro>
                        ().SetText (pecsList [i].PECSText);
            }
            for (int i = pecsList.Length; i < pecsPanels.Length; i++) {
                    pecsPanels [i].SetActive (false);
            }

            AdjustPECSPositions (pecsList);
    }

    // Adjust where PECs are depending on how many are loaded
    private void AdjustPECSPositions (PECS[] pecsList) {
            int pecsCount = pecsList.Length;
            if (pecsCount == 1) {
                    pecsPanels [0].transform.localPosition = panelPos1;
            } else if (pecsCount == 2) {
                    pecsPanels [0].transform.localPosition = panelPos2;
                    pecsPanels [1].transform.localPosition = new Vector3 (-panelPos2.x,
                        panelPos2.y, panelPos2.z);
                    //pecsPanels [1].transform.position = ;
            } else {
                    pecsPanels [0].transform.localPosition = panelPos3;
                    pecsPanels [1].transform.localPosition = panelPos1;
                    pecsPanels [2].transform.localPosition = new Vector3 (-panelPos3.x,
                        panelPos3.y, panelPos3.z);
            }
    }

    public void MoveDialogue (GameObject target) {
            Vector3 startPos = new Vector3 ( target.transform.position.x, yPos,  target.
                transform.position.z );
            Vector3 targetPos = new Vector3 ( playerReference.transform.position.x, yPos,
                playerReference.transform.position.z );
            gameObject.transform.position = Vector3.MoveTowards (startPos, targetPos,
                moveDistance);
            gameObject.transform.LookAt (playerReference.transform.position);
            ShowDialogue();
    }

    public void HideDialogue () {
            //Debug.Log (”yello not show”);
            gameObject.SetActive (false);
    }

    public void ShowDialogue () {
        //Debug.Log ("yello show");
        gameObject.SetActive (true);
    }

    public bool IsActive () {
        return gameObject.activeSelf;
    }

    public void ActivateChoices () {
        // Activate Box Colliders on Panels
        //Debug.Log("Activate Choices");
        foreach (GameObject panel in pecsPanels) {
            if (panel.activeSelf)
                panel.GetComponent<BoxCollider> ().enabled = true;
        }
    }

    public void DeactivateChoices () {
        //Debug.Log("Deactivate Choices");
        // Activate Box Colliders on Panels
        foreach (GameObject panel in pecsPanels) {
            panel.GetComponent<BoxCollider> ().enabled = false;
        }
    }
}
```

ParticleController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ParticleController : MonoBehaviour {

    public List<ParticleSystem> particleList;

    public void PlayParticle (string particleName) {
        if (particleList != null) {
            foreach (ParticleSystem p in particleList) {
                if (p.name == particleName) {
                    p.Clear ();
                    p.Play ();
                    return;
                }
            }
            Debug.Log ("Particle␣System:␣" + particleName + "not␣found!");
        }
    }
}
```

PECS.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;

[System.Serializable]
public class PECS {

        // What image will be shown in the PECS
        public Texture PECSImg;

        // What text is connected to this image
        [TextArea(1, 1)]
        public string PECSText;

}
```

LabelController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;

public class LabelController : MonoBehaviour {

        private TextMeshPro labelText;
        public GameObject playerReference;
        // Use this for initialization
        void Awake () {
                labelText = transform.GetChild (1).GetComponent<TextMeshPro> ();
        }

        public void UpdateLabel (string text) {
                ShowLabel ();
                labelText.SetText (text);
        }

        public void ShowLabel () {
                gameObject.SetActive (true);
        }


        public void HideLabel () {
                gameObject.SetActive (false);
        }
}
```

Notification.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;

public class Notification : MonoBehaviour {

        private TextMeshPro notifText;
        public float displayTime;

        // Use this for initialization
        void Awake () {
                notifText = gameObject.GetComponent<TextMeshPro> ();
                notifText.faceColor = new Color (1, 1, 1, 0);
                displayTime = 2F;
                //notifText.Set
        }

        public void UpdateNotification(string notif) {
                //notifText.text = notif;
                //Debug.Log(”New notif:” + notif);
                notifText.SetText (notif);
                StartCoroutine (FadeInAndOutText());
```

```
    }

    IEnumerator FadeInAndOutText() {
        //Fade In Text
        for (float i = 0; i <= 1; i += Time.deltaTime) {
            notifText.faceColor = new Color (1, 1, 1, i);
            yield return null;
        }
        yield return new WaitForSeconds (displayTime);
        yield return StartCoroutine (FadeOutText());
    }

    public void UpdateNotification(string notif, float newDisplayTime) {
        //notifText.text = notif;
        //Debug.Log(”New notif:” + notif);
        notifText.SetText (notif);
        StartCoroutine (FadeInAndOutText(newDisplayTime));
    }

    IEnumerator FadeInAndOutText(float newDisplayTime) {
        //Fade In Text
        for (float i = 0; i <= 1; i += Time.deltaTime) {
            notifText.faceColor = new Color (1, 1, 1, i);
            yield return null;
        }
        yield return new WaitForSeconds (newDisplayTime);
        yield return StartCoroutine (FadeOutText());
    }

    IEnumerator FadeOutText() {
        for (float i = 1; i >= 0; i -= Time.deltaTime) {
            //Debug.Log (i);
            notifText.faceColor = new Color (1, 1, 1, i);
            yield return null;
        }
        notifText.faceColor = new Color (1, 1, 1, 0);
    }
}
```

InventoryController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class InventoryController : MonoBehaviour {

    private List<string> inventory = new List<string>();
    public Notification notifReference;

    // Use this for initialization
    void Start () {
         //AddItem (”Letter”);
    }
```

```
        public bool IsInInventory (string toFind) {
                if ( inventory.Contains (toFind) ) {
                        return true;
                }
                return false;
        }

        public void RemoveItem (string toRemove) {
                inventory.Remove (toRemove);
                notifReference.UpdateNotification  (”You␣removed␣” + toRemove + ”!”);
        }

        public void AddItem (string toAdd) {
                inventory.Add (toAdd);
                notifReference.UpdateNotification  (”You␣got␣” + toAdd + ”!”);
        }
}
```

PlayerController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System;

public class PlayerController : MonoBehaviour
{
        public Camera cam;
        public float movementSpeed = 3.0f;
        public Vector3 heldPos = new Vector3 (0, 0, 0);
        public string barrierText, restText1, restText2;
        public Notification notifReference;
        public GameObject heldObj;
        [Range(1,20)] public int restInterval;
        private Vector3 prevHeldPos;
        private Quaternion prevHeldRot;
        private Rigidbody playerBody;
        private Transform heldParent;
        private DateTime oldDate, currentDate;
        private float timePassed;
        private int intervalCounter;

        void Start() {
                playerBody = transform.GetComponent<Rigidbody> ();
                oldDate = System.DateTime.Now;
                timePassed = 0;
                intervalCounter = 1;
        }

        void Update()
        {
                timePassed += Time.deltaTime;

                Debug.Log (timePassed);
                if (timePassed >= ( float ) restInterval * 60.0f * intervalCounter ) { //
                     restInterval * 60.0f to translate minutes to seconds
                notifReference.UpdateNotification (restText1 + (restInterval *
                    intervalCounter).ToString() + restText2, 5.0f);
                intervalCounter++;
            }

            if ( Input.GetButton (”Fire1”) ) {
                Vector3 moveForward = cam.transform.forward;
                moveForward.y = 0;
                //playerBody.AddForce (moveForward * Time.deltaTime * movementSpeed);
                playerBody.MovePosition(gameObject.transform.position + (moveForward *
                    Time.deltaTime * movementSpeed));
                //transform.position += moveForward * Time.deltaTime * movementSpeed;
            } else {
                var x = Input.GetAxis (”Horizontal”) * Time.deltaTime * 150.0f;
                var z = Input.GetAxis (”Vertical”) * Time.deltaTime * 3.0f;

                transform.Rotate (0, x, 0);
                transform.Translate (0, 0, z);
            }
        }

        void OnCollisionEnter(Collision colInfo) {
            if (colInfo.gameObject.name.Equals(”Barrier”)) {
                notifReference.UpdateNotification (barrierText);
            }
        }

        void OnCollisionExit(Collision colInfo) {
            playerBody.velocity = Vector3.zero;
            playerBody.angularVelocity = Vector3.zero;
        }

        public void HoldObject (GameObject obj) {
            prevHeldPos = obj.transform.localPosition;
            prevHeldRot = obj.transform.rotation;
            heldParent = obj.transform.parent;
            obj.transform.SetParent (cam.transform);
            obj.transform.localPosition = heldPos;
            if (obj.transform.GetComponent<Collider> () != null)
                obj.transform.GetComponent<Collider> ().enabled = false;
            heldObj = obj;
        }

        public void ReleaseObject () {
            if (heldObj != null) {
                heldObj.transform.SetParent (heldParent);
                heldObj.transform.localPosition = prevHeldPos;
                heldObj.transform.rotation = prevHeldRot;
                if (heldObj.transform.GetComponent<Collider> () != null)
                    heldObj.transform.GetComponent<Collider> ().enabled = true;
                heldObj = null;
                heldParent = null;
            }
        }
}
```

Quest.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

[System.Serializable]
public class Quest {

    // nextQuestID is -1 if this is the last quest in the chain
    public int ID, nextQuestID;

    // For guardian character
    [TextArea(3, 10)]
    public string tip;

    // Commands are played as soon as this quest is now the current quest
    /* COMMAND LIST
     * ----------------------
     * Inventory Commands:
     * > ADDOBJECT: source/target
     * > REMOVEOBJ: source/target
     *
     * Dialog Commands:
     * > MOVDIALOG: source/target
     * > PUTDIALOG: text
     *
     * Notification Commands:
     * > MAKENOTIF: text
     *
     * Animation/Timeline Commands:
     * > STARTANIM: animname
     *
     * Arrow Commands:
     * > SHOWARROW: source/target
     * > HIDEARROW
     * ---------------------
     */
    [TextArea(1, 1)]
    public string[] commands;

    // Limited to 3 images
    public PECS[] pecsList;

    // Connected to pecs, refers to next quest corresponding to pecs
    public int[] choices;

    // What object will activate the next quest
    public GameObject questSource, questTarget;
}
```

QuestManager.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine.SceneManagement;
using UnityEngine.VR;
using UnityEngine;

public class QuestManager : MonoBehaviour {

        public InventoryController invReference;
        public QuestGiver givReference;
        public PlayerController playerReference;
        public DialogueController dialogueReference;
        public ArrowController arrowReference;
        public ToyController toyReference;
        public TimelineController tlReference;
        public ParticleController particleReference;
        public bool getPecNames;

        [SerializeField]
        private Quest currentQuest;
        public Quest[] questList;

        private Queue<string> commands;

        void Awake() {
                UnityEngine.XR.XRSettings.enabled = true;
        }

        void Start() {
                System.Array.Sort (questList, QuestIDSort);
                commands = new Queue<string>();

                // 1 if play tutorial, 2 if dont play
                // SetCurrentQuest(0) if no tutorial, other number if tutorial
                if (PlayerPrefs.GetInt (”tutorial”) == 1) {
                        SetCurrentQuest (-10);
                } else {
                        SetCurrentQuest (0);
                }

                if (getPecNames)
                 GetPECNames();
        }

        public Quest GetCurrentQuest() {
                return currentQuest;
        }

        public void SetCurrentQuest(int questID) {
                //Debug.Log (”Next quest is of ID:” + questID);
                currentQuest = FindQuest (questID);
                LoadCommands ();
                ProcessCommands ();
                LoadOrHideChoices ();
                if (questID == 21)
                        toyReference.RandomizeToyPositions();
        }

        private Quest FindQuest(int questID) {
                foreach (Quest q in questList) {
                        if (q.ID == questID) {
                                //Debug.Log (”Found Quest #:” + q.ID);
                                return q;
                        }
                }
                return null;
        }

        private void LoadCommands() {
                commands.Clear ();
                foreach (string s in currentQuest.commands) {
                        //Debug.Log (s);
                        commands.Enqueue (s);
                }
        }

        private void LoadOrHideChoices() {
                if (currentQuest.choices.Length != 0 && currentQuest.choices != null) { // Choices
                      exist for this quest
                        dialogueReference.ActivateChoices ();
                } else if (dialogueReference.pecsPanels[0].activeSelf) { // If Choices don't exist
                      for this quest, but choices are visible in dialogue
                        dialogueReference.DeactivateChoices ();
                }
        }

        /* COMMAND LIST
         * ——————————————————
         * Quest Commands:
         * > NEXTQUEST
         *
         * Inventory Commands:
         * > ADDOBJECT:source/target/name
         * > REMOVEOBJ:name
         * > HIDEOBJCT:source/target/name
         * > SHOWOBJCT:source/target/name
         *
         * Dialog Commands:
         * > MOVDIALOG:source/target/name
         * > NEWDIALOG:text
         * > SHODIALOG
         *
         * Notification Commands:
         * > MAKENOTIF:text
         *
         * Animation/Timeline Commands:
         * > PLAYANIME:name
         *
         * Arrow Commands:
```

```
 * > SHOWARROW:source/target
 * > HIDEARROW
 *
 * Particle System Commands:
 * > PLYPARTCL:name
 * ------------------
 */
private void ProcessCommands() {
        while(commands.Count > 0) {
                string command = commands.Dequeue ();
                //Debug.Log (command);
                if (command.Length > 9) {
                        GameObject item;
                        string param = command.Substring (10, command.Length - 10);
                        command = command.Substring (0, 9);
                        switch (command)
                        {
                        case ”ADDOBJECT”:
                                //Debug.Log (currentQuest.questTarget.transform.name);
                                if (param == ”source”) {
                                        invReference.AddItem (currentQuest.questSource.
                                             transform.name);
                                        item = currentQuest.questSource;
                                        playerReference.HoldObject (item);
                                } else if (param == ”target”) {
                                        invReference.AddItem (currentQuest.questTarget.
                                             transform.name);
                                        item = currentQuest.questTarget;
                                        playerReference.HoldObject (item);
                                } else {
                                        GameObject obj = FindInScene (param);
                                        if (obj != null) {
                                                invReference.AddItem (param);
                                                item = obj;
                                                playerReference.HoldObject (item);
                                        }
                                }
                                break;
                        case ”REMOVEOBJ”:
                                if (invReference.IsInInventory (param)) {
                                        invReference.RemoveItem (param);
                                        playerReference.ReleaseObject ();
                                }
                                break;
                        case ”HIDEOBJCT”:
                                if (param == ”source”)
                                        currentQuest.questSource.gameObject.SetActive (
                                             false);
                                else if (param == ”target”)
                                        currentQuest.questTarget.gameObject.SetActive (
                                             false);
                                else {
                                        GameObject obj = FindInScene (param);
                                        if (obj != null)
                                                obj.SetActive (false);
                                }
                                break;
                        case ”SHOWOBJCT”:
                                if (param == ”source”)
                                        currentQuest.questSource.gameObject.SetActive (
                                             true);
                                else if (param == ”target”)
                                        currentQuest.questTarget.gameObject.SetActive (
                                             true);
                                else {
                                        GameObject obj = FindInScene (param);
                                        if (obj != null)
                                                obj.SetActive (true);
                                }
                                break;
                        case ”MOVDIALOG”:
                                if (param == ”source”)
                                        dialogueReference.MoveDialogue (currentQuest.
                                             questSource);
                                else if (param == ”target”)
                                        dialogueReference.MoveDialogue (currentQuest.
                                             questTarget);
                                else {
                                        GameObject obj = FindInScene (param);
                                        if (obj != null)
                                                dialogueReference.MoveDialogue (obj);
                                }
                                break;
                        case ”NEWDIALOG”: // Update Dialogue #1 and #2
                                if (currentQuest.pecsList == null || currentQuest.pecsList
                                    .Length == 0 ) { //PECS do not exist/empty, Update #1
                                        dialogueReference.UpdateDialogue (param);
                                } else { //PECS do exist, Update #2
                                        dialogueReference.UpdateDialogue (param,
                                             currentQuest.pecsList);
                                }
                                break;
                        case ”PLAYANIME”:
                                tlReference.PlayTimeline (param);
                                break;
                        case ”SHOWARROW”:
                                if (param == ”source”)
                                        arrowReference.TargetArrow (currentQuest.
                                             questSource.transform);
                                else
                                        arrowReference.TargetArrow (currentQuest.
                                             questTarget.transform);
                                arrowReference.ShowArrow();
                                break;
                        case ”PLYPARTCL”:
                                particleReference.PlayParticle (param);
                                break;
                        default:
                                Debug.Log (”COMMAND: ” + command + ”_NOT_VIABLE”);
                                break;
                        }
                        //Debug.Log (command + ”−” + target);
```

```
        } else {
            switch (command)
            {
            case ”HIDEDIALG”:
                dialogueReference.HideDialogue ();
                break;
            case ”HIDEARROW”:
                arrowReference.HideArrow ();
                break;
            case ”NEWDIALOG”: // Update Dialogue #3
                dialogueReference.UpdateDialogue (currentQuest.pecsList);
                break;
            case ”NEXTQUEST”:
                SetCurrentQuest (currentQuest.nextQuestID);
                break;
            default:
                Debug.Log (”COMMAND:” + command + ”␣NOT␣VIABLE”);
                break;
            }
        }
    }
}

// Checks if the current quest had a command toCheck in its command list
public bool HasCommand(string toCheck) {
    if (currentQuest.commands.Length != 0) {
        foreach (string command in currentQuest.commands) {
            string c = command.Substring (0, 9);
            //Debug.Log (”Compare: ” + c + ”?=” + toCheck);
            if (c.Equals (toCheck)) {
                //Debug.Log (”Command:” + toCheck + ” found!”);
                return true;
            }
        }
    }
    //Debug.Log (”Command:” + toCheck + ” not found :(”);
    return false;
}

private GameObject FindInScene(string toFind) {
    GameObject foundObj = GameObject.Find (toFind);

    if (foundObj == null) {
        foreach (GameObject obj in Resources.FindObjectsOfTypeAll<GameObject>())
        {
            if (obj.hideFlags == HideFlags.NotEditable || obj.hideFlags ==
                  HideFlags.HideAndDontSave)
                continue;
            if (obj.name == toFind) {
                foundObj = obj;
                break;
            }
        }
    }

    /*
```

```
                if (foundObj == null)
                        Debug.Log (toFind + " not found!");
                else
                        Debug.Log (toFind + " found!");
                */

                return foundObj;
        }

        private int QuestIDSort(Quest a, Quest b) {
                return a.ID.CompareTo (b.ID);
        }

        private void GetPECNames() {
                List<string> pecNames = new List<string> ();
                foreach (Quest q in questList) {
                        if (q.pecsList != null) {
                                foreach (PECS p in q.pecsList) {
                                        if (!pecNames.Contains(p.PECSText)) {
                                                pecNames.Add (p.PECSText);
                                        }
                                }
                        }
                }

                foreach (string s in pecNames)
                        Debug.Log (s);
        }
}
```

ToyController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ToyController : MonoBehaviour {

        private Vector3[] toyPositions;
        private Quaternion[] toyRotations;
        private int[] random;

        void Start () {
                toyPositions = new Vector3[5];
                toyRotations = new Quaternion[5];
                random = new int[5];

                int i = 0;
                foreach (Transform toy in transform.transform) {
                        toyPositions [i] = toy.localPosition;
                        toyRotations [i] = toy.rotation;
                        random [i] = i;
                        i++;
                }
        }
```

```
    // Randomizes the positions of the toys in the Play Area
    public void RandomizeToyPositions() {
        foreach (Transform toy in transform.transform) {
            if (!toy.gameObject.activeSelf)
                toy.gameObject.SetActive (true);
        }

        // Knuth shuffle
        for (int t = 0; t < random.Length; t++) {
            int tmp = random [t];
            int r = Random.Range (t, random.Length);
            random [t] = random [r];
            random [r] = tmp;
        }

        int i = 0;

        foreach (Transform toy in transform.transform) {
            toy.localPosition = toyPositions [random [i]];
            toy.rotation = toyRotations [random [i]];
            i++;
        }
    }
}
```

QuestGiver.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class QuestGiver : MonoBehaviour, TimedInputHandler {

public Animator m_animator;
public PlayerController playerReference;
public QuestManager questReference;
public GameObject parentPositioner, burger;

[Range(0.5f,10)]
public float lookSpeed;
[Range(1,30)]
public float maxDistance;
private int lastRot;
//private bool lookAtPlayer;

// Use this for initialization
void Start () {
    lastRot = (int) parentPositioner.transform.rotation.eulerAngles.y;
}

void Update () {
    if (m_animator.GetBool(”LookAtPlayer”)) {
        float time = Time.time;
        int currentRot = (int) parentPositioner.transform.rotation.eulerAngles.y;
        int rotChange = lastRot - currentRot;
        int turnDirection = m_animator.GetInteger ("Turn");

        if (rotChange > 0) { // Turn left
            if (turnDirection != 1) { // Not turning left
                m_animator.SetInteger ("Turn", -1);
                //Debug.Log (time + "Turn right" + rotChange);
            } // else do nothing, since already turning left
        } else if (rotChange < 0) { // Turn right
            if (turnDirection != -1) { // Not turning right
                m_animator.SetInteger ("Turn", 1);
                //Debug.Log (time + "Turn left" + rotChange);
            } // else do nothing, since already turning right
        } else {
            if (turnDirection != 0) { // Not standing still
            m_animator.SetInteger ("Turn", 0);
            //Debug.Log (time + ":Stand still:" + rotChange);
        }
    }

    Vector3 target = playerReference.transform.position - parentPositioner.transform.position;

    float step = lookSpeed * Time.deltaTime;
    Vector3 newRot = Vector3.RotateTowards (parentPositioner.transform.forward, target, step,
         0.0F);
    newRot.Set (newRot.x, 0, newRot.z);
    Debug.DrawRay (parentPositioner.transform.position, newRot, Color.red);

    Quaternion newQuat = Quaternion.LookRotation (newRot);
    parentPositioner.transform.rotation = newQuat;
    //Debug.Log(newQuat);
    float dist = target.magnitude;
    if (dist > maxDistance) {
        //lookAtPlayer = false;
        m_animator.SetBool("LookAtPlayer", false);
        m_animator.SetInteger ("Turn", 0);
        return;
    }
    lastRot = currentRot;
    //Debug.Log (dist);
    }
}

    public void HandleGazeEnter () {
        //lookAtPlayer = true;
        m_animator.SetBool ("LookAtPlayer", true);
    }

    public void HandleTimedInput() {
    if (burger.activeSelf)
        m_animator.Play ("Eat");
    else
        m_animator.Play ("Wave");
    }
}
```

XI. Acknowledgements

First and foremost, I'd like to thank God for everything I've experienced thus far in my college life. All the blessings and boons, trials and tribulations, wins and losses. For every laugh, tear, smile, and frown I've shared with my peers, I thank You. It is because of these things that I am where I am now in life, and I stand here proud and content in light of all the experiences I've undergone.

Next, I thank my loving and caring family. To my parents, thank you for all the support you've shown me and my decisions in life, despite most of them going against what you planned for me years ago. I hope that my accomplishments these past few years have made the both of you proud. Thank you also for raising me to be a kind and understanding young man and I apologize for the times when I forget that. I love you both so much and I promise I'll repay everything you've given me a hundred fold in due time. To my wonderful siblings, thank you for always being there for me whenever I need someone to lean on. Even though we aren't always together, with half of us in the US and the other half here in the Philippines, I can always feel your love. Whether it be through video chat, embarrassing comments on my fb posts, or even Dota 2 games. Thank you for loving me, Ate Izel, Ate Irene, Kuya Mike, and Kuya Ken. Love, bunso.

Next, I thank all my great friends in Block 10. Cheers to all the amazing moments we've had these 4 years. I know that there is great success awaiting all of you in the future, like a wise man once said, "Tiwala lang."

To the D buddies, Sigfreed, Kharl, Kenneth, and Cess, salamat sa mga heart to heart na nakaraan at sa mga hangouts na parating. Kahit biglaan lang nabuo grupo natin, sobrang fond na talaga para sa akin yung mga memories na magkasama tayo. To more walwa– study sessions in the future.

To my Chorale family. See you on tour! Thank you for keeping me sane throughout my college life. Salamat sa lahat ng rehearsals, gigs, concerts, lakads, atbp. Here's

to many more! Let's make this tour a successful and memorable one. Thank you to the execomm and my section leader of understanding whenever I had to skip or be late to rehearsals so I could work on my thesis. I promise that the extra time I was given has been definitely worth it, and I'll repay all your kindness and understanding in full this coming tour.

To my ates from 1002, thank you for always being there when I need a shoulder to cry on, an ear lent, a study buddy to motivate me, or even just a friend in my time of need. Thank you for helping me whenever times get tough, and for always being there. Wuvyu, Ate Kei, Ate Ei, and Ate Cj! Also, super special thanks to ate Cj for helping test and build upon my thesis time and time again! You've been such a huge help in helping me learn more about the subject matter so I can have a clearer vision of what I wanted to accomplish. Thanks again and hope to celebrate soon! ¡3

To my adviser, Sir Marvin, thank you for taking me in and believing that I could finish my SP on time. With your help, I was even able to propose first in my block and create an SP I can be proud of. You've helped instill in me a greater sense of purpose and made me think more about what I can leave behind after I graduate.

To Team VR, mamimiss ko mga cafe outings natin (kahit most of the time nang-gagago lang kami ni Romeo). Congrats sa ating lahat! Good luck sa lahat ng future endeavors niyo ¡3

To our domain expert, Ma'am Quilantang, a huge thank you for the amazing help and guidance you've provided us! Without your insight and experience, my thesis wouldn't remotely resemble what it is today. Thank you for being so supportive as well, and I hope that my thesis can make even just a bit of difference for the better in the field.