

Tech document

1. SinWave

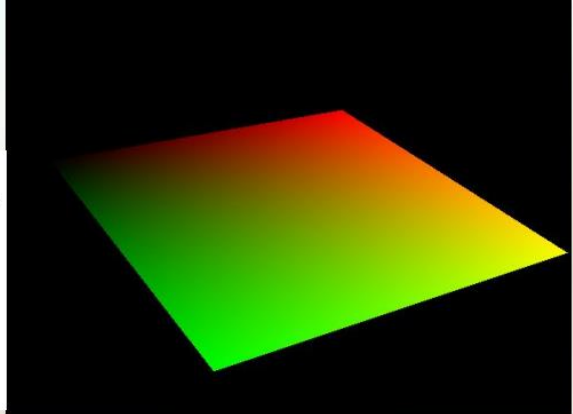
對一個平面的座標點做 \sin 函數運算來做出起伏，並利用時間 t 來讓函數可以連續。

```
if(wave_mode == 1){
    float k = 2 * PI * frequency;
    float w = k * (position.x)+t;
    vec3 p = position;
    p.y = amplitude * sin(w);
    v_out.normal = mat3(transpose(inverse(model_matrix)))*normal;
    gl_Position = proj_matrix * view_matrix * model_matrix * vec4(p, 1.0f);
    v_out.position = vec3(model_matrix * vec4(p, 1.0));
    v_out.texture_coordinate = texture_coordinate;
}
```

參考資料: <https://catlikecoding.com/unity/tutorials/flow/waves/>

2. Interactive wave-function-based height maps

1. 把 texture 座標轉換為顏色，再讀取顏色來轉換成點到的位置。



Pick

- Pick mouse position in screen space to texture space
 - Ray cast -> fast but need to compute intersection
 - Off-line render texture coordinate of wave
 - Read pixel value
 - Add drop

```
this->pool->pick(mx, h() - my - 1);

void pick(int x, int y)
{
    glBindFramebuffer(GL_READ_FRAMEBUFFER, uv_buffer.fbo);
    glReadBuffer(GL_COLOR_ATTACHMENT0);

    glm::vec3 uv;
    glReadPixels(x, y, 1, 1, GL_RGB, GL_FLOAT, &uv[0]);

    glReadBuffer(GL_NONE);
    glBindFramebuffer(GL_READ_FRAMEBUFFER, 0);

    if (uv.z != 0.0)
        this->addDrop(glm::vec2(uv), 0.03f, 0.01f);
}
```

2. 儲存點擊到的顏色(也就是座標)與時間 uv_t 。

3. Uniform 點擊到的座標與時間 uv_t 帶入函式來產生波型

```
sin((dist-t_c)*clamp(0.0125*t_c,0,1))/(exp(0.1*abs(dist-t_c)+(0.05*t_c)))*1.5;
```

備註:波型產生函式參考自蘇泓嘉

4. 傳入時間 t 與剛剛儲存的 uv_t 做差來 $update$ 即可產生連續的波型

3. Skymapping reflection

用 GLSL 的 function Reflect and Refract 來達成效果

```
vec3 norm=-normalize(cross(dFdy(f_in.position),dFdx(f_in.position)));

vec3 result={0.0,0.0,0.0};
vec3 viewDir = normalize(viewPos - f_in.position);

float ratio = 1.00 / 1.52;
vec3 I = normalize(f_in.position - viewPos);

if(reflect_enable){
    result += reflect(I, normalize(norm));
    f_color = vec4(texture(skybox, result).rgb, 1.0);
}
else if(refract_enable){
    result += refract(I, normalize(norm), ratio);
    f_color = vec4(texture(skybox, result).rgb, 1.0);
}
else{
    f_color = vec4(texture(skybox, result).rgb, 1.0);
}
```

參考資料: <https://learnopengl.com/Advanced-OpenGL/Cubemaps>