Aplicación para Kinect

Nuevos Paradigmas de la Interacción

Universidad de Granada

Profesor: Marcelino Cabrera Cuevas

Alumnos:

Daniel Álvarez González

Rebeca Vallejo Fernández

Eleuterio Risoto Roldán

Introducción

Con este proyecto, se busca desarrollar una aplicación sencilla para Kinect que permita realizar una tabla de ejercicios, de modo que el usuario haga cada ejercicio un número de veces establecido por el programa. Así, el usuario deberá realizar el movimiento descrito en pantalla y, cuando lo haya realizado correctamente en un número determinado de repeticiones, pasará al siguiente ejercicio hasta terminar con la tabla.

En cada momento, se mostrará en pantalla no sólo el estado de la cámara, sino el movimiento que debe realizar el usuario, así como las repeticiones correctas que lleva hasta el momento.

Ejercicios propuestos

Una descripción básica de los movimientos que forman la tabla de ejercicios de la aplicación:

- Ambas manos por encima de la cabeza: Se trata de levantar ambos brazos a la vez, de forma que los puños queden por encima de la cabeza.

- Subir la pierna derecha casi a la altura de la cadera y luego la izquierda.

- Dar un puñetazo con la mano derecha y luego la izquierda: Se trata de hacer un ejercicio de boxeo, de forma que se vaya alternando cuando un puño está por delante del tronco y el otro detrás.

- Mariposa: con los brazos abiertos y las manos por encima de los hombros, se mueven los puños hacia atrás hasta colocarlos un poco por detrás del pecho.

Codificación y ejecución

En primer lugar, para controlar la llamada a nuestros ejercicios, debemos hacerlo desde el manejador de eventos para los sensores de Kinect. En nuestro caso, esto se produce en la función *SensorSkeletonFrameReady*, que se ejecuta para cada *frame* del dispositivo.

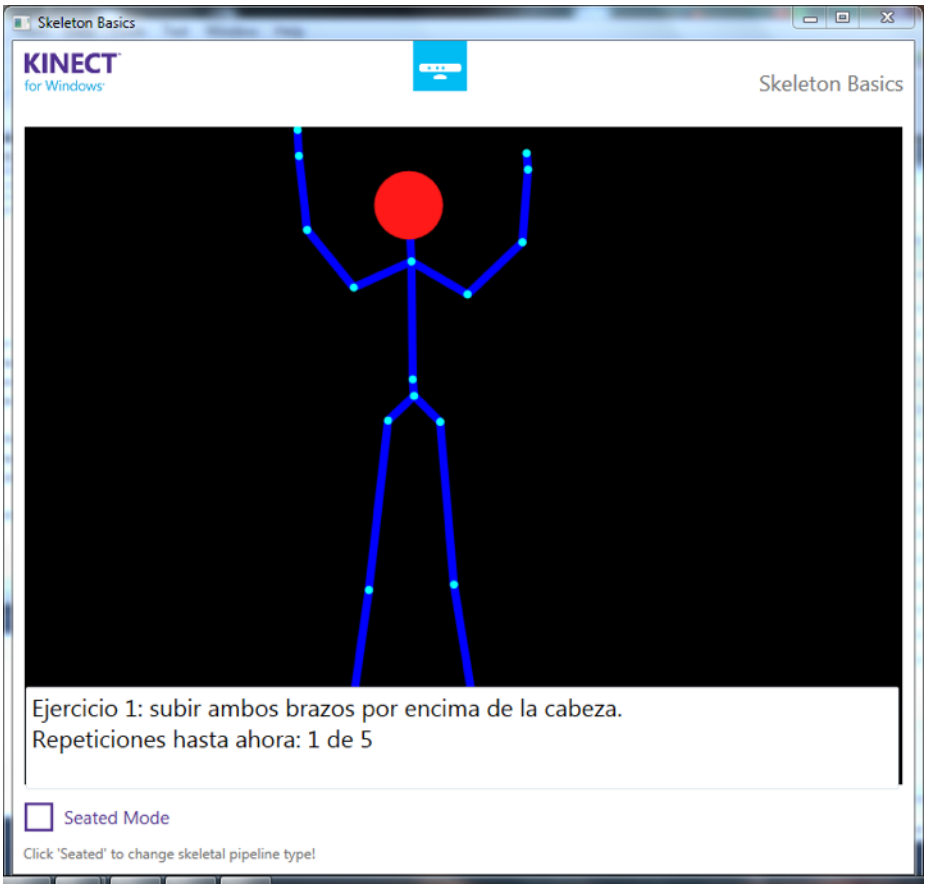
Así, utilizando un *switch* sobre nuestra variable global ***ejercicio***, podemos ir de uno a otro ejercicio, según corresponda. Así, para el primer ejercicio, tenemos el siguiente código:

```
switch (ejercicio) {
    case 0:
        // Ejercicio 1
        this.trackedBonePen = new Pen(Brushes.Blue, 6);
        anterior = actual;
        actual = Ejercicio_1(skel);
        if (anterior && !actual) {
            conteo++;
            this.trackedBonePen = new Pen(Brushes.Green, 6);
            textBox1.Text = "Ejercicio 1: subir ambos brazos por encima de la cabeza.
                \nRepeticiones hasta ahora: " + conteo.ToString() + " de " +
                repeticiones.ToString();
        }

        if (conteo >= repeticiones) {
            ejercicio++;
            conteo = 0;
        }
        break;
    case 1:
        ...
}
```

Que realiza una llamada a la función *Ejercicio_1*:

```
private bool Ejercicio_1(Skeleton s) {
    // Levantar ambas manos por encima de la cabeza

    //Sacamos los datos (puntos de las articulaciones) del skeleton
    Joint rightHand = s.Joints[JointType.HandRight];    //Obtenemos la mano derecha
    Joint head = s.Joints[JointType.Head];              //Obtenemos la cabeza
    Joint leftHand = s.Joints[JointType.HandLeft];      //Obtenemos la mano izquierda

    //Aquí, obtenemos la coordenada Y de cada una de las articulaciones obtenidas
    double rightHandY = rightHand.Position.Y;
    double headY = head.Position.Y;
    double leftHandY = leftHand.Position.Y;

    return rightHandY > headY && leftHandY > headY;
}
```

Y que genera la siguiente ejecución:



Del mismo modo, para los ejercicios 2 y 3:

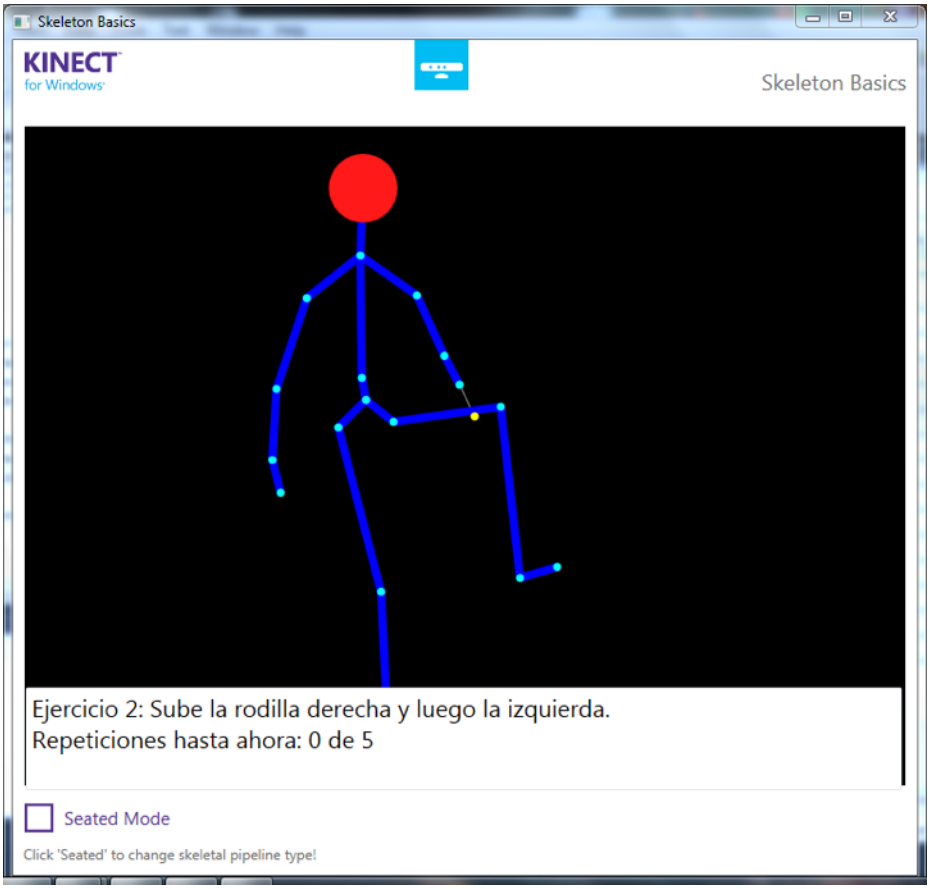
```
switch (ejercicio) {
    case 0:
        ...
    case 1:
        // Ejercicio 2
        this.trackedBonePen = new Pen(Brushes.Blue, 6);
        aux = Ejercicio_2(skel, estado);
        if (aux == 1) estado = 1;
        if (aux == 2) estado = 2;
        if (aux == 0) estado = 0;

        textBox1.Text = "Ejercicio 2: Sube la rodilla derecha y luego la izquierda.
            \nRepeticiones hasta ahora: " + conteo.ToString() + " de " +
            repeticiones.ToString();
        if(estado == 0){
            conteo++;
            this.trackedBonePen = new Pen(Brushes.Green, 6);
            estado = -1;
            aux = -1;
            if(conteo >= repeticiones){
                ejercicio++;
                conteo=0;
            }
        }
        break;

    case 2:
       // Ejercicio 3
        this.trackedBonePen = new Pen(Brushes.Blue, 6);
        aux = Ejercicio_3(skel, estado);
        if (aux == 1) estado = 1;
        if (aux == 2) estado = 2;
        if (aux == 0) estado = 0;

        textBox1.Text = "Ejercicio 3: Da un puñetazo con la mano derecha y luego la
            izquierda. \nRepeticiones hasta ahora: " + conteo.ToString() + " de " +
            repeticiones.ToString();
        if(estado == 0){
            this.trackedBonePen = new Pen(Brushes.Green, 6);
            conteo++;
            estado = -1;
            aux = -1;
            if(conteo >= repeticiones){
                ejercicio++;
                conteo=0;
            }
        }
        break;

    case 3:
        ...
}
```

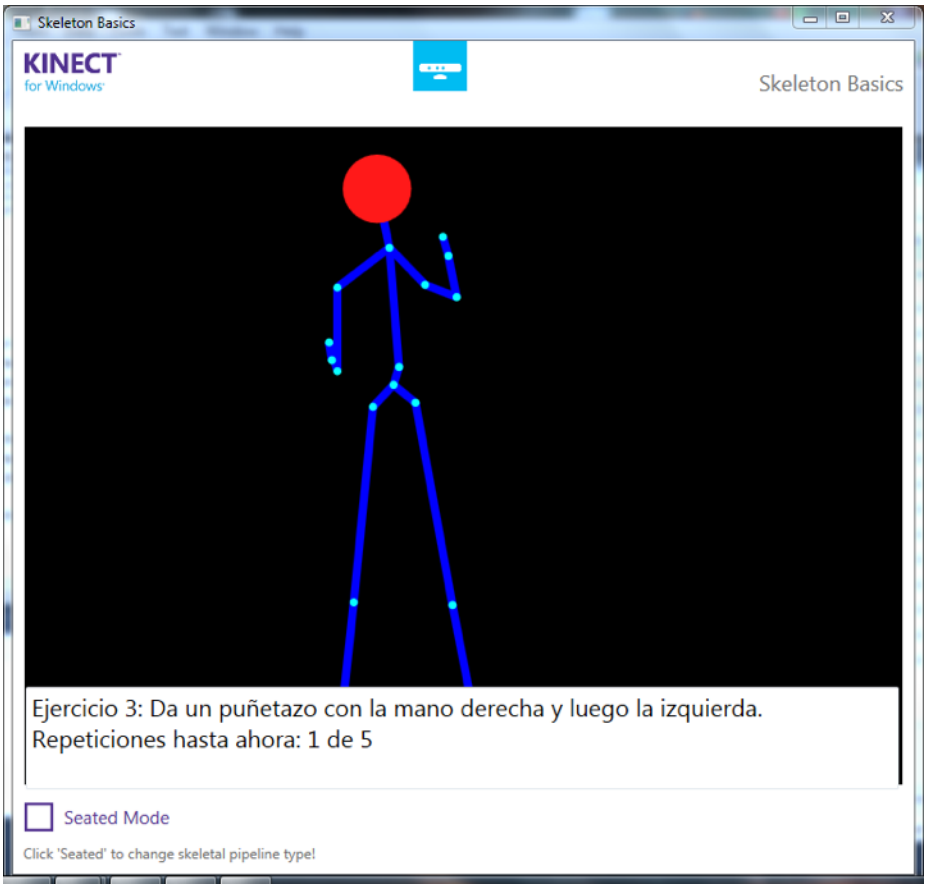
En este caso, puesto que para completar estos ejercicios es necesario hacer tres movimientos (para el primero, subir la pierna derecha, reposo y subir la otra pierna; y para el segundo, dar un puñetazo con una mano, reposo y luego con la otra mano), se recurre a un autómata finito determinista de tres estados. Así, el estado 0 representa el reposo después de haber completado una repetición del ejercicio, por lo que en ese caso, se aumenta la variable de conteo y se cambia momentáneamente el color del esqueleto a verde.

En este caso, se producen llamadas a las funciones *Ejercicio_2* y *Ejercicio_3*, que funcionan de forma semejante al tratar los distintos estados por los que pasa el autómata:

```
private int Ejercicio_2(Skeleton s, int estado){

    double rightHipY = s.Joints[JointType.HipRight].Position.Y;
    double leftHipY = s.Joints[JointType.HipLeft].Position.Y;
    double rightKneeY = s.Joints[JointType.KneeRight].Position.Y;
    double leftKneeY = s.Joints[JointType.KneeLeft].Position.Y;

    // Consideramos que se ha levantado la pierna cuando la rodilla
    // está al 60% de la altura de la cadera
    rightHipY = rightHipY*0.6;
    leftHipY = leftHipY*0.6;

    switch (estado){
        case -1:
        case 0:         // Hay que levantar la pierna derecha
            if(rightKneeY >= rightHipY)
                return 1;
            else
                return -1;
        case 1:          // Hay que levantar la pierna izquierda
            if(leftKneeY >= leftHipY)
                return 2;
            else
                return 1;
        case 2:          // Se ha completado el ejercicio y se vuelve al reposo
            if(leftKneeY < leftHipY && rightKneeY < rightHipY)
                return 0;
            else
                return 2;
        default:
            return -1;
    }
}
```

Y que tienen las siguientes ejecuciones:





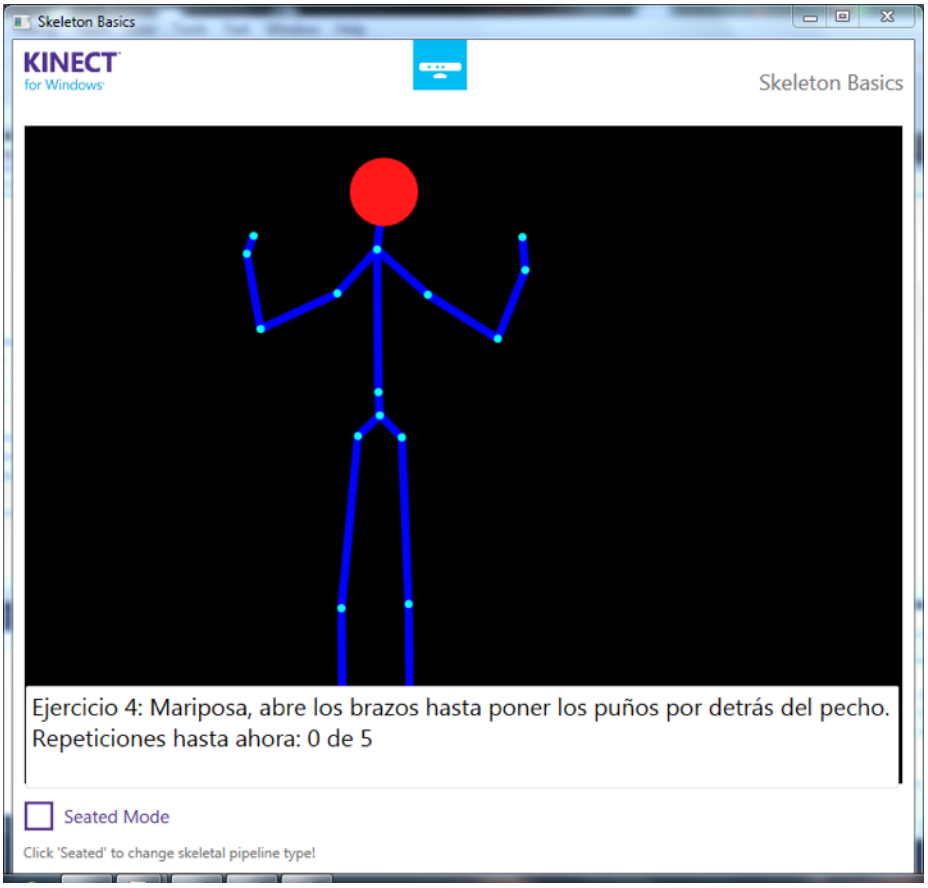
Por último, para el ejercicio 4, tenemos el resto del *switch* dentro de la función *SensorSkeletonFrameReady*. Además, cuando se termina con la tabla de ejercicios, se pasa al *case 4*, donde cambiamos el color del esqueleto y mostramos un mensaje.

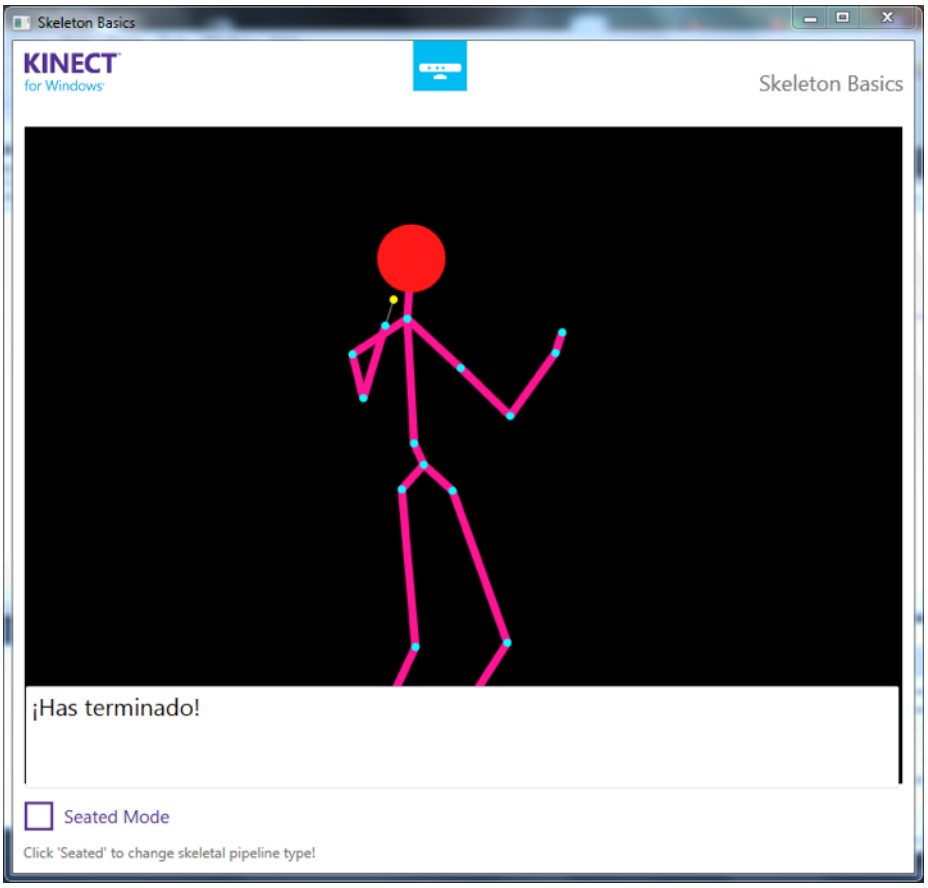
```
switch (ejercicio) {
    case 0:
        ...
    case 1:
        ...
    case 2:
        ...
    case 3:
        // Ejercicio 4
        this.trackedBonePen = new Pen(Brushes.Blue, 6);
        anterior = actual;
        textBox1.Text = "Ejercicio 4: Mariposa, abre los brazos hasta poner los puños
            por detrás del pecho. \nRepeticiones hasta ahora: " + conteo.ToString() +
            " de " + repeticiones.ToString();
        actual = Ejercicio_4(skel);
        if (anterior && !actual) {
            this.trackedBonePen = new Pen(Brushes.Green, 6);
            conteo++;
        }
        if (conteo >= repeticiones) {
            ejercicio++;
            conteo = 0;
        }
        break;

    case 4:
        this.trackedBonePen = new Pen(Brushes.DeepPink, 6);
        textBox1.Text = "¡Has terminado!";
        break;

    default:
        textBox1.Text = "Error.";
}
```

Y la llamada que se realiza a la función *Ejercicio_4*:

```
private bool Ejercicio_4(Skeleton s) {
    Joint rightHand = s.Joints[JointType.HandRight];
    Joint leftHand = s.Joints[JointType.HandLeft];
    Joint ShoulderCenter = s.Joints[JointType.ShoulderCenter];

    double rightHandZ = rightHand.Position.Z;
    double leftHandZ = leftHand.Position.Z;
    double shoulderCenterZ = ShoulderCenter.Position.Z;

    return (rightHandZ > shoulderCenterZ) && (leftHandZ > shoulderCenterZ);
}
```

De esta forma, las ejecuciones del último ejercicio y la pantalla de finalización, quedan así:





Referencias

Microsoft Developer Network; Kinect for Windows SDK: http://msdn.microsoft.com/en-us/library/hh855347.aspx

Visual Studio Gallery; Kinect Application Project Template: http://visualstudiogallery.msdn.microsoft.com/1e220317-8f73-43ed-afdd-1794b91200c5

Ray Chambers; Kinect Applications: http://raychambers.wordpress.com/kinect-applications/