PRACTICA 1 KINECT

Reconocimiento de Posiciones y Gestos

Enrique Garcia Gonzalez y Rubén Peralta Díaz

Problema abordado

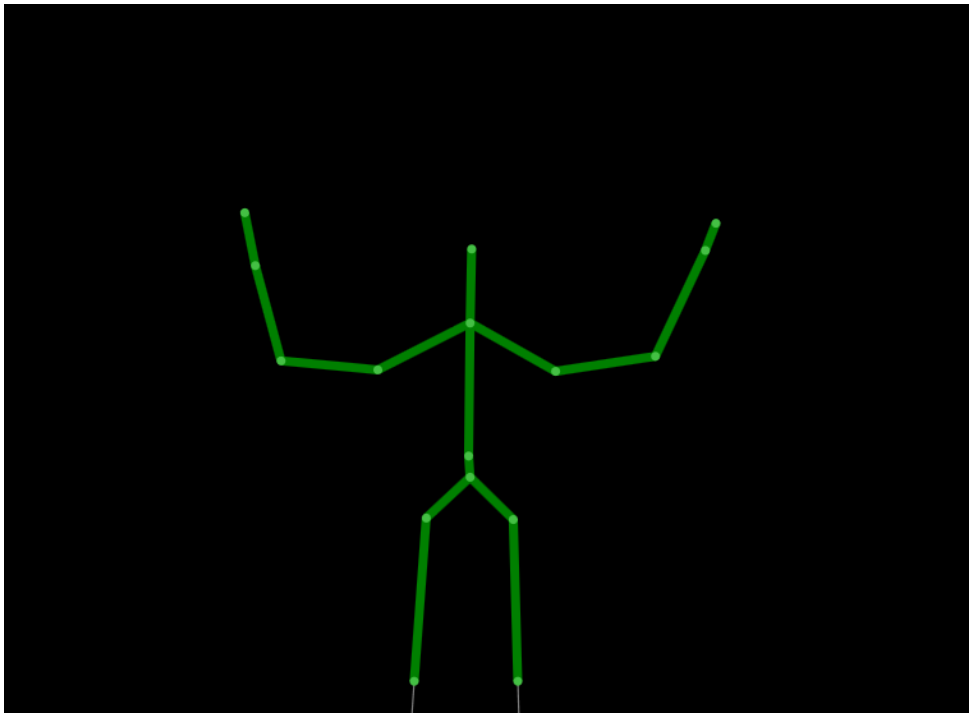
Para esta práctica hemos utilizado un dispositivo Kinect con el propósito de familiarizarnos con la API proporcionada por Microsoft y utilizar el dispositivo para el reconocimiento de posiciones y gestos.

El Kinect que hemos usado para esta práctica es el Kinect 1, dispositivo que viene con la Xbox 360 y la API utilizada es la 1.8. Con las funciones proporcionadas por la API de Kinect se puede obtener directamente la posición de diversas partes del cuerpo y generar una estructura llamada skeleton que simula la silueta de una persona.

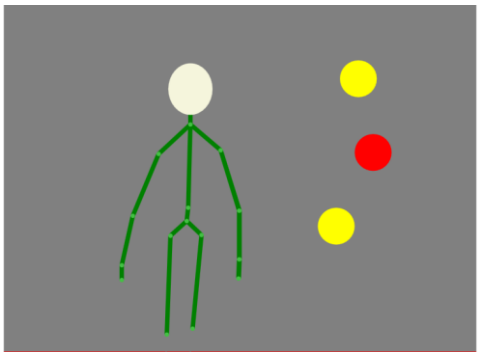
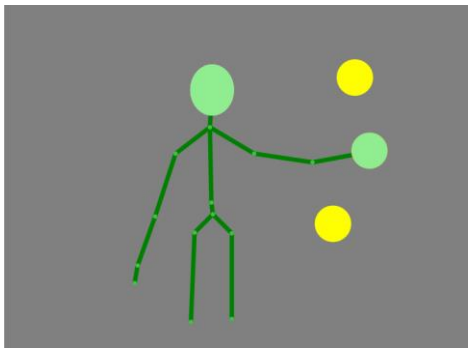
Nuestra aplicación usa esta estructura para detectar una posición que consiste en tener el brazo en horizontal y después el gesto consiste en subir o bajar el brazo a partir de esta posición. Para ello hemos utilizado 3 circunferencias, una roja y dos amarillas, situadas encima y debajo de la roja. Al tocar la bola roja se pondrá de color verde y ahora si pasamos la mano por encima de una de las bolas amarillas el fondo cambiará.

Descripción de la solución

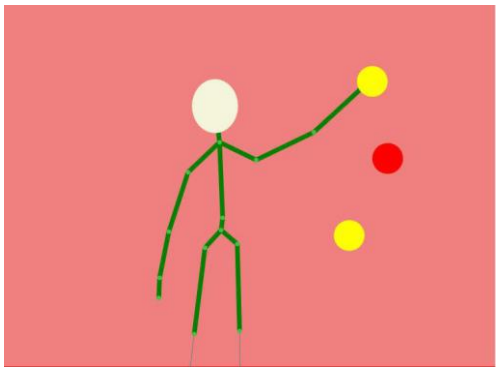
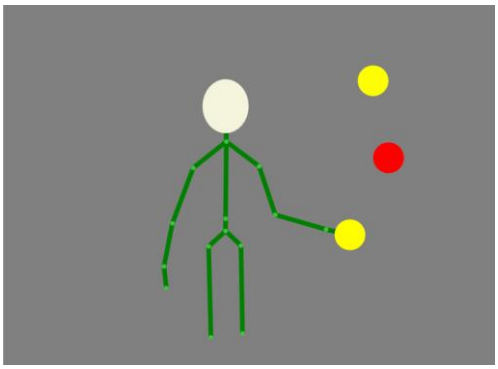
Nuestra solución parte del ejemplo en C# proporcionado en la API de Kinect llamado skeletonBasic. Este proyecto desde el que hemos partido lo que hace es detectar si hay un dispositivo Kinect conectado al PC y si lo detecta muestra una ventana negra. Si el dispositivo detecta una persona muestra una estructura en forma de maniquí en el fondo negro con la postura de la persona que detecta. Puede detectar hasta a dos personas.



A partir de este código hemos añadido una circunferencia en un lado de la pantalla la cual, si el usuario coloca su mano por encima de ella, podremos interaccionar con otras dos circunferencias situadas en la parte superior e inferior.

Ahora si el usuario coloca la mano en una de estas dos esferas el fondo cambiara de color. Si coloca la mano en la circunferencia superior cambiará al color siguiente, mientras que si colocas la mano en la circunferencia inferior cambiará al fondo anterior.

En primer lugar, hemos creado un vector para almacenar los distintos colores que tenemos para cambiar de fondo.

```
// Valor que controlara el color de fondo activo
private int fondo = 0;

SolidColorBrush[] colores = new SolidColorBrush[7];

private bool inicioReconocido = false;
```

Después, en la función “public MainWindow()” hemos inicializado las componentes del vector

```
public MainWindow()
{

        colores[0] = Brushes.Gray;
        colores[1] = Brushes.LightCoral;
        colores[2] = Brushes.LightSalmon;
        colores[3] = Brushes.MediumPurple;
        colores[4] = Brushes.Olive;
        colores[5] = Brushes.Orange;
        colores[6] = Brushes.Black;
    InitializeComponent();
}
```

El resto del código que hemos implementado ha sido dentro de la función "`private void SensorSkeletonFrameReady(object sender, SkeletonFrameReadyEventArgs e)`". Esta función en el proyecto original es la encargada de capturar los datos del sensor y pintar los esqueletos en la ventana.

Primero hemos creado 3 datos de tipo Point para que almacenan la información del punto sobre el que se dibujaran las circunferencias

```
private void SensorSkeletonFrameReady(object sender, SkeletonFrameReadyEventArgs e)
{
    Skeleton[] skeletons = new Skeleton[0];
    Point puntoPosInicial, puntoNextBG, puntoPrevBG;
    puntoPosInicial = new Point(500.0, 200.0);
    puntoNextBG = new Point(480.0, 100.0);
    puntoPrevBG = new Point(450.0, 300.0);
```

Para pintar las circunferencias utilizamos la función DrawEllipse, que recibe como parámetros el color del objeto, el centro de la elipse, el radio en x y el radio en y, para pintar una elipse. En el siguiente código se ve como pintamos los puntos de la pantalla y otro que representa la cabeza.

```
if (skeletons.Length != 0)
{
    foreach (Skeleton skel in skeletons)
    {
        RenderClippedEdges(skel, dc);

        if (skel.TrackingState == SkeletonTrackingState.Tracked)
        {

            this.DrawBonesAndJoints(skel, dc);
            dc.DrawEllipse(Brushes.Beige, null, SkeletonPointToScreen(skel.Joints[JointType.Head].Position), 30.0, 35.0);
            dc.DrawEllipse(Brushes.Red, null, puntoPosInicial, 25.0, 25.0);
            dc.DrawEllipse(Brushes.Yellow, null, puntoNextBG, 25.0, 25.0);
            dc.DrawEllipse(Brushes.Yellow, null, puntoPrevBG, 25.0, 25.0);
```

En el siguiente código se detecta cuando se toca la circunferencia central, y si ya la hemos tocado también cambia el fondo al pasar la mano por uno de las otras circunferencias. En el primer if si la mano esta encima de la circunferencia cambiara el color de esta a verde y pondrá inicioReconocido a true, que nos indicara si tenemos activas las otras dos circunferencias. Los otros dos if se encargan de cambiar el fondo si las otras dos circunferencias están activas y pasamos la mano por encima de una de ellas.

```
if (tocaBola(manoDerecha, puntoPosInicial))
{
    dc.DrawEllipse(Brushes.LightGreen, null, puntoPosInicial, 25.0, 25.0);
    dc.DrawEllipse(Brushes.LightGreen, null, SkeletonPointToScreen(skel.Joints[JointType.Head].Position), 30.0, 35.0);
    inicioReconocido = true;
}
if (inicioReconocido && tocaBola(manoDerecha, puntoNextBG))
{
    fondo = (fondo + 1) % 7;
    dc.DrawEllipse(Brushes.LightGreen, null, puntoNextBG, 25.0, 25.0);
    dc.DrawRectangle(colores[fondo], null, new Rect(0.0, 0.0, RenderWidth, RenderHeight));
    inicioReconocido = false;
}
if (inicioReconocido && tocaBola(manoDerecha, puntoPrevBG))
{
    fondo = (fondo - 1) % 7;
    dc.DrawEllipse(Brushes.LightGreen, null, puntoPrevBG, 25.0, 25.0);
    dc.DrawRectangle(colores[fondo], null, new Rect(0.0, 0.0, RenderWidth, RenderHeight));
    inicioReconocido = false;
}
```

Por último, la función tocaBola que indica si estamos tocando con una mano una circunferencia determinada utilizada en la parte del código anterior es la siguiente.

```
private bool tocaBola(Point manoDer, Point bola)
{
    bool toca = false;
    if (Math.Abs(manoDer.X-bola.X)<20.0f)
        if (Math.Abs(manoDer.Y-bola.Y)<20.0f)
            toca = true;
    return toca;
}
```

Problemas encontrados

El principal problema que hemos tenido ha sido el de añadir las instrucciones en texto en la pantalla, cosa que no hemos podido solucionar.

Nos llevó también bastante tiempo entender el código proporcionado por los ejemplos de la API.

Por último, intentamos hacer que las circunferencias dependiesen de la posición del esqueleto, aunque lo conseguimos, ya teníamos todo montado.

Lecturas Recomendadas

A la hora de entender el código, nos ha ayudado mucho una serie de desafíos subidos por Microsoft, en general el resuelto en el siguiente enlace https://msdn.microsoft.com/es-es/windows/hh545503

Y para pintar las elipses https://msdn.microsoft.com/es-es/library/system.windows.media.drawingcontext(v=vs.110).aspx

Referencias

Las referencias usadas son el proyecto SkeletonBasic en C# que se encuentra en la API de Kinect.

El proyecto se encuentra alojado en: https://github.com/NPIRubenyEnrique/Nuevos-Paradigmas-de-Interaccion