Introducció als algoritmes de Ray casting

Josep Boncompte Moya

7 de febrer de 2022

1 Objectius

En aquest treball vull estudiar els algoritmes ray casting que creen imatges digitals amb un alt grau de realisme traçant raigs. Analitzarè l'algoritme amb les diferents variants que es poden derivar, veient les ventatges i inconvenients d'aquests. També implementarè aquest algoritme aplicant-li optimitzacions.

2 Principis de l'algoritme

En aquesta secció explicarè els conceptes basics d'un Ray-tracer.

Començem explicant com veiem imatges a la vida real.

Primer de tot necesitem una font d'il·luminació, la llum és essencial per poder-hi veure. Aquesta font d'il·luminació desprén ratjos de llum que impacten contra objectes i reboten. Al rebotar canvien el seu color. El que nosaltres percevem, pertant, són els ratjos que van a parar al noste ull.

En un Ray-tracer s'intenta simular el mateix. Creem una font d'il·luminació, uns objectes, una quadricula i un ull. La quadricula representarà el conjunt de pixels de la imatge resultant. La única diferencia és que en aquest cas, els ratjos van en sentit contrari. Surten de l'ull i reboten contre els objectes.

Per cada pixel de la quadricula llencem un raig a traves, ens guardem el color dels objectes amb els que impacte. Per poder efectuar aquest algoritme ens caldrà calcular:

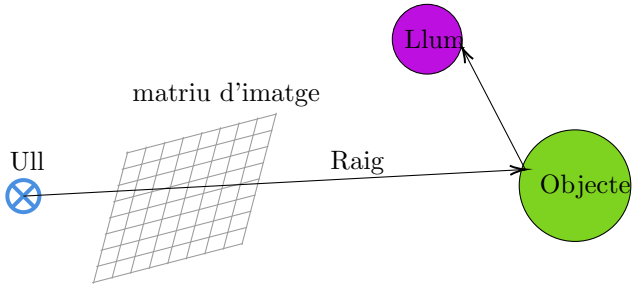

Figura 1: Representació del funcionament basic de l'algoritme de Ray casting

a) La intersecció d'un raig amb un obejecte.

b) rebot d'un raig amb una superficie.

c) repetir a) i b) amb un nou raig.

2.1 intersecció del raig amb objectes

Per saber si un raig intersecta amb un objecte ho podem fer de dues formés diferents:

2.1.1 Ray tracing

Ray tracing és una variant dels algoritmes de ray casting que resol un sistema d'equacions per trobar la intersecció del raig amb l'objecte. Aquesta variant és molt eficient però esta limitada a superficies que podem parametritzar o que tinguin una formula sensilla.

a) Començarem dibuixant esferes, ja que és el objecte més simple. Podem definir un raig amb l'equació d'una recta $f(t) = dt+p$ on $d = (d_1, d_2, d_3)$ és el vector director normalitzat, i $p = (p_1, p_2, p_3)$ és el punt de la recta quan $t = 0$. Més endavant utilitzarem rajos no rectes per simular la curvatura de l'espai. Podem expresar la recta de la forma següent:

$$\begin{aligned} x &= p_1 + td_1 \\ y &= p_2 + td_2 \\ z &= p_3 + td_3 \end{aligned}$$

Per una esfera de radi r i centre $c = (c_1, c_2, c_3)$ podem definir-la com:

$$(x - c_1)^2 + (y - c_2)^2 + (z - c_3)^2 = r^2$$

Per trobar els punts d'intersecció només cal substituir x, y, z obtenint:

$$\begin{aligned} ((p_1 - c_1) + td_1)^2 + ((p_2 - c_2) + td_2)^2 + ((p_3 - c_3) + td_3)^2 = r^2 \iff \\ (td_1)^2 + (td_2)^2 + (td_3)^2 + (p_1 - c_1)^2 + (p_2 - c_2)^2 + (p_3 - c_3)^2 + \\ 2(p_1 - c_1)td_1 + 2(p_2 - c_2)td_2 + 2(p_3 - c_3)td_3 - r^2 = 0 \iff \\ \|d\|^2 t^2 + \|p - c\|^2 + 2t(< p - c, d >) - r^2 = 0 \end{aligned}$$

Obtenint una equació de segon grau $at^2 + bt + c = 0$ on $a = \|d\|^2 = 1$ per definició, $b = 2(< p - c, d >)$ i $c = \|p - c\|^2 - r^2$.

Aquesta formula ens donarà 0, 1 o 2 solucións, que representen el temps t en el que la recta talla la esfera.

Per que el algoritme tingui sentit hem d'escollir la solució positiva més petita.

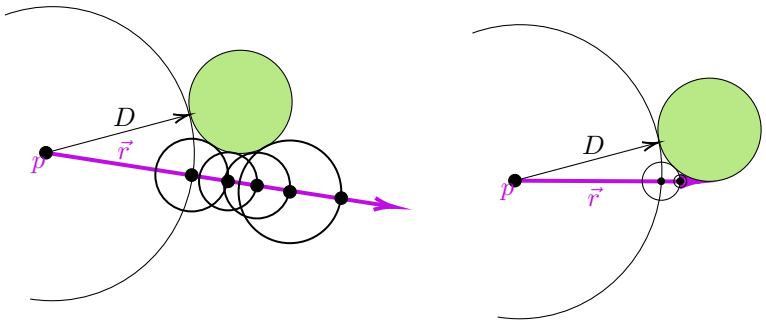

Figura 2: Calcul d'intersecció utilitzant la variant Ray marching

b) Per calcular la intersecció d'un raig amb un cub tindrem els següent conceptes presents:

...

2.1.2 Ray marching

Ray marching és una altre variant dels algoritmes de ray casting que calcula la intersecció d'un raig de forma recursiva a partir de la distancia del raig al objecte. El metode és el següent:

a) Creem un raig $f(t) = dt + p.$ amb d la direcció i p la posició.

b) calculem la distancia D minima entre p i el objecte.

c) abançem en la direcció d aquesta distancia i definim la nova posició com

$$p' = D \cdot d + p$$

d) repetim aquest procés fins que la $D = 0$ o fins que $D = \infty$ (en el cas que no intersectin).

Aquesta variant de Ray casting és costosa computacionalment, pero ens permet utilitzar objectes més complexos dels quals no podem trobar algebraicament la intersecció amb el raig.

2.2 Reflexió d'un raig amb una superficie

Donat un raig i una superficie que intersecten, volem trobar el vector director resultat de la reflexió.

Ens caldrà, pertant trobar el plà tangent de la superficie en el punt d'intersecció.

Sabem que troar el plà tangent és equivalent a trobar el vector normal de la superficie en aquell punt.

En el cas de l'esfera, el vector normal és el mateix que el vector posició.

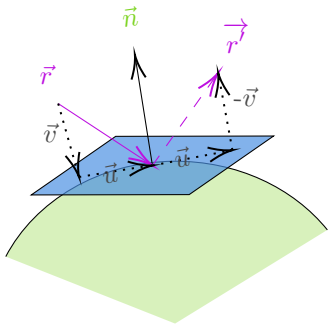

Figura 3: Reflexe d'un raig amb una esfera

Seguint la terminologia del la figura ref volem trobar el vector r' que descompon en $u - v$ tals que v sigui la projecció de r sobre N i $u = r - v$. El reflexe de r sobre la superficie només canvia la component v (de sentit). La component u es manté igual.

Anem doncs a calcular v.

$$v = \|r\| \cos(\theta) n \tag{1}$$
$$\cos(\theta) = < r, n > \tag{2}$$
$$u = r - v \tag{3}$$

on θ és l'angle que forman el vector r i n (producte escalar). Per tant el vector resultant és

$$r' = r - 2v$$

El problema que tindrém més endavant serà trobar el vector normal de superficies no parametriques (conjunt de Julia).

2.3 Repetir 2.1 i 2.2 per un nou raig

un com hem calculat la intersecció del raig amb l'objecte i tenim el vector de la reflexió, repetirem el proces amb el nou raig $f(t) = dt + p$ on d és el vector resultat de la reflexió i p és el punt d'interseció. Depen de l'escena que estiguem representant és possible que el raig reboti infinitament, pertant, haurem de posar-li un limit, el qual si s'assoleix, el color d'aquest pixel serà negre.

2.4 Suavitzat d'imatge

Explicar Millor

Amb el algoritme descrit anteriorment, només tenim un raig per cada pixel. això ens dona un resultat d'aquest estil: Si volem suavitzar una mica els limits dels objectes per donar-li un toc més de realisme haurem de repetir el procés i assignar a cada pixel la mitjana dels colors obtinguts cada cop.