DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico I

Métodos Númericos
2do Cuatrimestre - 2017

Integrante	LU	Correo electrónico
Jonathan Seijo	592/15	jon.seijo@gmail.com
COMPLETAR	COMPLETAR	COMPLETAR
COMPLETAR	COMPLETAR	COMPLETAR
COMPLETAR	COMPLETAR	COMPLETAR

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2160 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (54 11) 4576-3359
http://www.fcen.uba.ar

Índice

1. Introducción

Este trabajo consiste en la digitalización de objetos 3D basándose en imágenes producidas con cámaras tradicionales, utilizando la técnica de *fotometría estéreo*. Mostraremos que utilizando luces provenientes de diferentes ángulos, podemos aproximar las normales a la superficie y estimar las profundidades de cada punto.

Para esto debemos resolver varios sistemas de ecuaciones lineales, los cuáles resolveremos algorítmicamente de forma matricial. Usaremos en un primer caso el método clásico de Eliminación Gaussiana, y veremos como utilizando factorización LU podemos reducir los tiempos de cómputo. Luego, utilizaremos la factorización de Cholesky.

Los experimentos .... []

Completar breve pantallazo a experimentos

2. Calibración

Un paso anterior al cálculo de las profundidades, es el cálculo de los vectores normales a todo punto de la superficie. Para esto, se eligen tres luces diferentes (por ejemplo si elegimos 1, 2 y 3, nuestros vectores luz serán s^1, s^2 y s^3 respectivamente) y utilizando las intensidades ya registradas (I_i) queremos encontrar los m que son solución. Cabe aclarar que el sistema se deberá resolver para cada pixel de la imagen, por lo que ya conocemos las intensidades del pixel dado para toda imagen. Nos encontramos entonces con el primer problema, porque los vectores luz no son un dato conocido.

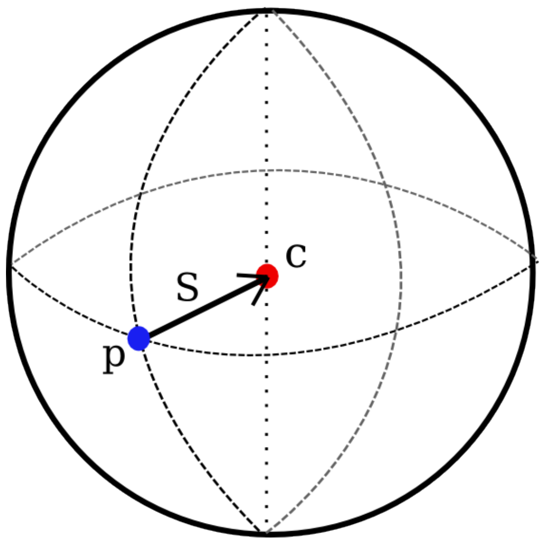
El sistema que queremos resolver es el siguiente:

$$\begin{pmatrix} s_x^1 & s_y^1 & s_z^1 \\ s_x^2 & s_y^2 & s_z^2 \\ s_x^3 & s_y^3 & s_z^3 \end{pmatrix} \begin{pmatrix} m_x \\ m_y \\ m_z \end{pmatrix} = \begin{pmatrix} I_1 \\ I_2 \\ I_3 \end{pmatrix}$$

Pero no conocemos los s_j^i.

Tenemos que $S = (s_x^i, s_y^i, s_z^i)$ es el vector luz en la imagen i. Dado que vamos a explicar el cálculo para una imagen cualquiera, omitiremos el supraíndice i para una notación mas relajada.

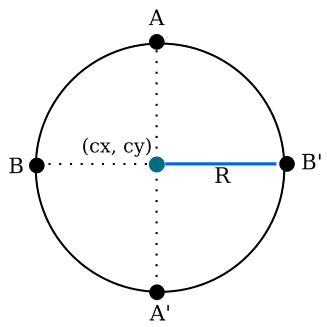
Llamemos $c = (c_x, c_y, c_z)$ al centro de la esfera. Pensemos la luz como un vector que apunta hacia el centro. El vector S toca la superficie en un cierto punto $p = (p_x, p_y, p_z)$, pero p no es un punto al azar, sino que es el punto más iluminado de la esfera. Por lo tanto, la dirección de luz que nos interesa es $S = c - p$.



Vemos que el vector que nos interesa es $S = c - p$

Mas precisamente, $S = (c_x - p_x, c_y - p_y, c_z - p_z)$. En principio no conocemos ninguno de estos valores, pero concentrémonos en calcular sobre el eje x e y. De la imagen de la esfera (2D) podemos conocer algunos datos. En la implementación utilizamos la máscara provista por la cátedra para simplificar los cálculos.

Como los píxeles en la máscara son blancos o negros, es muy sencillo identificar los puntos que pertenecen a la esfera con sólo ver su color. Recorriendo todos los puntos y tomando máximos y mínimos obtenemos 4 puntos clave: El punto del círculo que está mas a arriba A, el más abajo A', el de más a la izquierda B y el de más a la derecha B'.



Puede verse fácilmente que el radio R del círculo es la mitad de la distancia entre B y B'

$$R = \frac{|B'_x - B_x|}{2}.$$

Además, sabiendo que c es el centro del círculo,

$$c_x = B_x + R$$
$$c_y = A_y + R$$

Es fácil obtener p en una imagen 2D, basta con recorrer los pixeles de la imagen y quedarnos con el (p_x, p_y) con mayor iluminación. Tenemos entonces los datos de c_x, c_y, p_x, p_y y R. Veamos cómo podemos despejar lo que nos falta.

Recordemos lo que queríamos calcular:

$$S = (c_x - p_x, c_y - p_y, c_z - p_z).$$

Sólo la tercer componente de S es una incógnita. Sabemos que el radio del círculo es igual al radio de la esfera y el radio de la esfera es igual a la distancia euclidea entre el centro y un punto en la superficie. En particular, el radio es igual a la distancia entre p y c. Es decir:

$$\|c - p\| = R$$

Despejando..

$$\sqrt{(c_x - p_x)^2 + (c_y - p_y)^2 + (c_z - p_z)^2} = R$$

$$(c_x - p_x)^2 + (c_y - p_y)^2 + (c_z - p_z)^2 = R^2$$

$$(c_z - p_z)^2 = R^2 - (c_x - p_x)^2 - (c_y - p_y)^2$$

$$(c_z - p_z) = \sqrt{R^2 - (c_x - p_x)^2 - (c_y - p_y)^2}$$

Finalmente conseguimos lo que buscábamos. Repitiendo este procedimiento, podemos obtener todas las componentes del vector de luz para todas las imagenes. Podemos escribir a cualquier vector de luz utilizando datos conocidos:

$$s_x = c_x - p_x$$
$$s_y = c_y - p_y$$
$$s_z = \sqrt{R^2 - (c_x - p_x)^2 - (c_y - p_y)^2)}$$

Ya estamos en condiciones de resolver el sistema original pues conocemos todos sus coeficientes. Analizaremos en las siguientes secciones diferentes formas para resolverlo.

3. Métodos

3.1. Eliminación Gaussiana

Tenemos un sistema listo para resolver, dónde sólo los m_i son incógnitas:

$$\begin{pmatrix} s_x^1 & s_y^1 & s_z^1 \\ s_x^2 & s_y^2 & s_z^2 \\ s_x^3 & s_y^3 & s_z^3 \end{pmatrix} \begin{pmatrix} m_x \\ m_y \\ m_z \end{pmatrix} = \begin{pmatrix} I_1 \\ I_2 \\ I_3 \end{pmatrix}$$

La pregunta ahora es ¿cómo lo resolvemos?. Dado que en principio no sabemos cómo, nos gustaría llevarlo a una forma equivalente que sea mas fácil de resolver. Podemos hacer esta conversión a un sistema equivalente usando el algortimo de eliminación de Gauss. Lo que hace este algortimo es llevar una matriz a su forma triangular superior, de dónde luego es muy sencillo hacer los despejes finales.

El pseudocódigo del algoritmo de Gauss es el siguiente:

---

function ELIMINACIONGAUSSIANA(Matriz M[n][m])
  for $k \in [1..min(n,m)]$ **do**
    for $i \in [k+1..m]$ **do**
      if M[k][k] $\neq 0$ **then**
        mult $\leftarrow$M[i][k] / M[k][k]
        for $j \in [k+1..n]$ **do**
          M[i][j] $\leftarrow$ M[i][j] - mult*M[k][j]
      else
        Hay un cero en la diagonal!

Como puede verse, funciona correctamente solo **suponiendo que no hay ceros en la diagonal**. Es claro que puede modificarse para que realice intercambios de filas y no tenga el problema del cero, veamos que para nuestro problema no es importante. En nuestra implementación aplicaremos el algoritmo de Gauss en la siguiente matriz ampliada:

$$\left(\begin{array}{ccc|c} s_x^1 & s_y^1 & s_z^1 & I_1 \\ s_x^2 & s_y^2 & s_z^2 & I_2 \\ s_x^3 & s_y^3 & s_z^3 & I_3 \end{array}\right)$$

Nuestros s_j^i incialmente son todos distintos de cero. Además, tomamos todas las combinaciones posibles de tres luces y vimos que siempre son linealmente independientes, por lo tanto nunca vamos a hallar un cero en la diagonal y no es necesario aplicar permutaciones. Esto cobrará importancia cuando querramos aplicar factorización LU.

3.2. LU

Que es esto

Por que nos sirve para nuestro problema, hablar de repeticion de calculos

3.3. Cholesky

algo

4. Cálculo de normales

Explicar que es lo que tenemos que resolver

Explicar como aplicamos los metodos listados arriba para resolver el problema, y por que podemos hacerlo

5. Estimacion de profundidades

Hablar del ultimo sistema de ecuaciones de la cual (aun) no tengo idea

6. Resultados

Dar imagenes minimo del resultado final

7. Experimentación

PENSAR MAS

Mostrar como con diferentes luces obtenemos diferentes normales

Diferencias de tiempos EG vs LU vs mascara y combinaciones

8. Discusión/Conclusiones