Prototipo traductor de señales manuales a texto legible, utilizando Kinect

Translator prototype of hand signals a text using kinect

Claudia Patricia Rodríguez Rocha[1]*; Jhon Alexander Pineda Arias[2]*; Diego Sánchez Villamil[3]*

[1]*Ingeniera de Sistemas, Especialista en Base de Datos y Gerencia educativa; Experiencia en docencia e investigación. Coodinadora del semillero de investigación SICORVIR Fundación Universitaria de San Gil - UNISANGIL. Chiquinquirá, (Boy.), Colombia. *crodriguez2@unisangil.edu.co*

[2]*Ingeniero de Sistemas, egresado de UNISANGIL, Sede Chiquinquirá. Chiquinquirá (Boy.), Colombia. *jhonalex@unisangil.edu.co*

[3]*Ingeniero de Sistemas, egresado de UNISANGIL, Sede Chiquinquirá. Chiquinquirá (Boy.), Colombia. *diegofabian@unisangil.edu.co*

Fecha de recepción del artículo: 05/03/2013 Fecha de aceptación del artículo: 29/10/2013

Resumen

El prototipo desarrollado es capaz de tomar los puntos de una escena captada por medio del sensor de profundidad de Kinect, aplicar un filtro para eliminar datos innecesarios y posteriormente mostrar el Mesh o maya que reconstruye la imagen de la escena final en 2D, estableciendo la diferencia de distancias con el cambio de color. El aplicativo detecta la localización de la o las manos en el Mesh, luego toma imágenes del gesto manual y las compara con imágenes almacenadas en archivo; estas cuentan con la traducción respectiva. Si encuentra una imagen con alto grado de coincidencia se devuelve el texto correspondiente. Los resultados de la investigación permiten reconocer las fases, librerías y plataformas óptimas para el procesamiento de imágenes, implementado en el prototipo.

Palabras clave

Kinect, Nubes de puntos, OpenCV, Procesamiento de imágenes, Señas manuales, PLC.

Abstract

The prototype developed it is able for taking the scene points captured through the depth Device of Kinect, apply a screening for delete unnecessary piece of information and after, shows the mesh (something that rebuilds the picture of the final scene in 2D), establishing the difference of distances with the color change. This detects the location of the hand or hands in Mesh, then takes pictures of manual signals and compares it with the pictures stored in the file, these have the respective translation, if it finds a picture with a higher level of coincidence it returns with the correct text. The results of the investigation allow

Key words

Kinect, OpenCv, pictures processing, manual signals.

1. Introducción

El desarrollo de herramientas tecnológicas lleva consigo el desarrollo de las regiones; las orientaciones y diseños propuestos han permitido que personas con discapacidades puedan desenvolverse mejor en un medio que no está preparado para ellos. Las personas no oyentes sufren las consecuencias de manejar un lenguaje diferente al oralismo; ellos deben aprender la dactilología, lenguaje basado en

las posiciones de la mano, no obstante, mucha gente de la comunidad en general, lo desconoce, y es ahí donde la comunicación interpersonal se ve truncada.

La tecnología útil y que está dispuesta para tal fin, es el análisis de imágenes, el cual está definido en el artículo "Procesamiento de imágenes digitales", de la Universidad Autónoma de Puebla, en México, como un proceso que consiste en la extracción de características y propiedades de las imágenes, así como la clasificación, identificación y reconocimiento de patrones, indicando la importancia científico-técnica de este campo, no solo en las ciencias, sino principalmente en la sociedad; asimismo, las empresas y desarrolladores de software han enfocado sus estudios en el reconocimiento facial: la principal razón es la necesidad de aplicaciones de seguridad y vigilancia utilizadas en diferentes contextos. Los sistemas de reconocimiento facial pertenecen a las técnicas FRT (Face Recognition Thecniques), las cuales, por medio del método de las EigenFaces, considera las propiedades globales del patrón, mientras que las eingenfeautres consideran un conjunto de características geométricas de la cara. En el proyecto se reenfocaron e implementaron los métodos y técnicas propias del reconocimiento facial para el análisis de las posiciones de la mano.

Finalmente, el artículo considera los siguientes asuntos de forma sucesiva: metodología, resultados, aportes y conclusiones.

2. Metodología

Para el funcionamiento de Kinect se utilizaron los controladores Primesense [1] y OpenNi [2]. Durante las pruebas se desarrolló un aplicativo que contó con una interfaz que permitió inicializar el sensor, activar el flujo de la cámara de profundidad a una resolución de 640x480 y con una frecuencia de 30 fps (imágenes por segundo), utilizar la función Skeleton [3] para detectar las articulaciones de un cuerpo humanoide (controlar la inclinación que abarca de -27 a 27 grados). Como resultado de las pruebas, el aplicativo mostró tres imágenes procesadas por el sensor de Kinect, las cuales corresponden al DepthStream o flujo de profundidad, flujo RGB y a función Skeleton (ver Figura 1).

Figura 1. Resultados de las pruebas del sensor de Kinect.

Una vez evaluado el sensor, se inició la programación. Point Cloud Library (PCL) [4] y OpenCV con el wrapper Emgu [5] fueron las herramientas de desarrollo seleccionadas, debido a la amplia gama de funciones implementadas para el procesamiento de imágenes.

La adquisición de los puntos representa la información de la escena como una constelación de puntos, datos que necesariamente son capturados por el sensor de profundidad, creando un patrón de puntos que corresponden a la profundidad de los objetos (Figura 2).

Con el fin de conocer la distancia óptima de captura de datos, se realizaron experimentos en los cuales se capturaban tres escenas consecutivas de la palma de la mano, a distancias de 30 cm, 70 cm y 100 cm, con respecto a la cámara de Kinect. Se concluyó que a distancias menores de 30 cm no es posible captar los puntos propios de la mano, ya que a esta distancia no hay convergencia entre la cámara de profundidad y el láser, quienes tienen un umbral mínimo de 70cm. Si la mano se sitúa antes del umbral, genera una sombra que no se puede procesar. Si se aumenta la distancia, el sensor captura una mayor cantidad de puntos de la escena, sin embargo, la densidad de puntos sobre la mano disminuye, por lo tanto, se establece que la distancia óptima de captura es 70 cm respecto a la cámara de Kinect, ya que en ese punto se tiene menos grado de dispersión, y la densidad de puntos es significativa para definir la silueta morfológica de una mano. Debido a que los puntos capturados son muchos, se hizo necesario implementar un mecanismo para la reducción de la nube de puntos; este proceso se conoce como filtrado. Se utilizó el filtro Pass Through [6], que permitió desechar los puntos que están por encima de una distancia establecida; la reducción aplicada es de 750 mm sobre el eje Z. El resultado permitió definir la silueta morfológica de la mano.
El procesamiento de imágenes con nubes de puntos se abandonó porque al presentar un mayor grado de precisión en la comparación de imágenes, exige que los puntos y la cantidad de los mismos sean iguales en la imagen dada en tiempo real y la almacenada en la base de datos, lo cual no es posible debido a la diferencia en el tamaño de las manos de los usuarios y la información de la escena o medio en donde este el usuario. Estos aspectos hacen que la cantidad de puntos, así se trate de una misma señal, sean diferentes.

2.2 OpenCV (Open Source Computer Vision).

Esta herramienta se seleccionó por el paquete de librerías que contiene para el procesamiento de imá-

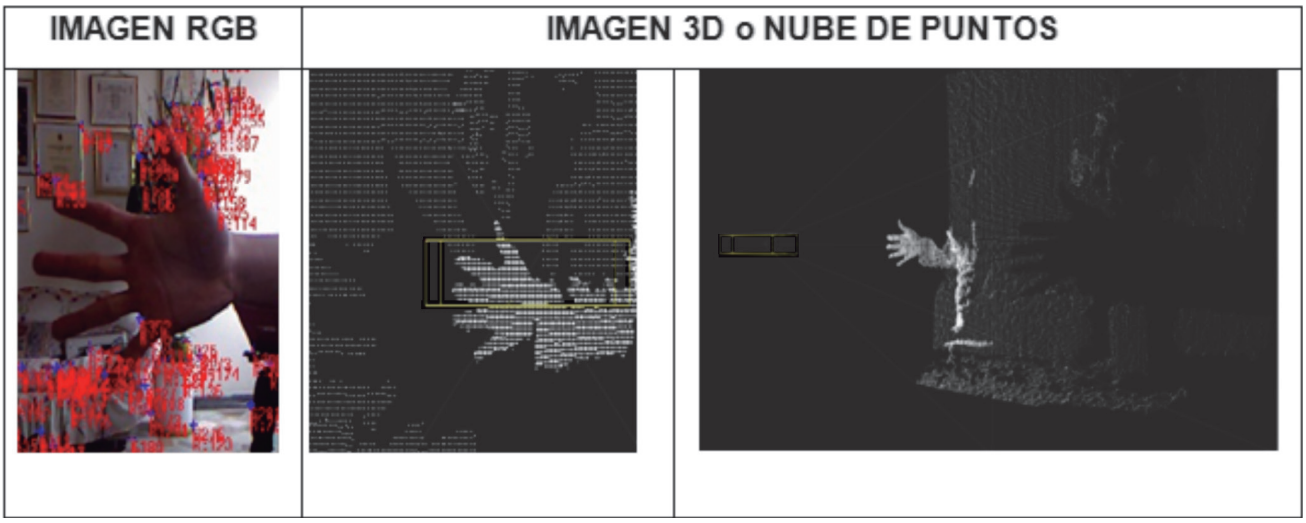


Figura 2. Comparación entre una imagen RGB y una nube de puntos.

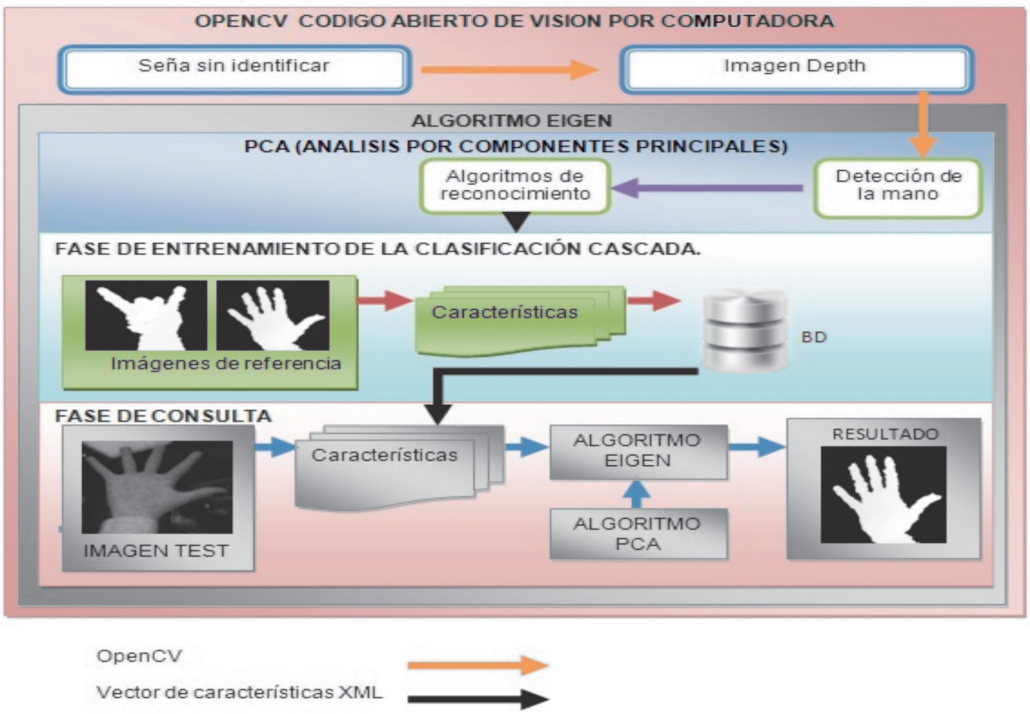


Figura 3. Fases para el desarrollo del intérprete

genes. Las fases para el desarrollo del intérprete se muestran en la Figura 3.

2.2.1 Seña sin identificar. Esta fase es enfocada en el reconocimiento de la estructura de la mano. La función principal de este proceso fue HandDataSource. La distancia de toma fue de 75cm, ya que de acuerdo a las especificaciones técnicas de Kinect, el umbral mínimo de reconocimiento de profundidad es 70cm. Como resultado se visualizó el clustering de la mano, proceso que consiste en trazar líneas sobre los bordes de la mano, como se muestra en la Figura 4.

2.2.2 Imagen Depth. Se representaron los objetos en escala de grises donde las zonas más oscuras equivalen a las mayores distancias con respecto al sensor. Esta escala permitió un proceso de comparación rápido y preciso, ya que se obtuvo la menor cantidad de datos en contraste con las imágenes RGB, las cuales contienen demasiadas características en el color (pixeles) y la información varía según el ambiente.

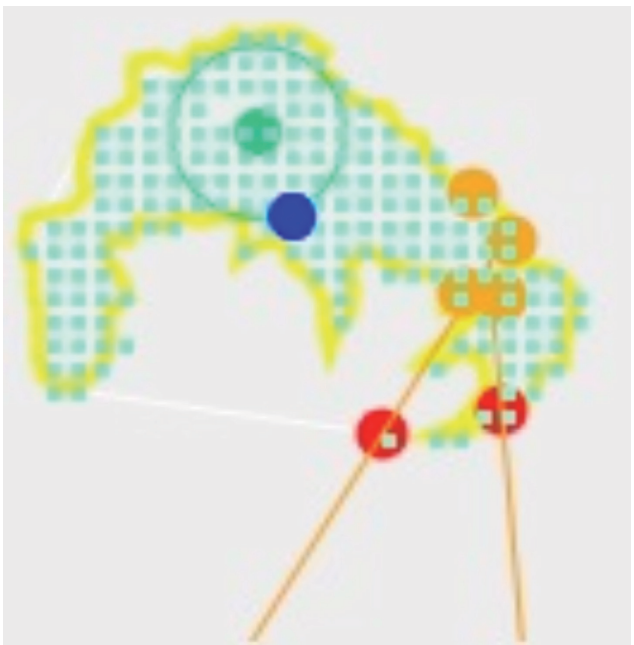


Figura 4.Clustering de la mano.

2.2.3 Algoritmo EIGEN.

1) Detección de la mano: localiza en la imagen Depth la/las manos abiertas o cerradas, con

el fin de devolver características como ubicación y tamaño. En el proyecto se hizo uso de los métodos basados en el aspecto, con algoritmos como el Eigenfaces[7], Fisherfaces y Local Binary Patterns Histograms; estas clases detectan y extraen imágenes de la pose de la mano haciendo uso del método PCA[8] (Principal Components Analysis).

2) Análisis por componentes principales (PCA): busca exhaustivamente en la imagen de entrada una mano, reduciendo su escala según características de la mano. Descarta las regiones donde no existe la posibilidad de encontrarla; esto se realiza mediante la aplicación de heurísticas sencillas que eliminan las regiones no uniformes. El resultado de este proceso es la ubicación de la mano. Estos datos alimentarán la siguiente fase, denominada Haarcascade [9].
3) HaarCascade y base de datos. Las redes neuronales están definidas como una estructura formada por muchos procesadores simples llamados nodos o neuronas, conectados por medio de canales de comunicación o conexiones; estas redes deben tener asociadas reglas de aprendizaje o entrenamiento [10]. Estas redes aprenden a partir de ejemplos [11].

HaarCascade es considerado una red neuronal porque clasifica la presencia o ausencia de la mano. La fase previa de entrenamiento se basa en la implementación de imágenes positivas y negativas, para posteriormente realizar el reconocimiento del gesto manual. Las imágenes positivas son aquellas que contienen la información de los gestos de la mano y se utilizan en el entrenamiento de la red neuronal como referencia para la extracción de características o descriptores; mientras que las negativas no tienen relación con la silueta morfológica de la mano, por lo tanto, son consideradas en el proceso de ubicación de la misma, ya que se tiene una referencia de lo que no es una mano. La Figura 5 muestra las etapas del HaarCascade.

- Imagen de Entrada o Preprocesado: imagen en tiempo real tras aplicarle un filtro de distancia para eliminar ruido.
- Detección: recibe como entrada la imagen preprocesada y devuelve la seña detectada en la imagen. Aplica el algoritmo de detección. Como resultado de este proceso se obtiene la imagen integral, que es una extracción a diferentes escalas de la zona donde se encuentra la mano.
- Extracción de características: recibe como entra-

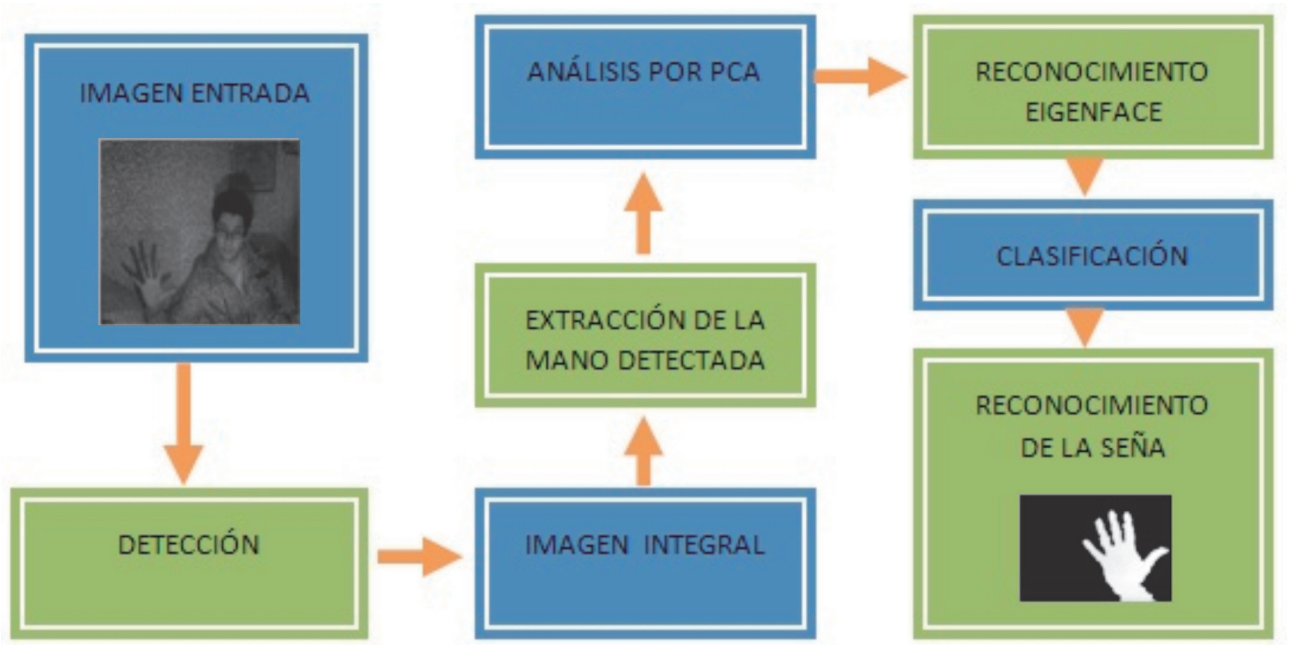


Figura 5. Etapas HaarCascade

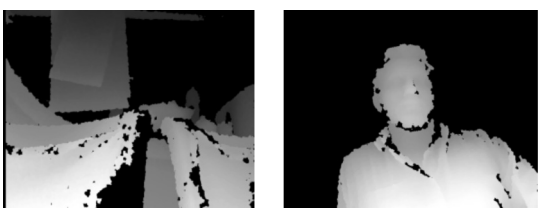
da la imagen integral de la mano o seña manual detectada y devuelve un vector de características. Consiste en obtener la información necesaria para identificar la seña que aparece en la imagen capturada en tiempo real.

- Clasificación: recibe como entrada el vector de características de la imagen y devuelve la seña de la base de datos a la que más se parece. Para el proyecto se implementaron dos clasificadores: uno para la mano abierta y cerrada, y el otro para el reconocimiento de la seña.

La red neuronal implementada está compuesta por cuatro nodos, y fue entrenada con 26 imágenes positivas y seis negativas; este proceso tardó 11,42 seg.

2.2.4 Fase de entrenamiento o aprendizaje. Se entrenó la red neuronal para que funcionara como un sistema de reconocimiento. Para los clasificadores se crearon dos carpetas: una correspondiente a las imágenes positivas, y la otra a las negativas. Estas imágenes fueron cargadas desde la interfaz del sistema (ver Figura 6).

a) Imágenes positivas



b) Imágenes negativas

Figura 6. Ejemplos de imágenes positivas y negativas. a) Señas o imágenes que el sistema debe llamar desde archivo y con las cuales comparará las obtenidas en tiempo real. b) Imágenes donde no existe presencia de la mano.

Teniendo las imágenes negativas y positivas se procedió a generar el archivo con las direcciones de las mismas. Luego fueron agregadas como parte final del entrenamiento del sistema.

En esta fase también se adicionaron señas y su interpretación en texto. Las imágenes son captadas por la aplicación que las almacena en un carpeta, en donde para cada seña se realizaron 10 capturas; además, se registró en un archivo Xaml etiquetas con el nombre de la seña y el nombre del archivo para generar una estructura que usara el training. Para agregar estas señas el aplicativo contó con una ventana que permitió adicionar y digitar su significado.

3. Resultados y análisis

El prototipo está conformado por imágenes de la mano de dos personas que muestran nueve señas básicas definidas en el diccionario básico de Lenguaje de señas colombiano, elaborado por INSOR. Cada foto se encuentra en formato JPG a ocho bits con un tamaño de 100x100 pixeles y una resolución de 96 ppp, y un peso promedio de 2,05kb. La interfaz de usuario se muestra en la Figura 7.

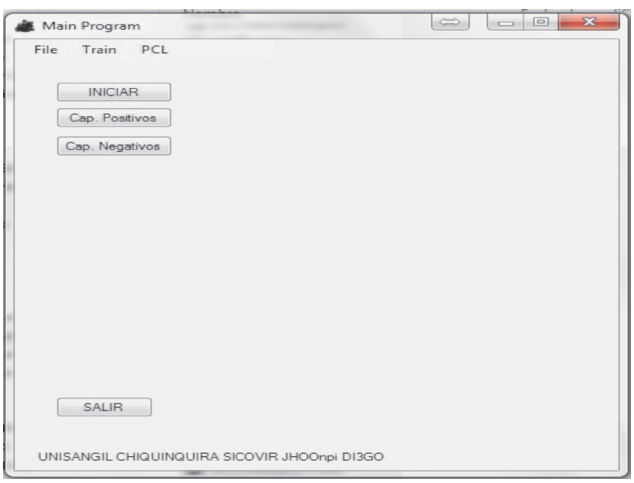


Figura 7. Interfaz de usuario del intérprete de señas.

En ella se presenta tres opciones de menú y tres botones de control:

- Opciones de Menú:
  - File: Permite salir del sistema de reconocimien-

to y detener todos los módulos de Kinect.
- Train: Carga el formulario para el entrenamiento del reconocedor de señas.
- PCL: Carga el formulario que permite visualizar la imagen en nubes de puntos.

- Botones de Control:
  - Iniciar: inicia el dispositivo Kinect y permite visualizar la imagen Depth.
  - Cap. positivas: permite el entrenamiento del reconocedor de la mano y agrega imágenes positivas.
  - Cap. negativas: permite el entrenamiento del reconocedor de la mano y agrega imágenes negativas.
  - Salir: permite salir del sistema de reconocimiento.

Se realizaron pruebas con dos personas. Cada persona mostró una serie de señas en donde se midió el tiempo de respuesta del sistema y el porcentaje de error en el reconocimiento de la misma. En el caso de presentar señas que no estuvieran registradas el sistema, no muestra respuesta alguna. Los resultados se muestran en la **Tabla 1**.

Tabla 1. Resultados de la detección de la seña.

	Persona 1	Persona 2
No Imágenes	9	9
No de detecciones	9	8
No de Fallos	0	1
% de Detección	100 %	89 %
Tiempo de detección promedio	0.0016s	0.0020s

Además de la identificación de señas, el prototipo cuenta con la opción TRAIN, la cual permite agregar nuevas señas al sistema.

En la Figura 8 se muestra un ejemplo en el cual se agrega la seña correspondiente a la letra P.

Es necesario colocarse a una distancia de 75 cm, para que Kinect pueda reconocer la seña. La opción "agregar seña" permite reforzar el reconocimiento de una seña previamente almacenada en el sistema, permitiendo agregar una nueva imagen con la seña. Se debe recordar que el reconocimiento de una seña es óptimo si se almacenan por lo menos 10 imágenes de la misma seña. Si se utiliza esta opción para agregar una nueva seña, solo almacenará una imagen; por lo tanto, el reconocimiento no será óptimo, es decir, no reconocerá la seña cuando el usuario la esté realizando. "Capturar 10", es la opción apropiada para agregar una nueva seña, toma diez imágenes de la seña y las almacena para luego utilizarlas en el proceso de reconocimiento.

Adicionalmente, existe la opción de eliminar todas las señas que se encuentran en el sistema y así comenzar desde cero.

La opción PCL (manejo de imágenes con nubes de puntos RGB) es un campo que se deja inicializado para posteriores investigaciones. Con este se puede:

- Capturar nube de puntos RGB - Capturar nube de puntos sin RGB.

Visualizar nube RGB - Visualizar nube.

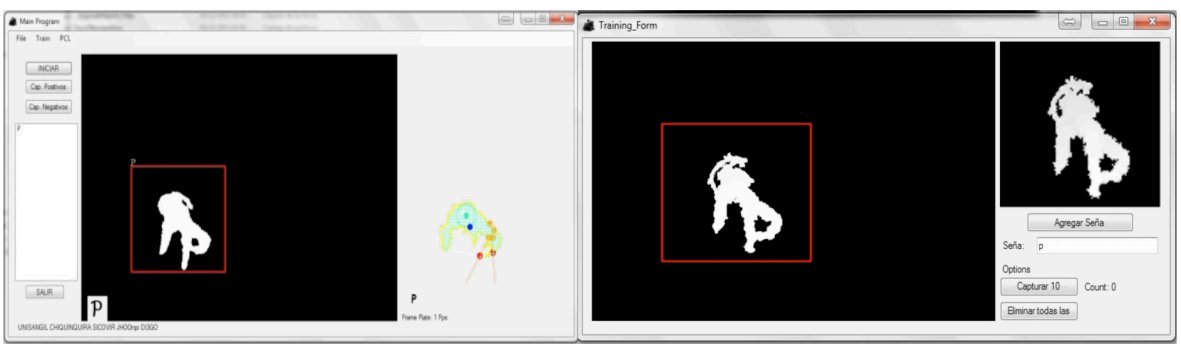


Figura 8. Esquema del entrenamiento del HaarCascade con su correspondiente resultado.

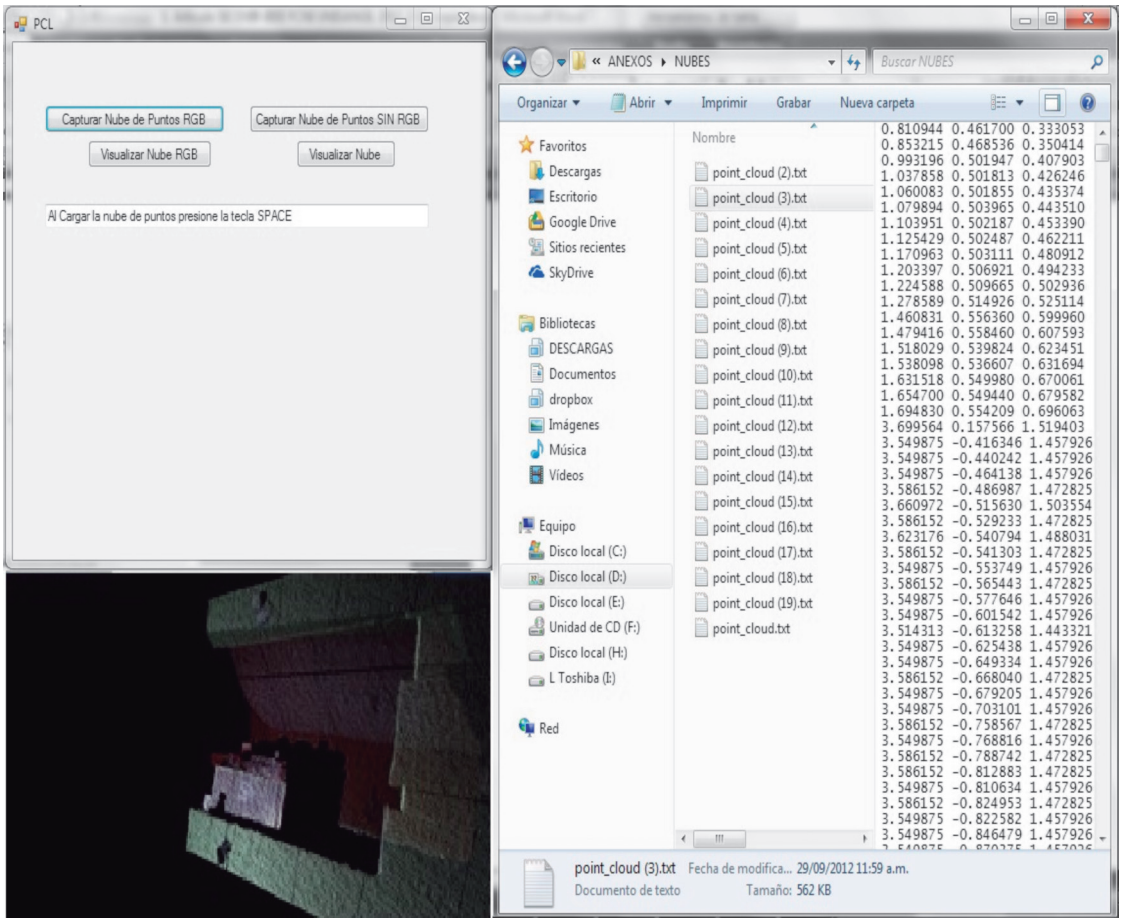


Figura 9. Proceso de captura, almacenamiento y visualización de una nube de puntos.

Las nubes visualizadas ya han sido almacenadas en el sistema por medio de la opción "capturar", y luego son cargadas desde el lugar de almacenamiento como lo muestra la Figura 9**.**

Las dos últimas opciones permiten agregar imágenes positivas y negativas. En la Figura 10 se muestra la captura de una imagen negativa, en la cual no debe existir una mano.

Las imágenes positivas permiten el reconocimiento de una mano en un espacio y no el reconocimiento de una seña.

Figura 10. Proceso captura imagen negativa.

4. Conclusiones

Con la implementación de las librerías que ofrece OpenCV se obtuvieron mejores resultados en el análisis de imágenes debido al gran número de funciones que maneja en lo relacionado a visión por computador, siendo la ejecución menos costosa en cuanto a tiempo y al consumo de recursos computacionales. Mientras el proceso de nubes de puntos trabaja archivos con las coordenadas de cada punto en un espacio 3D, OpenCV permite trabajar con

imágenes en escala de grises, con la ventaja de que una imagen de este tipo siempre permanece con la misma información, al contrario de una imagen manejada a color en donde la información depende de la luz reflejada en el objeto. OpenCV realiza el reconocimiento con archivos de clasificación, lo cual amerita el entrenamiento de una red neuronal, por medio de la cual el proceso de reconocimiento es más rápido y sencillo.

El análisis de las posiciones de la mano está basado en los estudios y avances realizados para el reconocimiento de rostros, analizando estos métodos, técnicas y algoritmos se consiguió el diseño final del prototipo.

La principal desventaja de Kinect es el manejo de distancias, ya que no captura imágenes por debajo de los 70 cm ni por encima de los 4 m. Para el aplicativo se estableció una distancia de captura de 75 cm.

Los sistemas de reconocimiento tienen hoy en día varias dificultades por superar ya que la información de una imagen de un mismo objeto puede variar de acuerdo al ambiente, a la distancia e incluso al ángulo de percepción, lo que marca sustancialmente los resultados dados por el reconocedor; limitaciones como estas deben ser analizadas para determinar mejoras en la implementación de este tipo de sistemas.

Referencias

1. Primesense, 2013. PrimeSense Documentación. Consultado el 21 de Marzo de 2012. http://www.primesense.com.
2. Openni, 2013. What is openni?. Consultado el 28 de marzo de 2012 http://www.openni.org/.
3. Kinectfordevelopers, 2012-2013. Skeleton Tracking – MapSkeletonPointToDepth Deprecated. Consultado el 19 de Febrero de 2012 http://www.kinectfordevelopers.com/2013/01/31/skeleton-tracking-mapskeletonpointtodepth-deprecated/.
4. Pointclouds, 2012. PCL. Consultado el 19 de marzo de 2012 http://pointclouds.org/documentation/ /.
5. EMGU, 2012. Main Page. Consultado el 3 de Abril de 2012 http://www.emgu.com/wiki/index.php/Main_Page.
6. Universidad de Málaga, (2010). Pass-Through del tipo de cambio. Consultado el 12 de Mayo de 2012 http://externos.uma.es/cuadernos/pdfs/papeles38.pdf .
7. Turk, M., 2001. Eigenfaces for Recognition: Journal of Cognitive Neurosicence, Vol. 3, No. 1, pp. 71-86.
8. Viola Paul and Jones Michael J., 2001. Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE CVPR.
9. Universidad del Quindio, 2010. Técnicas para la Detección de Rostros en Secuencias de Imágenes basadas en enfoques Holísticos. Consultado el 4 de Junio de 2012 http://www.uniquindio.edu.co/uniquindio/revistadyp20111013_2/Articulos/5ta%20Edicion/articulo_final.pdf
10. Romero Luis A. and Caro Teodoro C., 2011. Redes Neuronales y Reconocimiento de Patrones.
11. Vidal Ruiz Enrique, 2001. Aprendizaje y Percepción: Preproceso y Extracción de Características.