AR Chicken Invaders

Developers:

Gal Katz

Tamir Daniel

Ari Amitai

Supervisors:

Yaron Honen

Boaz Sternfeld

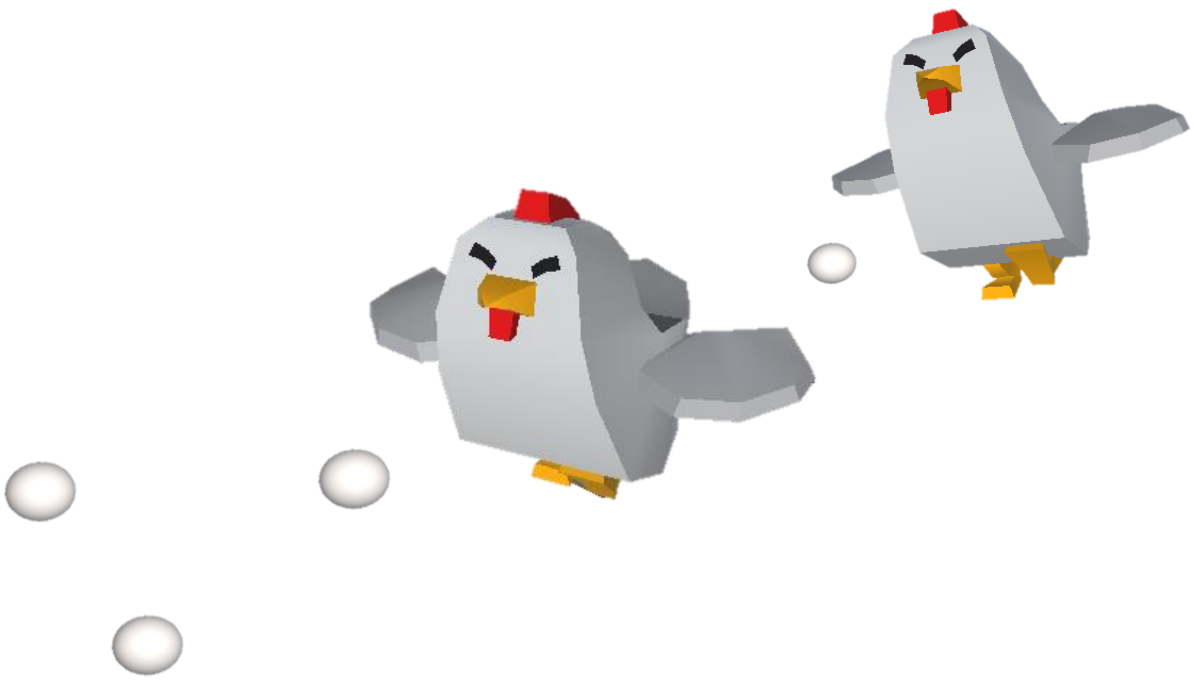


Table of Contents:

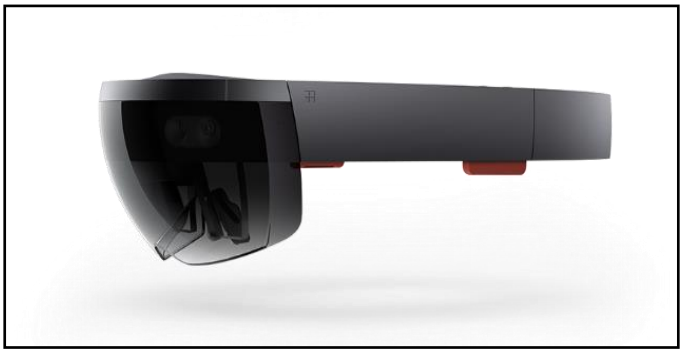
Introduction:

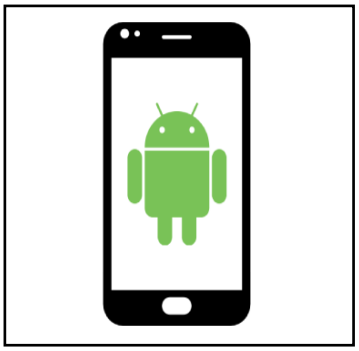
Our project is inspired by our childhood game, Chicken Invaders. Chicken Invaders is a series of shoot 'em up video games created by Greek indie developer Konstantinos Prouskas and released in 1999. The original game features chickens that emerge out of portals in space, and the user's objective is to zap shoot them all. In our game, instead of watching the battle field on a computer monitor, we let the user take part in the action, and after scanning the user's surrounding, chicken-launching portals appear in the room and start launching real life sized chickens. The chickens throw eggs at the user, like in the original game, but this time the eggs are 3d rendered eggs that move in 3d space. The user zaps the chickens with our special laser weapon and saves his house from the invading chickens.

The application is built as a AR game in unity and runs on Microsoft's HoloLens 1 headset, and on android devices (a special mobile phone friendly version of it).

Equipment:

The physical equipment we used consists of Microsoft's HoloLens 1 headset and an Android 11+ device.





Technology Used:

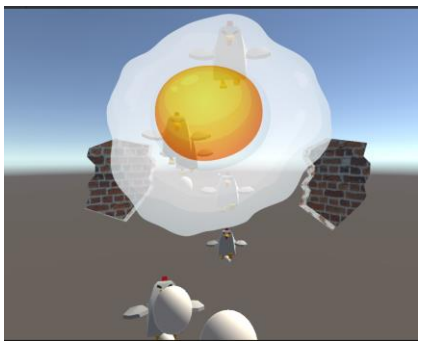
- Unity- Unity is a cross-platform game engine developed by Unity Technologies. It is particularly popular for iOS and Android mobile game development and is considered easy to use for beginner developers. The engine can be used to create three-dimensional (3D) and two-dimensional (2D) games, as well as interactive simulations and other experiences.

- Unity AR Foundation - Unity's AR Foundation is a cross-platform framework that allows you to write augmented reality experiences once, then build for either Android or iOS devices without making any additional changes.

- Microsoft Mixed Reality ToolKit - MRTK-Unity is a Microsoft-driven project that provides a set of components and features, used to accelerate cross-platform MR app development in Unity. It provides the cross-platform input system and building blocks for spatial interactions and UI, enables rapid prototyping via in-editor simulation that allows you to see changes immediately and operates as an extensible framework that provides developers the ability to swap out core components.

Game description:

The game is a VR FPS game against enemy chickens who emerge from holes in the walls around the player and drop eggs as projectile weapons. The player uses a hand gesture or screen tap (depending on the platform) to shoot a laser at the chickens. When chickens are defeated, they drop drumsticks or roasted chicken, which stays on the environment (floor, table). When there are too many chickens who shoot too many eggs, an egg will splash on the user's screen (or POV), preventing their sight for a short period of time.



- **Game proccess**

i. Enviroment scanning: As the game begins, a scanner recognizes elements in its surroundings.

ii. Wall detection: Upon finishing the scan, our algorithm filters the objects and detects vertical walls in the scanned environment.

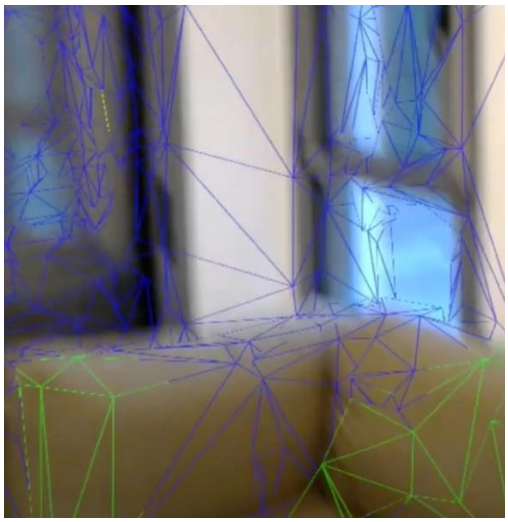
iii. Portal Placement: As soon as the walls have been detected, our algorithm determines which walls are best suited for portal placement.

iv. Game: After all preparations are made, the game begins and is managed by our Game Manager.

- **Key challenges**

Environment scanning and wall detection:

We encountered many problems with environment scanning and wall detection during the development of the project. For example, tables, walls, and floors were not recognized. Our problems were due to incompatibility between the code of Mixed Realty ToolKit (MRTK) that was designed for Microsoft's HoloLens 2 hardware and our HoloLens 1. Our solution involved refactoring some code to fit HoloLens 1 hardware (of course, many HoloLens 2 features wouldn't work on HoloLens 1, but it satisfied our goal of recognizing walls in our environment)



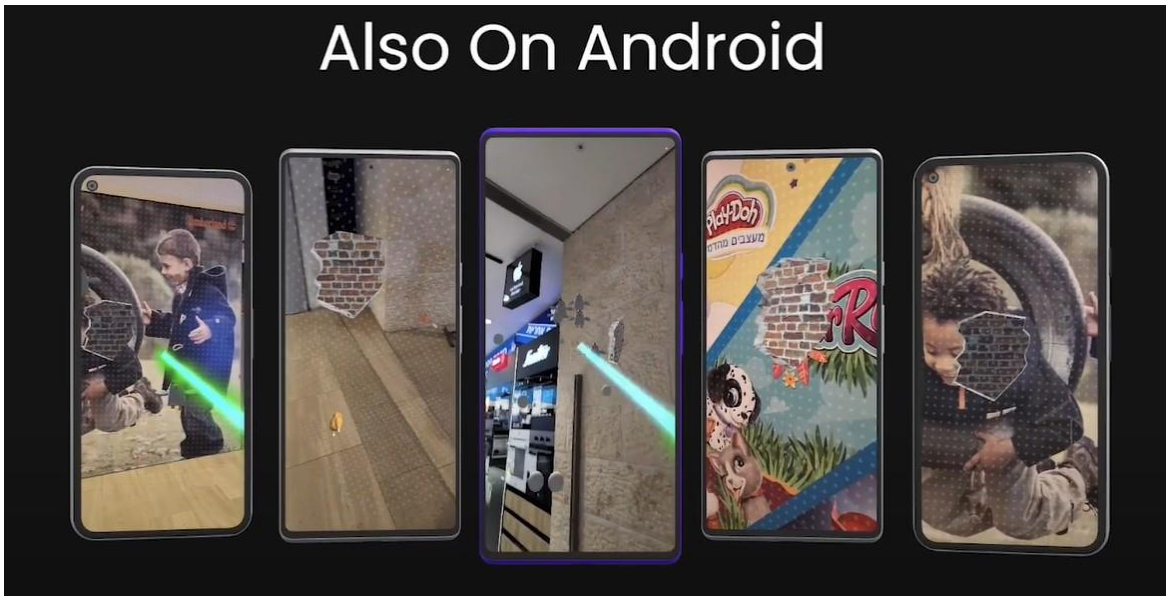
Preformance:

At the time of its launch in March 2016, the HoloLens 1's CPU, GPU, and memory were of high quality, but today they are considered low-end technology. While developing the game, we encountered a few performance problems caused by graphic elements, including high GPU and CPU consumption, low rendering, etc.

We resolved our rendering and performance issues by using low polygon assets. Additionally, we reduced the number of objects displayed on the screen by "killing" objects after some time (for example, removing eggs and baked chickens) or stopping chicken generation once there are a certain number of chickens. In addition, we stopped scanning once our requirements were met, which decreased our CPU consumption.

Android Development:

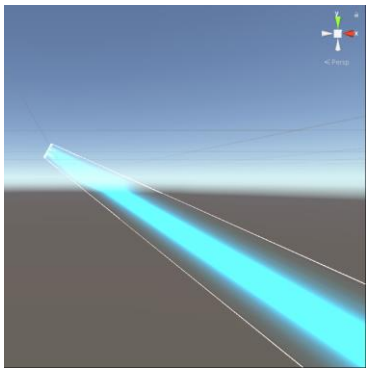
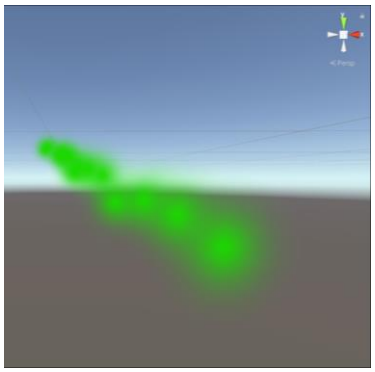
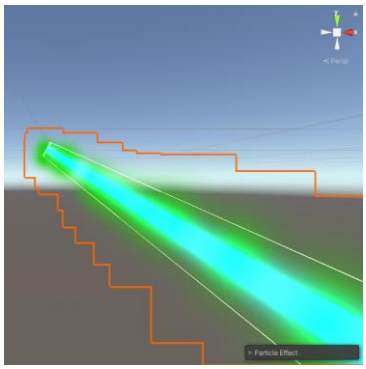
When porting to Android Platform, we encountered difficulties detecting walls and objects. These issues were caused by the differences between Unity AR Foundation and MRTK. As well as providing an infrastructure, the MRTK also creates a 3D environment mesh utilizing the sensors and cameras of the HoloLens. However, when using Unity AR Foundation, the infrastructure uses only one camera (the main camera) and does not make use of other sensors and cameras (such as depth sensor) even if they are present in the device, and Unity AR Foundation cannot detect plain surfaces (this is a known issue). Due to these limitations, a picture frame, TV, sofa, or non-plain wall will sometimes be selected as a portal location by our algorithm.



- **Visual AR effects**

  Shooting:

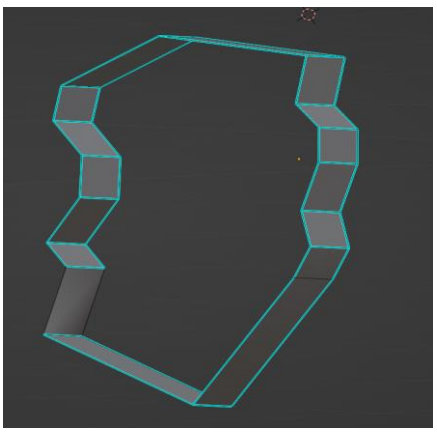
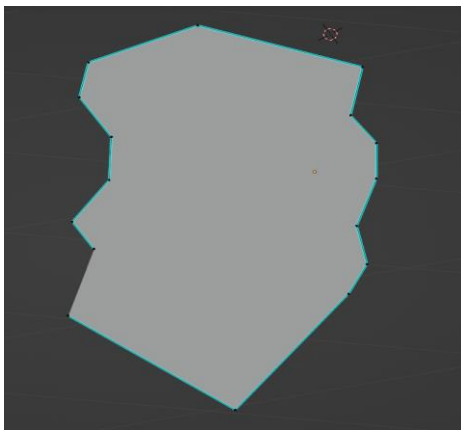
  Shooting down chickens is a major part of the game. To create the laser effect we used two parts – a static image and particles system. Both of them are visible for 0.26 seconds, and together with the laser sound create the effect. In order to keep it suitable for low resource systems, we used only 14 particles simultaneously.

  xIn addition, we used two different shooting methods in our project. On Hololens, we only used Raycast to tell if the laser hit a chicken since the field of view is relatively small and the white dot is used as a sight. On Android, we combined the Raycast with a box collider that wraps the laser beam, which enables 'slicing' chicken mid-air, while they fly. We decided not to use the box collider on Hololens as we noticed a slight performance decrease and didn't see any improvement in the player experience.

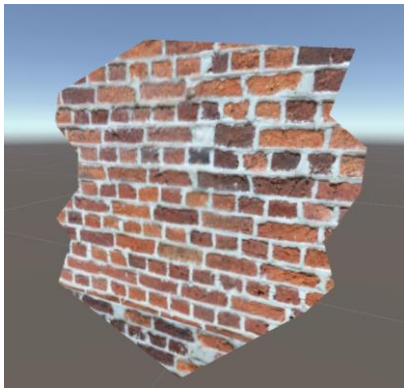
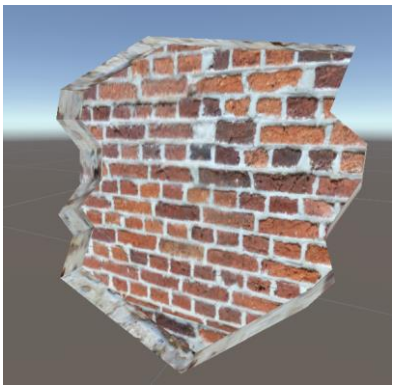
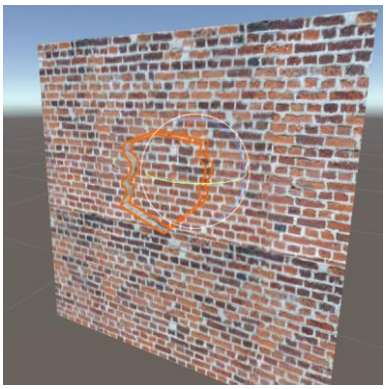
  Portals:

  Our project features portals, which are 'holes' we add to the walls around the player. One of the visual effects we wanted to implement was creating visual depth to the holes, visible only when looking into the hole. The wall 3D model consists of two parts – the boundaries (on the left) and the flat hole shape:

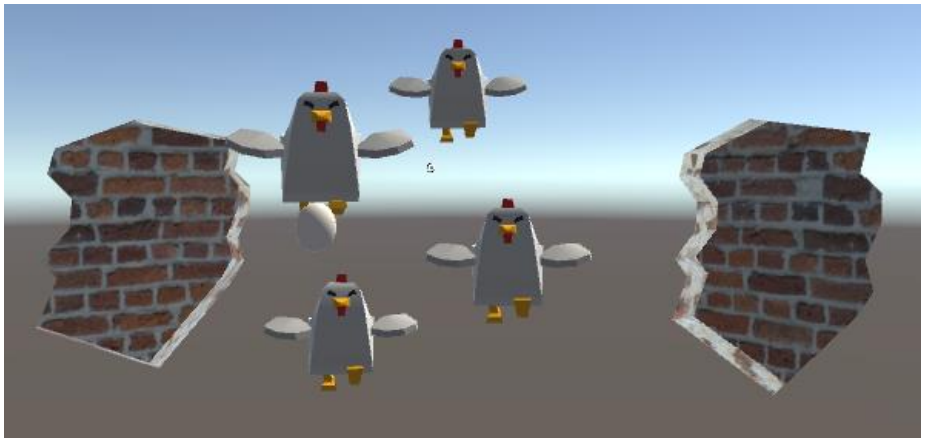
To create the effect, we used MRTK shader to create a stencil effect which makes the object invisible when looking from angles bigger than 90 or smaller than -90 from the flat shape normal. We added concrete texture to the boundaries to make it look like an actual hole in the wall. To create depth, we added a brick wall picture behind the flat shape and using the MRTK shader stencil options made the flat shape render the brick wall, which is always behind it and simulates depth. This way we add only 3 objects to our scene but gain a real depth effect.

To the left – the flat hole shape renders the bricks image behind it. In the center– concrete boundaries visible. To the right – entire bricks image visible with a hole shape on its enter.

Chickens:

The chickens int the game low-poly objects.
In order to make the game more challenging, the chickens move in a realistic essance, we used wing-animation and a semi-random sinusoidic movement, while keeping the chicken's face towards the camera (player's POV).



Closing thoughts:

We had fun working on this project and we learned a lot. We had no previous experience with these technologies and we enjoyed learning them as well. We belive AR development has huge potential, and that it’s an important skill to learn.