Grasp-Shell vs Gesture-Speech: A comparison of direct and indirect natural interaction techniques in Augmented Reality

Thammathip Piumsomboon*, David Altimira, Hyungon Kim, Adrian Clark, Gun Lee, Mark Billinghurst

HIT Lab NZ, University of Canterbury, Christchurch, New Zealand

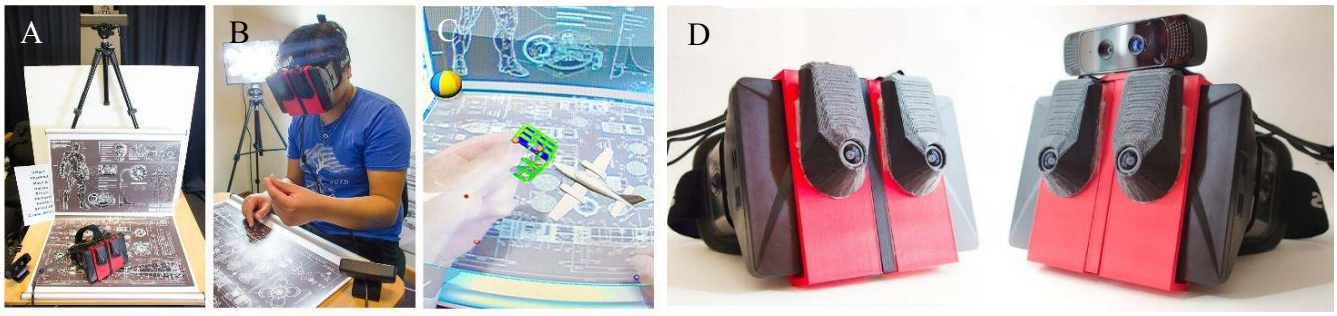

Figure 1: (A) Experimental setup, (B) A user is grasping onto a tiny virtual rubik's cube, (C) The user's view, and (D) Two versions of AR-Rift.

ABSTRACT

In order for natural interaction in Augmented Reality (AR) to become widely adopted, the techniques used need to be shown to support precise interaction, and the gestures used proven to be easy to understand and perform. Recent research has explored free-hand gesture interaction with AR interfaces, but there have been few formal evaluations conducted with such systems.

In this paper we introduce and evaluate two natural interaction techniques: the free-hand gesture based *Grasp-Shell*, which provides direct physical manipulation of virtual content; and the multi-modal *Gesture-Speech*, which combines speech and gesture for indirect natural interaction. These techniques support object selection, 6 degree of freedom movement, uniform scaling, as well as physics-based interaction such as pushing and flinging.

We conducted a study evaluating and comparing Grasp-Shell and Gesture-Speech for fundamental manipulation tasks. The results show that Grasp-Shell outperforms Gesture-Speech in both efficiency and user preference for translation and rotation tasks, while Gesture-Speech is better for uniform scaling. They could be good complementary interaction methods in a physics-enabled AR environment, as this combination potentially provides both control and interactivity in one interface. We conclude by discussing implications and future directions of this research.

Keywords: Augmented reality, natural interaction, multi-modal interface.

Index Terms: H.5.2. User Interfaces: Interaction styles, user-centered design.

1 INTRODUCTION

Augmented Reality (AR) overlays virtual content into the real environment [1], removing the boundary between the physical and virtual, and creating more engaging experiences with virtual content. Beyond the visual augmentation aspect of AR, heightening interactivity and increasing precision of interaction are crucial in enhancing the user experience. People naturally interact in the real world using speech and free-hand gestures, so we are interested in how similar gesture and speech methods can be used for connecting real and virtual worlds.

Previous research has explored a number of different natural interaction techniques in AR. Of particular relevance is the recent work on indirect, gesture and speech techniques [2], and direct free-hand gesture manipulation [3, 4]. These prior works raise several questions we would like to explore: When should we use one natural interaction technique over another in AR? Does the directness and indirectness of the interaction impact usability? For a given task, is there an easy to perform interaction for both free-hand gesture and multimodal input that would eliminate the need for arbitrary mapping of commands by the interface designer? Is it possible to achieve both interactivity and precision with direct manipulation using free-hand gesture as the primary input?

In this paper, we define interactivity as the user's ability to manipulate virtual objects and precision as the level of control the user has when interacting. An example measure of precision would be how small can an object be until a user can no longer grasp it, or how accurately can the user rotate or translate an object to match a target. Recent research [3, 4] has suggested that high levels of interactivity and precision can be achieved through natural interaction and demonstrations of this work has shown benefits in application areas such as interactive games, 3D modelling, rapid prototyping, and remote collaboration.

Consequently, we developed our own custom interaction engine, G-SIAR (Gesture-Speech Interface for Augmented Reality, pronounced "g-seer") that uses gesture and speech as the primary inputs, provides visuals cues such as shadows and hand occlusion through graphic shaders, supports a physics-enabled environment in AR, and offers seamless object creation. The AR experience was delivered on a wide field-of-view (*fov*) video see-through head-mounted display, known as the AR-Rift[1], which was made by mounting wide angle stereo cameras on the Oculus Rift display.

In AR, free-hand gesture input has not been well studied, and in this paper we present one of the first formal comparisons of gesture and multimodal input in AR. There are a number of

* thammathip@yahoo.com

LEAVE 0.5 INCH SPACE AT BOTTOM OF LEFT COLUMN ON FIRST PAGE FOR COPYRIGHT BLOCK

[1] Steptoe. "AR-RIFT", http://willsteptoe.com

2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 10-12 Sep 2014. 73-82.

attributes unique to AR, such as the need to provide virtual depth cues in the real world, occlusion of virtual objects by real hands, precise tracking of the users real hands relative to the virtual content, etc, which make it valuable to conduct studies that may be similar to those already conducted in Virtual Reality (VR) or other contexts. For this reason we believe that our techniques within the AR context provide important results complimentary to previous non-AR techniques.

By using current generation hand tracking and speech recognition technology, high precision free-hand gesture and speech interface input is made possible. In this research, our focus was on designing, formally evaluating, and comparing natural interaction techniques based on existing research guidelines [2, 5]. This lead to the development of two techniques: a free-hand gesture technique named *Grasp-Shell* (*G-Shell*) and a multimodal technique named *Gesture-Speech* (*G-Speech*). We conducted a user study to evaluate these techniques for single object relocation, multiple object relocation, and uniform scaling. From the results, we found difference in terms of efficiency, usability, and user preference. We discuss the impact of the directness and indirectness of interaction in each task, and explore some implications of the results.

This work makes the follow original contributions:

(1) Guidelines for implementing an interactive system, G-SIAR that uses gesture and speech as the primary input in AR.

(2) The design and implementation of the G-Shell and G-Speech interaction techniques.

(3) Results from a formal user study comparing the two interaction techniques.

(4) Discussion of the findings and their implications.

2 Related Work

2.1 Free-hand Gesture Interfaces in AR

While prior research has demonstrated the use of free-hand gesture in AR, there is no consensus on how the combination of these technologies can best serve users. For example, Handy AR [6] demonstrated intuitive hand interaction but the gestures were limited to an opened/closed hand for object selection and a small degree of hand rotation for object inspection.

The introduction of consumer depth sensors such as the Kinect made real-time 3D image processing possible. Using depth input, 6D hands [7] demonstrated six degree-of-freedom (*dof*) bimanual hand tracking for a Computer Aided Design application. The work involving simulated grasping using a physics engine for interaction [8, 9]. However these systems were limited to interaction on a 2D tabletop, and not in 3D AR/VR space.

In situated AR, the focus has been on interaction in the tabletop arm-reach distance space. Recent research [3, 4] has shown that allowing physical interaction between free-hand and virtual objects enhances user experience and increases believability of the existence of virtual content in the real world. This research provided a realistically simulated physical environment, using particle proxies representing the hand for physical interaction with virtual content. With this technique, the user could grasp onto the virtual object using the simulated force and friction between the hand proxies and the virtual object. However only limited support for gesture recognition was offered and there was no account for the wide range of expressive hand gestures that could potentially be used for input commands. Furthermore, this technique demonstrated a trade-off between interactivity and precision for the direct manipulation method using free-hand gestures.

Commercially, the G-Speak[2] platform supports multi-user gesture input either with gloves or using depth sensors aimed at multi-screen wall/tabletop displays. However, the system was not designed for 3D AR/VR space. Another offering is the Meta Pro head mounted display[3], which combines a stereoscopic optical see-through display with a depth sensor to support content viewing and natural hand interaction. Although this is a promising solution, the hand tracking technology is lacking compared to 6D hands [7] due to lower accuracy and fewer *dof* for interaction.

To support natural interaction in AR, we demonstrate a novel free-hand interaction technique, G-Shell that is based on 6D hands technology. G-Shell introduces the novel concept of an interaction shell, which offers both precision and interactivity.

2.2 Multimodal Gesture and Speech Interfaces in AR

Past studies of multimodal interfaces for AR mainly use hands for pointing during object selection to provide context for spoken deictic terms such as “that”, and “there” [10, 11]. In [12], multimodal information visualization was developed supporting 2D free-hand gesture, however there was no support for object manipulation in 3D space and no usability study was conducted.

Irawati et al. [13] developed a computer vision based AR system with multimodal input, allowing a user to pick and place virtual furniture in an AR scene using a combination of paddle gestures and speech commands. In the evaluation study they found that multimodal input enabled subjects to complete a task faster than with gesture alone. However this system required a handheld paddle, and did not use free-hand input.

A more recent study compared speech-only, gesture-only, and multimodal input in AR for translation and changing shape and color tasks [2]. They found that the multimodal condition was more usable than the gesture-only interface, and was more satisfying to use than speech-only conditions. However, the study was only conducted for a translation task on a 2D surface.

In this paper, we present a multimodal interaction technique, G-Speech, that supports both deictic gestures such as hand pointing, and metaphoric gestures, which offer spatial or movement information. We demonstrate its use for translating, rotating, and scaling single and multiple objects in 3D space in section 4.2. Although G-Speech is not as novel as G-Shell, it serves as a baseline representing multimodal interaction in this study.

2.3 Existing Interface Guidelines

In [14], a Wizard of Oz (WOz) study was conducted for a multimodal interface in AR to measure speech and gesture timing, and the types of gestures and speech used in virtual object manipulation tasks, such as selection, moving, and changing object shape/color. They found that 65% of all the gestures used were deictic, followed by 35% for metaphoric gesture. Based on this, they provide design guidelines such as using an accurate gesture-triggered system that is adept at recognizing pointing and metaphoric gestures, and to use context-based multi-signal fusion. They also emphasized that phrase-based speech commands should be used, that audiovisual feedback is crucial, and that learning modules should be applied in the multimodal fusion architecture.

Similarly, another participatory design study for user-defined gestures for AR [5] found that most of the gestures elicited from the subjects were physical gestures (39 %) for tasks such as move, rotate, scale, delete etc. These results matched those of Wobbrock et al. [15], whose original study for surface computing inspired the latter work. The authors pointed out that while the approach had already been applied [3, 4], these works had only supported basic manipulation with limited precision and control over the virtual content. They proposed that better control could be achieved by manipulating the physics engine’s constraints, or by making use of the collision detection component without dynamic simulation for tasks such as object selection, scaling etc.

[2]www.oblong.com/g-speak

[3]www.spaceglasses.com

We based the designs of our interaction techniques on these interface guidelines, as they offered empirically tested suggestions to support ease of use, naturalness and intuitiveness.

3 System Design and Implementation

In this section, we give an overview of our custom built interactive AR framework called G-SIAR. We discuss the current hardware and software choices in Section 3.1 and 3.2, respectively. The general design goals of G-SIAR that we set out to achieve were to:

i. Support the use of natural gesture and speech as inputs for AR interaction.
ii. Provide an interactive and precise environment that can supports a wide range of applications.
iii. Offer experiences across the reality-virtuality continuum from AR to VR.

3.1 G-SIAR Hardware

The hardware design goals were as follows:

i. Support free-hand tracking.
ii. Provide highly immersive and large viewing coverage of the interaction space.
iii. Support transitions across the reality-virtuality continuum from AR to VR.

The hardware used in our current system includes:

A. AR-Rift (Oculus Rift HMD mounted with wide-angle stereo cameras)
B. PrimeSense Carmine 1.09 (depth sensor)
C. Creative Senz3D camera (depth sensor)
D. Image-based marker (A1 paper size)
E. Alienware 17 laptop (Intel Core i7-4800MQ with 2.70Ghz CPU and Nvidia GeForce GTX 780M)

3.1.1 AR-Rift, Customized Video See-through HMD

We chose to use the Oculus Rift HMD as a display device due to its large *fov*. By attaching wide angle stereo cameras to the Rift, we can deliver a highly immersive user experience across the whole Mixed Reality (MR) spectrum [16]. The AR-Rift implementation was based on the design of Steptoe[4], who created the wide-angle stereo camera setup that matched the display properties of the Rift. We used Logitech C270 cameras with 1280 x 960 resolution and video capturing capability of 800 x 600 at 30 fps. We replaced the original Logitech C270 lenses with Genius WideCam F100 wide-angle lenses, resulting in a horizontal and vertical *fov* of approximately 116° and 94° respectively. By rotating the camera 90°, video could be captured at 600 x 800 in 3:4 aspect ratio and after padding the image horizontally by 20 pixels on both sides and shifting the image depending on calibration for the user, the resulting image matched the Rift display for each eye of 640 x 800. The camera mounts and protective covers were custom designed and 3D printed. Two version of the AR-Rift were developed, one similar to the original and another with a depth sensor mounted on top as shown in Figure 1-D. We discuss the implication of both models in Section 3.2.5. Figure 3-A illustrates the AR-Rift view.

3.1.2 Interaction Space

Hand tracking was performed using a PrimeSense Carmine 1.09 depth sensor. This was positioned on a tripod pointing down 700 mm above the interaction surface. The range and *fov* of the depth sensor defined the size of our interaction space, which was 600 x 450 x 450 mm (width x depth x height). An image-based marker was created to allow for positional tracking of the AR-Rift (see Figure 1-A), and doubled as a visual boundary of the interactive space. In the user study, the high quality microphone array in a Creative Senz3D camera captured speech commands.

3.2 G-SIAR Software

When we designed G-SIAR, these were our goals:

i. High *dof* hand pose estimation for free-hand tracking.
ii. Support for physics-enabled simulation with high accuracy.
iii. Support for real-time and seamless object creation and interaction experience.
iv. Support for realistic rendering with shadows, hand occlusion, and correct distortion for display on the Rift.

3.2.1 Architecture Overview

Figure 2 illustrates a simplified version of the G-SIAR architecture, inputs to the system include raw hand poses data from 3Gear Nimble SDK[5], audio and depth images captured by Creative Senz3D, and dual video streams from the AR-Rift stereo cameras. The output includes visual feedback through the AR-Rift and audio feedback through speakers or headphones.

G-SIAR is multi-threaded and every module runs in its own thread. The key software components include 3Gear's Nimble SDK (hand tracking), OpenSceneGraph[6] (graphics), GLSL (shader), Bullet[7] (physics), OpenCV[8] (image processing), osgBullet[9] (graphics and physics interface), OPIRA[10] (marker tracking), Intel Perceptual SDK[11] (camera capture and speech recognizer), FMOD[12] (audio output), Oculus SDK[13] (display), and Boost[14] (threading and data structure). In the remaining subsections of this section, we described the most important component in more detail.

3.2.2 Gesture Recognition

The key to high precision free-hand gesture interaction is high accuracy and high *dof* hand pose estimation. The 3Gear Nimble

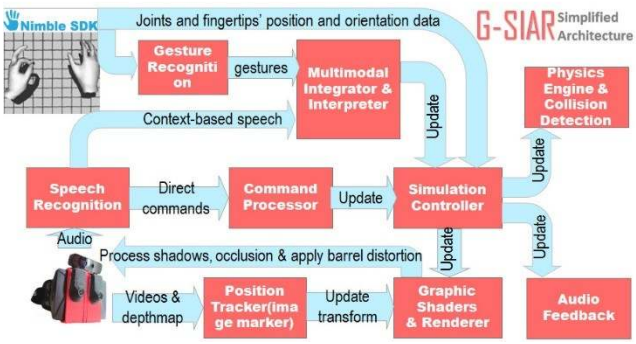

Figure 2: G-SIAR architecture.

SDK is based on the research of 6D hands [7], which demonstrated six *dof* bimanual manipulation using a single depth camera. It provides very accurate six *dof* free-hand tracking; tracking the wrist, 15 joints, and 5 fingertips for both hands with millimeter-level precision for every finger at 30 fps.

3.2.3 Direct Inputs and Multimodal Inputs

Gesture and speech can be used for both individual and multimodal input. By itself, speech input can be used to issue system commands that do not require context, such as "enable gravity" to turn on gravity in the simulation. Gesture input can provide hand position and orientation for direct physical interaction, for example physically pushing objects with the

[4] www.willsteptoe.com
[5] www.threegear.com
[6] www.openscenegraph.org
[7] www.bulletphysics.com
[8] www.opencv.org
[9] www.osgbullet.vesuite.org/
[10] www.hitlabnz.org, search for "OPIRA"
[11] software.intel.com/en-us/vcsource/tools/perceptual-computing-sdk
[12] www.fmod.org
[13] www.oculusvr.com/
[14] www.boost.org

fingertips. For multimodal input, G-SIAR includes a multimodal integrator, which is responsible for determining the action required as a result of combined speech and gestures. The integrator acts when a verbal command is given that requires gestural context. For example when the user points at an object and says "change the color to red", the integrator informs the interface controller of the action, the selected object and the color information, which then turns the selected object's color to red.

3.2.4 Contacts Points and Dynamic Simulation

In AR, physics-based simulation can enhance the user experience in terms of realism and believability, and can also improve usability. In the real world, when a user grasps an object, interaction between the fingertips and the object is initiated at the contact point(s). In G-SIAR, we track the user's fingertips and use a physics engine's collision detection to monitor the occurrence of contact in 3D space. Using this information, we can interpret the user's intention and provide the appropriate interaction.

3.2.5 Shaders for Shadows, Occlusion, and Distortion

GPU shaders were used to provide realistic hand occlusion, shadows, and visual distortion. The rendering framerate was approximately at 60fps. In order to achieve this, we had to explore several recognition and rendering techniques, as described below.

A. Skin segmentation and per-pixel occlusion

The depth data provided by the Creative Senz3D mounted on top of the AR-Rift is retrieved and matched to a skin color segmented image to create a depth map containing only values assumed to belong to the hands. This depthmap is passed to the shader to test the depth of each pixel at render time. If the hand pixel was closest then no virtual content rendered, at this location and the texture of the hand is shown instead (see Figure 3-B).

The advantage of this technique is sharp per-pixel occlusion; however there are two significant disadvantages. Firstly, the mismatch in the *fov* between the Creative Senz3D and the AR-Rift stereo cameras means that occlusion can only be applied for a small portion of the whole scene. Secondly, the Creative Senz3D made the AR-Rift heavier and this reduced the overall user experience. For these reasons, this approach was abandoned.

B. Skin segmentation and ray-casting

In this technique, skin color segmentation was applied to the stereo camera images. Occlusion is determined using ray-casting from the viewing camera position to each fingertip. If no object was intersected by the ray, any pixel that was of skin tone in the shader is discarded and the background texture is displayed.

This technique removes the need for the Creative Senz3D, however it was less accurate as occlusion is based on binary decision rather than by depth. This issue could be overcome by calculating a disparity map from the stereo cameras, however this would be a computationally expensive process (see Figure 3-B).

C. Semi-transparent hand reconstruction

The final technique involves using the hand model that was reconstructed for hand shadows as a semi-transparent proxy for the actual hand. This allows users to see their actual hand with the virtual hand overlaid on top of it, while still having shadows cast on the virtual objects. The main advantage of this method compared to the others is that there was no additional hardware or computation required. An additional benefit was that virtual objects are still partially visible even when occluded so the user could tell what was behind their hands (see Figure 3-C). In order to get the best results, an additional calibration step is required to align the virtual hands with the real hands as precisely as possible.

We evaluated techniques B and C during a pilot study, and participants reported that they felt they could perform the task equally well using either method. As technique C required less computation than B, we chose to use the semi-transparent hand reconstruction method for occlusion in the experiment.

Figure 3: (A) AR-Rift's view, (B) Hand occlusion of method A and B, and (C) Hand occlusion of method C.

3.2.6 Seamless Object Creation and Interaction

G-SIAR supports dynamic creation of objects and every object can be interacted with using the G-Shell and G-Speech interaction methods. In the current version, we support object creation using a "solid of revolution" and "solid of extrusion" tool, where user can draw the outline of an object and a solid model will be generated. We also support the loading of external models in various formats, and even model exporting, so that users can export their created models for 3D printing.

4 Designing the Interaction Techniques

Within the G-SIAR framework, we designed and implemented two interaction techniques, Grasp-Shell and Gesture-Speech. The design approach was modular, so that either one or both techniques could be applied at any time.

4.1 Grasp-Shell (G-Shell)

A common assumption in AR is that virtual objects overlaid in the real environment should offer the same interactive affordances as real objects. As mentioned in Section 2, a user-defined gestures study [5] found that 39% of all the gestures elicited from participants for interaction in AR were "physical", meaning that they were gestures that acted physically on the virtual object as if it was a real object. As a result, our design goals for G-Shell were:

i. Demonstrate direct manipulation through free-hand gesture interaction.
ii. Learn from the user-defined gesture for AR study [5] and apply the gestures elicited when possible.
iii. Support both interactivity-oriented and precision-oriented tasks.

4.1.1 Contacts, Shell and Penetration Distance

Natural physical interactions such as grasping require points of contact between the hand and virtual object. However without any tactile feedback it is difficult to know if the hands are in contact with the object. Previous research [3, 4] applied a constraint that disallowed virtual hand proxies from penetrating the object. However, in a physics-enabled simulation, the response to a collision between two objects is that one or both of the objects will move apart in a direction defined as the contact normal. We refer to this behaviour as being in "*dynamic mode*". Because of this reaction, when attempting to grasp an object in a physically simulated environment, the user may unintentionally nudge the object out of reach. To resolve this, a "*kinematic mode*" is introduced, where the physical contact response of the object is disabled and the user has a full control over the object.

In order to provide a natural interaction technique, such as grasping, that universally works for object of all shapes and sizes, we developed the concept of a Grasp-Shell (G-Shell). G-Shell allows natural free-hand grasping of virtual objects using an invisible "Interaction Shell". When contact occurs between the user's hands and the interaction shell the object's mode changes from "dynamic" to "kinematic", allowing for precise control and supporting multiple types of physical gestures.

The interaction shell is essentially a collision shape that approximates the hull of the model. The simpler shape means reduced computation for contact points and better performance. Conversely, a collision shape with greater details can offer more precise contacts for manipulation at a cost of more computation.

With G-Shell, the interaction shell is always larger than the object's collision shape, so that collision with the shell occurs before collision with the object. We define "shell thickness" as the distance from shell surface to the objects collision shape, and the distance penetrated by the finger into the shell as the *Penetration Distance in Percentage of Shell Thickness* (*PDST)*.

Division of the shell into multiple layers permits us to offer both kinematic mode actions where the physical collision response is disabled for full user control and dynamic mode actions where the physical collision response is enabled. From a preliminary study, we found the optimal shell thickness for alternating between these modes to be 5mm. G-Shell supports bimanual operation where two hands can be used at the same time to perform independent actions on separate objects. Listing 1 describes the G-Shell algorithm.

Listing 1: Simplified G-Shell Interface Handler Routine

```
IF contact between finger(s) and shell(s) > 0
    Set each object to kinematic mode.
    IF object is not selected THEN select it ELSE deselect it.
    IF No thumb contact
        IF finger's PDST > 0% and ≤ 50% THEN
            Check for kinematic gestures
        ELSE IF finger's PDST > 50% and ≤ 120% THEN
            Object is being pushed
        ELSE IF finger's PDST > 120% THEN
            Check for kinematic gestures
    ELSE there is a thumb contact
        IF thumb and one of contacted fingers distance < threshold
            THEN Grasping = true
            ELSE Grasping = false
            Check PDST of each finger for other actions
IF Grasping on both hands is true and on the same object
THEN resize the object based on the change in distance
            between hands
ELSE
    Move object(s) on each hand that Grasping is true
    IF fling condition is met THEN
        Disable kinematic mode and apply
        impulse, J = m * Δv = Δp on the object(s)
```

4.1.2 Actions in Kinematic Mode

Figure 4 shows an interaction shell cross section for a Porsche model. We define the "safe zone" as a *PDST* greater than 0% but less than 50%. This safe zone is intended for object selection (see Figure 5-A), and when activated the model of the object turns semi-transparent and a red outline is shown around the object to indicate that the kinematic mode is activated. In the kinematic mode, the physical collision response is disabled and object selection is toggled. Object selection allows for non-physical manipulation, such as changing the color of the object.

At a *PDST* over 120%, the object is again in the kinematic mode. A single handed grasp action in this zone would cause the object to be moved as if attached to the hand (see Figure 5-B). We define a grasp as a gesture where the distance between the thumb and the opposing finger that made the contact is smaller than 20mm so that objects of various size can easily be grasped and released.

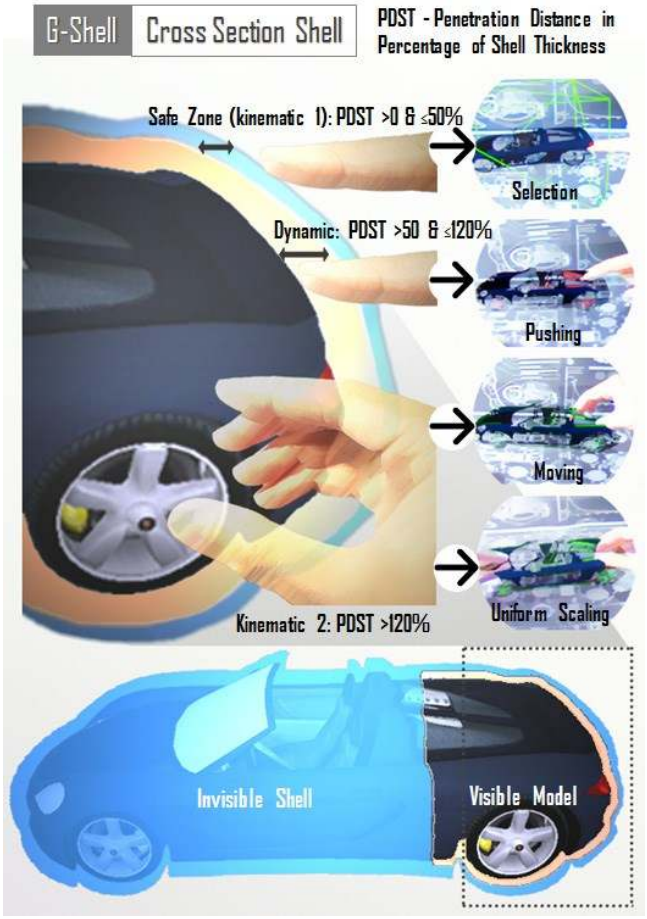

Figure 4: G-Shell's cross section and visualization.

In the event of a successful grasp, the object's outline turns from red to green. The relative transformation between the original object and the hand's frame of reference is stored and any translation and rotation of the object is applied on top of this relative transform. Moving is a 6 *dof* manipulation, where the hand can translate and rotate the object at the same time. Moving is also supported for multiple objects simultaneously (Figure 5-B).

In the case of bimanual actions, i.e. grasping with both hands on the same object, the system allows for uniform object scaling (see Figure 5-F, G). The change in size is determined by the difference between the current and initial positions of the thumbs. At the end of the resizing, the user can commit the change by releasing the grasp on their dominant hand first, or cancel by releasing the grasp on the non-dominant hand first.

4.1.3 Actions in Dynamic Mode

Currently, two dynamic actions are supported in G-Shell, pushing and flinging. Pushing is enabled at a *PDST* between 50% and 120%. Allowing a penetration of up to 120% guarantees a collision between a fingertip and the object's collision shape and the object will move a small distance in response to this slight penetration. During pushing, any contact made by the thumb disables the dynamic mode and enables the kinematic mode, as

we assume all gestures involving the thumb to be kinematic (e.g. grasping uniform scaling).

To fling an object, the user must first grasp the object, then quickly release it while moving their hand in the direction of the fling. A fling is only executed when the velocity between the thumb and index finger exceeds a given threshold of 0.3m/s. The impulse applied to the flung object was taken as the mass times the change in hand velocity, which is equivalent to the change in momentum of the object. The velocity vector is taken as the difference between the initial and final position of the hand.

4.2 Gesture-Speech (G-Speech)

In a recent WOz study [14], the authors suggested the best gesture-speech interaction approach is to use context-based multi-signal fusion and emphasized phrase-based speech input. For these reasons, our design goals for G-Speech were to:

i. Demonstrate indirect manipulation through free-hand gesture and speech.
ii. Use design recommendation from the WOz study [14].
iii. Support fast detection and interpretation of deictic gestures.
iv. Apply suitable metaphoric gestures learned from the user-defined gestures for the AR study [5].

4.2.1 Deictic Gestures

Deictic gestures are pointing gestures. They require understanding of real world geometry and physics. For example, when issuing a command to move an object, we might point at a place and say "move it there" and we would expect that the object would be placed in an appropriate location, for example placing a flower pot on a table or hanging a picture on a wall.

In a virtual 3D environment, these limitations and assumptions may not apply; the target location might be floating in 3D space where there is nothing for the pointing ray to intersect with. Consequently, for this study G-Speech was designed to support continuous actions such that when the user selected an object and said "move it", the object would be attached to the ray extending from the user's index finger (see Figure 5-H). The user could move the object and say "release" to commit the movement or "cancel" to cancel the action.

Pointing detection is based on the hand's geometry. We restrict the pointing gesture to the index finger or index and middle fingers only. By sampling the distance between the index, ring and pinky fingertips from a stable reference point such as the wrist, we can determine the probability that a pointing gesture is being made. Visual feedback is provided as a ray cast from the index finger into the distance, which is coloured red for the left hand and blue for the right (see Figure 5-C). When the ray intersects an object, it turns green, a yellow box appears around the object, and the user can issue voice commands to perform actions such as "move it". Objects which are pointed at can be selected by saying "select", at which point the yellow bounding box turns green. Multiple objects can be selected consecutively, otherwise the "select all" command can select everything. To deselect, the user can point at the object and say "deselect", or use the "cancel selection" command to deselect everything.

4.2.2 Metaphoric Gesture

We define metaphoric gestures as those that can convey spatial and/or movement information. When observing how people interact indirectly using gesture and speech in the real world, we noticed that translation and rotation operations were usually separated. For example saying "move the table there" while pointing at the floor, followed by "rotate it by this much" and showing the amount of rotation by twisting the hand. For this reason, we chose to separate translation and rotation operates in G-Speech, compared to the 6 *dof* interaction in G-Shell.

We designed the gestures for rotation and uniform scaling based on metaphors used in real life. For rotation, the user can say "turn it" with the amount of rotation indicated by a change in orientation of the user's hand. Rotation can also be performed for multiple objects at the same time as shown in Figure 5-D, E.

In the case of scaling, the metaphor used describes the object's size using the distance between the two hands or the thumb and index finger of one hand. For our implementation the scale factor is determined by the difference between the current and initial distances between the hands. The user is given continuous feedback showing the change in rotation and scale in the graphics model (see Figure 5-I, J).

4.2.3 Gesture and Command Pair

Although G-SIAR supports an unlimited pairing of gestures and commands, for the user study we limited the number of commands to the nine shown in Listing 2. This list was

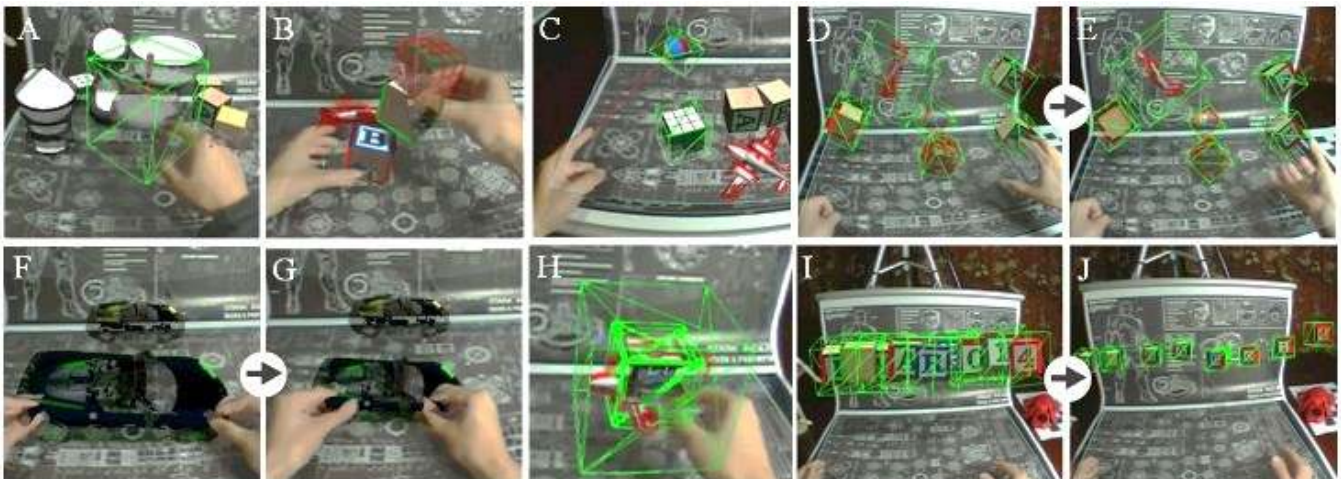

Figure 5: ***(A)*** G-Shell selection, ***(B)*** G-Shell - left hand moving a single object and right hand moving multiple objects, ***(C)*** G-Speech selection, ***(D, E)*** G-Speech multiple objects rotation, ***(F, G)*** G-Shell scaling, ***(H)*** G-Speech moving multiple objects, and ***(I, J)*** G-Speech scaling multiple objects.

determined in a pilot study as containing a sufficient number of commands to complete the task, while not requiring significant learning time or overloading the user's memory.

Listing 2: List of commands used in the experiment.

1: **SELECT**	6: **RELEASE**
2: **DESELECT**	7: **CANCEL**
3: **MOVE IT / TRANSLATE**	8: **SELECT ALL**
4: **TURN IT / ROTATE / TWIST**	9: **CANCEL SELECTION**
5: **RESIZE**	

5 User Study

In this study we were interested in the usability and user impression of the two interaction techniques. It was our belief that each technique had merits and weaknesses, but that neither technique would be universally better as different tasks would be better suited to different levels of directness of interaction. We predicted that relocation tasks would benefit from the direct interaction of the G-Shell technique, as it allows the user to apply their real-world experience in grasping and moving object(s). In contrast, resizing an object may be more challenging using direct manipulation due to the fine hand control required when resizing small objects, and as such be easier completed with G-Speech.

Because of this, we proposed the following hypotheses:

(*H1*) There is a difference in the resulting performance, usability, and preference between G-Shell and G-Speech in relocating object(s) in 3D space.

(*H2*) There is a difference in the resulting performance, usability, and preference between G-Shell and G-Speech in resizing an object.

5.1 Experiment

The environment our user study was conducted in was an arm-reachable near-space environment 600mm wide, 450mm deep, and 450mm high. Virtual objects ranged from 30mm to 400mm in size. We designed and divided the experiment into three tasks; (1) *single object relocation*, (2) *multiple object relocation* and (3) *uniform scaling*. We decided to focus this study on these three tasks because they contained fundamental actions that are the basis of many other functions, e.g. touching, grasping, moving and releasing for G-Shell as well as pointing, moving, uttering commands for G-Speech. We made sure the task load was appropriate to complete within a single session of the experiment given the number of actions and commands that the subjects needed to learn and remember for both interaction techniques.

5.1.1 Participants

Twenty one participants were recruited for the study, eleven males and ten females, with a mean age of 24.9 (SD = 6.09) years. Seventeen of the participants were right handed, two were left-handed and two could use both hands equally well. Nine participants had some experience with AR but only one considered himself knowledgeable on the subject. Twelve had some experience with gesture interfaces from playing with Kinect or Wii. Twelve had some experience with speech interfaces from using Siri or Xbox. There were nine native, seven fluent and five intermediate English speakers. All participants could communicate well and understood the experimental protocol.

5.1.2 Tasks

Task 1 focused on relocation of a single object, and featured ten subtasks, involving both translation and rotation. Task 2 focused on relocation of three objects and was comprised of five subtasks involving translation, and combined translation and rotation. Task 3 focused on resizing objects of varying shapes and sizes and was comprised of five subtasks.

For Task 1 and 2, the virtual object size ranged from 60mm to 100mm. The target matching condition was that the position and orientation difference between the object and the target had to be less than 20mm and 15°, respectively. For Task 3, the requirement was that the object's scale had to be within 10% of the target's scale. These limits were obtained in two pilot studies, where the matching requirement values were varied between 10, 15, and 20mm for position, 5°, 10°, and 15° for orientation, and 6%, 8%, and 10% for scaling. During these pilot studies, we measured the average time taken by users with no prior experience with either interaction techniques, and decided on the final values based on time taken. We were also able to determine that expert users could easily achieve matching conditions of 5mm, 5° and 5%.

5.1.3 Procedure

To counterbalance conditions, we alternated the presentation order of the interaction and for each order we distributed the gender equally. The experimental setup is shown in Figure 1.

Each experiment took 1 to 1.5 hours to complete. Before commencing the experiment, we gave each participant 5 minutes per interaction to learn in a sandbox program and another 5 minutes in a practice session that was similar to the experiment. During this learning period, we calibrated the system to make sure that it worked well for each participant. This calibration involved adjusting parameters to ensure accurate hand pointing regardless of hand size, and selection of verbal commands that the speech recognizer could accurately determine for each participant.

At the beginning of each task, the participant had a 3 second countdown, which displayed in the center of their view. As each task began and for every successful target matched, a sound of a bell was played. The object was displayed with opaque textures and a red outline, while the target was 50% translucent with a yellow outline. The task completion time started after the bell rang and stopped when all the targets in the scene were matched.

5.1.4 Data Collection and Analysis

The data recorded during the experiment included the task completion time for each subtask, the discontinuity occurrence of G-Shell (number of times grasped and released), the use of each command for G-Speech and the distance the hands travelled for both interaction techniques. The questionnaire included a 7-point Likert scale usability rating and NASA TLX for each condition, and a user preference for each task. Videos of the experiment from the participant's point of view were recorded.

We compared task completion time (**tct)**, usability rating, and NASA TLX between G-Shell and G-Speech for each task. We analyzed each task independently and applied a paired T-test for **tct**. We applied a Wilcoxon signed-rank test with continuity correction for the usability rating, and NASA TLX.

5.2 Results

In the following subsections, we report the results of the comparison of the two interaction techniques for **tct**, NASA TLX, 7-point Likert scale usability rating, preference, and general user feedback.

5.2.1 Task Completion Time (**tct**)

The G-Shell technique was significantly faster than G-Speech for single and multiple object manipulation, but not for scaling. The T-test showed a significant difference between the two interaction techniques in term of **tct** for Task 1 *single object relocation*, with M = 23.77s (SD = 25.72) for G-Shell and M=

42.69s (SD = 73.5) for G-Speech where t(209) = -3.78, p < 0.001. The same was true for Task 2, *multiple object relocation*, with M = 56.94s (SD = 58.03) for G-Shell and M = 103.17s (SD = 87.2) where t(94) = -6.91, p < 0.001. For Task 3, there was no significant difference in **tct** where M = 16.0 (SD = 22.57) for G-Shell and M = 12.21 (SD = 10.17) for G-Speech.

5.2.2 NASA TLX

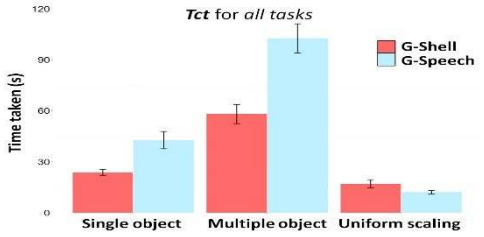

Figure 6: Task completion time, error bars represent +/-SEM.

G-Shell required significantly less *effort*, *frustration*, *mental*, *physical* and *temporal demand*, and provided significantly higher performance for Task 1. G-Shell required significantly less *temporal demand* for Task 2. G-Speech was significantly less *frustrating* for Task 3.

A Wilcoxon signed-rank test with continuity correction yielded a significant difference between the two interaction techniques in every category for Task 1, which comprised of *mental demand* (V = 37.5, p = 0.012), *physical demand* (V = 22, p = 0.033), *temporal demand* (V = 8.5, p = 0.01), *performance* (V = 104, p = 0.001), *effort* (V = 38.5, p = 0.013), and *frustration* (V = 19, p = 0.012). For this task, G-Shell required lower *mental demand*, *physical demand*, *temporal demand*, *effort*, *frustration*, and higher *performance*. However, a significant difference could only be found for *temporal demand* in Task 2 (V = 20, p = 0.043), which favoured G-Shell, and *frustration* in Task 3 (V = 66.5, p = 0.034) which favoured G-Speech.

We also applied the same test to compare Task 1 and 2 for each interaction. The test gave a significant difference between Task 1 and 2 for G-Shell in two categories, they were *mental demand* (V = 29, p = 0.014) and *temporal demand* (V = 13.5, p = 0.027).

5.2.3 Usability Ratings

G-Shell was rated significantly better for *single object relocation* and G-Speech was rated better for uniform resizing. There was no significant difference for the *multiple object relocation* task.

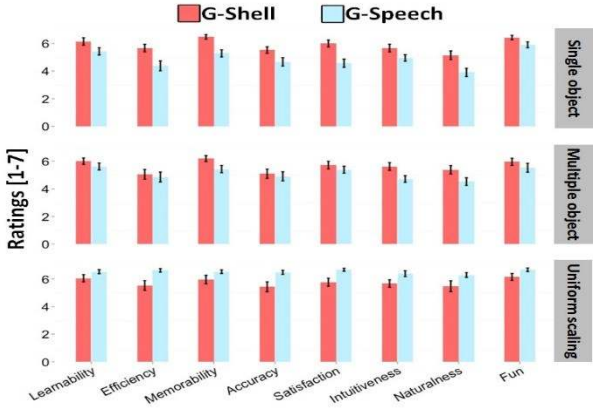

Figure 7: Usability rating, error bars represent +/-SEM.

A Wilcoxon signed-rank test with continuity correction was applied for usability ratings and resulted in a significant difference between the two interaction techniques in every attribute for Task 1, which were *learnability* (V = 87, p = 0.029), *efficiency* (V = 138, p = 0.003), *memorability* (V = 110.5, p = 0.004), *accuracy* (V = 96, p = 0.041), *satisfaction* (V = 144, p = 0.001), *intuitiveness* (V = 98, p = 0.028), *naturalness* (V = 123, p = 0.004), *fun* (V = 64.5, p = 0.042) where G-Shell had higher ratings.

For Task 2, only *memorability* (V = 95, p = 0.006), and *intuitiveness* (V = 114, p = 0.016) were significant in favour of G-Shell. For Task 3, *efficiency* (V = 14, p = 0.016), *accuracy* (V = 16, p = 0.023), *satisfaction* (V = 13.5, p = 0.013), and *intuitiveness* (V = 4, p = 0.009) were rated significantly higher for G-Speech.

The *goodness* score for G-Shell and G-Speech, was calculated as the average of all ratings for each task. The ratings for Task 1 were 5.88 and 4.89, for Task 2 were 5.63 and 5.12, and for Task 3 were 5.75 and 6.52, for G-Shell and G-Speech respectively. There was a significant difference between the two interaction techniques for Task 1 in favour of G-Shell (V = 211.5, p < 0.001) and Task 3 in favour of G-Speech (V = 32, p = 0.037).

5.2.4 Preference

Figure 8 shows that G-Shell was more preferable for *single and multiple object relocation* tasks, while G-Speech was preferred for uniform resizing task. We found that participants with prior gesture interface experience performed significantly better than those without in terms of **tct**. We cross-referenced this with the votes for each task, and found that for Task 2 *multiple object relocation*, the preferences of G-Shell, G-Speech, and Neither was 6, 1, and 4 respectively for users with gesture interface experience, and 4, 5, and 1 respectively for those without.

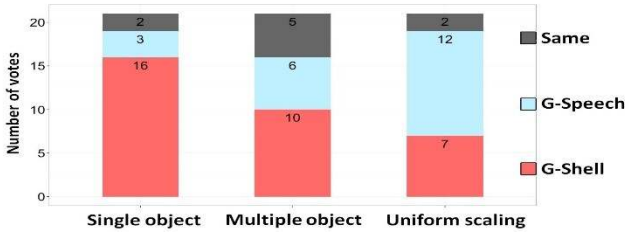

Figure 8: Preferences for all tasks.

5.2.5 General Feedback

We found common themes throughout the subjective feedback and summarized them into 7 motifs as follows.

Challenging but enjoyable and fun. Some participants found that performing certain tasks with certain interaction techniques was challenging for them. Nevertheless, all participants thought that the experiment was enjoyable and fun to do. For example, after performing the *multiple object relocation* task with G-Speech subject *P2* commented, *"It was challenging a bit, which was part of the fun but can also be irritating sometimes"*.

Natural and intuitive to grasp. Most of the participants found G-Shell's 6 *dof* movement and bimanual scaling, natural and intuitive. Subject *P13* gave the following comment, *"Love it. The interface is natural and requires little extra thought to manipulate. There is a very high level of similarity to reality, with the hand movements being intuitive"*.

Difficult imagining an indirect rotation. Some participants found it hard to conceptualize rotation remotely with G-Speech. *P9* expressed *"the rotation, again, was hard to imagine"* and *P11*

commented, *"I like this interface but would like more practice with rotation"*.

Easy and intuitive to resize. All of the participants found G-Speech's scaling interaction, easy and intuitive to use. Subject *P12* said, *"Very easy to use, intuitive, and allows for very accurate object manipulation, very efficient, and looks to be easy to use on multiple objects. Can think of no drawbacks! "*.

Smaller is harder. Some participants found it challenging to directly resize a small object with G-Shell. *P14* commented, *"Getting the finger in the right place for both hands is tricky"*.

Better control and thus more precision. The ability to move in 6 *dof* in G-Shell gave more freedom but reduced precision, particularly when the user wished to change only the position or rotation. Regarding G-Speech, *P3* stated, *"I liked this interface better because I felt like I had more control over the manipulations when I had to rotate or move the object with my hands. Mostly, this was because the objects also responded to my voice, so I didn't need to worry about not having effect fine motor skills in this program"*.

To speak or not to speak. The preference of interaction varied between participants and one of the key factors for this was the ability to use speech. *P4* favoured G-Shell for every task stating that, *"I don't like to keep talking all the time during the tasks"*, while *P7,* who favoured G-Speech for every task stated that, *"Because actions without the voice commands takes more effort, It was easier with voice commands to an extent"*.

6 Discussion

6.1 G-Shell vs G-Speech: Results of Our Hypotheses

The differences that were found in the statistical analysis of the two interaction techniques can be summarized in five main points:

(1) G-Shell was more efficient on average for combined translation and rotation tasks. We believe that the main reason for this is due to G-Shell supporting 6 *dof* interaction, where a single action can move and orient an object as opposed to two separate actions required by G-Speech.

(2) For single object relocation, G-Shell required less effort, frustration, mental, physical and temporal demand, and provided higher performance. However, for multiple object relocation G-Shell only required less temporal demand and for uniform resizing G-Speech was less frustrating. The results for temporal demand correlated to the time taken for single and multiple object relocation where G-Shell took less ***tct*** on average. After reviewing videos of the experiment, we believe the higher frustration with G-Shell for uniform scaling task is due to the impact of having to directly manipulate objects of varying shapes and sizes.

(3) In term of task load index, participants perceived that the multiple object relocation task was more mentally and temporally demanding than single object relocation task when using G-Shell. Introducing multiple objects has an impact on the G-Shell interaction technique as it requires direct object contact and hence more space to manoeuvre than G-Speech. This may explain the significant rise in perceived *mental* and *temporal demand* in the two tasks for G-Shell.

(4) In terms of usability, both interaction techniques were rated positively in all categories in every task, while G-Shell was rated significantly better for single object relocation and G-Speech was rated better for uniform resizing. Although, G-Shell was rated higher on average than G-Speech in the multiple object relocation task, there was not a significant difference between them.

(5) In term of preference, the majority of participants voted for G-Shell for single and multiple object relocation tasks, while G-Speech had the majority voting for the uniform resizing task. This matches our expectation that neither technique would be universally better than the other for every task.

6.2 Implications and Design Recommendations

Based on the results, we propose some implications and design recommendations for (1) direct free-hand interaction in AR, (2) multimodal interaction in AR, and (3) natural interaction in AR.

6.2.1 Direct Free-hand Interaction in AR

G-Shell demonstrated that direct free-hand interaction in AR can offer a high level of usability, which was confirmed by the average usability rating scores for all tasks being above 5 points and subjective feedback of being *Natural and intuitive to grasp*. The participants could directly manipulate virtual objects as if they were real, and expressed that it was believable, enjoyable and immersive. We observed that G-Shell was good for bimanual manipulation of single or small groups of objects. Nonetheless, direct manipulation of an object has limits with respect to the object's size and how cluttered the scene is. We observed that some participants had difficulty manipulating objects that are too small or when there were too many objects cluttered together as pointed out in *Smaller is harder*. We propose that this can be overcome by providing zoom functionality where the whole scene can be resized and manipulation can be performed at a more manageable scale. Therefore we provide the following design recommendations:

Free those hands. Use direct free-hand interaction to improve naturalness, intuitiveness, and interactivity of the AR interface.

Zoom the world. Allow zooming to scale the virtual scene while the hand's size remains constant relative to the real world.

6.2.2 Multimodal Interaction in AR

The G-Speech interaction technique combined gesture and speech input to offer a high level of usability (with 6 points for scaling), an above average level of usability (with above 4.5 for relocation tasks) and good user feedback, as described in, *Easy and intuitive to resize* and *Better control and thus more precision*. Indirect manipulation offers remote interaction where the hands are not required to interact directly with the object. Removing the hands from the scene can be beneficial where the interaction space may be limited.

We found that G-Speech is effective at both single and multiple object manipulation, with no limit to the number of objects being manipulated at the same time. However, separating the interaction from the scene and from the object also has drawbacks, and we found that inexperienced participants were not be able to predict the result of the indirect action as described in *Difficult imagining an indirect rotation*. We suggest offering a surrogate object close to the user but away from the scene that the user can interact directly with. Furthermore, certain participants had difficulty with certain commands, and a range of commands should be offered. Our design recommendations are:

Redundancy. Use several speech commands for the same action for better usability

Telekinesis. Use indirect manipulation to remotely interact with distant objects or many objects at the same time.

Surrogate. Support a direct interaction on a distant object using a surrogate object that is close to the user. Any action applied to the surrogate is applied to the remote object being manipulated.

6.2.3 Natural Interaction for AR

Although the majority of participants liked the 6 *dof* movement offered by G-Shell, in certain cases, more control was desired, as mentioned by a participant in *Better control and thus more*

precision. For this reason, we recommend that the interface offer choices for simplified actions as well as combined actions.

The results showed that G-Shell and G-Speech both had strengths and weaknesses. For natural interaction, it is challenging to design an interface suitable for every user and task, as described in *To speak or not to speak*. Therefore we propose that to enhance usability and user experience, offer complementary interactions within a single interface so that the user has a choice. Our design recommendations are:

Divide and conquer. Provide both combined and divided actions such as translation, rotation, and both for better control.

More is greater than one. Offer a choice of interaction techniques that complement each other.

6.3 Limitations

Our study has a number of limitations that we hope to address in future work. First, there are limits in the tracking resolution and speech and gesture recognition accuracy of the 3Gear and Intel Perceptual Computer SDKs. We attempted to address these problems during the practice session by calibrating the hand tracker for each user and providing commands that worked well for each subject. During the experiment, we observed that only a small number of errors, typically below 5%, were encountered in a session regardless of interaction technique, reflecting the high usability rating for both interaction techniques.

A further limitation was that we only compared two tasks, moving and scaling. However, as we explained in Section 5.1, both of these involved several fundamental actions (e.g. grasping, pointing, etc.), that are required in other, more complicated interactions.

7 Conclusion and Future Works

In this work we present a new AR interaction framework, G-SIAR, and two new interaction techniques, G-Shell and G-Speech. Using G-Shell, we demonstrated that a direct free-hand interaction technique can be natural, intuitive, interactive and precise. With G-Speech, we showed that ease of use and control is achievable for interactions without direct contact. In an empirical user study, we found that G-Shell was better for object relocation while G-Speech was easier to use for scaling. Therefore, we recommend that both interaction techniques can be combined in a single AR interface to improve usability and enhance user experience.

In the future, we hope to develop both techniques to support more interactions. We will explore hybrid interaction techniques that combine the advantages of both direct and indirect manipulation and evaluate them with experienced and novice users to gain an insight on the best way to transition novice into experts over time.

References

[1] R. Azuma, "A Survey of Augmented Reality," *Presence,* vol. 6, pp. 355-385, 1997.

[2] M. Lee, M. Billinghurst, W. Baek *et al.*, "A usability study of multimodal input in an augmented reality environment," *Virtual Reality,* vol. 17, no. 4, pp. 293-305, 2013/11/01, 2013.

[3] O. Hilliges, D. Kim, S. Izadi *et al.*, "HoloDesk: direct 3d interactions with a situated see-through display," in Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems, Austin, Texas, USA, 2012, pp. 2421-2430.

[4] H. Benko, R. Jota, and A. Wilson, "MirageTable: freehand interaction on a projected augmented reality tabletop," in Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems, Austin, Texas, USA, 2012, pp. 199-208.

[5] T. Piumsomboon, A. Clark, M. Billinghurst *et al.*, "User-Defined Gestures for Augmented Reality," *Human-Computer Interaction – INTERACT 2013*, Lecture Notes in Computer Science P. Kotzé, G. Marsden, G. Lindgaard *et al.*, eds., pp. 282-299: Springer Berlin Heidelberg, 2013.

[6] T. Lee, and T. Hollerer, "Handy AR: Markerless inspection of augmented reality objects using fingertip tracking," *Proceedings - International Symposium on Wearable Computers, ISWC.* pp. 83-90.

[7] R. Wang, S. Paris, J. Popovi *et al.*, "6D hands: markerless hand-tracking for computer aided design," in Proceedings of the 24th annual ACM symposium on User interface software and technology, Santa Barbara, California, USA, 2011, pp. 549-558.

[8] A. D. Wilson, "Simulating grasping behavior on an imaging interactive surface," in Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces, Banff, Alberta, Canada, 2009, pp. 125-132.

[9] O. Hilliges, S. Izadi, A. D. Wilson *et al.*, "Interactions in the air: adding further depth to interactive tabletops," in Proceedings of the 22nd annual ACM symposium on User interface software and technology, Victoria, BC, Canada, 2009, pp. 139-148.

[10] E. Kaiser, A. Olwal, D. McGee *et al.*, "Mutual disambiguation of 3D multimodal interaction in augmented and virtual reality," *ICMI'03: Fifth International Conference on Multimodal Interfaces.* pp. 12-19.

[11] G. Heidemann, I. Bax, and H. Bekel, "Multimodal interaction in an augmented reality scenario," *ICMI'04 - Sixth International Conference on Multimodal Interfaces.* pp. 53-60.

[12] M. Kolsch, R. Bane, T. Hollerer *et al.*, "Multimodal interaction with a wearable augmented reality system," *IEEE Computer Graphics and Applications,* vol. 26, no. Compendex, pp. 62-71, 2006.

[13] S. Irawati, S. Green, M. Billinghurst *et al.*, "An evaluation of an augmented reality multimodal interface using speech and paddle gestures," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).* pp. 272-283.

[14] M. Lee, and M. Billinghurst, "A Wizard of Oz study for an AR multimodal interface," in Proceedings of the 10th international conference on Multimodal interfaces, Chania, Crete, Greece, 2008, pp. 249-256.

[15] J. O. Wobbrock, M. R. Morris, and A. D. Wilson, "User-defined gestures for surface computing," in Proceedings of the 27th international conference on Human factors in computing systems, Boston, MA, USA, 2009, pp. 1083-1092.

[16] P. Milgram, and F. Kishino, "Taxonomy of mixed reality visual displays," *IEICE Transactions on Information and Systems,* vol. E77-D, no. Compendex, pp. 1321-1329, 1994.