UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Comparación entre puntero infrarrojos y puntero láser, y creación de ejemplo de interacción mediante OpenCV

Apellidos, Nombre	Navarro Huerta, Álvaro (lvanahue@inf.upv.es)
Departamento	Estudiante Ingeniería de Computadores
Centro	Universidad Politécnica de Valencia

Table of Contents

Resumen ideas clave

En este articulo pretendemos comparar la interacción producida mediante el uso de un puntero LED y de un mando de luz infrarroja sobre el escritorio con el fin de comprobar cual de las dos resulta mejor para desarrollar, mediante el uso de librerías de OpenCV. Emplearemos la versión de OpenCV 2.3.1, la desarrollaremos en un entorno de maquina virtual GNU/Linux sobre el entorno Virtual Box v5.0.16.

Introducción

Este articulo mostrara como crear y realizar una interacción con el usuario mediante el uso de un mando. El usuario empleará un mando para pasar al computador la información, este recogerá esta información y la procesará mostrándola por la pantalla una sucesión de puntos.

Analizaremos el análisis de la información de los mandos, así como la forma en la que trabaja OpenCV, en cuanto a la adquisición y procesado de imágenes.

Parte de este documento se inspira en código adaptado de un trabajo[1] de la asignatura IMD,

Objetivos

Los objetivos tras leer este documento serán los siguientes:

- Iniciar el entorno de desarrollo
- Instalar las librerías de OpenCV
- Comparar las señales emitidas por los mandos LED e infrarrojos
- Diseñar un breve programa mediante interacción con OpenCV

Instalación del entorno de trabajo

En primer lugar voy a hacer una breve introducción para explicar como configurar mi entorno de trabajo. Para descargar Virtual Box hay que ir a la pagina de Oracle Virtual Box. De todas las versiones, es mejor descargar la mas actual(v5.0.16) ya q versiones anteriores a 4.1.36 no disponen de acceso a la

webcam.
A continuación debemos descargar la maquina virtual en la que trabajaremos. La maquina que yo elegí es Ubuntu 12.04. El proceso para asignar la maquina al entorno Virtual Box es intuitivo. Abrir Virtual Box ->Agregar maquina, y seleccionamos nuestra maquina. Para configurar la cámara hay que habilitar los puertos USB y añadir la entrada de webcam. Una vez dentro del entorno Ubuntu. En el menú Dispositivos -> añadir webcam. Este ultimo paso habrá que realizarlo cada vez.

Configuración de librerías OpenCV

Una vez tenemos el entorno y la cámara. Debemos agregar las librerías de OpenCV. Para ello hay que escribir el la consola: *sudo apt-get install opencv-libcdev*. Las librerías instaladas son la versión 2.3.1, en caso de no disponer de estas, hay alguna posibilidad de que sea necesario modificar alguna función. De este modo nuestras librerías estarán configuradas correctamente para que podamos comenzar con nuestro código.

Desarrollo en OpenCV

En nuestro código encontramos diversas funciones aparte del main, en las que entraremos en detalle mas adelante.
Comenzando por el main() encontramos la función *cvCaptureFromCAM(0);* que captura la imagen de la cámara, el valor de la función depende de la cámara que pretendamos usar en caso de disponer mas de una.
Lo siguiente que hacemos es capturar una imagen de la pantalla y creamos las tres escenas que mostraremos. Para el correcto funcionamiento es importante añadir la funcion *cvWaitKay*, para que permita la devolución y manejo de eventos.
Modificamos la primera escena con la función *cvCvtColor(imgOrg,imgGris,CV_BGR2GRAY);[2]* para pasarla a imagen de grises. Esta escena la umbralizamos para obtener los puntos donde más brillo se obtiene, si hemos umbralizado con valores elevados solo deberíamos obtener el valor de nuestro puntero.
En este documento hemos querido comparar las tecnologías de infrarrojos y el puntero láser mediante el atributo de umbral usado en las imágenes. En nuestro código hemos umbralizado usando la función *cvThreshold (imgGris, imgUmbral, valorUmbral, 255, CV_THRESH_BINARY);*. Una comparativa se encuentra en la siguiente tabla:

Valor umbral	Puntero láser	Mando infrarrojo	Entorno sin luz
150	No funciona	No funciona	El entorno es demasiado brillante
175	Si funciona	Si funciona	Funciona dependiendo de la posición.
225	Si funciona	Si funciona	El punto del infrarrojo es mayor
251	Si funciona	Si funciona	Posición apenas visible

Tabla1.Comparación entre tecnologías

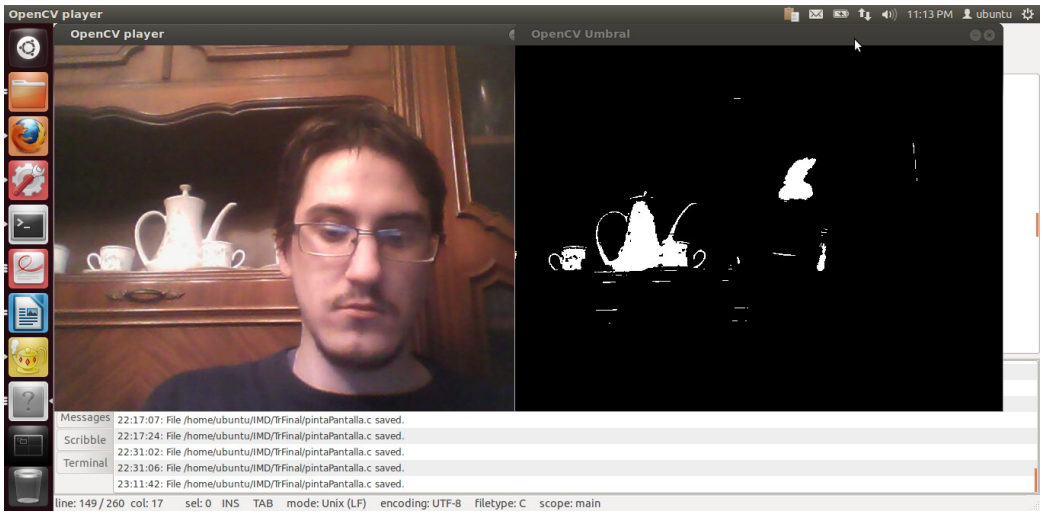

Imagen1.Valor umbral bajo(150)

Como se puede intuir por la gráfica, los valores de umbral por debajo de 175 no permiten ningún tipo de interacción con la maquina mediante puntero(Imagen1). Mientras que los valores por encima pueden responder como deseamos. Por otro lado confirmamos que nuestro sistema no funciona correctamente al emplear cualquier tipo de luz natural, el brillo que emite impide que la cámara capture correctamente nuestras señales. Así que podemos definir unas condiciones para el uso de los dispositivos, estas serian un entorno sin luz natural y un valor umbral elevado(Imagen2). Preferiblemente el entorno será poco luminoso.

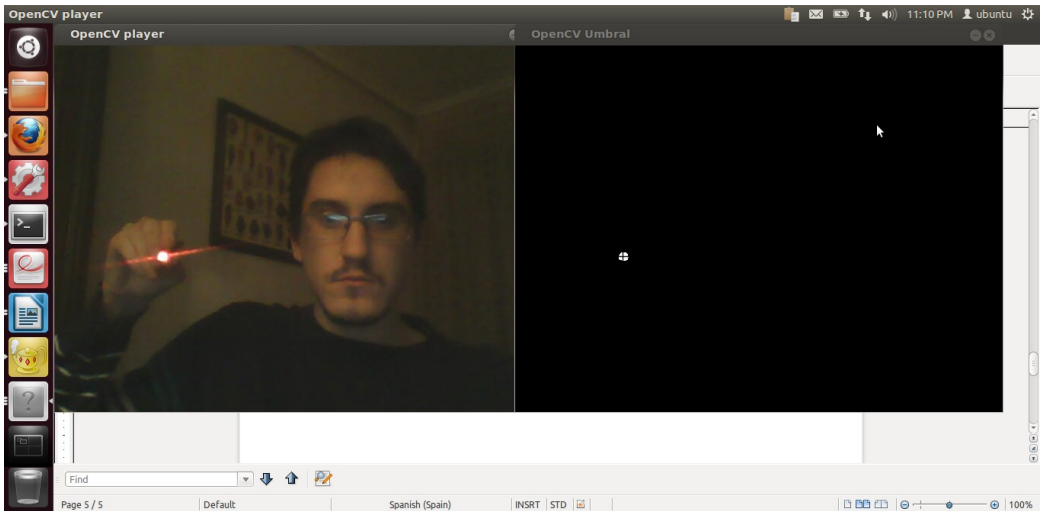

Imagen2. Valor umbral normal(225)

Comparándolos sin luz natural. No observamos mucha diferencia entre la captación de señales mediante ambos dispositivos. Sin embargo al compararlos mediante la distancia a la cámara receptora comprobamos como el mando infrarrojo emite un punto de luz mas amplio en distancias cortas, que disminuye de manera notable cuanto mas se aleja. Mientras que el puntero láser permanece con un grosor mas constante durante todas las distancias.

Desarrollo programa prueba

Luego de haber comprobado que el uso dispositivo no tiene relevancia sobre nuestro código. pasamos a crear un proceso de interacción que incluya una de la tecnología antes probada,notros elegimos el puntero.
De ese modo hemos creado añadidos en nuestro código. Nos hemos propuesto crear una aplicación que cree un sistema de tipo el videojuego Snake. Como no es el objetivo de nuestro documento el diseño de algoritmo para poner puntos, así como para simplificar el código generado, hemos asignado tres puntos arbitrarios en pantalla el usuario podrá interaccionar mediante puntero láser guiando a la serpiente hasta los puntos.
Para comenzar comentaremos las funciones auxiliares del código creado.
Las funciones crear lista, asignarALista y recorreLista son funciones que recorren la lista anidada de puntos que nos muestra la cámara por la pantalla. Estas funciones has sido adaptadas a partir del código de una web[2].

La función recorreSerpiente(...) ubica en la pantalla los punto mostrados por la lista enlaza creada anteriormente.

Las función centroGravedad(...) encuentra la zona donde se encuentran los puntos blancos no umbralizados y elige el punto en la posición central, y la función pintaCruz(...) genera una cruz en el punto hallado por centroGravedad(...).

La función generaPuntoObjetivo(...) inicializa un punto en el listado de puntos objetivos, mientras que la función ponPuntoObjetivo(...) es la encargada de situar el punto en la pantalla, además de encargarse de la aparición y desaparición de puntos.
La función objetivoPintado(...) es la función que comprueba la proximidad de la serpiente o puntero al objetivo, si esta lo bastante cerca devolverá un 1 y pondrá el punto en estado inalcanzable. Si no devolverá un 0.
La ultima función empleada pintameeeTexto(...) esta relacionada con la funcionalidad de la aplicación, muestra en pantalla el texto de puntuación actual y lo actualiza a medida que se incrementa la puntuación.

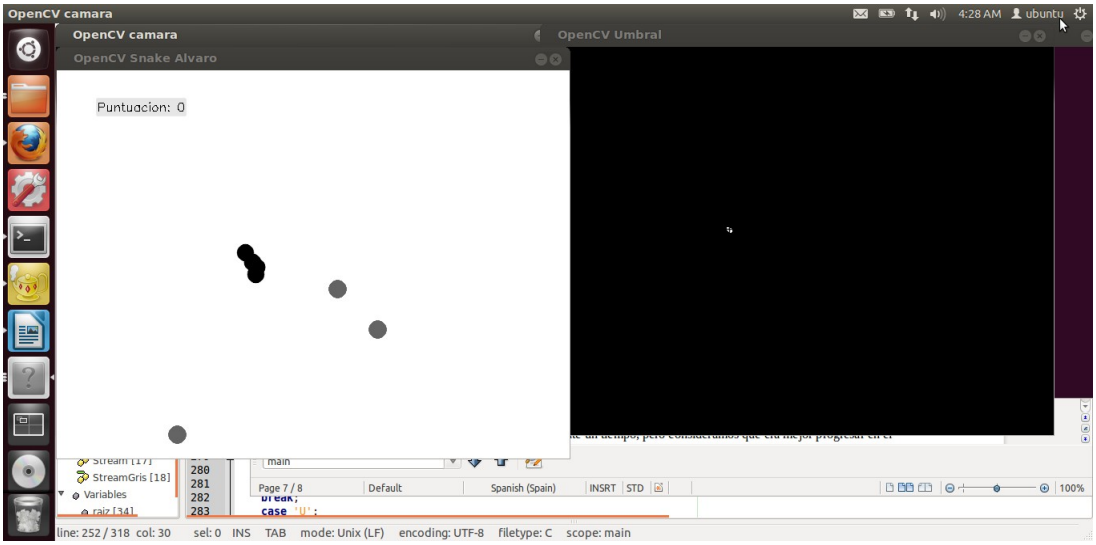

Imagen3.Muestra de la aplicación

A continuación expondremos como desarrollamos la aplicación en la función main(...), ya hemos expuesto lo que realiza en el comienzo antes de entrar al bucle principal, en el bucle realizamos las mismas funciones de cvCvtColor(...) y Threshold(...), pero para que sea mas intuitivo para el usuario se añade la función cvFlip(...), con la finalidad de ser mas intuitivo de usar para el usuario.
Usaremos la funciones de centroGravedad(...) y pintaCruz(...) para establecer el punto en la imagen umbralizada, comprobamos si nuestro punto esta cercano a uno objetivo con objetivoPintado(...) y añadimos el punto a la lista con asignarALista(...). Por ultimo volvemos a pintar los puntos en la pantalla si el valor de reaparece en los puntos objetivos es mayor de 0.
Por ultimo se pinta sobre la pantalla la puntuación obtenida, de este modo, la puntuación queda por encima del resto de puntos y es siempre visible.(Imagen3)

Conclusiones

En este documento hemos expuesto una comparación entre los mecanismos de puntero láser y mando infrarrojo, los cuales tienen una intensidad aproximada en las distancia corta, sin embargo al alejarnos el puntero láser resulta mas brillante, pues el infrarrojo pierde intensidad. Por otro lado se ha mostrado un breve ejemplo del juego Snake para su interacción mediante los medios antes propuestos.

Como una opción de mejora puede ser el desarrollo de un algoritmo para que aparezcan puntos aleatoriamente en el mapa.

Bibliografía y referencias

Solo se han empleado referencias electrónicas

[1]<Interacción con OpenCV: detección de movimiento para realizar un instrumento virtual con OpenCV + OpenAL >
https://www.riunet.upv.es/handle/10251/12684

[2]<OpenCV wiki>
http://www.docs.opencv.org/2.3/

[3]<Algoritmos de listas enlazadas>
http://www.thegeekstuff.com/2012/08/c-linked-list-example/