Paper Toss AR

Manuel Kugler*
HCC Student

ABSTRACT

This paper describes the design and implementation of the mobile game "Paper Toss" in augmented reality. Paper Toss was originally a popular mobile-only game. Unity and Vuforia made it possible to take the game concept into augmented reality.

Index Terms: Human-Centerd-Computing, Augmented Reality, Unity, Vuforia, Mobile Game——
—

1 INTRODUCTION

Paper Toss AR is a mobile game developed with Unity that uses the Vuforia engine to provide an augmented reality experience. The goal of the game is to throw as many balls as possible into the trash can within a certain time. The trash can is located on a marker that is detected using the Vuforia engine. The game was developed for mobile devices and uses the camera for the augmented reality part and swipe and tap gestures to control the game. The developed game is strongly inspired by the existing mobile game "Paper Toss", which is developed by Backflip Studios for iOS and Android.

2 DESIGN

In order to create a good AR version of Paper Toss, it is crucial to examine all features and requirements of the game. This knowledge is used to create the design [2].

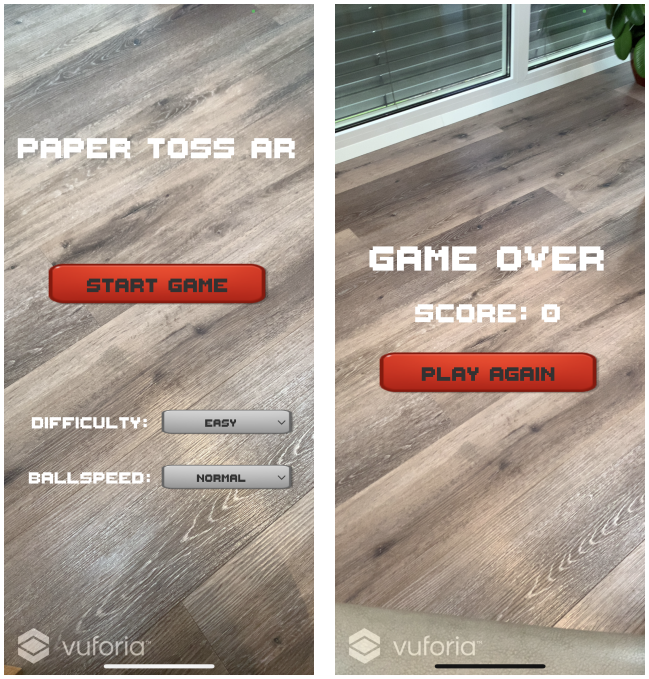
2.1 Requirements and Possible Features

- The Game is designed to run on mobile devices. So there should be a modern Android or IOS Smartphone available.
- The application itself should be interactive and in real-time. Virtual Buttons and user interfaces will be provided to increase the interactivity of the game.
- Game logic should not be neglected. There should be enough game modes etc. to provide players a good gaming experience. Approximately 650 LoC are needed for this purpose.
- The game ball is to be moved via touchscreen. Swiping or tapping the touch screen is to determine the force with which the ball is moved towards the trash bin.
- After all Paper Toss AR is a mobile game. Features like score, high-score and game sound would be pretty entertaining.

2.2 User Interaction

Paper Toss AR features a full user interface that lets you navigate through three different game screens. A virtual button has also been added to display the current score and the total high score. Start- and Game Over screens are displayed in Fig. 1a and Fig. 1b.

The start screen of the graphical user interface gives players the opportunity to choose between six difficulty levels and four level of ball speed. All buttons are used to switch among all game scenes. "Start Game" , for example, switches to the current game scene to the game scene. At the end of the game, the application switches to the game over screen. There the final score is displayed. Of course, the player also has the option to replay the game.



(a) Start Screen (b) Game Over Screen

Figure 1: The Start and Game Over Screen of the Game

2.3 Gameplay

The game starts at the start screen and waits for the player to choose their settings and press the "Start Game" button. When this happens, the application switches to the game screen shown in Fig. 2. The Player is now in-game and must throw the ball into the trash bin. The score and the total high score can be viewed by pressing and holding the virtual button. If the button is green, it's pressed. If it's red, it's not pressed. The game time should be 60 seconds. After these 60 seconds of game time, the game will switch to the game over screen. There you will see the final score and the "Play again" button. Pressing the button will take you back to the starting point.

2.4 Score and Gamesound

Each hit increases the current score in the game by five. At the end of the game, the score is compared to the high score. The high score is overwritten if the new score exceeds the high score.

Paper Toss AR is a mobile game. Each scene got its own sound to enhance the game experience. These sounds are to convey the vibe of a good mobile game. All game sounds were recorded

*e-mail: manuel.kugler@mailbox.org

Figure 2: Game Screen

and produced in a professional studio environment.

3 Implementation

Paper Toss AR was implemented with Unity 2019.2.2f1 and Vuforia. The following sections show the implementation of the previous discussed designs. All scripts were implemented in C#. The application uses two image targets. One for spawning the trash bin and another for the virtual button. Depending on the difficulty the trash bin can stay in place or move in different directions. Here is a brief overview of the functionality of all difficulties:

- Practice: Trash bin stays still at the image target.
- Easy: Trash bin slowly moves on horizontal axis.
- Normal: Trash bin slowly moves on horizontal and vertical axis
- Intermediate: Trash bin normally moves horizontal, vertical or in a circle
- Hard: Trash bin moves fast horizontal, vertical or in a circle. Furthermore the trash bin can now get smaller or bigger
- Godmode: Trash bin has random speed and random movement on horizontal and vertical axis.

3.1 Game Process

First, a scene manager script was created. This script is needed to switch the scenes under certain circumstances. For example when a button is pressed. Fig. 3 graphically shows the flow of the game and its scripts.

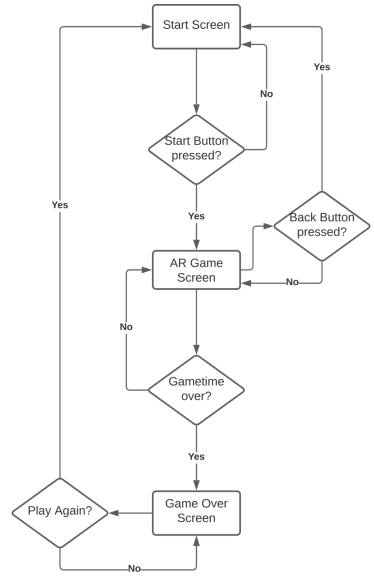


Figure 3: Game Process

3.2 Throwing the Ball

The ball in Paper Toss is thrown via touch screen. The function measures how long the user's finger touches the screen. The greater the touch, the less force the ball is thrown. So if you swipe the screen slowly, the ball won't be thrown as far as if you swipe the screen quickly. The function within the ball physics script looks like this [3]:

```
void throw_ball(){
    if (Input.touchCount > 0 && Input.GetTouch(0).phase
         == TouchPhase.Began)
    {
        //starts the timer when finger touches the screen
        timer_start = Time.time;
    }

    if (Input.touchCount > 0 && Input.GetTouch(0).phase
         == TouchPhase.Ended)
    {
        //ends the timer when finger no longer touches
             the screen
        timer_finish = Time.time;
        timer_interval = timer_finish - timer_start;
        rb.isKinematic = false;

        //calculating force in dependence of touch time
             interval
        rb.AddForce(0, throw_force_x_y * 2f,
             throw_force_z / timer_interval * 1f);
        rb.useGravity = true;
        //reset ball after two seconds
        Invoke("reset_ball", 2);
    }
}
void reset_ball()
{
    rb.isKinematic = true;
    rb.useGravity = false;
    transform.position = init_pos;
    rb.velocity = Vector3.zero;
    application.hit_detected = false;
}
```

Listing 1: Functions for throwing and reset the ball

The Force with which the ball moves can be adjusted by changing the ball speed at the start screen of the game.

3.3 Hit Detection

A game object was created inside the trash bin that detects if the ball hits its target. The applications notice when the ball object collides with the hit detection object. The hit detection object is shown purple in Fig. 4. The functionality for detecting a hit, increase the game

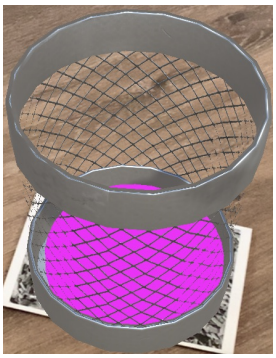


Figure 4: Hit Detector inside the Trash Bin

score and playing a hit sound is shown here [1, 3]:

```
//function will be called when the hit detection game
     object is colliding with something
void OnTriggerEnter(Collider other)
{
    application.hit_detected = false;
    //will only be true if the collision object was our
         game ball
    if(ball == other)
    {
        //play collison sound
        audioS.PlayOneShot(myAudio, 0.7F);
        //update in-game score
        score_update();
    }
}

void score_update()
{
    if(application.hit_detected)
    {
        Debug.Log("match detected!");
        return;
    }

    application.hit_detected = true;
    application.score += score_value;
}
```

Listing 2: hit detection functionality

4 Future Work

A lot could be done in this game. A few ideas could be:

- A good feature would be the creation of obstacles. Walls or other objects could be integrated to give players new challenges.
- Another idea is to use more than one image target for the trash bin. So two objects could be hit or a trash bin switches between image targets.
- Persistently save all scores and integrate a scoreboard to get a better overview of all high-scores in all difficulties.

5 Conclusion

Paper Toss AR is a mobile game developed with Unity that uses the Vuforia engine for an augmented reality experience. The game design was taken from the mobile game ”Paper Toss”, which was developed by Backflip Studios. The principle is quite simple: hit the trash bin as many times as possible in 60 seconds to score points. Paper Toss AR has several difficulty levels. After all, every user should find his challenge. Depending on the difficulty level, the trash bin moves fast/slow and changes direction after a hit. The game could be expanded with some more features in the future. New obstacles or a persistent local high score would be a possibility.

Acknowledgments

Special thanks to my friend Marco C. Platzl for the support, the production and the recording of these insane in-game sounds.

References

[1] I. PTC. Vuforia developer library, 2021.
[2] B. Studios. Paper toss, 2009.
[3] U. Technologies. Unity scripting reference, April 2020.