# AI for Toggling the Linearity of Interactions in AR

Jing Qian
*Brown University*
Providence, RI, USA
jing_qian@brown.edu

Laurant Denoue
*FXPAL*
Palo Alto, CA, USA
denoue@fxpal.com

Jacob Biehl
*FXPAL*
Palo Alto, CA, USA
biehl@fxpal.com

David A. Shamma
*FXPAL*
Palo Alto, CA, USA
aymans@acm.org

***Abstract*—Interaction in Augmented Reality or Mixed Reality environments is generally classified into two modalities: linear (relative to object) or non-linear (relative to camera). Switching between these modes can be arduous in cases where someone's interaction with the device is limited or restricted as is often the case in medical or industrial applications where one's hands might be sterile or soiled. To solve this, we present Sound-to-Experience where the modality can be effectively toggled by a noise or sound which is detected using a modern Artificial Intelligence deep-network classifier.**

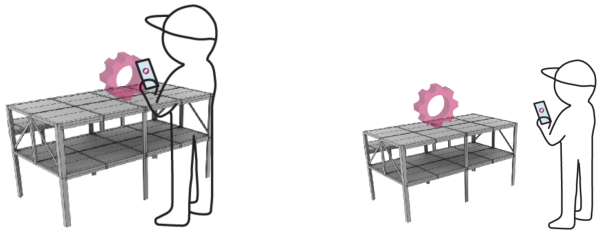
***Index Terms*—AR, AI, Augmented Reality, Artificial Intelligence, Mixed Reality, Interaction, Modality.**

## I. Introduction

In Augmented Reality (AR) and Virtual Reality (VR) environments, there are two main interaction types: Linear and Non-linear. The first type requires the user to approach the virtual objects and interact within a distance reachable with our arm (e.g., similar to how we interact with real-life objects). The second type does not require close-by interactions and allows the user to interact at any distance as long as the virtual objects are in the line of sight. Linear interactions provides an opportunity for a one-to-one scale direct manipulation of virtual objects with our hands or controllers. In a way it is an intimate and expressive communication between the user and the virtual objects, yet its limited interaction space requires users put extra effort from physically approaching every interaction scenarios. The non-linear interaction module, on the other hand, frees us from the distance limitations but loses ability to directly manipulate the virtual objects. Futhermore, gauging physical depth from a smartphone screen is difficult for Linear AR interactions. This is due to lack of stereo display on a smartphone and the perspective differences between the user's eyes and the smartphone. The depth issue is less sever in Non-linear interactions, where depth cue is omitted due to unlimited interaction distance available in this mode.

When an AR system runs on a smartphone, users can benefit from switching the linearity of the interactions. With the added mobility of a smartphone, switching between Linear and Non-linear modes provides extra freedom and efficiency for the user's preferences in AR interaction. Some users can perform better when they are close to the AR objects while others prefer a remote control.

Switching linearity becomes a challenge when the touch interface is restricted on a smartphone (e.g., factory workers with non-conductive gloves or medical staff in a sterile environment).Existing alternatives are hand tracking, smartphone position and orientation sensing, and eye tracking. If the primary alternative method is hand tracking [1], switching linearity can be compromised by the tracking quality or accidental triggering from unintended hand movements. The eye tracking methods suffer from the similar effects of unwanted eye movements and in turn result in disrupted AR experience.



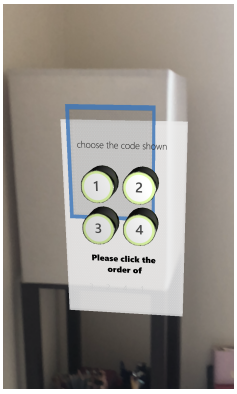
(a) Linear. (b) Non-linear.

Fig. 1 Examples of Linear (a) and Non-linear (b) interactions in AR.

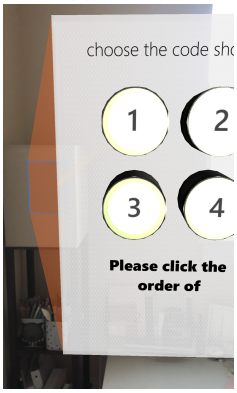
Using voice to switch linearity on smartphone AR does not possess the issues from other hands-free methods described above. Voice has been used as auxiliary input in the early stage of AR and VR development [2] and is usually used as a set of pre-defined commands [3], [4] that execute tasks or operations on the device. These commands often comprised of special words that are hard to be triggered by accident. Modern smartphones are optimized for running deep Artificial Intelligence classifiers (such as sound or voice recognition) with low latency response time, therefore enabling a wide-range of voice interface possibilities.

## II. Sound-to-Experience

In this demonstration, we present *Sound-to-Experience* as a method of toggling the Linearity of AR interactions. A specific sound or voice is used for both input and output to provide a seamless interaction experiences in touch-less AR interfaces. The input voice toggles between the current linearity of the interaction with the alternative ones, and the output voice confirms the action. The voice interaction mimics our daily experiences such as snapping the finger. (i.e., when a person snaps a finger, it makes a loud sound for attention) We used two sounds, *beep* and *boop*, combination to obtain the attention




(a) Linear.




(b) Non-linear.

Fig. 2 An interaction with a combination safe. Here a linear interaction (a) requires a user to be close to the lockbox while a non-linear interaction (b) can be remotely operated. An orange line is anchored to the physical object connect the interface to the object it controls.

from the AR system, and result in switching between the linear and non-linear interactions.

We integrated the Apple's ARKIT 1.5 in Unity3D and iOS Native CoreML to create this real time demo. ARKIT is a platform released by Apple to support mobile AR applications. It has ability to perform real-time localization, feature extraction, and vertical and horizontal plane registration. Unity3D is a game development environment that facilitates rendering and programming of 3D objects, and it has a native integration with ARKIT 1.5. Finally, We add IOS's coreML for its robust detection and data classification in real-time.

The integration between the Unity3D and CoreML is done through creating a shared object written in Objective-C. This object is created from XCode 9, and served as an external library for the Unity3D but native to CoreML. With this shared object, Unity3D can call and receive callbacks from CoreML in real time. During the run-time, auditory signals will be handled by the script using CoreML to filter the special sounds, ( *beep* and *boop*) and invoke the method in shared object to signal Unity3D toggling the correct linearity mode.

This demonstration displays AR interactions in both Linear and Non-linear modes. In Linear interaction mode, virtual objects take in the transformation matrix from the close-by anchor planes. These objects are registered in the 3D space and creates a one-to-one movement relationship to the smartphone and the hand (when using hand tracking). See Figure 2. The Non-linear interaction mode is created by applying only the position matrix from the AR camera in ARKIT to the virtual objects, therefore the distance between the virtual objects and the virtual camera stays the same yet the camera is still able to move out of the Virtual objects.

We set the initial orientation (i.e., rotation) of the virutal objects in-line with a target anchor plane. The system does not allow for triggering the Non-linear mode if the target anchor plane is not within the line of sight. Meanwhile, a virtual line is drew between the center of Non-linear AR objects to the Linear counter part (see Figure 2b). This way we visually enhence the connection that this Non-linear AR object is associated with a specific real-life object. When the toggling signal is received from the CoreML, we swap the global position of the virtual objects based on the current Linearity.

## III. Future Work and Other Embodiments

In controlled environments, tones or sounds (even background music) can be used as selectors to toggle the enabled state or interaction linearity. Tones or music in various rooms can be used to hint the interaction type of all the devices in a room. Additionally, the system can be expanded to use computer vision to toggle the interaction linearity. For example, certain devices (like a lockbox or safe) could be visually recognized and disable non-linear interactions, thus requiring an operator to be physically next to the object for security reasons.

## References

[1] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 63:1–63:8, Jul. 2009. [Online]. Available: http://doi.acm.org/10.1145/1531326.1531369

[2] M. R. Mine, "Virtual environment interaction techniques," *UNC Chapel Hill CS Dept*, 1995.

[3] B. A. Delail, L. Weruaga, and M. J. Zemerly, "Caviar: Context aware visual indoor augmented reality for a university campus," in *Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology-Volume 03*. IEEE Computer Society, 2012, pp. 286–290.

[4] F. Roesner, T. Kohno, and D. Molnar, "Security and privacy for augmented reality systems," *Communications of the ACM*, vol. 57, no. 4, pp. 88–96, 2014.