TFG ADRIÁN GONZÁLEZ HERRERA

DEFINICIONES:

API: (Application Programming Interface). Conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstacción.

FUNDAMENTOS TEÓRICOS:

La detección de gestos es un ámbito de estudio recurrente en la ciencia de la computación, haciendo uso de algoritmos matemáticos para ser capaces de detectar e interpretar dichos gestos. Enfocado, mayoritariamente, en el reconocimiento de expresiones faciales y gestos con las manos. Interpretando el lenguaje de signos [4], mediante cámaras y sensores, las máquinas consiguen ser capaces de entender e interpretar los gestos realizados por las personas, logrando de ese modo un mayor *HCI.*

Mediante la detección de gestos, algunos desarrolladores pretenden sustituir (o convivir) con el uso del teclado y del ratón.

Leap Motion es un pequeño sensor de control gestual que nos permite capturar con mucha precisión nuestras manos, incluyendo dedos, articulaciones y objetos. Esta interfaz 3D nos da la opción de interactuar con nuestro ordenador sin ni siquiera tocarlo, dejando de lado dispositivos convencionales como son el teclado y el ratón, o más modernos como las pantallas táctiles.

Algunas de las más destacadas son:

 Navegar por Internet, leer artículos, ver fotos, vídeos o reproducir música con sólo mover un dedo.

 Dibujar, pintar, diseñar con la punta del dedo. Además, se puede utilizar un lápiz real o un pincel.

 Esculpir, moldear, estirar, doblar y construir objetos 3D. Desarmar y volver a unir objetos.

 Introducir la mano en diferentes mundos, agarrar objetos, girarlos.

 Jugar a infinidad de juegos usando tan solo las manos.

El diseño de este dispositivo, consiste en un aparato con unas dimensiones muy reducidas, que se conecta mediante USB a nuestro ordenador. Después tan solo hay que configurar el dispositivo con el software específico y listo.

La API de Leap Motion viene con pocos gestos incorporados, por lo que para aplicaciones complejas es necesario que el usuario de-

fina su propio conjunto de interacciones a usar. Estamos acostumbra-dos a estados binarios de seleccionado o no, arriba o abajo. Con Leap Motion, la experiencia se define menos por los estados individuales y más por las transacciones entre dichos estados. Debe responder cons-tantemente a los movimientos del usuario (feedback), para dar a enten-der que nos está escuchando y entendiendo. Esto se aleja los controla-dores tradicionales, en los que la interfaz solo cambia cuando el usuario actúa directamente sobre el dispositivo.

TECNOLOGÍAS PREDECESORAS:

El primer dispositivo de control de movi-miento fue proporcionado por Nintendo Wii. Se in-trodujo la detección de movimiento aplicada a di-versos juegos.

Más tarde el Kinect de Microsoft amplió esta idea incluyendo el movimiento de todo el cuerpo además de eliminar cualquier elemento que se interpusiera entre la pantalla y el usuario como man-dos u otros. Esto introdujo una nueva ge-neración de sensores con control gestual.

HARDWARE:

El LEAP Motion contiene en una caja de metal de 75 mm de largo, 25 mm de ancho y 11 mm de alto los componentes hardware necesarios para su correcto funcionamiento. Entre ellos, encontramos dos cámaras, tres luces leds, un microcontrolador y un controlador USB. Este dispositivo es compatible con los sitemas operativos Linux, Macin-tosh y Windows.

SOFTWARE:

El procesado de los gestos y movimientos capturado por el LEAP Motion se realiza en el dispositivo al que está conectado, en un software llamado “Leap Motion Service”. Los sensores captan las dis-tintas secuencias de movimientos en imágenes, que son enviadas al ordenador de destino. Una vez el ordenador las recibe, analiza y com-pensa las imágenes con la iluminación y herramientas necesarias, eli-minando el “ruido” de fondo como la cabeza, muebles, luces… u otros elementos que puedan interferir en la posterior detección de gestos.

A continuación, se procede a extraer de la imagen capturada una reconstrucción en 3D de lo que realmente el dispositivo “ve”, inten-tando obtener los dedos y las manos de las imágenes ya “limpias” (sin el ruido de objetos de fondo). En este proceso se aplican técnicas de filtrado para asegurar una lectura del movimiento suave, sin cortes y coherente a tiempo real.

Una vez el LEAP Motion Service tiene el movimiento “limpio” guardado, lo transmite mediante un protocolo de transporte (TCP o WebSocket, dependiendo de dónde se use el dispositivo) sobre una co-nexión local. La librería del cliente organiza los datos obtenidos (el aná-lisis del movimiento) en la estructura API creada por los desarrolladores, y los enlaza; de este modo se asocia el movimiento capturado con la clase o función que le corresponde en el API. Una vez vinculado el gesto grabado, y encontrada la clase o función, se procede a ejecutar la llamada a la acción vinculada.

FUNCIONAMIENTO INTERNO:

El LEAP Motion, como bien hemos explicado anteriormente, úni-camente se encarga de capturar los distintos movimientos con sus sen-sores, no los procesa; de ello se encarga la computadora a la que se conecta. El procesado de gestos pasa por 3 etapas para llegar a ser interpretado por el ordenador: una vez las imágenes son capturadas por los sensores, se aplica una correción de la distorsión que producen los sensores. Más tarde se aplica un modelo matemático para determinar la configuración de cada mano, junto a un algoritmo de visión estereos-cópica entre cada pareja de imágenes con la finalidad de conocer la posición en el plano que representa LEAP tridimensional.

Para identificar los distintos gestos de manera correcta, cuando se detecta movimiento sobre el dispositivo, éste ilumina la zona a la que enfocan las cámaras con una luz infrarroja emitida por sus luces LEDs. Cuando esta luz refleja en las manos, los sensores incorporados en el controlador reciben esta reflexión y la almacenan en una matriz como una imagen digitalizada. Los sensores captan el valor de intensidad lu-minosa por cada píxel de la imagen capturada, y se guardan en un buf-fer. Este valor de la intensidad luminosa se cuantifica a 8 bits (256 va-lores distintos de luminosidad) para producir una imagen RAW en es-cala de grises. Cada imagen tiene una medida de 640x120 píxeles, por lo que en total hay unos 76.800 píxeles por cada imagen. Es en este momento del proceso donde el microcontrolador ajusta a la resolución adecuada la matriz con la imagen digitalizada.

Una vez ajustada correctamente la resolución, los datos de los sensores se envían directamente al driver instalado en la computadora conectada por USB. Una vez el driver tiene esta información, la analiza y trata de identificar las manos y los dedos a partir de una secuencia matemática de caracterización anatómica. También se obtiene la pro-fundidad a la que se encuentran mediante otro algoritmo distinto, que se explicará también a continuación.

Distintos tipos de distorsiones.

Las lentes del dispositivo producen una distorsión en la imagen óptica de lo que realmente capta; se trata de una mezcla entre la dis-torsión de barril y la distorsión de cojín: una distorsión compleja. Esta distorsión deforma el objeto observado, y para optimizarla, LEAP Mo-tion incorpora una opción de calibrado mediante la cual se obtiene un mapa de mallado de puntos que se superpone a la imagen capturada por cada sensor.

El dispositivo trata estos datos de la siguiente manera: cada buf-fer de datos de imagen que se envía al driver contiene otro buffer que contiene los datos de distorsión. Mediante una rejilla de 64x64 puntos con dos valores de 32 bits cada uno se representa la distorsión de la lente. Así, cada punto representa un rayo proyectado en la cámara, y el valor de un punto del mallado define la luminosidad de un píxel en la imagen. Se puede obtener el valor del brillo para cualquier rayo proyec-tado, al igual que los datos de luminosidad de todos los píxeles, pero este último dato se obtiene mediante interpolación.

La siguiente imagen nos permite explicar cómo se aplica la co-rrección de la distorsión compleja que crea la lente; como se puede apreciar, no es una imagen rectangular, pues al estar obtener la imagen deformada y posteriormente aplicar la corrección, se deforma su figura. La imagen se reconstruye mediante el cálculo de las pistas horizontales y verticales representados por cada píxel y se puede encontrar el valor de brillo verdadero. El valor del brillo que realmente proporciona el re-flejo de la mano se puede encontrar en el mapa de calibración.

Una vez contamos con la proyección de la imagen con la distor-sión corregida, el driver tratará de identificar las manos y dedos de esa secuencia. Una vez las detecte, lo primero que determinará será su po-sición en el sistema de coordenadas en el que trabaja LEAP. Se trata de un sistema de coordenadas cartesianas a través de técnicas de vi-sión estereoscópicas; como su sistema de detección de gestos está ba-sado en la visión binocular, podremos obtener distancias “fácilmente”.

El sistema de visión binocular capta dos imágenes con cierta disparidad (diferencias mínimas entre las dos fotos). Funciona de la misma manera que si las cámaras fueran nuestros ojos: al encontrarse un poco separados, la imagen que ambos captan no es la misma, pero sí muy similar. Mediante un sistema de dos ecuaciones, considerando que la distancia focal en ambas cámaras es la misma y conociendo la distancia a la que se encuentran separadas, podemos obtener la dis-tancia a la que se encuentran las manos respecto al LEAP Motion.

API:

─ Hand:

Esta clase implementada en el API del LEAP Motion aporta informa-ción sobre las manos que detecta. Representada en base ortonormal, aporta datos como los grados de inclinación, radio de curvatura de la mano, posición relativa, probabilidad del movimiento de la mano del si-guiente frame…

─ HandList:

Se trata de una lista de manos que contiene las manos detec-tadas simultáneamente (en Objetos hand). También existe una list para los objetos tipo finger, que es FingerList, con métodos similares en ambas clases.

─ Finger:

Aporta información (también en base ortonormal) sobre cada dedo detectado en cada una de las manos (mano a la que le corres-ponde, posición, inclinación, vector de dirección, anchura y longitud...).

─ Bone:

Refleja el esqueleto de la mano representada, concretamente las falanges de los dedos. La posición de la articulación, anchura, lon-gitud o posición son algunos de los datos que se pueden extraer de esta clase.

─ Frame:

Es el conjunto de manos o dedos representados en grupo, en un único frame.

─ Listener:

La clase Listener se encarga de responder a los eventos pro-porcionados por el objeto Controller. Estas respuestas pueden ser configuradas y personalizadas para cambiarse por otras según el gusto del usuario. Para controlar los eventos de movimiento se instan-cia una subclase de Listener y se le asigna una instancia del controla-dor. Este controlador llama a la función establecida del Listener cuando el evento se activa.