HAND TRACKING CON KINECT Y PROCESSING

Encontrar el punto medio de pixeles que se encuentren a cierto umbral de profundidad.

Nivel: Intermedio

Tiempo: 30–45 minutos

Recursos: Kinect modelo 1414.
Computador Mac con Processing instalado. (Probado con la versión 3.2.1)

Para computadores que usan Windows o Linux, revisar referencias.

PASOS A SEGUIR:

1. Entender qué es Kinect y cómo funciona.
2. Instalar librería en Processing.
3. Mostrar imágen que proviene del sensor.
4. Crear un rango de profundidad para mostrar sólo una parte del cuerpo.
5. Encontrar el punto medio de los pixeles que se muestran para hacer traking

OBJETIVO:

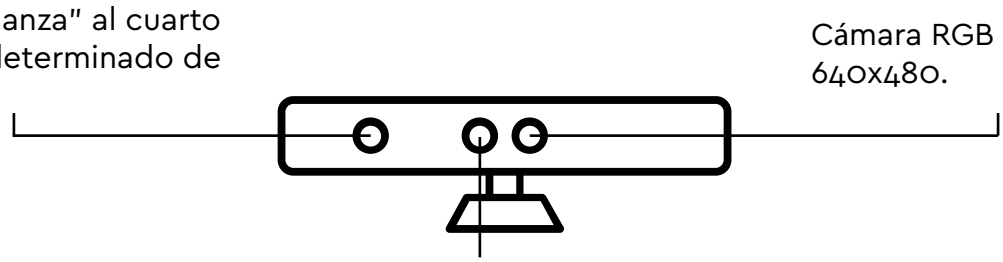
Se aprenderá a instalar y utilizar la librería que permite usar el sensor infrarojo del Kinect junto con Processing. Además, se utilizará la función de Depth Treshold para mostrar la imágen proveniente de la cámara infraroja y se creará un rango de profundidad que permitirá conectar Processing y Kinect con más precisión, al revelar sólo una parte de la imágen. Por último, se encontrará el punto medio de los pixeles, permitiendo hacer *tracking* de una parte específica del cuerpo, en este caso la mano.

En un video aparte, se dará referencia de cómo utilizar Kinect con otras aplicaciones y sistemas operativos.

PASO A PASO

Qué es Kinect y cómo funciona:

Kinect es un controlador de juegos desarrollado por MIcrosoft. Utiliza tecnología infraroja que permite medir profundidad.



Cámara infraroja que lee y analiza la distorisión en el patrón para saber que tan lejos del sensor se encuentra cada punto.

Instalar librería en Processing:

1. Abra el programa
2. Vaya a Sketch / Importar Biblioteca / Añadir Biblioteca
3. Busque la librería Open Kinect
4. Seleccionela y da click en Instalar
5. Reinicie Processing

Utilizar la función DepthImage

1. Declare funciones setup() y draw() en el sketch.
2. Importe librería de OpenKinect (Se escribe arriba de todo el código).

```
import org.openkinect.freenect.*;
import org.openkinect.processing.*;
```

3. También arriba de todo, declare la variable que va a permitir utilizar los componentes del kinect.

```
Kinect kinect;
```

4. Dentro de setup(), escriba el tamaño del sketch.

```
size(640,480); En este caso se utiliza la resolución del Kinect
               que es de 640x480.
```

5. Indíquele a Processing que empiece a usar el kinect y la función específica que quiere utilizar.

```
kinect = new Kinect(this);
kinect.initDepth();
```

6. Crear una nueva variable PImage que nos permita ver lo que reconoce el Kinect:

6.1 Declare la variable arriba de todo el código:

```
PImage prof;
```

6.2 En setup(), llame la variable e indique que sea del tamaño de la resolución del kinect:

```
prof= new PImage(kinect.width, kinect.height);
```

6.3 En draw(), indique que se carguen y se actualicen los pixeles de la imágen y dibujela en el canvas:

```
prof.loadPixels();

prof.updatePixels();
image(prof, 0, 0);
```

7. Utilizar los datos de profundidad que reconoce el kinect y crear un rango que permita dibujar solo los objetos que se encuentren en él.

7.1 Arriba de todo el código, declare variables para el valor mimino y maximo del rango:

```
int profMin= 100;
int profMax= 700;
```

7.2 En draw(), entre `prof.loadPixels(); y prof.updatePixels();`, cree un arreglo que guarde los valores de la lectura de profundidad.

```
int[] profundidad= kinect.getRawDepth();
```

7.3 Cree un *for loop* para usar cada valor de X y Y dentro de la imágen que recibe el kinect.

```
for (int x = 0; x < kinect.width; x++) {
     for (int y = 0; y < kinect.height; y++) {
```

7.4 Dentro del for, cree una variable que le permita hacer coincidir cáda pixel con la posición de la lectura de profundidad que se encuentra en el arreglo.

```
int offset = x + y * kinect.width;
int p = profundidad[offset];
```

7.5 Utilice las variables del rango para hacer visíbles sólo los pixeles que se encuentren dentro de este.

```
if (p >= profMin && p <= profMax) {
     prof.pixels[offset] = color(255);
}

else {
     prof.pixels[offset] = color(0);
}
```

8. Encontrar el punto medio de los pixeles que se dibujan dentro de acuerdo al rango:

8.1 en draw(), declare variables que reunan la suma de los valores de los pixeles en X y en Y, además de la suma total de los pixeles.

```
float sumX = 0
float sumY = 0;
float total = 0;
```

8.2 Sume los valores de X, Y y el total. (Esto se hace dentro del condicional para utilizar sólo los pixeles que se dibujan de acuerdo al rango).

```
for (int x = 0; x < kinect.width; x++) {
     for (int y = 0; y < kinect.height; y++) {
          int offset = x + y * kinect.width;
          int p= depth[offset];

     if (p >= profMin && p <= profMax) {
          prof.pixels[offset] = color(255);

          sumX += x;
          sumY += y;
          total++;
     }

     else {
          prof.pixels[offset] = color(0);
     }
```

8.2 Encuentre el punto medio de los pixeles dividiendo la suma de los pixeles en X (sumX) y en Y (sumY) entre el total de los pixeles dibujados (total)

```
float promX = sumX / total;
float promY = sumY / total;
```

8.3 Utilice estas variables para dibujar una elipse que se encuentra en el punto medio de los pixeles que se dibujan.

```
noFill();
stroke(255,0,0);
ellipse(promX, promY, 30, 30);
```

** En caso de cualquier duda, revise el video que acompaña el tutorial, el código final del ejercicio o la documentación y los ejemplos de la librería Open Kinect for Processing.*

```
import org.openkinect.freenect.*;     // Importar la librería que instalamos.
import org.openkinect.processing.*;

Kinect kinect;     // Variable que nos va a permitir usar todas las funciones del Kinect.

PImage prof;     // Variable que nos permite ver la imágen que genera el sensor.

int profMin= 100;     // Variables que delimitan el rango de profundidad
int profMax= 700;

void setup() {

  size(640, 480);
  kinect = new Kinect(this);     // Decirle a Processing que empiece a utilizarr el kinect
  kinect.initDepth();            // Específicar que funcion del kinect queremos utilizar

  prof = new PImage(kinect.width, kinect.height);     // Variable que muestra la imágen
}

void draw() {

  prof.loadPixels();     // Carga los pixeles de la imágen de kinect

  int[] profundidad = kinect.getRawDepth();     //arreglo que organiza los datos de
                                                //profundidad provenientes del sensor

  float sumX=0;     // Variables que permiten sumar los valores de los pixeles
  float sumY=0;
  float total=0;

  for (int x=0; x<kinect.width; x++) {
    for (int y=0; y<kinect.height; y++) {
      int offset = x+y*kinect.width;      // variable que permite coincidir los pixeles con
                                          // los valores del arreglo

      int p = profundidad[offset];        // variable que relaciona el offset con el arreglo

      if (p>= profMin && p<= profMax) {   // condicional que permite dibujar los pixeles
                                          // dependiendo de la profundidad
        prof.pixels[offset] = color(255);

        sumX+=x;  // suma de los pixeles en eje X y Y el total que se dibuja.
        sumY+=y;
        total++;

      } else {
        prof.pixels[offset] = color(0);
      }
    }
  }

  prof.updatePixels();        // Actualiza los pixeles de la imágen del kinect
  image(prof, 0, 0);          // Dibuja la imágen en el canvas

  float promX= sumX/total;    //Variables que permiten encontrar el punto medio de los pixeles
  float promY= sumY/total;

  noFill();                   // elipse para indicar el punto medio
  stroke(255,0,0);
  ellipse(promX, promY, 30, 30);

}
```

REFERENCIAS Y OTRAS APLICACIONES

Daniel Shiffman y OpenKinect:

http://shiffman.net/p5/kinect/
https://github.com/OpenKinect

SimpleOpenNi:

Esa es una librería que permite utilizar Kinect con Processing (v. 2.2.1) en Windows. Es recomendable descargar los drivers y el SDK de Kinect para Windows.

http://openni.ru/files/simpleopenni/index.html
https://developer.microsoft.com/en-us/windows/kinect

Synapse for Kinect:

Este programa permite controlar Ableton Live con el Kinect. Se necesita instalar Live y Max MSP y sirve tanto para MAC OSX como para Windows.

http://synapsekinect.tumblr.com/
https://www.ableton.com/en/live/
https://cycling74.com/products/max/

ofxKinect for OpenFrameworks:

Este addon permite controlar Kinect dentro del marco de OpenFrameworks (viene preinstalado en el paquete de OF). Se necesita instalar XCode (MAC OSX) o Visual Studio (Windows).

https://github.com/ofTheo/ofxKinect
http://openframeworks.cc/
https://developer.apple.com/xcode/
https://www.visualstudio.com/es/