Desarrollo de una interfaz de realidad virtual para videojuegos mediante una cámara Kinect

Emmanuel Rojas Fredini, Cristian Yones, Lucas Genzelis, Enrique Albornoz y César Martínez

Facultad de Ingeniería y Ciencias Hídricas, Universidad Nacional del Litoral, Santa Fe, Argentina
{erojasfredini,cyones77,lucas.genzelis}@gmail.com
{emalbornoz,cmartinez}@fich.unl.edu.ar
http://fich.unl.edu.ar/

Resumen Este trabajo presenta un método que permite identificar la información esqueletal de una mano en tiempo real. La mano es un objeto pequeño, que presenta muchas articulaciones, por lo que es altamente sensible a errores de segmentación. El reconocimiento de su estructura es, por lo tanto, un problema considerablemente desafiante. Para afrontarlo, se propone un algoritmo original que, utilizando el canal de profundidad del dispositivo Kinect, logra obtener resultados aceptablemente precisos en tiempo real, con muy bajo costo computacional. Finalmente, se utilizan estos resultados para presentar una interfaz de realidad virtual que ejemplifica un posible uso de la información obtenida.

Keywords: Reconocimiento esqueletal, realidad virtual, cámara de profundidad, Kinect, OpenKinect

1. Introducción

En la actualidad la industria de los videojuegos cuenta con un impulso importante. Mucho esfuerzo es dedicado al desarrollo de consolas y periféricos cada vez más avanzados, utilizando tecnologías cada vez más potentes. Uno de los elementos más notorios de los últimos desarrollos es el dispositivo Kinect para la consola XBOX 360. Dicho dispositivo se ha utilizado ya en numerosos juegos, y aun en trabajos de investigación, donde se han logrado cuantiosos resultados.
En el presente trabajo se ha pretendido explorar las funcionalidades de Kinect, utilizando la información de profundidad suministrada en un programa de aplicación, apoyándose en librerías Open Source. Específicamente, el objetivo propuesto fue que la aplicación reconozca la estructura de una mano, la posición y dirección de los dedos. Esta información puede utilizarse en diversas aplicaciones[1][3], como por ejemplo, para la interacción entre un usuario y una computadora.

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
E. Rojas Fredini, C. Yones, L. Genzelis, E. M. Albornoz & C. E. Martínez; "Desarrollo de una interfaz de realidad virtual para videojuegos mediante una cámara Kinect"
3er Workshop Argentino sobre Videojuegos (WAVi), pp. 17-24, 2012.

2. Justificación

El rastreo de objetos articulados en 3D, puntualmente de la estructura de la mano, es un problema ampliamente estudiado, que ya cuenta con varios métodos exitosos para la captura del movimiento. Algunos de estos métodos utilizan hardware especializado[6], pudiendo ser aprovechados únicamente en aplicaciones donde el costo y la complejidad del hardware no sean un limitante. También existe la posibilidad de utilizar marcadores visuales[8], con la desventaja de que éstos interfieren en la escena observada. Otra alternativa más reciente, utiliza la información de la cámara de profundidad de Kinect y resuelve luego un problema de optimización[4]. La desventaja clara de este último método es el alto costo computacional que conlleva.
En este trabajo, se propone un método que, utilizando sólo el canal de profundidad de Kinect, consigue resultados aceptablemente precisos en tiempo real, con muy bajo costo computacional. Este método tiene amplias ventajas en aplicaciones donde no es necesaria una gran precisión, como ser, por ejemplo, en la interacción entre un usuario y una computadora.

3. Materiales

El principal recurso utilizado en la realización del presente trabajo fue el dispositivo Kinect de Microsoft. Este dispositivo está compuesto por dos cámaras que operan a 30 fps, un emisor de rayos infrarrojos y un arreglo de micrófonos. La primer cámara se comporta como una cámara de video tradicional, proveyendo información RGB. La segunda cámara opera en la porción infrarroja del espectro electromagnético, capturando una grilla de puntos proyectados por el emisor de rayos, la cual se utiliza para calcular información de profundidad (i.e. distancia a la cámara).
Se utilizó la librería OpenKinect, la cual permite acceder, desde el programa desarrollado, a las matrices RGB y de profundidad. La precisión de dichos datos es de 8 bits por cada componente de color, y de 11 bits en el componente de profundidad. Los mismos se representan en matrices de 480 filas por 640 columnas. También fueron utilizadas las librerías OpenGL y CImg, con propósitos de visualización.
Por último, se utilizó la librería Bullet, como motor físico para la implementación de realidad virtual.

4. Metodología

En primer lugar, debe destacarse que, siendo éste un primer acercamiento, no se ha utilizado la información de color, basando todo el procedimiento exclusivamente en la información de profundidad. La razón detrás de esta simplificación es que al estar las dos cámaras en posiciones diferentes (separadas aproximadamente 10 cm), la información de un elemento en una posición dada de las matrices de color no corresponde necesariamente a la misma posición de

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
E. Rojas Fredini, C. Yones, L. Genzelis, E. M. Albornoz & C. E. Martínez; "Desarrollo de una interfaz de realidad virtual para videojuegos mediante una cámara Kinect"
3er Workshop Argentino sobre Videojuegos (WAVi), pp. 17-24, 2012.

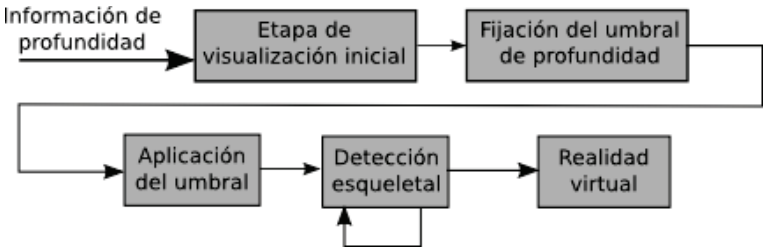

Figura 1. Diagrama en bloques del proceso

Figura 2. Paleta de color

la realidad que el elemento en la misma posición en la matriz de profundidad. La metodología desarrollada puede resumirse en el diagrama en bloques representado en la Figura 1. Se parte de una etapa de visualización inicial, en la que se muestra la captura de video. Luego se procede a la fijación del umbral de profundidad, donde se detecta el movimiento para permitir que el usuario indique su distancia a la cámara. Acto seguido, se procede a la aplicación del umbral, donde se procesa la imagen, mostrando sólo aquellos objetos en un rango dado alrededor de la distancia previamente calculada. Después se avanza a la etapa de detección esqueletal, donde se detecta el centro de la mano y las líneas centrales de los dedos. Finalmente, se pasa a la etapa de realidad virtual, donde se muestra una ilustración de un uso que puede hacerse de la información esqueletal.
A continuación, se explicará en detalle cada uno de estos puntos.

4.1. Etapa de visualización inicial

En esta etapa, simplemente se muestra la captura de la cámara de profundidad, aplicando la paleta de color que se observa en la Figura 2. Esto permite que el usuario ubique el dispositivo Kinect de forma que capture de forma deseable su posición. Un ejemplo de visualización puede verse en la Figura 3.

4.2. Fijación del umbral de profundidad

En esta parte, el usuario debe indicar su distancia a la cámara, de forma de poder calcular el valor del umbral a aplicar en la siguiente etapa. Sin embargo, para evitar al usuario la tediosa tarea de tener que medir su distancia exacta para luego ingresarla manualmente a la aplicación, el enfoque utilizado consiste en calcular este valor mediante la detección de movimiento. De esta forma, el usuario sólo debe pararse a la distancia deseada y realizar algún movimiento (por ejemplo, saludar), y el algoritmo desarrollado se encarga del cálculo de su distancia a la cámara.

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
E. Rojas Fredini, C. Yones, L. Genzelis, E. M. Albornoz & C. E. Martínez; "Desarrollo de una interfaz de realidad virtual para videojuegos mediante una cámara Kinect"
3er Workshop Argentino sobre Videojuegos (WAVi), pp. 17-24, 2012.

Figura 3. Ejemplo etapa visualización inicial

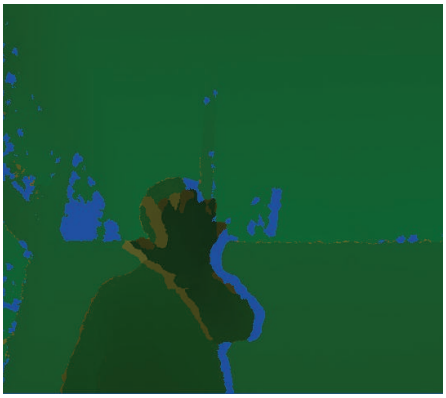

Figura 4. Ejemplo etapa fijación del umbral de profundidad

Para esto, se detecta el movimiento utilizando la diferencia entre las imágenes correspondientes a cuadros sucesivos. Se toman todas las profundidades de los pixels para los cuales el valor absoluto de la diferencia supera un cierto umbral, y se calcula la mediana de estas profundidades. Este valor calculado es el que se toma como umbral de profundidad.
Un ejemplo de lo que se visualiza en esta etapa puede verse en la Figura 4, donde se muestra en color naranja los pixels donde se ha identificado movimiento.

4.3. Aplicación del umbral

En esta etapa, se procesa la imagen de profundidad, asignando un color negro a todos los elementos cuya profundidad supera al umbral, y un valor de gris proporcional a la profundidad al resto de los elementos. Además, se aplica un

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
E. Rojas Fredini, C. Yones, L. Genzelis, E. M. Albornoz & C. E. Martínez; "Desarrollo de una interfaz de realidad virtual para videojuegos mediante una cámara Kinect"
3er Workshop Argentino sobre Videojuegos (WAVi), pp. 17-24, 2012.

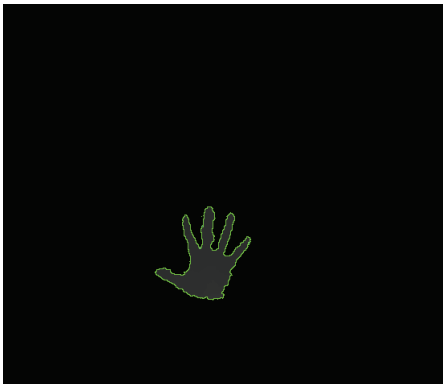

Figura 5. Ejemplo etapa aplicación del umbral

filtro pasa bajos de forma de eliminar las frecuencias más altas, correspondientes en general a ruido en la medición. Por último, utilizando un filtro pasa altos de suma cero se detecta el contorno de la imagen, el cual se muestra en la imagen con un color diferente, tal como puede observarse en la Figura 5.

4.4. Detección esqueletal

A partir de este punto, se asume que el único objeto que está a una distancia de la cámara menor al umbral, es la mano del usuario (o, en general, una porción de su antebrazo que incluye su mano).
Todos los pasos descriptos a continuación se ejecutan cíclicamente, para cada frame capturado, hasta que el usuario decide avanzar al punto final.
En primer lugar, se realiza el cálculo de la posición del centro de la mano, utilizando para esto un promedio de las posiciones de todos los pixels cuya profundidad es menor a la del umbral.
Luego, se arma una representación 1D de la mano como se explica en [2] o [5], pero utilizando una variación de la función de distancia al centroide, que aprovecha la información de la profundidad. El elemento de índice θ dentro de este vector v representa el peso de un segmento con origen en el centro de la mano, y con un ángulo θ (en grados sexagesimales) respecto a la horizontal. El peso de cada segmento se calcula según la Ecuación 1, donde (C_x, C_y) son las coordenadas del centro, (u_x, u_y) es un vector unitario con dirección θ y Max es la máxima cantidad de valores a considerar para cada segmento. Sólo se suma para aquellos elementos cuya profundidad es inferior a la del umbral.

$$v\left[\theta\right] = \sum_{i=0}^{Max} \frac{i}{profundidad\left[C_x + u_x i\right]\left[C_y + u_y i\right]} \tag{1}$$

Como se ve, el peso asignado a cada elemento es inversamente proporcional a su distancia a la cámara. Se procede de esta forma para dar mayor importancia a

los pixels que realmente son de la mano, en relación con los pixels del antebrazo, puesto que en general los primeros están más cerca de la cámara que los últimos. El ángulo θ que corresponde al segmento de peso máximo se toma como dirección principal de la mano. Luego de eso, se toma la porción de v correspondiente a 180º centrados en la dirección principal, y se truncan todos los elementos menores a la quinta parte del peso máximo.
Después se determina si la mano se está usando tiene el pulgar a la derecha o a la izquierda. Para ello, se calcula el ángulo medio pesando todos los ángulos que superaron el umbral, mediante la Ecuación 2.

$$media = \frac{\sum_i \theta_i v_i}{\sum_i v_i} \tag{2}$$

Luego, se compara el ángulo medio con el punto medio entre el máximo y mínimo ángulo que superaron el umbral. Si es mayor, se considera que el pulgar está situado del lado derecho, de otra forma se considera que está del lado izquierdo. En el algoritmo 1 se puede ver lo explicado anteriormente.

Algorithm 1 Detección esqueletal

$(C_x, C_y) = \frac{\sum_{i=0}^{Pixeles} (x_i, y_i)}{Pixeles}$
for $\theta \in [0, 360]$ **do**
  $v[\theta] = \sum_{i=0}^{Max} \frac{i}{profundidad[C_x + u_x i][C_y + u_y i]}$
end for
$\theta_{DirMano} = \arg\max_\theta (v[\theta])$
$\left(\theta', v'\left[\theta'\right]\right) = \{ \ (angDif((\theta_{DirMano} + 90) \ mod \ 360, \theta_i), v[\theta_i]) \quad | \ angDif(\theta_{DirMano}, \theta_i) \leq 90 \ \wedge \ v[\theta_i] \geq \frac{v[\theta_{DirMano}]}{5} \ \forall \ i \ \}$
if $\theta_{PosMed} > \frac{\arg\max_\theta(\theta') - \arg\min_\theta(\theta')}{2}$ **then**
  $KMeansPesado\left(\theta', ManoDerecha\right)$
else
  $KMeansPesado\left(\theta', ManoIzquierda\right)$
end if

Finalmente, se aplica una variante pesada del algoritmo k-means[7]. De esta forma se encuentran cinco centros de gravedad sobre la distribución de los valores truncados. La información respecto del pulgar se utiliza para fijar valores iniciales, mejorando con ello la velocidad de convergencia.
En la Figura 6 puede observarse una captura de la información esqueletal de la mano, con su correspondiente función $v[\theta]$, donde los máximos de k-means se han graficado en color verde.

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
E. Rojas Fredini, C. Yones, L. Genzelis, E. M. Albornoz & C. E. Martínez; "Desarrollo de una interfaz de realidad virtual para videojuegos mediante una cámara Kinect"
3er Workshop Argentino sobre Videojuegos (WAVi), pp. 17-24, 2012.

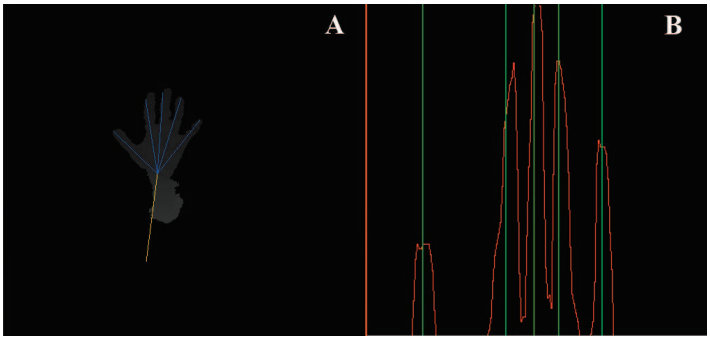

Figura 6. A) Etapa detección esqueletal. B) Vector v del esqueleto

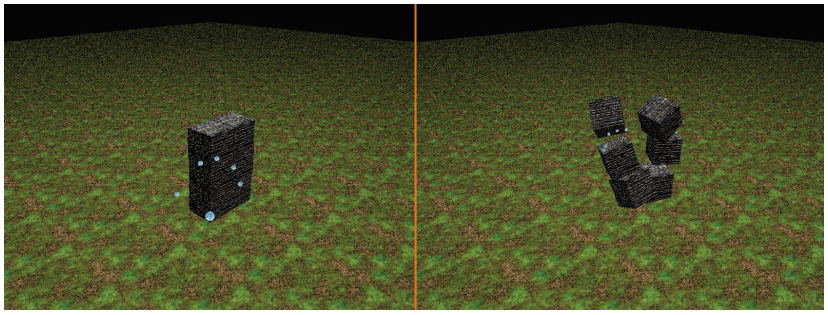

Figura 7. La mano en el espacio virtual imita a la mano del usuario y permite interactuar con el entorno

4.5. Realidad virtual

Se desarrolló una aplicación de realidad virtual, para mostrar cómo podría utilizarse la información obtenida previamente para un uso concreto. Esta aplicación permite controlar una mano virtual en un espacio tridimensional e interactuar con objetos del entorno.
Básicamente, la mano se representa mediante una esfera ubicada en el centro de la mano, y cinco esferas más pequeñas ubicadas una en el extremo de cada dedo. Además, se presenta una pila de bloques con los que es posible interactuar mediante las esferas mencionadas. En la Fig. 7 se ilustra la aplicación desarrollada.

5. Conclusiones y trabajos futuros

Como ha sido ilustrado en cada paso de la metodología expuesta, el método propuesto permite obtener buenos resultados, sin importar la posición o inclinación de la mano del usuario, o aun la posición relativa de sus dedos.

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
E. Rojas Fredini, C. Yones, L. Genzelis, E. M. Albornoz & C. E. Martínez; "Desarrollo de una interfaz de realidad virtual para videojuegos mediante una cámara Kinect"
3er Workshop Argentino sobre Videojuegos (WAVi), pp. 17-24, 2012.

Además, debido a que la información obtenida es capturada en el espectro de luz infrarroja, el método desarrollado es independiente de las condiciones de iluminación ambiente, lo cual presenta una ventaja importante respecto de los métodos tradicionales basados sólo en información de color.
En el futuro, podría utilizarse la información RGB para mejorar el método propuesto. Para esto deberían utilizarse transformaciones geométricas que permitan encontrar la correspondencia de los elementos de las matrices de color con los de la matriz de profundidad.
Además, podrían agregarse articulaciones adicionales al esqueleto, de forma de aumentar el nivel de detalle de la información calculada. Dentro de esto, además, podrían considerarse restricciones adicionales que ayuden a la correcta identificación (es decir, considerar que hay posiciones y movimientos imposibles).
Por otra parte, podrían utilizarse técnicas de segmentación para identificar cuándo se están utilizando dos manos en lugar de sólo una, y adaptar el algoritmo desarrollado para que trabaje con ambas manos a la vez.

Agradecimientos

Los autores agradecen al Dr. Néstor Calvo, quien proveyó del dispositivo Kinect que se utilizó durante el desarrollo del trabajo realizado.

Referencias

1. Ali Erol, George Bebis, Mircea Nicolescu, Richard D. Boyle, and Xander Twombly. Vision-based hand pose estimation: A review. In *Computer Vision and Image Understanding*, 2007.
2. Hannu Kauppinen, Tapio Seppänen, and Matti Pietikäinen. An experimental comparison of autoregressive and fourier-based descriptors in 2-d shape classification. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1995.
3. Thomas B Moeslund, Adrian Hilton, and Volker Kru. A survey of advances in vision based human motion capture and analysis. In *Computer Vision and Image Understanding*, 2006.
4. I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *BMCV*, 2009.
5. Z. Ren, J. Yuan, and Z. Zhang. Robust hand gesture recognition based on finger-earth movers distance with a commodity depth camera. In *ACM MM*, 2011.
6. Mark Schneider and Charles Stevens. Development and testing of a new magnetic-tracking device for image guidance. In *SPIE Medical Imaging*, 2007.
7. George C. Tseng. Penalized and weighted k-means for clustering with scattered objects and prior information in high-throughput biological data. In *Bioinformatics*, 2007.
8. Robert Y. Wang and Jovan Popoví. Real-time hand-tracking with a color glove. In *ACM Transactions on Graphics*, 2009.