Claude Taliana

Creating an interactive application

Computer Interface Design Principles

Claude Taliana

14

Table of Contents

Task 1: *Which are the most important criteria when it comes to choosing a game engine?*

- **Cost –** One has to make sure the budget he has to build a game, as most of the game engines are expensive. Some costs that one has to keep in mind are training, integration time, support and maintenance fees, and add-ons.

- **Functionality –** The user has to check which engine systems are the most important before starting to build his game. Some of these are: multi-threading system, rendering pipeline, animation system, physics system, streaming system, and networking design.

- **Flexibility –** When choosing a game engine, the user has also to make sure what range of games he can produce with that specific engine. So, he must speak to people who have used this engine before. The game developer must know well the programming language that the chosen game engine works with.

- **Integration into current technology –** The user has to make sure that there will be an existing custom technology to be integrated with the game engine he builds his game with. Also, middleware technology is an important fact when it comes to integration into technology such as: Bink, FMOD, Havok and Wwise. Middleware technologies are often used to eliminate the problem of integration. Basically, it is a layer between two systems that makes it easy for both of them to communicate.

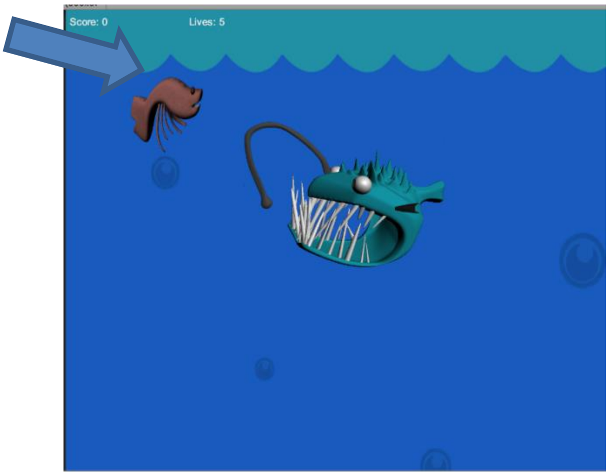
- **Tools and Content Pipeline –** Some of the most important engine tools are profiling system, live preview on target platform, standalone world builder, particle system editor, and scripting system (run-time script debugger). The Content Pipeline is designed to be extensible, so that the user can easily support new input file formats and new types of conversion.

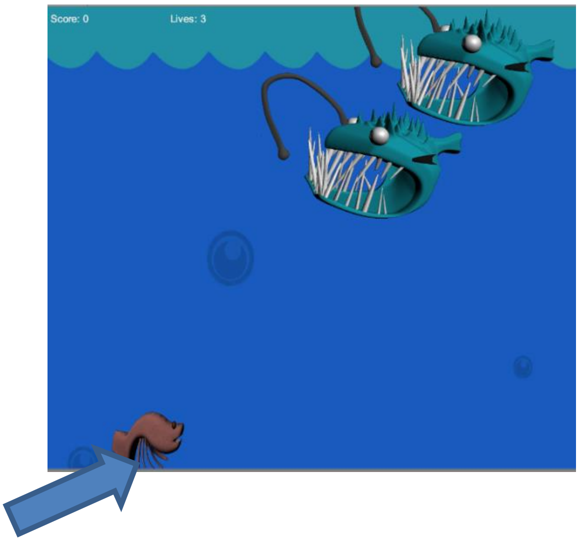
Task 2: *Evaluate constraints affecting interface design*

Code:

```
function Update () {

        transform.position.y =

        Camera.main.ScreenToWorldPoint(Input.mousePosition).y;


        //transform.Translate(Vector3.right * Time.deltaTime * Input.GetAxis("Horizontal"));

        if (transform.position.y > 2.8){

                transform.position.y = 2.8;

        }

        if (transform.position.y < -4.5){

                transform.position.y = -4.5;

        }

}
```

This code shows the position of the character and does not let it get out of the border of the sea and also does not let the character leave the bottom of the sea.





Task 3: *Implementation of sound effect*

This code represents the sound that the game produces when the up button is being pressed.

Code:

```
function Update () {

        if (Input.GetKeyUp("space")){

                if(!audio.isPlaying){

                        audio.Play();

                }



        }

}
```

Task 4: *Reference to the audio folder*

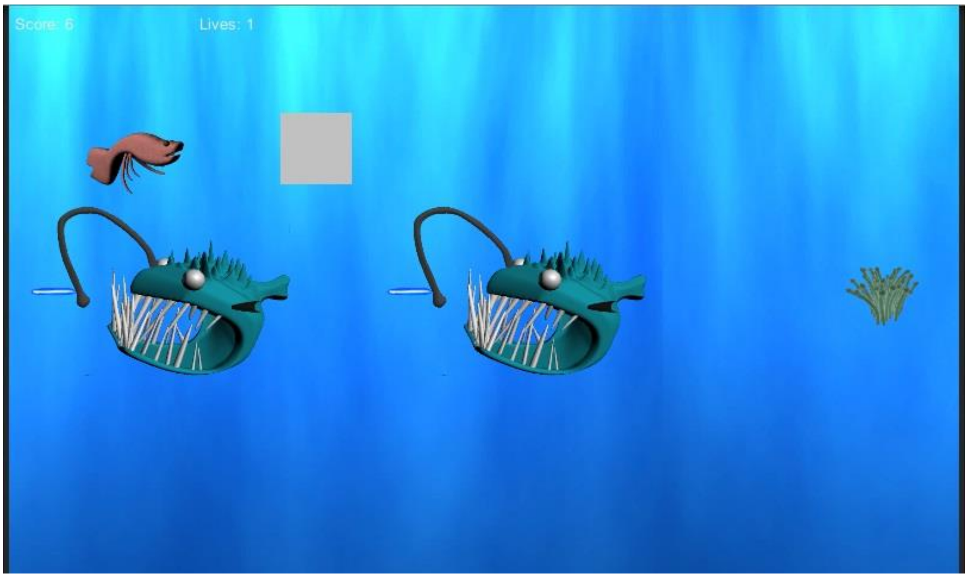
Claude_Taliana_2HND2 > Assets > Materials > Sound

Effect: deEsser & deNoiser in Adobe Premiere.

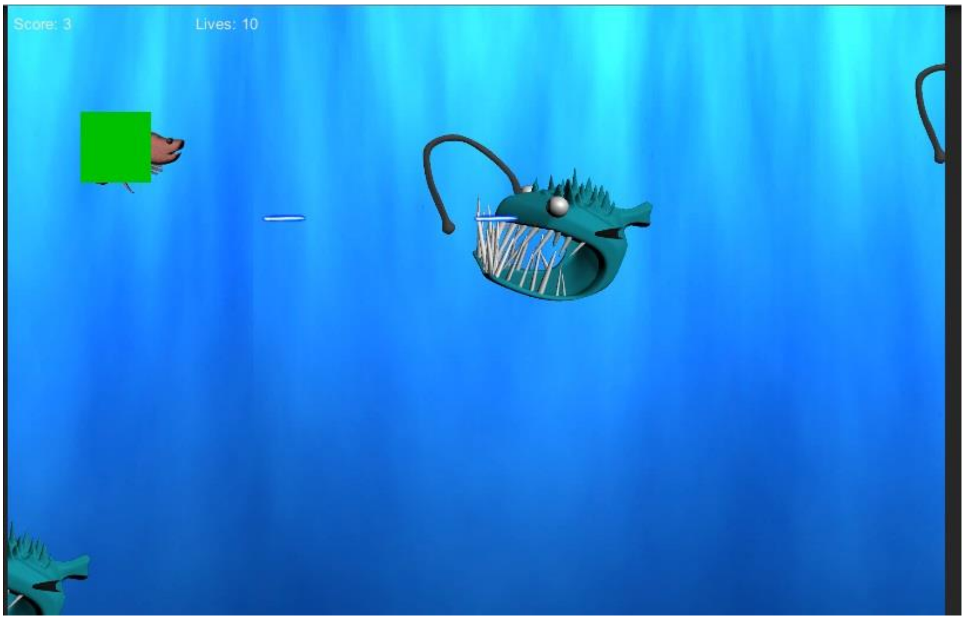
Task 5: OnTriggerEnter and OnTriggerExit functions

The code below shows that when the shrimp hits the box, the color of the box will change to green.

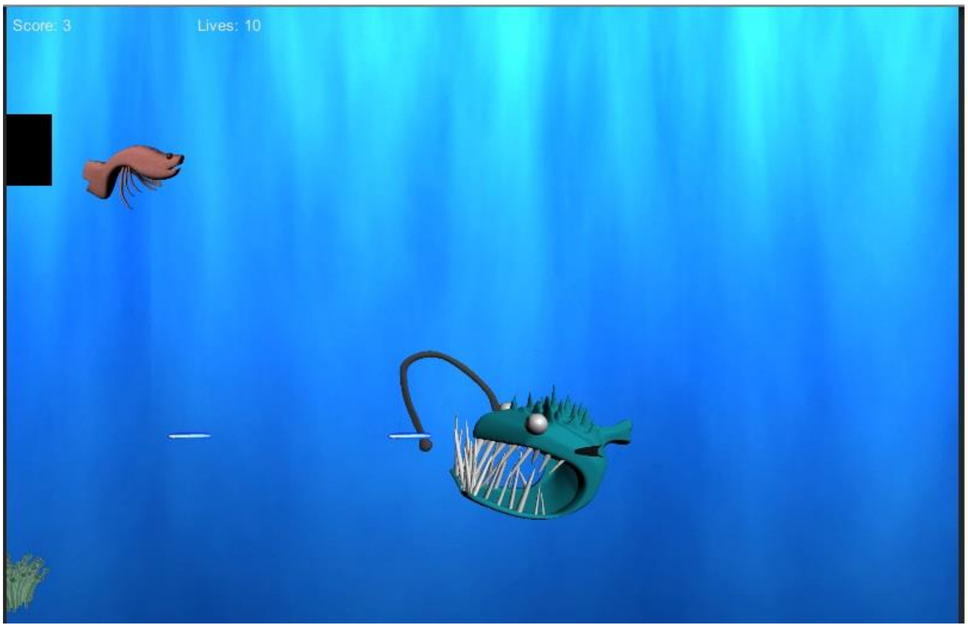
```
function OnTriggerEnter(other:Collider)
{
//if the player hits the square, the following will happen:
if(other.tag == "Player"){
//score=score+2;
PlayerCont3.score+=1;//Level3
//give green material
renderer.material.color = Color.green;
}

}
```



Before hitting the box



Hitting the box



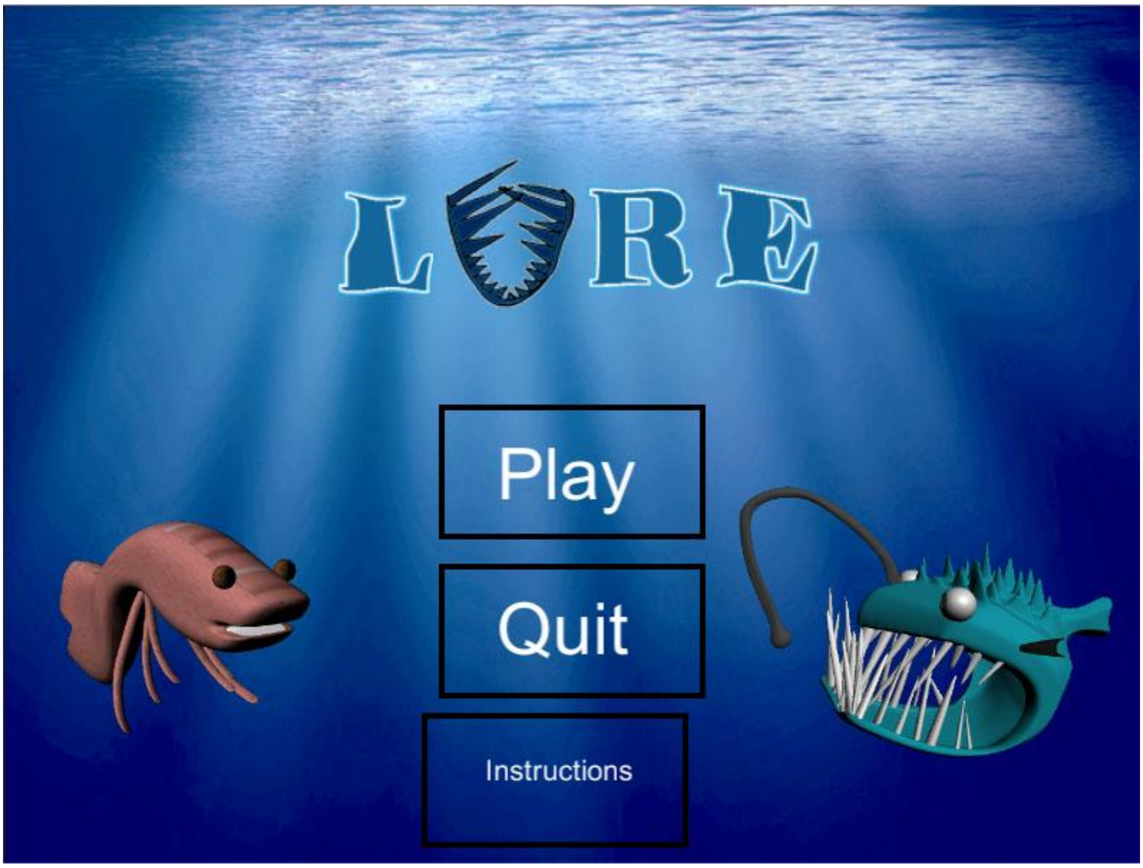
Leaving the box

The code below shows that when the shrimp is about to leave the box, it will change to black.

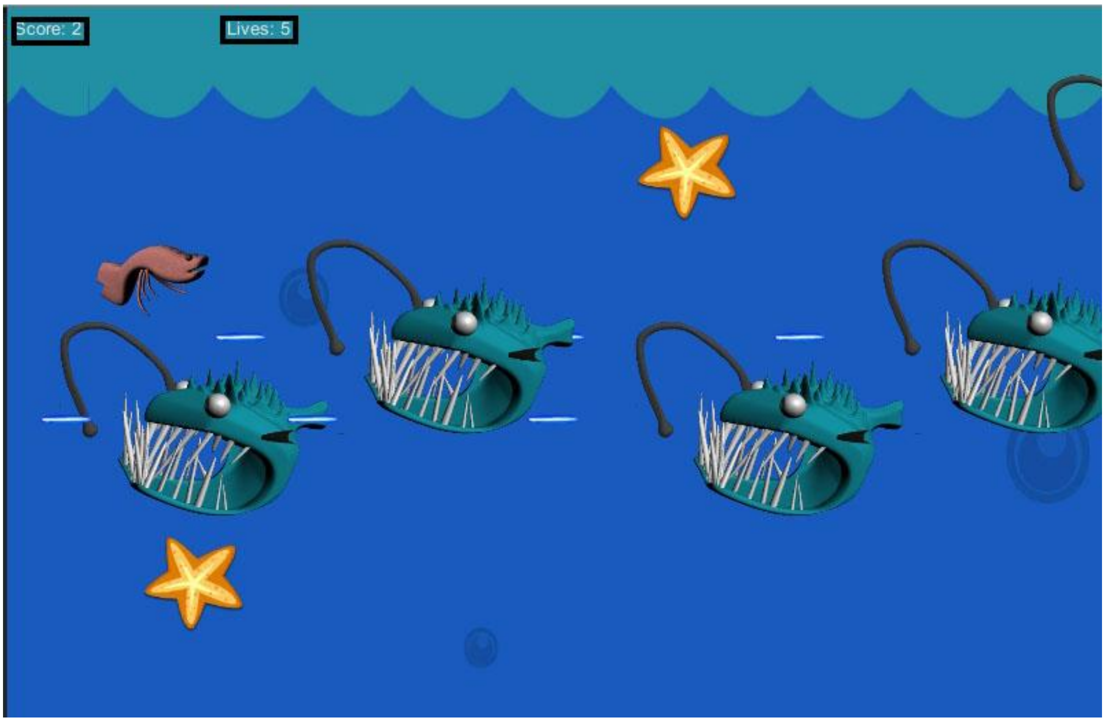
```
function OnTriggerExit(other:Collider)
{
if(other.tag == "Player")
{
//give black material
renderer.material.color = Color.black;
}
}
```

Task 6: Screens of the Game

Main Menu:



Game Play:



Task 7: Conceptualize an interactive system by explaining the reason for the following three methods

Start() – This is mainly used on a frame when the script is enabled before any of the Update methods will be called. Start() will be only called once in a script.

```
function Start () {
//ufos will shoot half a second after they spawn and every second after
InvokeRepeating("shootLaser",0.5,0.5);

}
```

Update() – This type of function is used when the user needs to keep track of game logic and interactions, animations, camera positions, and much more. Update() is called once per frame. It is the main process of a function for frame updates. It is also the most common function being used to implement any kind of game behaviour.

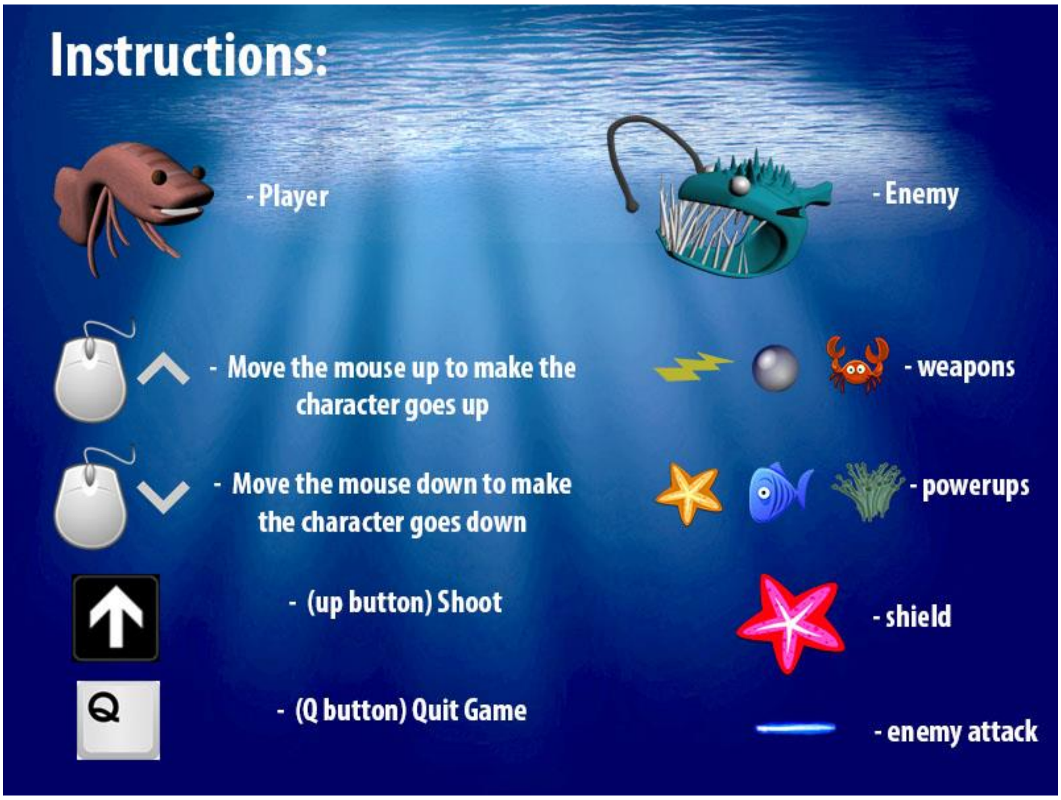
```
function Update () {
//enemy laser moves left
transform.Translate(Vector3.left * speedOfLaser * Time.deltaTime);
}
```

OnTriggerEnter() – This function is called when the Collider “other” enters the trigger. This function can be used if you have a rigidbody on the player controller so that the user can get the collision events. Also, the user generally should be using physics based movement, though that isn’t always useful if the colliders are large enough.

```
function OnTriggerEnter(other:Collider)
{
if (other.gameObject.tag == "Player")
{
//decrease lives by 1
PlayerCont1.lives--;
PlayerCont2.lives--;
PlayerCont3.lives--;
//remove the enemy laser from the scene
Destroy(this.gameObject);
}
}
```

Task 8: Implementation

The player will have control of the shrimp. Basically, the shrimp has to run away until it achieves certain amount of points (depends on the level) or else if the lives reach 0, the level will restart. If the shrimp hits an enemy, a life will be decreased. The powerups helps the player to increase the score and when 3 of the powerups are collected, a shield will appear and lets the player hit an enemy without decreasing a life. Also, the enemy shoots a laser that when it hits the character, the scores decreases by 1.



Instructions:

- Player

- Enemy

- Move the mouse up to make the character goes up

- weapons

- Move the mouse down to make the character goes down

- powerups

- (up button) Shoot

- shield

- (Q button) Quit Game

- enemy attack

Location of the executable file: Claude_Taliana_2HND2 > Lure Game

Task 9: Write a short paragraph explaining what improvements you would have implemented in the game if you had more time

If I had more time, I would have made the enemy (angler fish) and backgrounds animated as well. Also, I would have created the powerups and shooting materials myself. If I had more time I would surely have created a better gameplay and the game would be much more difficult such as: different type of enemies with different attacks, and the score must be increased or decreased depending on which enemy have attacked the player and which powerup the player have collected.

Bibliography:

- http://wiki.unity3d.com/index.php?title=The_Absolute_Beginner%27s_Guide_to_Unity
- https://www.youtube.com/watch?v=QUCEcAp3h28
- Basic Games done during the lecture
- http://www.slideshare.net/markdeloura/choosing-a-game-engine
- https://docs.unity3d.com/Documentation/ScriptReference/Collider.OnTriggerEnter.html
- http://www.w3schools.com/js/js_functions.asp
- http://answers.unity3d.com/questions/8908/is-function-update-short-for-function-start-whilet.html
- https://docs.unity3d.com/Documentation/ScriptReference/Collider.OnTriggerExit.html