ANÁLISIS DEL CONDUCTOR: ESTIMACIÓN DE LA DISTRACCIÓN Y SOMNOLENCIA MEDIANTE VISIÓN POR COMPUTADOR E INTELIGENCIA ARTIFICIAL USANDO TECNOLOGÍA TOF

Fernando Guevara C., Oswaldo Valencia B.

***Abstract*—El presente documento describe la investigación, diseño y ejecución de una alternativa tecnológica para detectar y prevenir accidentes de tránsito a causa de la distracción y/o somnolencia. Mediante el uso de la tecnología de tiempo de vuelo se determinará distancias de objetos con la emisión de una señal infrarroja; conjuntamente con la inteligencia artificial del modelo *CANDIDE*, se diseñará un algoritmo para seguimiento facial en tiempo real de un rostro actualizado parametrizado, que mediante algún cambio reflejado en el rostro, valores como unidades de acción y vértices cambiarán de estado. Para la detección de la distracción existen tres tipos de movimientos faciales: cabeceo, balanceo y giro; los mismos que en un estudio previo de los vértices del modelo, se elegirá puntos exactos que reflejan dichos movimientos para relacionar en un algoritmo de control los valores de datos de los vértices en cada situación de cambio de estado en el rostro del conductor. En cambio para la determinación de somnolencia se usará las unidades de acción. Las mismas que controlan los rasgos faciales de modo que entregan valores por cada expresión del rostro.**

***Index Terms*—Seguimiento, Visión, Inteligencia Artificial, Tiempo de vuelo, Distracción, Somnolencia, C Sharp, Wpf, Candide, Kinect, Visual Studio.**

I. INTRODUCCIÓN

La distracción y/o somnolencia al conducir son fenómenos complejos, que implican disminuciones en los niveles de alerta y conciencia de parte del que maneja. Estas situaciones conllevan accidentes eludibles ante la identificación de situaciones peligrosas y por cuanto a evitar tomar riesgos. El cansancio mental, como el físico, provoca el adormecimiento del conductor, y representa un factor que contribuye a los accidentes al menos en el 24 % de ellos.

La Agencia Nacional de Tránsito [1] en un estudio realizado en Noviembre del 2011 determinó que factor humano ocasiona el 89% de los accidentes de tránsito en el Ecuador. Los componentes del factor humano son:

- Negligencia de conductor 60% (distracción por celular, mal rebasamiento, invasión de carril, mal estacionamiento y pasar semáforo en rojo).
- Exceso de velocidad 14%.
- Embriaguez del conductor 8%.
- Imprudencia del peatón 7%.

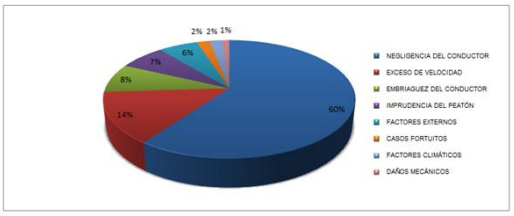


Figura 1. Porcentajes de las Causas de los Accidentes de Tránsito en el Ecuador

Un porcentaje menor equivalente al 11% se debe a factores externos tales como:

- Mal estado de la vía 6%
- Casos fortuitos 2%.
- Factores climáticos 2%.
- Daños mecánicos 1%.

A. Tecnologías aplicadas a la conducción.

Cuando un conductor no circula lo suficientemente atento o cuando se está quedando adormilado, presenta una serie de indicios en su conducción y en sus rasgos faciales, los cuales son causa de posibles accidentes. Los sistemas de detección de la somnolencia y/o distracción reconocen estos indicios y advierten al conductor de la situación de riesgo con la suficiente antelación.

Aunque prácticamente puede darse en cualquier situación, el mayor riesgo de sufrir un accidente por la pérdida de concentración en la conducción o incluso por quedarse dormido como consecuencia de la fatiga se produce en trayectos normalmente largos. Dichas situaciones de riesgo no son exclusivas de la noche, también la monotonía de la tarea de dirigir el vehículo (especialmente en autopistas) conlleva un aumento de la probabilidad de sufrir un accidente como consecuencia de la fatiga del conductor.

En muchos casos el propio conductor aunque es consciente de su cansancio, infravalora el riesgo de quedarse dormido al volante, cuando tan solo un segundo de sueño puede derivar en un accidente de graves consecuencias. Para evitar estos accidentes los sistemas de detección de la somnolencia del conductor reconocen los indicios que muestran la conducción bajo condiciones de cansancio o pérdida de concentración, alertando al conductor de que dicha situación se está produciendo e invitándole a que se tome el descanso necesario.

Los desarrollos tecnológicos que guardan alguna relación con la fatiga del operador al volante se han caracterizado en este trabajo de acuerdo con los siguientes objetivos:

- Prevención de la distracción y/o somnolencia.
- Detección de la distracción y/o somnolencia.

- Prevención de accidentes debidos a la distracción y/o somnolencia.

A continuación se describirán las tecnologías existentes por cuanto a estos tres enfoques:

TECNOLOGÍAS RELATIVAS A LA DISTRACCIÓN Y/O SOMNOLENCIA DEL CONDUCTOR	PREVENCIÓN	Comodidad	Vibraciones/Ruidos Atmósfera en Cabina Deslumbramientos
		Horas de Servicio	Cajas negras Registradores/Verificadores Calculadoras
		Esfuerzo Mecánico y Visual	Control aut. velocidad Control de cambios Trans. automáticas Despliegues binoculares
	DETECCIÓN	Monitoreo del Conductor	
		Monitoreo de la conducción	
	PREVENCIÓN DE ACCIDENTES		

Figura 2. Tecnologías Relativas a la Distracción y/o Somnolencia del Conductor

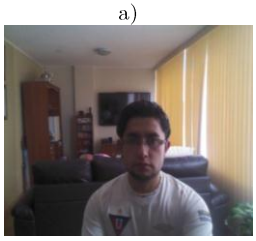
II. INTERFACE DE PROGRAMACIÓN

Las herramientas tecnológicas que nos ayudarán a diseñar el algoritmo de control que detecte la somnolencia y/o distracción del conductor, se basan en la tecnología de tiempo de vuelo para determinar distancias de objetos mediante el envío y recepción de una señal infrarroja, una Interface Natural de Usuario conformada por el sensor Kinect for Windows que consta de una cámara infrarroja para la detección de objetos en tres dimensiones y el modelo Candide que nos entrega un rostro actualizado parametrizado.

A. Tecnología de Tiempo de Vuelo.

El tiempo de vuelo (TOF) de imágenes se refiere al proceso de medición de la profundidad de una escena mediante la cuantificación de los cambios que una señal de luz emitida encuentra cuando rebota de objetos en una escena.

a)



b)

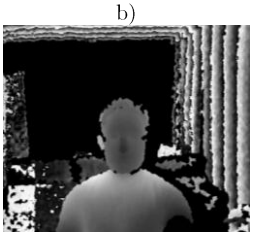


Figura 3. a) Imagen de una cámara regular b) Imagen de una cámara de profundidad ToF

Un escáner 3D de tiempo de vuelo determina la distancia a la escena cronometrando el tiempo del viaje de ida y vuelta de un pulso de luz. Un diodo láser emite un pulso de luz y se cronometra el tiempo que pasa hasta que la luz reflejada es vista por un detector. Como la velocidad de la luz c es conocida, el tiempo del viaje de ida y vuelta determina la distancia del viaje de la luz, que es dos veces la distancia entre el escáner y la superficie. Si T es el tiempo del viaje completo, entonces la distancia es igual a:

$$\frac{c(T)}{2} \tag{1}$$

Claramente la certeza de un escáner láser de tiempo de vuelo 3D depende de la precisión con la que se puede medir el tiempo *T: 3,3 picosegundos* (aprox.) es el tiempo requerido para que la luz viaje 1 milímetro. Se utilizan láseres visibles (verdes) o invisibles (infrarrojo cercano).

El distanciómetro láser sólo mide la distancia de un punto en su dirección de la escena. Para llevar a cabo la medida completa, el escáner va variando la dirección del distanciómetro tras cada medida, bien moviendo el distanciómetro o deflectando el haz mediante un sistema óptico. Este último método se usa comúnmente porque los pequeños elementos que lo componen pueden ser girados mucho más rápido y con una precisión mayor. Los escáneres láser de tiempo de vuelo típicos pueden medir la distancia de 10.000 ~ 100.000 puntos cada segundo [2].

Con la medición de la profundidad usando la tecnología ToF disponemos entre otras las siguientes ventajas:

- Sólo se requiere de una cámara específica.
- No se requiere de un cálculo manual para la profundidad.
- Adquisición de la geometría de la escena 3D en tiempo real.
- Menor dependencia de iluminación de la escena.
- Casi sin depender de textura de la superficie.

B. Natural User Interface (NUI).

Una interfaz natural de usuario o *Natural User Interface* (NUI), es una interfaz de usuario diseñada para reutilizar las habilidades existentes en los seres humanos de interactuar con el ambiente y enfocarlas a la interacción directa con el contenido informático.

Las NUI adquieren su carácter de "naturales" debido a su capacidad de ser intuitivas [3], ciertamente esto todavía es muy ambiguo pues sólo hemos pasado de una palabra a otra, en referencia a Bill Buxton, uno de los grandes expertos en la tecnología de las NUI una interface es natural si explota las habilidades que hemos adquirido a través de nuestra vida [4].

Interpretando estas palabras decimos que una interface es natural si aprovecha las capacidades de bajo nivel del ser humano (hablar, gesticular, reaccionar) más las habilidades aprendidas que hemos desarrollado a través de la interacción con nuestro propio entorno en la vida cotidiana.

Las siguientes características nos permiten definir e identificar los aspectos más relevantes de una NUI para su diseño:

- *Experiencia inmediata.* Los usuarios no necesitan aprender algo nuevo sólo valerse de las habilidades simples que ya dominan y aplicarlas en una nueva situación.

- *Especialidad.* Las habilidades pueden explotarse para un grupo específico.
- *Carga cognitiva.* El uso de habilidades simples permite al usuario concentrarse en la labor ejecutada y no en la forma de comunicación de datos.
- *Aprendizaje progresivo.* Incluso en procesos complicados se debe llevar una curva suave de aprendizaje, para tareas complejas la mejor forma de abordarlas es descomponerlas en subtareas que hagan uso de las habilidades simples.
- *Interacción directa.* El diseño de las interfaces es creado para manejar entidades informáticas como objetos sensitivos.

De la forma de interacción directa se desprenden nuevas características sobre la respuesta del sistema a los impulsos del usuario:

- *Proximidad espacial.* La proximidad física a un elemento informático que no tiene forma real provoca una sensación de familiaridad con él.
- *Proximidad temporal.* La interfaz reacciona al mismo tiempo que la acción del usuario.
- *Asignación concurrente.* Hay una relación entre cada grado de libertad de la acción del usuario y cada grado de libertad de la reacción de la interface.

El conjunto de estas características crea el efecto de sencillez, familiaridad y fluidez, siendo no sólo fácil manipular los elementos de un sistema, el usuario comprende de forma natural el proceso que se lleva a cabo y las limitantes a las que se afronta del mismo modo que comprendemos el ambiente que nos rodea en una forma no del todo profunda pero lo suficiente para darnos cuenta de lo que sucede [3].

C. El Sensor Kinect.

El sensor Kinect es un dispositivo inicialmente pensado como un simple controlador de juego, que gracias a los componentes que lo integran: sensor de profundidad, cámara RGB, arreglo de micrófonos y sensor de infrarrojos (emisor y receptor), es capaz de capturar la anatomía humana, reconocerla y posicionarla en el plano. Mediante la combinación de la salida de estos sensores, un programa puede rastrear y reconocer objetos en frente de ella, determinar la dirección de las señales de sonido, y aislarlos del ruido de fondo.

Figura 4. Sensor Kinect de Microsoft

Dentro del sensor Kinect se encuentra un mecanismo muy complejo que contiene una colección de sensores y una gran poder de procesamiento, de hecho, es tan grande el procesamiento que sucede en el interior de la barra de sensores que incluso hay un pequeño ventilador para evitar un sobrecalentamiento en el dispositivo.

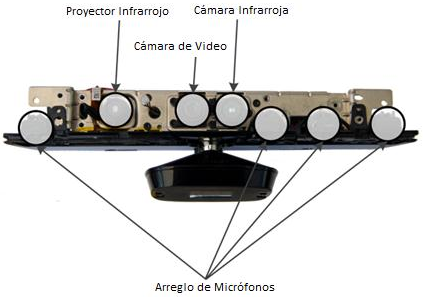


Figura 5. Estructura Interna del Kinect

1. *La Cámara de Video del Kinect:* La cámara de vídeo Kinect proporciona imágenes de alta resolución de vídeo suministrada al dispositivo receptor. Se puede trabajar en una variedad de resoluciones de hasta 1280x1024 píxeles, aunque la resolución que se utiliza normalmente es de 640x480.

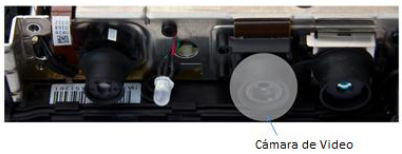


Figura 6. Imagen interna de la Cámara de Video del Kinect

2. *El arreglo de Micrófonos del Kinect:* La barra del sensor Kinect tiene cuatro micrófonos diferentes. Uno se encuentra a la izquierda de la fuente de luz infrarroja. Los otros tres están espaciados uniformemente a la derecha de la cámara de profundidad; éstos no son instalados de manera que el Kinect puede capturar multi-canal de sonido estéreo. En su lugar, los micrófonos se utilizan para permitir al Kinect centrarse en las fuentes de sonido individuales. El Kinect contiene Señales Digitales de Procesamiento (DSP) que procesan el sonido recibido de los micrófonos.

Figura 7. Arreglo de Micrófonos del Kinect

El software que se ejecuta en el interior del sensor conoce la separación física entre cada micrófono y la velocidad a la que viaja el sonido a través del aire. El software puede realizar el análisis de las señales entrantes para identificar la dirección de una fuente de sonido particular y permitir que el sensor rechace los sonidos no deseados y de los ecos [5], [6].

Todo el trabajo adicional que realiza el sensor es para dar una señal de audio lo más clara posible para analizar el reconocimiento de voz. También hace posible que el Kinect reconozca diferentes altavoces en una

situación en la que más de una persona está dando comandos de voz.

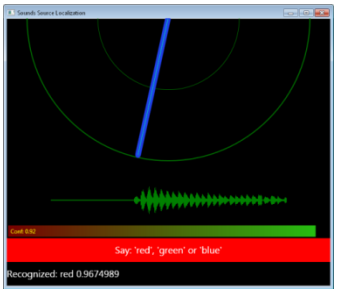


Figura 8. Ejemplo de direccionamiento de voz

La Figura 8 muestra el resultado de una de las aplicaciones de ejemplo que se proporcionan como parte del SDK de Kinect para Windows. Este programa utiliza la plataforma Microsoft Speech para reconocer el habla recibida por el sensor Kinect. El programa también muestra la dirección de la voz con respecto al sensor.

3. *El sensor de profundidad del Kinect:* Kinect tiene la capacidad única de ver en 3D. A diferencia de la mayoría de los otros sistemas de visión por ordenador, el sistema Kinect es capaz de construir un mapa de profundidad de la zona en frente de ella. Este mapa se produce totalmente dentro de la barra de sensor y luego se transmite por el cable USB al host en la misma forma que una imagen de cámara típica transmite su información, excepto que en lugar de la información de color de cada píxel de una imagen, el sensor transmite los valores de distancia [7].

   Se podría pensar que el sensor de profundidad utiliza algún tipo de radar o transmisor de sonido ultrasónico para medir los objetos están lejos de la barra de sensores, pero en realidad no es así. Esto sería difícil de hacer en una distancia corta. En su lugar, el sensor utiliza una técnica inteligente que consiste en un proyector de infrarrojos y una cámara que puede ver los pequeños puntos que el proyector produce.

Figura 9. Elementos de la cámara de profundidad del Kinect

Hay dos elementos en la cámara de profundidad del Kinect, como se indica en la Figura 9. El primero es el proyector de infrarrojos que proyecta un campo de puntos frente a la escena de la barra de sensores. El segundo elemento es una cámara infrarroja que ve los puntos de la escena. La posición de los puntos en la escena reflejada depende de la posición con la que la cámara refleja la escena [6], [7].

C. Modelo Candide.

CANDIDE es una máscara parametrizada específicamente desarrollada para la codificación de rostros humanos, basados en su modelo y constitución. Su bajo número de polígonos (aproximadamente 100) permite la reconstrucción facial rápida con una potencia de cálculo moderada.

El modelo es controlado por unidades de acción globales y locales (UA). Las globales corresponden a rotaciones alrededor de los tres ejes. Las unidades de acción locales controlan los rasgos faciales de modo que se pueden obtener diferentes expresiones.

El concepto de unidades de acción fue descrito por primera vez hace unos 40 años por Carl-Herman Hjortsjö en su libro *El rostro humano y el Lenguaje Mímico*. Este trabajo fue ampliado más tarde por Paul Ekman y Wallace Friesen del Departamento de Psiquiatría de la Universidad del Centro Médico de California lo cual recibió el nombre de Sistema de Codificación de Acción Facial (Facial Action Coding System FACS) [8].

El modelo *CANDIDE* fue creado por Mikael Rydfalk del Grupo de Codificación de Imágenes de la Universidad de Linköping en 1987 [9]. Este trabajo fue motivado por los primeros intentos para realizar una compresión de imágenes a través de la animación [10].

El modelo *CANDIDE* fue conocido por un público más amplio a través de varios artículos en revistas científicas. Ahora se encuentra a disposición del público y es utilizado por los grupos de investigación de todo el mundo.

El modelo original de *CANDIDE*, fue descrito por primera vez en el informe de M. Rydfalk, el mismo que contenía 75 vértices y 100 triángulos, acoplado sobre una plataforma en versión demo de Java; hoy en día esta plataforma es obsoleta y muy poco utilizada.

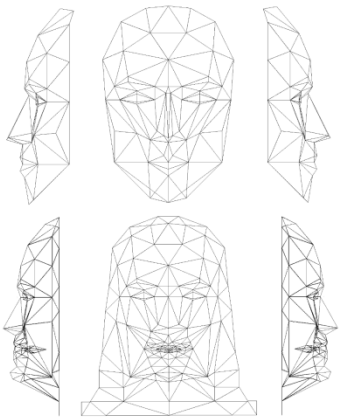


Figura 10. Arriba: CANDIDE-1 con 79 vértices y 108 triángulos 238 triángulos. Abajo: *CANDIDE-2* con 160 vértices y 238 triángulos

La primera versión difundida, por muchos años fue modelo facto estándar de *CANDIDE*, el cual fue un modelo ligeramente modificado con 79 vértices, 108 triángulos y 11 unidades de acción; este modelo fue creado por Mårten Strömberg, mientras

implementaba la plataforma de software llamada xproject [11], conocida como *CANDIDE-1.*

Más tarde, Bill Welsh de British Telecom [12], creó otra versión con 160 vértices, 238 triángulos y 6 unidades de acción que cubren toda la cabeza frontal incluyendo el cabello, los dientes y los hombros. Esta versión es conocida como *CANDIDE-2*, que también se incluye en la plataforma xproject.

1. *Modelo Candide-3*

El modelo *CANDIDE* sigue siendo ampliamente utilizado, ya que su simplicidad lo hace una buena herramienta para las tareas de análisis de imágenes y animaciones con baja complejidad. Sin embargo, con el transcurso del tiempo se ha visto en la necesidad de una actualización del modelo, debido a las siguientes razones [13]:

- La simplicidad en el diseño de los ojos y boca, hace que el modelo *CANDIDE* sea muy poco realista; lo cual con la adicción de unos pocos vértices mejoraríamos la calidad de manera significativa.
- Una norma para cada punto característico facial que se debe utilizar en la animación está en formato MPEG-4. Varios de los puntos característicos faciales (Facial Feature Points FFPs) no están incluidos en los vértices del modelo *CANDIDE.*

Por lo tanto se han realizado las siguientes modificaciones:

- La estructura de la boca ha sido modificada para obtener vértices correspondientes a los puntos característicos externos e internos del labio, y se han añadido también vértices entre las esquinas de la boca y el centro de los labios (vértices 79-89, MPEG-4 FFPs en el grupo 2 y 8).
- Los vértices correspondientes a cada mejilla (vértices 27 y 60) han sido reemplazados por dos vértices correspondientes a los puntos faciales característicos 5.2 y 5.4 (lado derecho) y 5.1 y 5.3 (lado izquierdo). Los nuevos vértices tienen los números 90 y 91, correspondientemente.

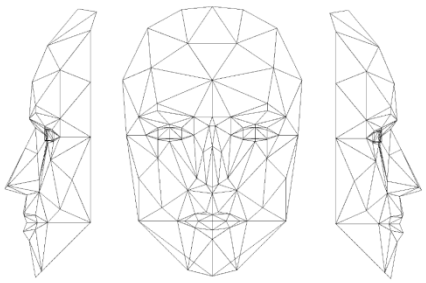


Figura 11. CANDIDE-3 con 113 vértices y 168 triángulos

- Existen tres vértices (92 - 94) en la parte media de la nariz, que corresponden a los nuevos FFPs agregados: 9.12, 9.13, y 9.14, y dos vértices (111.112) correspondientes a los nuevos FFPs agregados: 9.4 y 9.5.

- Se han añadido vértices al contorno de los ojos, haciendo que el diseño sea más realista (vértices 95-110).

- Las unidades de acción definidas en CANDIDE-1 se amplían para incluir nuevos vértices.
- La animación puede ser realizada por MPEG-4 FAPs (Facial Animation Parameters), así como por unidades de acción.

2. *Controlando el Modelo*

CANDIDE es un modelo similar a una malla de alambre con una textura mapeada en su superficie. Las coordenadas de los vértices se almacenan en un archivo *wfm*, y puede ser visto como un vector $\bar{g}$ *3-N* dimensional (donde N es el número de vértices) que contiene las coordenadas (x, y, z) respectiva de cada vértice.

El modelo entonces puede ser reconfigurado según [13]:

$$g(\sigma, \alpha) = \bar{g} + S\sigma + A\alpha \tag{2}$$

, donde el vector resultante $\bar{g}$ contiene los nuevos vértices de coordenadas (x, y, z). Las columnas de S y A son la Forma y Unidades de Animación, respectivamente, y por lo tanto los vectores σ y α contienen la forma y los parámetros de animación.

Como también queremos realizar un movimiento global, necesitamos algunos parámetros más para la rotación, el escalamiento, y la traslación. Por lo tanto, reemplazamos (2) con:

$$g = Rs(\bar{g} + S\sigma + A\alpha) + t \tag{3}$$

, donde

$$R = R(r_x, r_y, r_z) \tag{4}$$

, es la matriz de rotación, s es la escala, y

$$t = t(t_x, t_y, t_z) \tag{5}$$

, es el vector de traslación.

La geometría del vector está por lo tanto parametrizada por el vector de parámetros:

$$p = [v, \sigma, \alpha] = [r_x, r_y, r_z, s, t_x, t_y, t_z, \sigma, \alpha] \tag{6}$$

, donde v es el vector de los parámetros globales de movimiento. El formato de archivo admite tener diferentes escaladas en las tres dimensiones, ejemplo:

$$g = RS_3(\bar{g} + F\sigma + A\alpha) + t \tag{7}$$

, donde

$$S_3 = S_3(s_x, s_y, s_z) \tag{8}$$

Las Unidades de Animación pueden ser implementaciones de los MPEG-4 FAPs (Parámetros de Animación Facial) o los Vectores de Unidad de Acción implementaciones de las Unidades de Acción desde los FACS (Facial Action Coding System).

III. SEGUIMIENTO FACIAL, RECONOCIMIENTO DE DISTRACCIÓN Y/O SOMNOLENCIA

Una vez identificados los parámetros del modelo Candide-3, procederemos mediante inteligencia artificial proporcionada por el sensor Kinect, a realizar un algoritmo inteligente que detecte las condiciones anormales de conducción y alerte mediante una alarma al conductor informándole su errónea forma de conducir.

A. Kinect for Windows SDK.

Kinect para Windows SDK es un conjunto de bibliotecas que nos permite programar aplicaciones en una variedad de plataformas de desarrollo Microsoft utilizando el sensor Kinect como entrada. Con él, programamos aplicaciones WPF, WinForms, aplicaciones de XNA y con un poco de trabajo, incluso aplicaciones basadas en navegador que se ejecutan en el sistema operativo Windows [6].

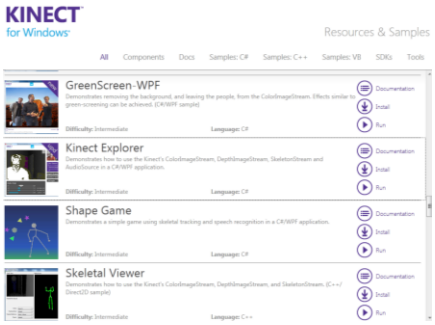


Figura 12. Kinect for Windows Developer Toolkit v1.5.2

Kinect para Windows consiste en el sensor Kinect, el Kinect para Windows Software Development Kit (SDK) y la concesión de licencias comerciales necesarias para el despliegue de la aplicación. El Kinect para Windows SDK es compatible con las aplicaciones generadas con C + +, C # o Visual Basic utilizando Microsoft Visual Studio 2010 o 2012. La última actualización de la Kinect para Windows SDK y las herramientas del desarrollador agregan características adicionales del sensor Kinect, mejorando la eficiencia de desarrollo y añade al sistema operativo varias herramientas de apoyo.

Con la instalación del Kinect para Windows SDK, se proporcionan recursos y ejemplos con la documentación respectiva para el entendimiento de los programadores en el uso del SDK a la aplicación respectiva.

B. Face Tracking.

El FaceTracking SDK de Microsoft conjuntamente con el SDK de Windows para la cámara Kinect for Windows, permite crear aplicaciones que puedan hacer seguimiento del rostro humano en tiempo real.

El motor de rastreo de rostros del FaceTracking SDK analiza la entrada de la cámara Kinect, para luego deducir la posición de la cabeza y las expresiones faciales, y hace que esa información esté disponible para realizar una aplicación en tiempo real. Por ejemplo, esta información puede ser usada para representar el seguimiento de la posición de la cabeza de una persona y sus expresiones faciales dentro de un avatar en un juego de video, o una aplicación de comunicación, o para manejar una interface natural de usuario.

1. Descripción del Proyecto Microsoft.kinect.toolkit.facetracking

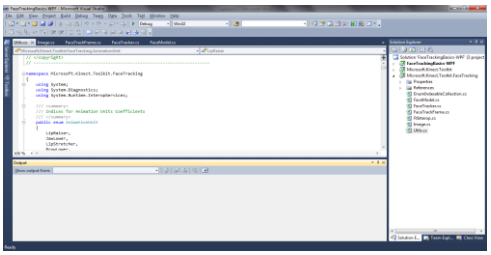


Figura 13. Clases proyecto Microsoft.kinect.toolkit.facetracking

Proyecto desarrollado por Microsoft liberado conjuntamente con el SDK de Windows, dentro de este proyecto encontraremos algunas clases que serán de utilidad para realizar proyectos relacionados con el seguimiento de rostros. Las funciones que realizan cada clase son las siguientes:

- *FaceModel.cs:* La interfaz FaceModel proporciona una manera para leer parámetros 2D y 3D, los mismos que pueden ser manipulados para realización de aplicaciones según la intención del usuario.
- *FaceTrack.cs:* La interfaz FaceTrack al ser instanciada arranca el motor de seguimiento de rostros y sigue el rostro de una persona devolviendo varios valores como unidades de animación o de acción, puntos 3D y triángulos en el rostro.
- *FaceTrackFrame.cs:* Representa los resultados del seguimiento de rostros en un cuadro (frame).
- *Image.cs:* Representa una imagen usada por el seguidor de rostros. Los usuarios pueden fijar esta clase a su buffer de imágenes o pueden llenar esta referencia con datos de la imagen.
- *Utils.cs:* Esta interfaz proporciona los valores correspondientes de cada uno de los vértices del modelo.

2. *Proyecto Facetracking Basic.*

Dentro del SDK de Windows encontramos una aplicación básica del seguimiento de rostros, la misma que usa las clases anteriormente mencionadas del proyecto Microsoft.kinect.toolkit.facetracking.

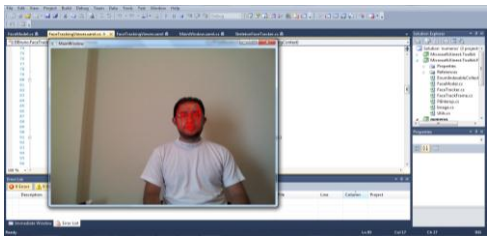


Figura 14. FaceTracking Basic

La aplicación consiste en formar una malla de triángulos en el rostro permitiendo el reconocimiento de algunas expresiones faciales como apertura y cierre de la boca o movimientos de la cabeza.

Figura 15. FaceTracking Basic Movimiento de la Cabeza

B. Numeración Facial.

Una vez obtenida la malla el siguiente paso será la obtención de puntos vértices del modelo *CANDIDE-3*. Para ello desarrollamos un algoritmo dentro del cual creamos una variable la misma que contendrá una lista de los puntos faciales, cada uno con su respectiva coordenada, mismos que nos son entregados por la clase SkeletonFaceTraker.

El paso siguiente será crear un lazo en el cual se tomaran uno a uno los puntos de la lista para luego ser graficados sobre el rostro. De tal forma que sin importar que el sujeto mueva su cabeza los puntos seguirán el movimiento del mismo. A continuación se puede observar todos los puntos graficados sobre el rostro del sujeto de prueba.

Figura 16. Puntos Faciales CANDIDE-3

En el modelo *CANDIDE-3* a cada punto facial le corresponde un vértice, para poder visualizarlas sobre el rostro del sujeto de pruebas, de igual forma se creará un lazo en el cual cada vez que se tome un punto de la lista para graficarlo, se graficará a su vez la unidad correspondiente.

Figura 17. Vértices del CANDIDE-3

Para la aplicación que se desarrolló no se necesitó todos los puntos del modelo CANDIDE 3, sino un grupo mínimo que permita conocer la posición actual del conductor. Para esto lo que se hizo fue dentro del algoritmo crear una variable que haga de filtro, dicho filtro contendrá vértices a ser utilizados. Como paso siguiente se creará un lazo y de igual forma fue tomando los puntos de la variable filtro para ser graficados sobre el rostro del sujeto de prueba.

Figura 18. Puntos filtrados

Para poder observar el vértice correspondiente a cada punto se realizará el procedimiento antes mencionado pero ahora utilizando la variable filtro en lugar de la lista entera.

Figura 19. Vértices numerados filtrados

Los vértices del modelo CANDIDE 3 utilizados son:

- 0 Parte alta del cráneo.
- 2 Parte media de la frente.
- 94 Mitad baja en el borde del hueso de la nariz
- 10 Parte inferior de la barbilla

C. Detección de la Distracción.

El algoritmo para detección de la distracción consta de variables para la inicialización de alarmas, contadores para la activación de alarmas y la secuencia para la alarma sonora. Cuando el programa principal se ejecuta se marcarán en el rostro del conductor los puntos principales, tomados del subcapítulo anterior con la variable filtro. A su vez las posiciones iniciales de dichos puntos se guardaran en distintas variables. La posición actual de cada punto será tomada como referencia para determinar las alarmas. Debido a que no todos los conductores poseen las mismas características físicas a partir de la posición inicial en la que los puntos se marquen en sus rostros se determinaran las alarmas para cada uno.

Las alarmas serán accionadas cuando el punto o los puntos de referencia se desplacen de su posición inicial hacia fuera del rango normal de conducción y haya transcurrido un tiempo considerable de distracción.

Para determinar el rango de normal de conducción se tomaron las posiciones de los puntos, cada cinco segundos, de varios conductores mientras se encontraban en el simulador y luego en un auto real. Después de esto se obtuvieron graficas de la posición de cada punto en función del tiempo. Obteniendo como resultado un rango aceptable de conducción normal.

Es complejo determinar cuánto sería un tiempo de distracción considerable, debido a que este dependería de la velocidad a la que se encuentra conduciendo el conductor, por lo tanto para poder realizar las pruebas se tomó un tiempo de dos segundos.

Además cada vez que una alarma sea accionada la cámara tomara una foto de la posición del conductor en ese momento y así dar veracidad al suceso.

1. Sistema de coordenadas

La cámara Kinect proporciona un sistema de coordenadas el mismo que es usado para el seguimiento facial obteniendo como resultado seguimiento en 3D. En la siguiente figura se puede observar como está establecido el sistema de coordenadas teniendo como origen el centro óptico de la cámara.

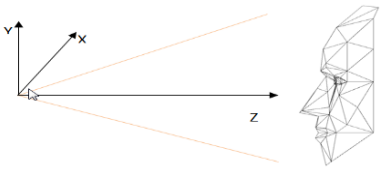


Figura 20. Sistema de coordenadas con origen en el centro óptico del Kinect

Las posiciones de los ejes X, Y, y Z de la cabeza del conductor se determinarán mediante un sistema de coordenadas utilizando la regla de la mano derecha. Las traslaciones se miden en metros. Es así que la posición de la cabeza del conductor se medirá por tres ángulos: giro, balanceo y cabeceo.

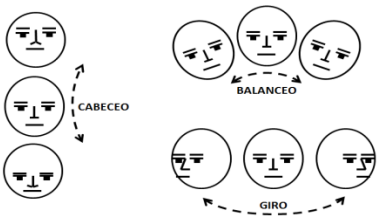


Figura 21. Posicionamiento de la cabeza

2. Alarmas del sistema de detección de distracción

Para el sistema de detección de distracción se creó tres alarmas denominadas de la siguiente manera:

- *Alarma de giro:* Teniendo como vértice referencial al que se encuentra en la mitad baja del borde del hueso de la nariz (94).

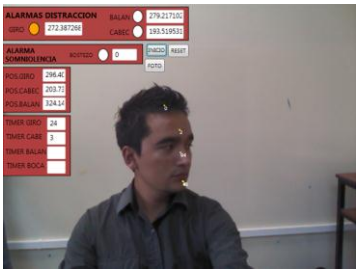


Figura 22. Alarma de giro encendida

- *Alarma de balanceo:* El vértice utilizado como referencia para esta alarma es el que está posicionado en lo más alto del cráneo (0).

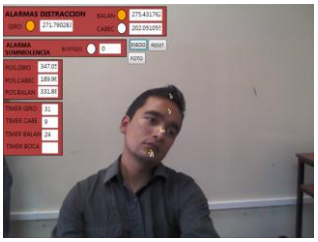


Figura 23. Alarma de balanceo encendida

- *Alarma de cabeceo:* Para esta alarma se tomó como vértice referencial al que está ubicado en la parte media de la frente (2).

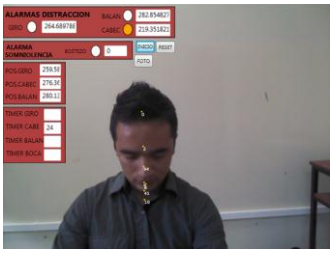


Figura 24. Alarma de cabeceo encendida

D. Detección de la somnolencia.

El primer síntoma de la somnolencia es el bostezo, el mismo que fue tomado como detonante de la alarma en el programa. Para detectar el bostezo del usuario, se tomó en cuenta la unidad de acción que interviene en la abertura de la boca y el tiempo en que la misma permanecía abierta. Para calcular la apertura de la boca, a diferencia de la detección de la distracción en la que se ocuparon los vértices del modelo *CANDIDE-3*, se trabajó directamente con el algoritmo que proporciona la unidad de acción correspondiente al movimiento de la mandíbula inferior. Al abrir y cerrar la boca el algoritmo nos entrega un valor entre cero, correspondiente a cuando la boca se encuentra cerrada y mayor a cero dependiendo la magnitud de apertura. Si la magnitud de apertura de la boca es mayor o igual a la magnitud de un bostezo promedio se tendría la primera condición para detonar la alarma.

La segunda condición seria el tiempo que el usuario permanecería con la boca abierta. Si este rebasase los tres segundos se consideraría un bostezo, dando paso a que la alarma de somnolencia se active.

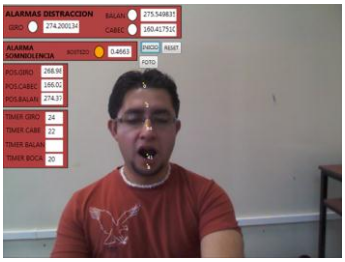


Figura 25. Alarma de Somnolencia encendida

IV. PRUEBAS Y RESULTADOS

Una vez que se realizó el código, el siguiente paso será instalar la cámara en un automóvil para la comprobación en una conducción real.

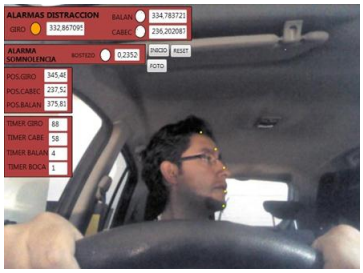


Figure 26. Alarma de distracción encendida, conducción real

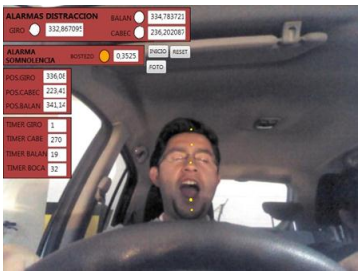


Figure 27. Alarma de somnolencia encendida, conducción real

Como indica la Figura 26 y Figura 27, el algoritmo sirve para una conducción real, alertando las situaciones erróneas del conductor y previniendo accidentes de tránsito.

V. CONCLUSIONES

- Las bondades que ofrece el kinect conjuntamente con sus librerías permiten que la programación se facilite, ofreciéndonos un sin número de posibilidades al momento de realizar aplicaciones de forma rápida y sencilla.
- El seguimiento de rostros ofrece una gran cantidad de aplicaciones, la mayoría ligadas al entretenimiento de todo tipo de edades, pero sin duda las aplicaciones orienorientadas a salvar vidas como la mencionada en este artículo como las que se dedican al control de robots de tal forma de realizar operaciones peligrosas para las personas son las que más se han desarrollado en los últimos años.
- El modelo CANDIDE es un modelo con más de 20 años de creación y ha sido ampliamente utilizado para el seguimiento de rostros en tiempo real, mediante los parámetros como las unidades de acción y vértices del modelo se logró determinar valores de coordenadas en los tres ejes para verificar distracción y somnolencia.
- La implementación de una alternativa tecnológica para la detección de situaciones erróneas al conducir un automotor usando el modelo CANDIDE es una aplicación interesante, ya que se usó la inteligencia artificial del sensor Kinect fusionada con la tecnología de tiempo de vuelo, desarrollando un algoritmo inteligente que detecte distracción y somnolencia en la conducción.

REFERENCIAS

[1] Comisión de Tránsito del Ecuador, http://www.cte.gob.ec
[2] R. Lange: 3D Time-of-flight distance measurement with custom solid-state image sensors in CMOS/CCD-technology. PhD Tesis, Universidad de Siegen, 2000.
[3] Alberto Beltrán Herrera, Sistema de tecnología de sexto sentido para dispositivos móviles, 2012.
[4] William Arthur Stewart "Bill" Buxton, Principal Researcher at Microsoft Research, http://www.billbuxton.com
[5] Robert Miles: Using Kinect for Windows with XNA, Kinect for Windows SDK, Marzo 2012, Edición 1.1
[6] Jarrett Webb, James Ashley: Beginning Kinect Programming with the Microsoft Kinect SDK, 2012
[7] Rob Miles: Learn Microsoft® Kinect API, 2012
[8] C.-H. Hjortsjö, *Människans ansikte och det mimiska språket* (In Swedish, "Man's Face and the Mimic Language", Studentlitertur, Lund, Sweden, 1969.
[9] P. Ekman and W. V. Friesen, *Facial Action Coding System*, Consulting Psychologist Press, 1977.
[10] R. Forchheimer and O. Fahlander; *Low bit-rate coding through animation*; *Proc. Picture Coding Symposium*, Davis, CA, USA, 1983.
[11] Sitio web del Grupo de Codificación de Imágenes de la Universidad de Linköping, http://www.icg.isy.liu.se/candide/, 1994.
[12] B. Welsh, *Model-Based Coding of Images*, PhD dissertation, British Telecom Research Lab, Jan. 1991.
[13] M. Rydfalk, *CANDIDE, a parameterized face*, Report No. LiTH-ISY-I-866, Dept. of Electrical Engineering, Linköping University, Sweden, 1987.

Alejandro Guevara nació en Quito-Ecuador en 1988. Recibió el titulo de bachiller en Electricidad-Electrónica en el colegio Don Bosco en el año 2006. En el año 2012 egresó de la Escuela Politécnica del Ejército de Ingeniería Electrónica en Automatización y Control. Actualmente se encuentra desarrollando su proyecto de tesis para la obtención del título de Ingeniería. Sus investigaciones de interés incluyen automatización industrial, domótica, redes industriales y energías renovables.

Sebastian Valencia nació en Quito-Ecuador en 1987, realizó sus estudios primarios en la escuela Borja 2 y sus estudios colegiales en el colegio Maristas de Quito. En el año 2012 egresó de la Escuela Politéc- nica del Ejército de Ingeniería Electrónica en Automatización y Control. Actualmente se encuentra desarrollando su proyecto de tesis para la obtención del título de Ingeniería.