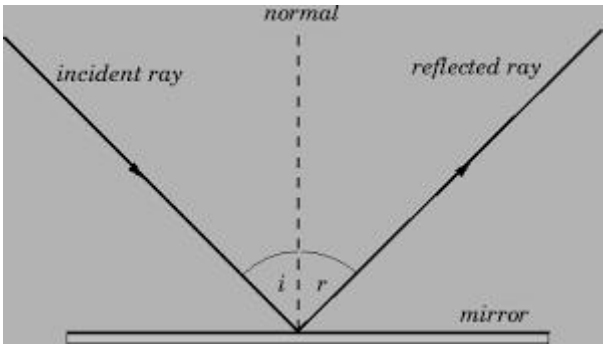
In this assignment, I researched how to simulate the reflection of the water for a pool. In the beginning, I researched how to use the mathematical method to get the value of every pixel. There is some related default function helping me. However, in this process. I got a new idea that I had ever made a mini map for a TD game. I can simulate the reflection of the water easier by using the similar way. It is more convenient and simple. However, the final version is a little fake. Thus, I choose to calculate the reflection by calculating the normal of view direction for every pixel and get the specific value on the Cubemap.



Cubemap is a very useful tool in Unity 3D. It is a little similar with the skybox. First, we should get the Cubemap from a specific point, which below our Camera. Why we have to set it below our Camera? It is because The angle of incidence equals the angle of reflection. We can see the picture right, i is equal to r. There is really great program on the unity official website creating cube map: https://docs.unity3d.com/ScriptReference/Camera.RenderToCubemap.html. What you should do is to put the script in the folder(created by yourself in Assets). Then you should create a new empty game object as the render position and create a new Cubemap(Create-Legacy-Cubemap). Then, you should open drag them into the rendering window (clicking GameObject-Render into Cubemap).

Before rendering, you should set your Cubemap readable first. Otherwise, you can't render it in your script. However, some version of Unity 3D has bugs. If your Unity crash when you click it. You should update your Unity 3D to newest version(2018.3.10f). Then, click "Render!". You will get the Cubemap.

Then, let's finish the water shader. In the struct "appdata", we should get the normal of every pixel.

```
float3 normal : NORMAL;
```

Then, in function "vert" ,increase sentences as below.

```
o.worldPos = mul(unity_ObjectToWorld, v.vertex);
o.worldNormal = mul(unity_ObjectToWorld, v.normal);
o.worldViewDir = normalize(_WorldSpaceCameraPos.xyz - o.worldPos.xyz);
o.worldRef = reflect(-o.worldViewDir, normalize(o.worldNormal));
```

"worldPos" is the position of every pixel in world space. "worldNormal"is the normal

of every pixel in world space.”worldViewDir”is the vector from Camera to every pixel(incidence ray). “worldRef”is the reflection vector(reflected ray). By using function “reflect”, we can get the value of the pixel on Cubemap, which is the intersection of the reflected ray and the Cubemap. Then, in the function “frag”, increase the following sentences.

```
float4 refltemp = texCUBE(_CubeMap, i.worldRef + noise * float3(sin(_Time.y
                                    * speed.x), sin(_Time.y * speed.y), 0.0f));
float4 fincol = lerp(refltemp, temp, _ReRate);
```

“_CubeMap” is the Cubemap we rendered just now. I add the noise to simulate the waee of the water. If you want to learn more details of the parameter I used in the scripts, you can read my last tutorials. There are more explains about them.