Índice general

Índice de cuadros

Índice de figuras

Capítulo 1

Introducción

El presente trabajo muestra el desarrollo de un sistema *multi-touch* para crear figuras básicas bidimensionales.

La interacción con el sistema contará con objetos físicos que representan herramientas para dibujar, además se cuenta con otros objetos como herramientas básicas para la edición del dibujo.

El desarrollo del sistema está enfocado en la integración de tecnologías, en éste caso particular se utilizará la herramienta *Kinect*™de *Microsoft*®conectada a una computadora personal, además el sistema contará con reconocimiento de patrones para la selección de herramientas de dibujo o de edición del mismo.

1.1. Kinect™

Kinect™es un dispositivo que combina una cámara *RGB*, un sensor de profundidad y un arreglo de micrófonos[**?**]. En el Cuadro 1.1 se listan los elementos principales de *Kinect*™junto con su función.

Elemento	**Función**
Arreglo de micrófonos	Detecta las voces y las aísla del ruido ambiental
Proyector de luz infrarroja	Dispara luz infrarroja.
Sensor de profundidad cámara infrarroja	Detecta la luz que lanza el proyector de luz infrarroja y genera un canal extra al *RGB* que trae información de la profundidad de la escena y es similar a un mapa de disparidad estéreo.
Motor de inclinación	Ajustar hacía donde están dirigidas las cámaras.
Salida de adaptador *USB* 2.0	Conectar un cable para conectar vía *USB* el dispositivo.
Cámara *RGB*	Reconocer los tres colores básicos. Rojo, verde y azul.

Cuadro 1.1: Elementos Principales de *Kinect*™[**?**].

Además cuentan con:

- *Triple Core PowerPC* 970*,* 3.2*GHz, Hyperthreaded,* 2 *threads/core.*
- 500 *MHz ATI graphics card.*
- 512 *MB RAM.*

1.2. OpenNI

Se utiliza *OpenNI framework (Open Source Natural Interaction)* versión 1.3.4.6 que es una *API* desarrollada por *OpenNI organization*, que provee una interfaz para dispositivos que ofrecen una interfaz natural para operar los componentes del *middleware*[**?**], en nuestro sistema lo utilizamos para poder comunicar la computadora personal con *Kinect*^{TM}y así utilizar de la mejor manera para lo que se diseño el dispositivo *Kinect*^{TM}, que es eliminar los dispositivos físicos para controlar, en este caso, la computadora personal.

Posee una biblioteca de código abierto para poder trabajar con casi todas las capacidades de *Kinect*^{TM}en *Windows*, *Linux* y *Mac*.

1.3. OpenCV

OpenCV (Open Source Computer Vision Library) Es una biblioteca *GPL*, orientada a la computación visual en tiempo real, utilizada principalmente en el campo de la Interacción Computadora - Humano por sus siglas en inglés *HCI (Human Computer Interaction)* creada y soportada por *Intel* desde 1999, y tiene amplia documentación.

Actualmente trabajamos con la versión 2.3.1 de la librería *OpenCV*.

Una de las características más importantes de *OpenCV* es que las funciones están totalmente optimizadas para los procesadores de arquitectura *Intel*. Existen versiones para diferentes arquitecturas de procesadores y para diferentes sistemas operativos[**?**].

1.4. Reconocimiento de Patrones

Para intentar definir que es el Reconocimiento de Patrones (RP) definiremos algunos conceptos[**?**].

Reconocimiento: Proceso de clasificación de un objeto en una o más clases.

Objeto: Es un concepto con el cual representamos los elementos sujetos a estudio. Pueden ser concretos o abstractos.

Patrón: Tras los procesos de segmentación, extracción de características y descripción, cada objeto queda representado por una colección (posiblemente ordenada y estructurada) de descriptores.

Clase: Es un conjunto de objetos. Al agrupa en clases, se puede hacer de dos formas distintas:

- *Por pertenencias duras:* Un objeto pertenece o no a una clase.
- *Por pertenencias difusas:* Los objetos pertenecen parcialmente a una clase. Existen clases con intersecciones no vacías.

Existen varios intentos para definir al reconocimiento de patrones.

- La disciplina dedicada a la clasificación de objetos y el pronóstico de fenómenos.[**?**]
- Rama del conocimiento, de carácter multidisciplinario, cuyo objeto de estudio son los procesos de identificación, caracterización, clasificación y reconstrucción sobre conjuntos de objetos o fenómenos, así como el desarrollo de teorías, tecnologías y metodologías relacionadas con dichos procesos.[**?**]

- Es la ciencia que se ocupa de los procesos sobre ingeniería, computación y matemáticas relacionados con objetos físicos y/o abstractos, con el propósito de extraer información que permita establecer propiedades de o entre conjuntos de dichos objetos.[**?**]

Con respecto al reconocimiento de patrones, se cuenta con objetos físicos los cuáles hacen la función de botones permitiendo eliminar éstos de la interfaz, éstos objetos tienen asignada una imagen binaria que denominamos *tag*(etiqueta), así la cámara de *Kinect*$^{\text{TM}}$captura la imagen del escenario, entonces el sistema procesa esa captura donde si se encuentra uno de los objetos físicos se decodifica la imagen y se activa la acción que se tenga asignada a dicha *tag*. Algunas de las *tags* tienen más de una acción que se activan con un giro en cierto ángulo esto hace que cambie la función que al principio tenia designada.

1.5. Etiquetas (Tag´s)

Denominamos *tag's*(etiquetas) a un conjunto de imágenes binarias en colores blanco y negro que indica el uso de una herramienta.

Analizamos tres tipos de imágenes binarias candidatas, las cuales son: *reacTIVision fiducials*, Código *QR* y *AR-ToolKit Marker*.

1.5.1. reacTIVision

Son marcadores especiales (Figura 1.1) que se desarrollaron en conjunto con el sistema *reacTIVision* que es un *software* creado especialmente para el rastreo de éstos marcadores que a grandes rasgos son imágenes binarias, específicamente, en blanco y negro.

Figura 1.1: Ejemplos de marcadores para reacTable.

Para la identificación de cada marcador se basa en una región grafica adyacente y los rectángulos de delimitación. El método combina coincidencias de patrones binarios de gráficos topológicos (Figura 1.2) para el reconocimiento y la identificación con simples técnicas geométricas para calcular la ubicación y orientación de los marcadores[**?**].

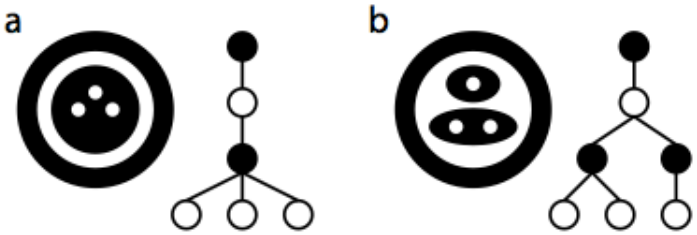

Figura 1.2: Algunas simples topologías y su correspondiente gráfico de región adyacente.

Se utilizan algoritmos genéticos para la identificación de cada marcador, para más detalles consultar[**?**].

Éstos marcadores están disponibles en PDF para su impresión así como el sistema *reacTIVision* en la página del proyecto[**?**] y no es necesario producir nuevos marcadores.

1.5.2. Código QR

El código de barras de respuesta rápida por sus siglas en inglés *QR code** (*Quick Response Barcode*, Figura 1.3) es un sistema que permite almacenar información en un código de barras bidimensional, esto quiere decir que tiene un patrón de arriba hacía abajo, de izquierda a derecha, y puede almacenar alrededor de 7,000 dígitos (véase el Cuadro 1.2) mucho más que un código de barras convencional, además con la ayuda de una cámara y un programa especial podemos recuperar la información de cada código. Éste código esta estandarizado *ISO/IEC 18004*.

Figura 1.3: Ejemplo de Código QR.

Numérico	Máximo 7,089 caracteres.
Alfanumérico	Máximo 4,296 caracteres.
Binario	Máximo 2,953 caracteres.
Kanji/Kana	Máximo 1,817 caracteres.

Cuadro 1.2: Capacidad de datos del código QR.

Existen versiones del código QR desde la 1 hasta la 40 y cada una tiene diferentes números de módulos (módulo se refiere a los puntos blancos y negros que conforman el código QR)[**?**].

* QR code es una marca registrada por DENSO WAVE INCORPORATED

Tiene la capacidad de corrección de errores (véase el Cuadro 1.3), si una parte del código está dañada, manchada o doblada puede ser interpretado de igual forma.

QR Code Error Correction Capability	
Level L	Approx.7 %
Level M	Approx.15 %
Level Q	Approx.25 %
Level H	Approx.30 %

Cuadro 1.3: Capacidad de corrección de errores Código QR[**?**].

La decodificación del código *QR* puede seguir varios algoritmos a continuación se describe un algoritmo general que puede utilizarse para algunos código de barras en 2D.

1. Binarización de la imagen.
   Método de Otsu[**?**].
2. Corrección de la inclinación.
3. Corrección geométrica de la imagen.
4. Obtención de los cuatro vértices de la imagen.
5. Obtención de los nuevos valores de los vértices.
6. Obtener el valor en cada nuevo pixel.
7. Normalización de la imagen.

Existen distintos sistemas que ofrecen la creación de códigos *QR* así como la decodificación del mismo como *ZXing*[**?**].

1.5.3. ARToolkit

Son plantillas de forma cuadrada, que se componen de cuadrado negro con un cuadrado blanco cuatro veces más pequeño que su centro y un dibujo sencillo en el interior del cuadrado blanco, como se muestra en la figura 1.4.

Figura 1.4: Ejemplo de ARToolKit Marker.

Para identificación de la platilla está basada en la detección de las esquinas con ayuda del algoritmo de *fast pose estimation*[**?**]. Los pasos para el tratamiento de estas plantillas son los siguientes:

1. La imagen capturada se transforma a una imagen binaria.
2. Identificamos el marco de color negro.
3. Extraemos los patrones del dibujo que se encuentra en el interior del marco negro.
4. Almacenamos los patrones.
5. Repetimos los primeros tres pasos.
6. Comparamos los patrones extraídos con los almacenados.
7. Aplicamos funcionalidad de la imagen.

1.6. Metodología

El desarrollo del proyecto se realizó aplicando el modelo incremental.[**?**] Esta metodología tiene la ventaja de ser dinámica y flexible, además permite usar las salidas de las etapas precedentes, como entradas en las etapas sucesivas y facilita corregir cualquier error detectado o llevar a cabo mejoras en los distintos productos que se generan a lo largo de su aplicación[**?**].

Esta metodología, se basa en la metodología en cascada. El uso de esta metodología dentro del desarrollo del proyecto proporcioná:

- Definición de actividades a llevarse a cabo en el tiempo de realización del Trabajo Terminal.
- Unificación de criterios en la organización para el desarrollo del proyecto.
- Puntos de control y revisión.
- Seguimiento de secuencias ascendentes o descendentes en las etapas del desarrollo.
- Cumplimiento de etapas o fases en paralelo, por lo que es más flexible que la estructurada.

1.6.1. Paradigma

El paradigma será Orientado a Objetos, porque la *API* usada de *OpenCV* está en lenguaje *C++*, que permite la manipulación de objetos, ya que primero definen objetos, para luego enviarles mensajes solicitándoles que realicen sus métodos por sí mismos.

El uso del paradigma proporciona:

- No modela la realidad, sino la forma en que las personas comprenden y procesan la realidad.
- Es un proceso ascendente basado en una abstracción de clases en aumento.
- Se basa en identificación de objetos, definición y organización de librerías de clases, y creación de macros para aplicaciones específicas.
- Utiliza menor cantidad de código.
- Es reutilizable.

1.7. Objetivos

1.7.1. Problemática

Ofrecer una alternativa al teclado y mouse con una interfaz de usuario natural donde ya no se interactué directamente con un dispositivo electrónico, junto con esto querer desarrollar algún sistema que probara que se podía utilizar *Kinect*™junto con la computadora personal.

1.7.2. General

Desarrollar un sistema con una interfaz de usuario natural, asistido con la herramienta de entretenimiento *Kinect*™y reconocimiento de patrones.

1.7.3. Particulares

- Lograr una configuración para usar *Kinect*™con la computadora personal, para desarrollar un sistema.
- Implementar una interfaz sin involucrar dispositivos electronicos como una pantalla tactil.
- Eliminar uso del *mouse* en nuestro sistema.
- Eliminar uso del teclado en nuestro sistema.

1.8. Justificación

Una de las principales características sobre la dificultad del desarrollo de los sistemas *multi-touch* basado en tecnología "reciente"en el caso de *Kinect*™era la falta de documentación fiable al momento de plantear este proyecto (Octubre - Diciembre 2011) ya que no se contaba con drivers capaces de explotar todas las características de *Kinect*™ni un entorno de desarrollo estable por parte de *MS* o de la comunidad de *software* libre. El desarrollo de éste sistema busca contribuir a la creación de documentación formal que permitirá que futuras generaciones tengan mayor cantidad de fuentes fiables y por lo tanto se interesen por la creación de sistemas basados en movimientos, logrando ser un aportador más al crecimiento de dichos entornos.

Además, permitir que los alumnos de la Escuela Superior de Cómputo que se encuentren cursando o tengan interés en el reconocimiento de patrones o semejantes, trabajen con los actuales dispositivos de captura de imagen, siendo en nuestro caso *Kinect*™de *Microsoft*®, dejando a un lado su complejidad y crear un mayor interés, buscando cambiar el enfoque de dicha herramienta en donde el alumno no la vea como un proyecto de Trabajo Terminal sino como prácticas semestrales, lo que brindará mayor competitividad e integración de nuevas tecnologías.

Esta integración de tecnologías ofrece una alternativa al *mouse* y al teclado pudiendo así evitar enfermedades ya conocidas causadas por éstos dispositivos como lo es el síndrome de túnel carpiano[**?**].

1.8.1. Estado del arte

Actualmente no se cuenta con un sistema de dibujo como el que se pretende realizar, a la fecha de la documentación del estado del arte (Marzo 2012) existen otros trabajos que también manejan *Kinect*™*, OpenCV y OpenNI*, elementos con los que se llevará acabo el desarrollo de nuestro sistema, de los cuales vamos a mencionar algunos a continuación.

Gracias a la aparición de *drivers* que permiten la interacción entre el dispositivo *Kinect*™, que primeramente era exclusivo para la consola de videojuegos *Xbox* 360 de *Microsoft*®, y la computadora, se comenzaron a realizar aplicaciones que permiten al usuario tener una interacción más natural, permitiendo ser ellos mismos el control de la aplicación. Debído a que era una tecnología "reciente"existía poca documentación formal acerca de proyectos relacionados.

Hand Tracking - Kinect with OpenCV 2.2 and OpenNI

Es una aplicación sencilla que en primera instancia reconoce la mano de un usuario, como un punto permitiendo realizar trazos a mano alzada (Figura 1.5), la aplicación puede cambiar el punto con que se realizar el trazo entre una mano y otra juntando las manos para volverlas a separar así queda realizado el cambio. Esta aplicación también permite la identificación del cuerpo (Figura 1.6) con lo que se toman a ambas manos como puntos de interés, uno de ellos se encarga de realizar el trazo y con el otro se puede seleccionar el color.

La relación que existe entre este trabajo y el que se pensó realizar es que ambos deben poder generar dibujos identificando un punto de interes con el cual se van a hacer los trazos además de seleccionar el color y agregar otra

Figura 1.5: Hand Tracking.

funcionalidades. Éste sistema no cuenta con una documentación y lo único que se puede obtener es lo que se visualiza en un video subido a la red[**?**].

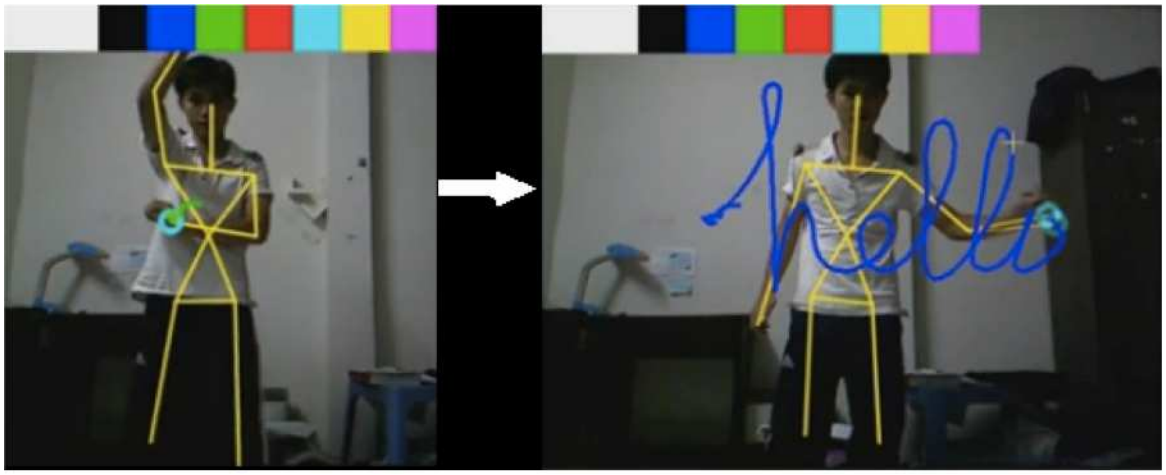

Figura 1.6: Reconocimiento del cuerpo.

Kinect Active Projection Mapping

Es una aplicación que trabaja con *Kinect*^{TM}*, OpenCV y OpenNI*, además comparte la idea de mantener un área de trabajo y una proyeción, de tal modo el usuario interactúa directamente sobre el área designada (Figura 1.7).

En este proyecto se utiliza el kinect para reconocer el cuerpo, la posición de las manos principalmente. El sistema crea un efecto visual en sobre las manos y entre ellas por medio de una imagen que es proyectada sobre una pantalla detras de el usuario[**?**].

Esta aplicación ha sido desarrollada en el *Computer Fusion Laboratory* como parte del programa de ingenieria de la *Temple University*. La página donde se dan más detalles del proyecto se encuentra aún en construcción. Y por el momento los recursos no están disponibles.

Figura 1.7: Proyección de imágen sobre un área de trabajo.

Aldebaran Nao Kinect Controller

Es un proyecto donde se controla por medio de *Kinect*™a un robot (Figura 1.8), organismo autónomo programable y de mediana estatura desarrollado por la empresa Francesa *Aldebaran Robotics*. Esta aplicación ha sido desarrollada en *Technical University Bergakademie Freiberg* en Alemania por Erik Berger y Heni Ben Amor. Existe información adicional de este proyecto en la página oficial de la universidad [**?**], Pero se encuentra en idioma Alemán.

Este proyecto no tiene mucha relación en cuanto a la funcionalidad del trabajo que se desea realizar, pero se considera por el hecho de también emplear los elementos que utilizaremos en nuestro sistema.

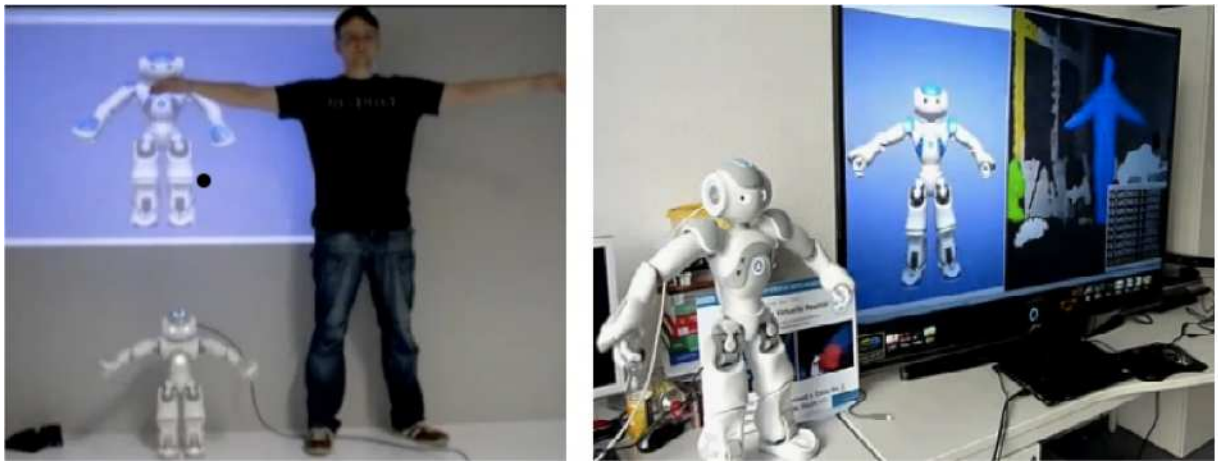

Figura 1.8: Interacción con robot automáta programable.