SHADOW GAMES

Bringing *Yu-Gi-Oh!* cards game to life using AR

Students: Or-Eli Pilosof, Yael Tsafrir, Rafi Cohen

Supervisors: Boaz Sterenfeld, Yaron Honen

Table of Contents

Introduction

As part of our BSc in Computer Science at the Technion, we heard about the AR experience with HoloLens and wished to try it ourselves.
When we began exploring the possibilities of the HoloLens, we recalled one of our favorite childhood TV series: *Yu-Gi-Oh!*[1]
In *Yu-Gi-Oh!*, the characters duel each other with monsters, and the monsters are summoned using special cards.
At some point in the series, the monsters can be summoned as full-size holograms, as if they were real.
There is also a trading cards game of *Yu-Gi-Oh!*[2] with real cards.

We created our own version of the *Yu-Gi-Oh!* game:

- The player uses a deck of cards from the original card game.
- Playing against a computerized opponent.
- We use the real cards to summon monsters in the AR game, creating the feeling that the game is alive, just like in the TV series.
- The monsters are interactable and the player may choose which action each monster will perform.

To make the AR experience more alive and real, we added shadows to the monsters in the game:

- The player can control the direction of the shadows to optimize his experience.

[1] Yu-Gi-Oh! Manga created by @Kazuki Takahashi. TV series by @NAS - TV Tokyo
[2] Yu-Gi-Oh! Trading Card Game – By Konami.

Development Tools

Unity:

A cross-platform game engine that can be used to create both 2D and 3D games, as well as simulations for computers, game consoles (e.g. Xbox), mobile applications and smart TVs.
Our scripts in Unity were written with C# and edited in Visual Studio.

Vuforia:

An AR software development kit (SDK) by PTC, that enables the creation of augmented reality applications. It uses computer vision technology to recognize and track planar images and 3D objects in real time.

HoloToolkit:

A collection of scripts and components intended to accelerate development of holographic applications targeting Windows Holographic.
HoloToolkit contains several feature areas, of which we used the Input features, which handles all user input in the HoloLens, and provides scripts and prefabs for AR buttons and interactable elements.

Required Equipment - Microsoft HoloLens:

A pair of mixed-reality smart-glasses developed and manufactured by Microsoft. HoloLens was the first head-mounted display running the Windows Mixed Reality platform under the Windows 10 computer operating system.

Development Process

When we began exploring the possibilities of the HoloLens, we recalled one of our favorite childhood TV series: *Yu-Gi-Oh!*
In *Yu-Gi-Oh!*, the characters duel each other with monsters, and the monsters are summoned using special cards.
At some point in the series, the monsters can be summoned as full-size holograms, as if they were real.

At the beginning, we didn't have yet a clear image of the game we will develop, but we knew what our first step would be: Finding a way to identify different cards with the HoloLens' camera, and project monsters based on these cards into the AR game. That is the main issue upon which our game is based.

We started searching the best way to achieve that, and we decided to use Vuforia SDK, as it enabled us to identify very complex and detailed images. Vuforia SDK provides a Unity object called "AR Camera", this object handles capturing and identification of images through the HoloLens camera. We created image targets from the pictures on the cards, each card is a unique image target. These image targets are then identified by the AR Camera and trigger a special event. We wrote our own event handler to handle what happens with each card that the AR Camera captures.
Using Vuforia SDK proved a bit of a challenge, because there were some program conflicts between Unity with Vuforia when running the game on the HoloLens. We eventually, through trial and error, found a version of Unity which fitted all our needs, including the HoloToolkit that we added later.

Our project supervisors advised us to add shadows to the game, and give the player the option of controlling the shadows' direction, so they will fit the surroundings and provide a more real experience, so our second step was handling the shadows inside the game.
We researched how we can control the direction of the shadows. In the Unity scene we have a directional light source. We found that when we change the X-axis and Y-axis rotation of the directional light – the shadows casted by it change accordingly.
We created a scene for shadows settings, containing a menu with 4 buttons (left, right, up, down), a plane and a character standing on top of it.
Pressing the buttons changes the rotation of the directional light, and the player can see the effect of each click on the shadows casted by the character on the plane.

Our next step was thinking about the game itself – the game flow, the design, which elements and actions should be included, etc.
Due to time limits and manpower, we couldn't recreate the entire cards game, so we decided which parts of the game are most important for us to have.
Once we had a clear scope of the game, we could move forward to build it.

We divided our game scope to smaller pieces on which we could work in parallel:

- Creating an interactable monster that the player can click on – This was done using a "Compound Button" script provided by the HoloToolkit, which handles any human interaction (targeting, pressing, etc.) with an AR object.
- Handling card capturing – Vuforia creates an AR object in the same location where it captured the image target. We wanted our monsters, that are summoned as AR objects by capturing cards' images, to be summoned at a specific location on our game board.

- A game manager component – we needed some component that will manage the entire gameplay – moving through the different game phases, handling the turns of the players, etc.
- A computerized opponent with basic AI – the player plays against the computer, but we wanted the game to feel real and challenging, so the computer doesn't repeat his actions, but rather follows some game strategy.
- A UI which will be both informative and easy-to-use for the player, to help navigating the game phases.

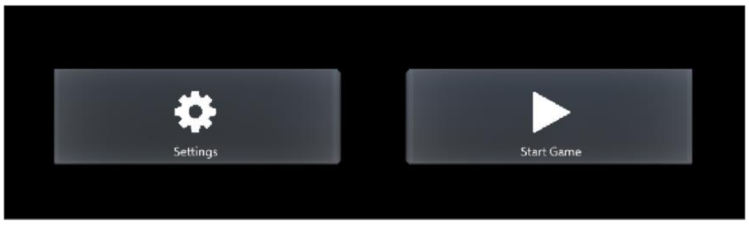
We relied quite a lot on the event system to manage the human player flow and his interactions with the monsters in the game, as any human input comes from outside our code and we cannot control the 'when' and 'what' here.

During the development we made sure to test the game flow by ourselves, as the game experience was very important for us. We tested all the cases and all the phases in the game to guarantee correct flow and behavior.
Near the end of the development, we also asked some friends to test our game and give us their feedback, to which we addressed and made necessary changes.

Application Overview

We will provide here an explanation of our application and how to use it:

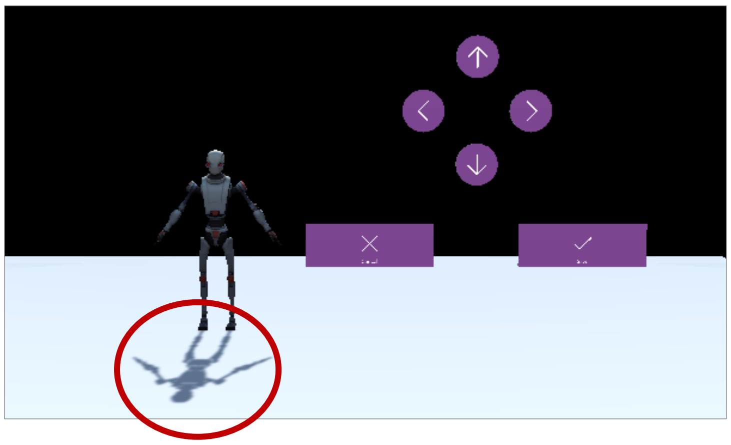
Opening screen:



- This a general opening menu, the player can choose either to change the settings of the game or start playing.

Settings:

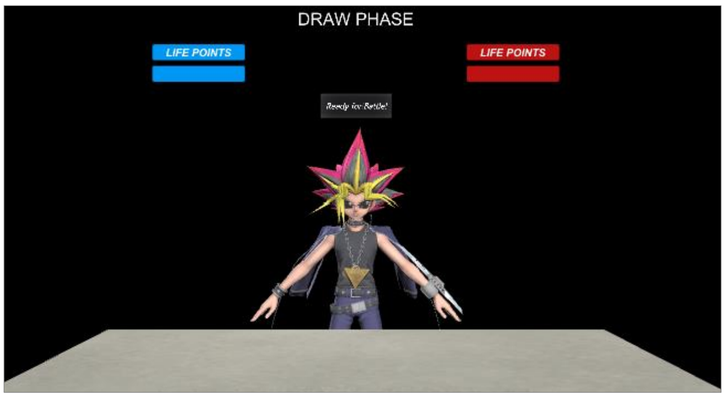
- Here the user can control the direction of the shadows in the game, as we mentioned before.



- Clicking on the arrow buttons will change the casted shadows. The player can immediately see the effect of his clicks, as the shadows casted by the robot character standing in the middle of the plane change accordingly.

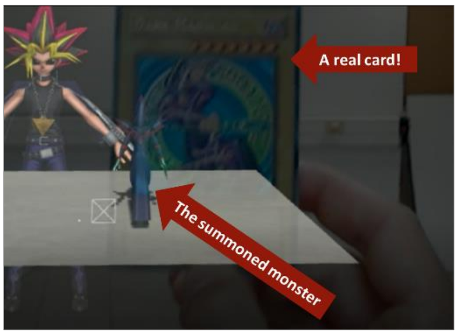
Game:

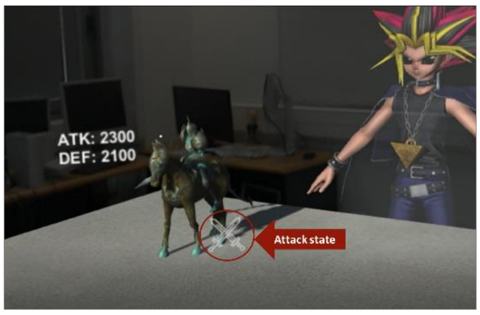
- This is the game scene.



- The model of Yugi represents the computer opponent, to give the player a feeling that he is playing against a 'real' opponent.
- There are two information boxes for life points in the top of the screen – the blue one belongs to the human player, and the red belongs to Yugi.
- In the center top of the screen, there is a title stating the current phase in the game. During the human player's turn, a button is present below the title, on which the user clicks to advance phases.

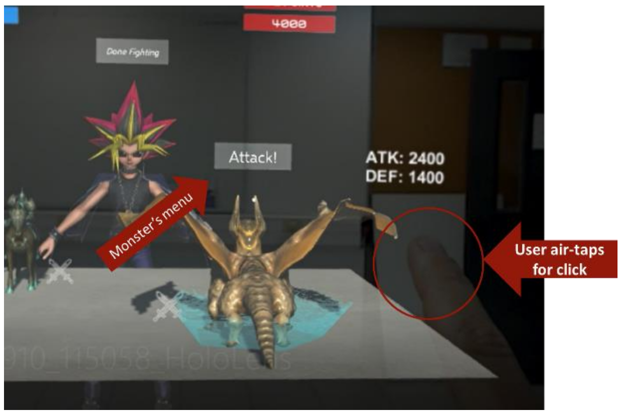
- The monsters are summoned on the game board, using the cards. To summon a monster – place the card in front of the HoloLens' camera.





- When player looks at a monster, he can see it's attack and defense strength data.
- Each monster has an icon next to it showing it's current state – swords for attack state, shield for defense state.

- When the player clicks on a monster – a menu of this monster will open, showing the actions that are available to do with the monster.
  The actions in the menu change in accordance with the phase of the game.



Game Flow

The game is played in turns. Each player has 4 phases in his turn:

1. **Draw Phase** – The player must draw a card from his deck and place in his hand. If the player doesn't have any more cards to draw – he loses and the game ends.
2. **Main Phase 1** – The player can summon a single monster using a card from his hand, if he has an empty location on the board (each player has 2 locations on the board).
   The player can also change the state of his monsters at this phase – from defense to attack state or vice versa.
3. **Battle Phase** – The player can now attack his opponent:
   - If the opponent doesn't have monsters at all – the attack reduces the opponent's life points directly.
   - If the opponents' monsters are in defense state – they may be destroyed but the opponent's life points are un-harmed.
   - If the opponent's monsters are in attack state – the attack points difference between the monsters may be reduced from the opponent's life points.
   - If a player reaches life points 0 or less – he loses and the game ends.
4. **Main Phase 2** – The player can change the state of his monsters again, but only if they haven't attacked during the battle phase.
   If the player hasn't summoned a monster during Main Phase 1, he can summon one now.

Looking Forward

Well, our project has ended, but it doesn't mean the end for this game.
We have had several thoughts and ideas throughout this project that unfortunately we didn't have the time nor the ability to do, but they can be used to enhance and develop the game even farther:

1. **Adding communication between 2 HoloLens devices, so 2 players can play against each other.**
2. Adding more card types to the game – we only used monster cards in our version of the game, but the original cards game includes different magic, trap, and other types of cards.
3. Automize monsters creation process – We edited manually each monster that we added to the game – setting up the menu to fit each monster, fitting their sizes, the collider sizes, etc.
   Perhaps this can be automized to be done using a script and not manually. This will enable adding many more cards to the game very quickly and easily.
4. Bring a different cards game to life – while some of the elements in our game were designed to fit the *Yu-Gi-Oh!* game, some are more general and the game's concept can be applied to several other card games.