UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

INTERFAZ HUMANO MÁQUINA CONTROLADA POR GESTOS

MEMORIA PARA OPTAR AL TITULO DE INGENIERO CIVIL ELÉCTRICO

ISMAEL FERNANDO ESCALONA NEIRA

PROFESOR GUÍA:
JAVIER RUÍZ DEL SOLAR

MIEMBROS DE LA COMISIÓN
PAUL VALLEJOS SÁNCHEZ
HÉCTOR AGUSTO ALEGRÍA

SANTIAGO DE CHILE
2014

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELÉCTRICO
POR: ISMAEL ESCALONA NEIRA
FECHA: 24 DE JUNIO DE 2014
PROF. GUÍA: JAVIER RUÍZ DEL SOLAR

INTERFAZ HUMANO MÁQUINA CONTROLADA POR GESTOS

El lenguaje corporal es importante para comunicarse fluidamente con las personas. En el ámbito de la interacción con máquinas, existen sistemas para reconocer automáticamente gestos, pero confunden cuerpos de color similar y sus capacidades son muy inferiores a las de los seres humanos. Para contribuir a la solución de este problema, se presenta una plataforma que sigue una esquina e identifica pulsaciones utilizando una webcam, independientemente del tono del objeto y del fondo, lo que se efectúa analizando variaciones luminosas.

El sistema registra imágenes con una cámara y las preprocesa para disminuir el ruido. Posteriormente, obtiene las zonas con cambios luminosos y reconstruye los objetos móviles. Luego, identifica las esquinas de los cuerpos, sigue la trayectoria de una de ellas y verifica si hay oscilaciones. La plataforma es complementada con módulos para configurar su funcionamiento, dibujar trayectorias y controlar un cursor. La programación se realiza en C++ y utiliza las librerías OpenCV para procesar imágenes y Qt para mostrar la interfaz de usuario.

El desempeño se evalúa con videos en que un dedo recorre trayectorias predefinidas. En las pruebas se utilizan varios tipos de iluminación, fondos, distancias a la cámara, posturas y velocidades de movimiento. Los resultados muestran que el algoritmo ubica el dedo con error promedio de 10 píxeles y detecta pulsaciones en el 82% de los intentos. Se producen fallas cuando hay fuentes de luz frente a la cámara, oscuridad o movimiento detrás de la mano. El programa se ejecuta a 30fps y utiliza el 16% de un procesador Intel Core i5-3337u.

La plataforma es capaz de distinguir objetos de tono similar, pero es poco tolerante a movimientos en el fondo. Una combinación de los métodos basados en variaciones luminosas y color puede corregir sus deficiencias mutuamente y habilitar aplicaciones que utilicen señales hechas con las manos para ordenadores personales y robots.

Dedicado a mi bisabuela y a mi mamá que me dieron los pilares fundamentales.

También a mis padrinos y seres queridos por apoyarme.

Agradecimientos

Expreso mi gratitud a los profesores Javier Ruíz del Solar, Paul Vallejos y Héctor Agusto por su confianza, consejos y voluntad para participar en una tesis propuesta por un alumno. A Roberto Román por responderme consultas y entregar mis avances en secretaría cuando estaba fuera de Santiago. También a mis tíos argentinos por ayudarme cuando se descompuso el computador en que trabajaba.

Tabla de contenido

Índice de tablas

Índice de ilustraciones

Capítulo 1: Introducción

1.1. Introducción

Una interfaz hombre máquina permite que las personas interactúen con aparatos. Se compone de mecanismos para controlar y consultar el estado del sistema. Los dispositivos más comunes y antiguos son de tipo electro-mecánico: botones, perillas, teclados, ratones, entre otros. Con el tiempo, estos sistemas se han ido haciendo intuitivos, de modo que las personas sin entrenamiento puedan utilizarlos con facilidad, como ocurre con los teléfonos celulares y ordenadores. La inteligencia artificial, reconocimiento de habla y visión computacional son objeto de investigación y en el futuro podrían engendrar aplicaciones en que personas y máquinas colaboren para realizar tareas eficientemente en hogares, oficinas, etc. [1], [2].

Existen herramientas para reconocer automáticamente indicaciones visuales simples. Algunas de estas iluminan la escena con luz infrarroja y examinan los reflejos para obtener gestos [3]. Otras aproximaciones emplean una cámara bidimensional y utilizan métodos de detección de piel [4] [5]. Ninguno de estos procedimientos equipara las capacidades humanas y ambos pueden fallar en presencia de objetos similares o en ciertos ambientes, debido a limitaciones de los sensores y algoritmos. Por otra parte, las webcams están presentes en prácticamente todos los ordenadores modernos, por lo que un desarrollo que haga uso de ellas y resuelva sus problemas podría tener utilidad inmediata para muchos usuarios.

El propósito de este documento es describir una plataforma para identificar gestos que utiliza una webcam. El programa detecta los objetos empleando los cambios luminosos que producen sus desplazamientos para prevenir que regiones de color parecido se confundan. El sistema puede rastrear una esquina y reconocer clics, lo que permite que el usuario controle el cursor de un ordenador -como con un mouse- y dibuje trayectorias utilizando su mano o un cuerpo con puntas.

La plataforma contribuye a entender si es ventajoso utilizar iluminación infrarroja para identificar gestos y si las limitaciones de los sistemas que utilizan una webcam se pueden superar. Por otra parte, el trabajo facilita desarrollos en la industria del entretenimiento, como controles para videojuegos y sistemas interactivos para ordenadores personales [6]. En robótica, el programa puede servir para reconocer indicaciones con las manos, como la señalización de objetos y comandos de acercamiento o alejamiento.

1.2. Objetivos generales

Desarrollar una plataforma para seguir la trayectoria de una esquina y detectar oscilaciones, independientemente de los colores del objeto y fondo.

1.3. Objetivos específicos

1. Caracterizar un cuerpo mediante los cambios luminosos que genera su movimiento.
2. Determinar la ubicación de las puntas.
3. Identificar la trayectoria y las pulsaciones efectuadas con un objeto.
4. Generar una aplicación para controlar un cursor y dibujar.

5. Evaluar el desempeño del programa en varias situaciones.

1.4. Estructura del documento

El Capítulo 2 revisa tecnologías para comunicar personas con máquinas. Primero se hace una reseña de dispositivos comunes y luego se presentan ejemplos basados en procesamiento de imágenes. Finalmente se discuten los requisitos que debe cumplir cualquier interfaz.

El Capítulo 3 presenta métodos de procesamiento de imágenes y de análisis de información relacionados con la plataforma. Se incluyen operaciones para mejorar cuadros, extraer puntos característicos y seguir objetos. También se describen herramientas de software para construir aplicaciones visuales.

El Capítulo 4 describe la implementación. Primero, se presenta una visión general sobre el funcionamiento de la plataforma. Posteriormente, se explica cada componente en detalle y se proporcionan los parámetros de funcionamiento.

El Capítulo 5 trata sobre el desempeño del sistema. Primero, se reportan y analizan los resultados de pruebas efectuadas en varias configuraciones y tipos de gestos. Posteriormente, se compara la plataforma con un trabajo que utiliza detección de piel.

Por último, el Capítulo 6 hace una conclusión general del trabajo. Se analizan los principales resultados y se presentan recomendaciones.

Capítulo 2: Revisión tecnológica

Una interfaz hombre máquina puede adoptar diversas formas. Hay dispositivos compuestos por botones que el usuario presiona para solicitar acciones puntuales. Otros sistemas utilizan objetos móviles, como ratones y lápices, que facilitan la interacción en entornos gráficos. Las interfaces basadas en voz y visión artificial permiten controlar máquinas sin contacto directo ni la utilización de accesorios. Cualquiera sea el caso, una interfaz bien hecha debe ser altamente confiable, fácil de utilizar y económicamente accesible.

2.1. Interfaces clásicas

Una de las primeras interfaces para computador fue el teclado, cuyo origen se remonta al telégrafo de fines del siglo XIX [7]. Posteriormente, se crea el lápiz óptico (1952) y el mouse (1963), con la finalidad de interactuar con gráficas [8]. A partir de los 80's, los ordenadores se integran a la vida cotidiana y emergen diversos mecanismos de interacción: joysticks, tabletas, pantallas táctiles, entre otros [9]. Algunos ya no se utilizan, mientras que otros se emplean en aplicaciones específicas. A continuación, se hace una reseña de algunos dispositivos usados en la actualidad.

- **Pantalla táctil**: Monitor para introducir información por contacto. Existen varios tipos de pantallas táctiles. Uno de ellos es la resistiva, que está compuesta por dos láminas de material medianamente conductivo. Estas tienen electrodos en los bordes y están separadas por aire. Cuando un objeto ejerce presión, las superficies chocan y se forma un divisor resistivo. Las coordenadas del punto de contacto se determinan con mediciones de voltaje para tensiones aplicadas en direcciones transversales.

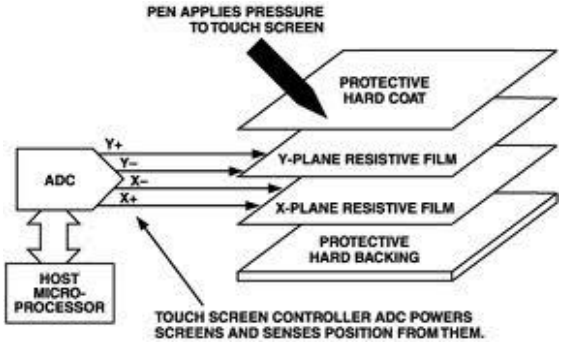

Ilustración 1: Pantalla táctil resistiva.

  La primera pantalla táctil fue construida en los 60's. Los ordenadores personales con esta tecnología surgieron en los 80's. En la última década, estos aparatos se han empleado masivamente en equipos celulares y computadoras [10].
- **Tableta digitalizadora**: Superficie que registra trazados hechos con un lápiz. En una implementación, el dispositivo está compuesto por un arreglo de alambres horizontales y verticales, que permiten determinar la ubicación y orientación del apuntador mediante inducción electromagnética.

Ilustración 2: Tableta digitalizadora.

La primera tableta para ordenador fue construida en 1957. Su uso se extendió en los 70's y 80's, principalmente en programas CAD y artes. Sus aplicaciones han ido aumentando hasta el presente [11].

- **Lápiz óptico**: Dispositivo que permite dibujar sobre una pantalla. En la punta, contiene un sensor luminoso que detecta el instante en que la imagen se actualiza. Esta información es transmitida a un ordenador, el que determina la posición sobre el monitor.

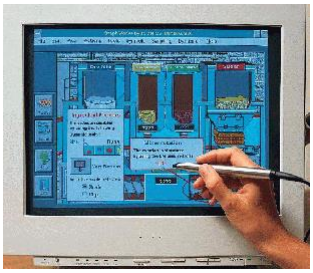

Ilustración 3: Lápiz óptico.

El primer lápiz óptico data de 1952 y fue de uso común en los 80's [12]. Actualmente, se emplea en diseño y artes, ya que permite dibujar con precisión.

- **Reconocimiento de habla**: Permite transformar un discurso en texto o comandos entendibles por una máquina. Utiliza un transductor para convertir el audio en una señal eléctrica, la que se procesa para obtener fonemas, palabras e instrucciones.

Ilustración 4: Reconocimiento de habla.

Los primeros sistemas de reconocimiento surgieron en los 50's y distinguían solamente números. En los 80's, era posible identificar miles de palabras habladas pausadamente. Posteriormente, se introdujeron las cadenas de Markov ocultas, que permiten identificar mensajes en base a probabilidades. Con esto, los errores disminuyeron y fue posible reconocer texto en discursos continuos. En los 90's surgen aplicaciones comerciales y en la siguiente década el sistema se incorpora a teléfonos celulares [13]. Si bien la tecnología ha avanzado bastante, aún requiere mayor confiabilidad e inteligencia.

2.2. Sistemas basados en imágenes

En las últimas dos décadas, la capacidad de cómputo de los ordenadores se ha multiplicado [14], lo que permite procesar gran cantidad de información y facilita el desarrollo de aplicaciones basadas en imágenes. A continuación se describen algunas interfaces de tipo visual desarrolladas en este período y que se encuentran disponibles al público.

- **LeapMotion**: Sistema de control con las manos. Utiliza un arreglo de emisores y sensores infrarrojos para triangular la posición de los dedos. Su uso requiere cercanía con el dispositivo y es intuitivo [15].

Ilustración 5: LeapMotion, sistema de control con las manos.

- **FLUTTER**: Aplicación para controlar música con las manos. Utiliza una webcam e identifica gestos estáticos. Requiere memorizar una serie de comandos y es compatible con algunas aplicaciones para ordenadores [16].

Ilustración 6: FLUTTER, sistema de control de música con las manos.

- **KINECT**: Sistema de interacción por gestos. Utiliza un emisor y receptor infrarrojos, una cámara y micrófonos. Permite identificar señales faciales, corporales y sonoras. Requiere de una consola Xbox o computadora [17].

Ilustración 7: KINECT, sistema de reconocimiento de gestos.

- **NPointer**: Programa para controlar un puntero con las manos. Utiliza una webcam y mueve el cursor según los desplazamientos del usuario. Su uso requiere calibración y es poco apto para movimientos precisos [18].

Ilustración 8: NPointer, software de control de puntero con las manos.

- **Enable Viacam**: Sistema para controlar un puntero con la cabeza. Utiliza una cámara web para adquirir imágenes y mueve un cursor según los desplazamientos del rostro. La aplicación está orientada a personas con discapacidad [19].

Ilustración 9: Enable Viacam, software de control de puntero con la cabeza.

- **PointGrab**: Sistema de control con las manos. Utiliza una webcam y reconoce gestos para mover un puntero y seleccionar objetos. Disponible a través de televisores inteligentes Samsung y computadoras Acer [20].

Ilustración 10: PointGrab, software de control de puntero con las manos.

2.3. Consideraciones

La construcción de una interfaz hombre máquina requiere variadas consideraciones. Por un lado, las características físicas y psicológicas de las personas deben ser tomadas en cuenta para que el dispositivo sea agradable y seguro. Por otra parte, el sistema se debe poder realizar con los recursos existentes, como presupuesto, tecnologías y conocimientos. Finalmente, el resultado debe garantizar un alto grado de confiabilidad para ser utilizable. Todos estos aspectos son críticos para que un HMI esté al alcance del público objetivo y sea confortable [21].

La ergonomía es una disciplina que estudia las capacidades intelectuales y físicas de las personas para la creación y mejora de productos, máquinas y entornos artificiales. Los resultados facilitan la utilización de dispositivos, aumentan la seguridad y mejoran la productividad. Cuando la ergonomía no es considerada, los usuarios se pueden cansar, necesitar largos períodos de aprendizaje y sufrir accidentes [22]. Por lo tanto, a la hora de diseñar equipamiento y sistemas interactivos, es importante organizar los recursos para que producto sea amigable con las personas.

Una interfaz hombre máquina se compone recursos y conocimientos provenientes de diferentes áreas. Por una parte, el hardware da forma física al dispositivo y proporciona los medios para su operación. Por otro lado, los procedimientos regulan el comportamiento de la aplicación ante las acciones del usuario o condiciones ambientales. A esto se suman consideraciones económicas, que determinan la factibilidad de realizar el diseño e inciden en la conveniencia de utilizar el dispositivo. El resultado, para que sea útil, debe ser accesible y cumplir con su cometido.

Una interfaz tiene que ser confiable para su utilización. Dependiendo del ámbito de aplicación, una falla puede generar desde molestia hasta accidentes. Por ejemplo, un vehículo autónomo puede manejarse a sí mismo y obedecer comandos verbales. En este caso, un error puede costar la vida de los ocupantes y peatones, por lo que este sistema tiene que funcionar a la perfección para que alguien se atreva a utilizarlo. Si bien es relativamente sencillo crear un prototipo para demostrar un concepto, llevarlo a la práctica requiere un nivel de desempeño superior.

Por lo tanto, una interfaz tiene que estar diseñada con consideraciones ergonómicas para facilitar y optimizar su utilización. También debe poseer una combinación de tecnología, conocimiento y recursos que produzcan los resultados deseados a un costo razonable. Por otro lado, el sistema requiere un alto grado de confiabilidad para ser utilizado en aplicaciones reales. Si se satisfacen estas necesidades, el dispositivo interactivo será amigable, accesible y certero.

Capítulo 3: Herramientas

La creación de una interfaz hombre máquina basada en imágenes demanda múltiples herramientas. Por una parte, las aplicaciones visuales utilizan un conjunto de métodos elementales para representar y transformar cuadros. Estos permiten construir detectores de movimiento, que sirven para localizar cuerpos dinámicos. Por otro lado, los sistemas de seguimiento requieren técnicas para identificar atributos -como esquinas y contornos- y examinar su comportamiento. Muchos de estos componentes se encuentran disponibles a través de bibliotecas de software, mientras que los otros se pueden implementar utilizando un lenguaje de programación.

3.1. Procesamiento digital de imágenes

3.1.1. Representación

Una imagen es una superficie en que la intensidad luminosa depende de las coordenadas espaciales. En los ordenadores, los cuadros se representan con una cantidad finita de puntos y el nivel de luz adopta valores discretos. Los retratos se organizan en tablas (Ilustración 11), donde cada celda se identifica mediante números de fila y columna y el contenido especifica el aspecto. Matemáticamente, una celda o píxel se simboliza como:

$$y(i,j)$$

Donde y es la intensidad luminosa para un punto ubicado en la fila i y columna j del cuadro.

100	100	0	0	0	0	255	255
50	0	50	50	50	50	255	255
50	0	50	255	255	255	255	255
50	0	50	255	255	255	255	255
50	0	50	255	255	255	255	255
50	0	50	50	50	50	255	255
100	100	0	0	0	0	255	255
100	100	50	50	50	50	255	255

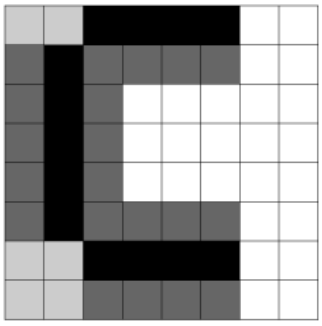

Ilustración 11: Imagen digital. (Izq.) Matriz. (Der.) Aspecto.

Habitualmente, se emplean 3 tipos de imágenes que difieren en los valores de sus celdas (Ilustración 12):

- **Imágenes binarias**: indican la presencia de atributos de interés mediante un valor que puede ser 0 o 1.
- **Imágenes en escala de grises**: codifican la intensidad luminosa total, usando un número que usualmente es entero y está en el rango que va desde 0 a 255.
- **Imágenes a color**: representan los tonos reconocibles por el ojo humano empleando 3 valores que especifican las intensidades luminosas para los colores rojo, verde y azul. Cada número es un entero entre 0 y 255.

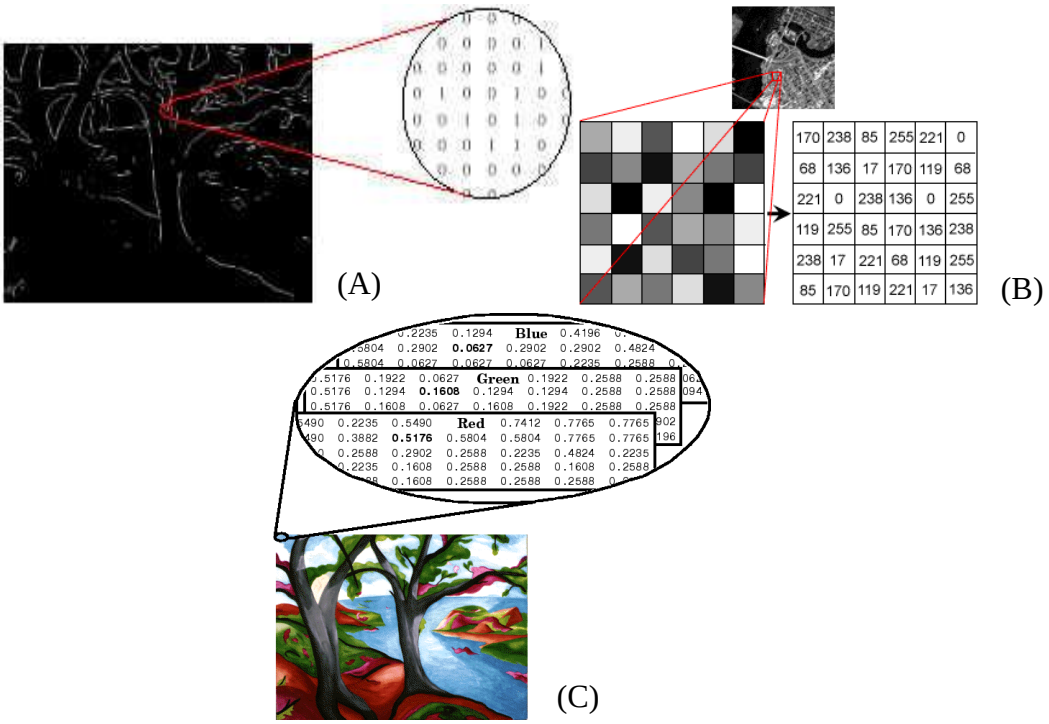

Ilustración 12: Imágenes (A) binaria, (B) en escala de grises y (C) a color.

3.1.2. Cambio de escala

Es una operación que modifica el número de píxeles que representa una imagen (Ilustración 13). Los nuevos puntos se calculan utilizando el cuadro original, intentando conservar el aspecto del retrato y minimizar la cantidad de cómputos.

Ilustración 13: Cambio de escala.

Existen varios métodos para cambiar la resolución. Uno de ellos es la interpolación al punto más próximo, que asigna a cada píxel el valor de la celda más cercana en el cuadro original (Ilustración 14).

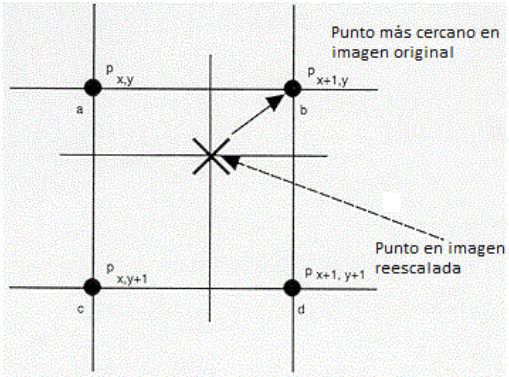

Ilustración 14: Aproximación al vecino más cercano.

Esta operación sirve para transformar una imagen al tamaño deseado y para comparar cuadros de dimensiones diferentes. También es útil cuando los objetos están representados por más píxeles que los necesarios, en cuyo caso permite reducir el número de celdas y aumentar la velocidad de procesamiento.

3.1.3. Binarización

Es una operación para imágenes en escala de grises, que asigna un valor binario a cada píxel según la intensidad luminosa. Una implementación, consiste en marcar las celdas en que la iluminación iguala o supera un umbral y borrar las demás. En este caso, la expresión matemática es:

$$y(i,j) = \begin{cases} 0 & , \ x(i,j) < u \\ 1 & , \ x(i,j) \geq u \end{cases}$$

Donde $x(i,j)$ son los píxeles de la imagen en escala de grises, u es el valor de referencia e $y(i,j)$ es el resultado. La Ilustración 15 muestra la binarización de una fotografía. La operación es útil para señalar en qué regiones están presentes los atributos de interés y simplificar objetos.

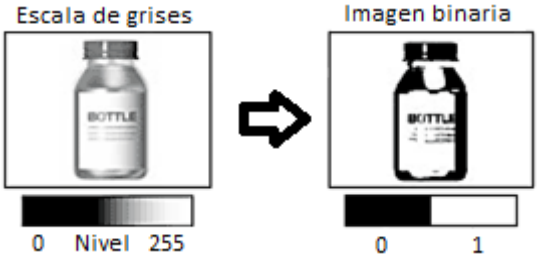

Ilustración 15: Binarización.

3.1.4. Convolución

Operación que calcula la suma ponderada de los píxeles contenidos en una vecindad [23]. Matemáticamente:

$$y(i,j) = \sum_r \sum_s k(r,s)x(i-r, j-s)$$

Donde $x(i,j)$ son los píxeles de la imagen a procesar, $y(i,j)$ es el resultado y $k(r,s)$ son constantes que multiplican los términos. Los factores se pueden representar por una matriz o kernel, que en el caso de una vecindad de 3x3 es:

$$k = \begin{pmatrix} k(-1,-1) & k(-1,-0) & k(-1,\ 1) \\ k(\ 0,-1) & k(\ 0,\ 0) & k(\ 0,\ 1) \\ k(\ 1,-1) & k(\ 1,\ 0) & k(\ 1,\ 1) \end{pmatrix}$$

El filtro requiere información que está fuera del cuadro para efectuar los cálculos en las orillas. En estas situaciones, se puede recortar la imagen de destino o extrapolar los datos faltantes.

El resultado de la convolución depende del tipo de kernel. Algunas posibilidades son efectos de difusión, reducción de ruido, realce de detalles, entre otros. Un ejemplo se muestra en la Ilustración 16, donde se utiliza una configuración para detectar bordes.

$$k = \begin{pmatrix} -1 & -1 & -1 \\ -1 & +8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Ilustración 16: Detección de bordes. (Arriba) Kernel. (Izq.) Imagen de prueba. (Der.) Resultado.

3.1.5. Dilatación

Expande el tamaño de los objetos en una imagen binaria. Esto se consigue marcando los alrededores de las celdas resaltadas. La vecindad se especifica mediante una máscara o elemento estructural. La expresión matemática es:

$$v(i,j) = \sum_r \sum_s k(r,s)x(i-r, j-s)$$

$$y(i,j) = \begin{cases} 0 & , v(i,j) = 0 \\ 1 & , v(i,j) > 0 \end{cases}$$

En la primera ecuación, v es una función que cuenta las celdas resaltadas alrededor del punto (i,j) en la imagen x. La vecindad se especifica mediante un elemento estructural k que es una matriz binaria. En la segunda fórmula, y es la imagen dilatada, que tiene marcados todos los píxeles en que v vale por lo menos 1. La Ilustración 17 muestra un ejemplo de la operación.

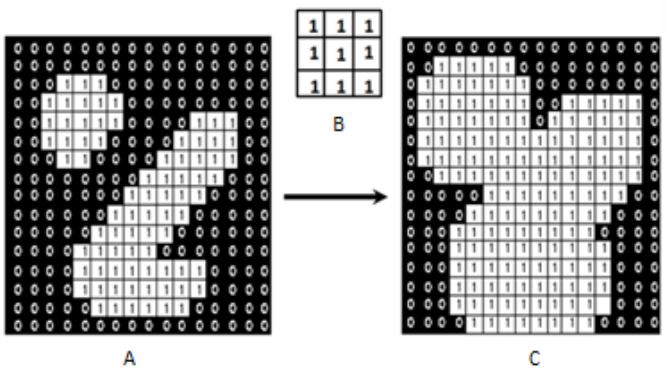

Ilustración 17: Dilatación. (A) Imagen de prueba. (B) Elemento estructural. (C) Resultado.

La dilatación permite eliminar orificios en estructuras y unir objetos fragmentados. Esto sirve para simplificar la representación y reducir perturbaciones. Mayor información se puede encontrar en [24].

3.1.6. Erosión

Reduce el tamaño de los objetos de una imagen binaria. Esto se efectúa borrando los puntos cercanos a celdas oscuras. La fórmula es:

$$v(i,j) = \sum_r \sum_s k(r,s)x(i-r, j-s)$$

$$y(i,j) = \begin{cases} 0 & , v(i,j) < M \\ 1 & , v(i,j) = M \end{cases}$$

Estas expresiones son similares al caso de dilatación. La diferencia es que la imagen erosionada y tiene borrados todos los píxeles en los que v no alcanza su valor máximo M. La Ilustración 18 muestra un ejemplo de la operación.

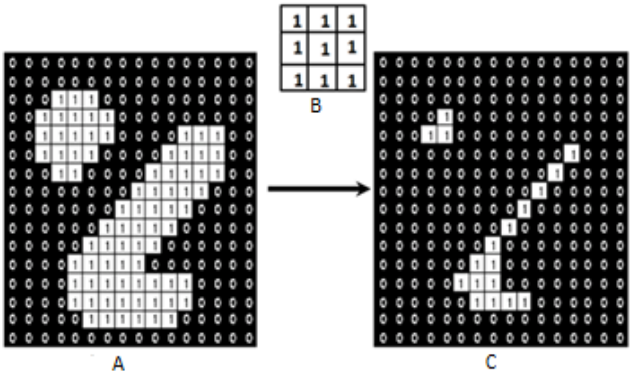

Ilustración 18: Erosión. (A) Imagen de prueba. (B) Elemento estructural. (C) Resultado.

La erosión ayuda a eliminar regiones pequeñas y desconectar estructuras. Esto es útil para filtrar ruido y deshacer uniones accidentales entre objetos inconexos.

3.1.7. Operaciones lógicas

Son cálculos booleanos que se realizan con las celdas de una o varias imágenes binarias. Estos procedimientos son útiles para comparar cuadros, combinar información, seleccionar objetos, entre otros. Algunas operaciones son:

- **Inversión**: $z(i,j) = \neg x(i,j)$
- **Unión**: $z(i,j) = x_1(i,j) \cup x_2(i,j)$
- **Intersección**: $z(i,j) = x_1(i,j) \cap x_2(i,j)$
- **Sustracción**: $z(i,j) = (x_1(i,j) \cup x_2(i,j)) \cap x_1(i,j)$

Donde $\neg$ es la negación, $\cup$ es la unión y $\cap$ es la intersección. Los píxeles de los cuadros de entrada son $x(i,j)$, $x_1(i,j)$ y $x_2(i,j)$. El resultado es $z(i,j)$. La Ilustración 19 muestra ejemplos de operaciones lógicas con imágenes.

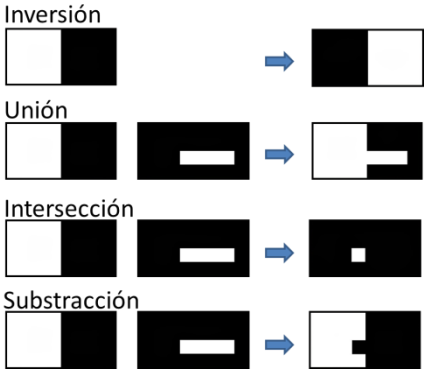

Ilustración 19: Operaciones lógicas.

3.2. Detección de movimiento

3.2.1. Diferencia de cuadros

Resta dos imágenes y registra los lugares en que los cambios luminosos superan un umbral. Cuando el recinto tiene iluminación estática, el resultado señala las zonas en que hay objetos desplazándose. Formalmente, la operación se puede expresar como:

$$y(i,j) = \begin{cases} 0 & , \quad |x_1(i,j) - x_2(i,j)| < u \\ 1 & , \quad |x_1(i,j) - x_2(i,j)| \geq u \end{cases}$$

Donde $x_1(i,j)$ y $x_2(i,j)$ son los píxeles de las imágenes que se comparan, $y(i,j)$ es el resultado de la operación y u es la diferencia mínima para marcar un punto. La Ilustración 20 muestra el procedimiento para dos cuadros de ejemplo.

Ilustración 20: Diferencia de cuadros.

El método es simple y registra cambios eficientemente. Sin embargo, puede fallar cuando la cámara vibra, los objetos se mueven lento o la iluminación varía. Por otra parte, el resultado entrega el área ocupada por un cuerpo en dos instantes, lo que impide conocer su forma exacta.

3.2.2. Doble diferencia de cuadros

Emplea tres imágenes 1, 2 y 3. Calcula la diferencia de cuadros entre las dos primeras, lo que marca las regiones del cuerpo móvil en los instantes 1 y 2. Luego, hace lo mismo con las dos últimas, lo que registra las formas en los momentos 2 y 3. Al final, los resultados son intersectados, lo que entrega el objeto en el cuadro 2. Matemáticamente:

$$z_2(i,j) = y_{12}(i,j) \cap y_{23}(i,j)$$

Donde y_{12} e y_{23} son las diferencias de cuadros 1-2 y 2-3 respectivamente, ∩ es la intersección y z_2 es el resultado para el instante 2.

El procedimiento identifica los objetos móviles de forma precisa. Sin embargo, los cuerpos se detectan en la penúltima imagen, lo que genera un retraso temporal en aplicaciones que se ejecutan en tiempo real. El algoritmo puede fallar cuando los objetos son uniformes, el movimiento cesa o la cámara se desplaza.

3.2.3. Substracción de fondo

Genera un modelo del fondo y calcula la diferencia con una imagen. El resultado destaca los objetos en primer plano. Formalmente:

$$m_t(i,j) = (1-\alpha)\, m_{t-1}(i,j) + \alpha\, x\,(i,j), \qquad 0 \le \alpha \le 1$$

$$y(i,j) = \begin{cases} 0 & , \quad |x(i,j) - m_t(i,j)| < u \\ 1 & , \quad |x(i,j) - m_t(i,j)| \ge u \end{cases}$$

Donde m_t es el modelo, x es una imagen, α es la tasa de aprendizaje, t es el tiempo, u es el umbral e y es el resultado. La primera ecuación genera una representación del fondo que se actualiza iterativamente con los cuadros registrados. La segunda expresión obtiene los objetos que se posan en la escena.

El procedimiento entrega la forma exacta de los cuerpos, sin retardos y tolera detenciones breves. Sin embargo, es sensible a vibraciones de la cámara y falla cuando se producen cambios bruscos en el fondo. Una aplicación de esta técnica se puede encontrar en [25].

3.3. Descripción de objetos

3.3.1. Esquinas

Son puntos donde se intersectan dos bordes. Existen varios algoritmos para la detección de esquinas. Uno que destaca por su eficiencia y simplicidad es FAST (*Features from Accelerated Segment Test*) [26], el cual reconoce como vértices los puntos que tienen varias celdas cercanas con brillo marcadamente distinto (Ilustración 21).

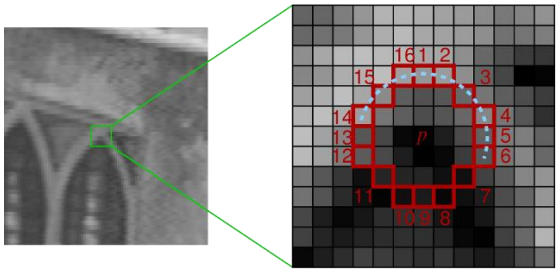

Ilustración 21: Detección de esquina con FAST.

3.3.2. Contornos

Un contorno se caracteriza por una discontinuidad en la iluminación o color entre dos regiones [27]. Habitualmente, coincide con bordes, sombras o estructuras de objetos. La Ilustración 22 muestra las aristas de una fotografía.

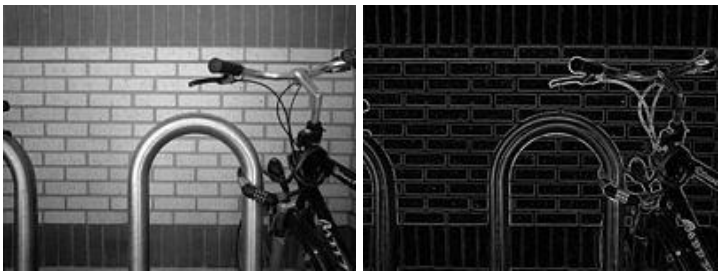

Ilustración 22: (Izq.) Imagen de prueba. (Der.) Contornos.

Los bordes pueden ser empleados para caracterizar cuerpos. En el computador se representan mediante una secuencia de coordenadas o segmentos orientados. También pueden ser descritos usando curvatura. A menudo, se emplean mecanismos para simplificarlos y procesarlos con mayor facilidad, como aproximaciones por segmentos rectilíneos y selección de puntos con curvatura extremas.

3.3.3. Caracterización.

El reconocimiento de un atributo en múltiples imágenes, requiere el empleo de identificadores descriptivos e independientes de la orientación, rotación, iluminación y escala. Este tipo de información permite comparar entidades y decidir si son la misma.

La información contenida en un descriptor depende del tipo de aplicación. Por ejemplo, las bolas de billar (Ilustración 23) se pueden caracterizar por sus colores y áreas de cobertura. Estos datos describen unívocamente a cada elemento y son invariantes a cambios en las escenas, lo que facilita su reconocimiento en diferentes cuadros.

Ilustración 23: Bolas de billar.

Hay algoritmos que generan descriptores aplicables a una gran cantidad de situaciones. Por ejemplo, SIFT (*Scale-invariant feature transform*) selecciona características independientes de la escala, orientación y variaciones parciales de iluminación y rotación [28].

3.4. Seguimiento

3.4.1. Correspondencias

Una forma de seguir un objeto en un video consiste en identificar sus puntos característicos en cada cuadro. Esto requiere comparar los descriptores del cuerpo con los de las imágenes, lo que se realiza mediante una métrica. La más común es la Euclidiana:

$$d(\vec{p},\vec{q}) = \sqrt{\sum_{i=0}^{N}(p_i - q_i)^2}$$

Donde $\vec{p}$ y $\vec{q}$ son descriptores de N elementos, p_i y q_i son sus componentes y $d(\vec{p},\vec{q})$ es la distancia entre ellos. Otra norma es la L_1, que destaca por su facilidad de cálculo:

$$d_{L_1}(\vec{p},\vec{q}) = \sum_{i=0}^{N}|p_i - q_i|$$

Un método para identificar un punto característico en una imagen es calcular la distancia con todos los descriptores del cuadro e identificar el más parecido. Existen algoritmos que hacen esto eficientemente, pero son sofisticados y su desempeño depende de las características de la indagación. Varios métodos de búsqueda están disponibles a través de la librería FLANN (*Fast Library for Approximate Nearest Neighbors*) [29], que puede, además, escoger automáticamente la técnica más adecuada para cada situación.

3.4.2. Filtro de partículas

Es un algoritmo que aproxima el estado de una planta en base a datos experimentales y modelos de comportamiento y medición [30]. La versión más simple del filtro, describe sistemas que se pueden representar mediante:

$$\vec{u}_k = \vec{g}(\vec{u}_{k-1}) + \delta\vec{u}_k \quad , \delta\vec{u}_k \sim W$$

$$\vec{z}_k = \vec{h}(\vec{u}_k) + \delta\vec{z}_k \quad , \delta\vec{z}_k \sim V$$

La primera ecuación es el modelo de comportamiento. En esta $\vec{u}_{k-1}$ y $\vec{u}_k$ describen el estado en los instantes $k-1$ y k respectivamente, $\vec{g}(\vec{u}_{k-1})$ es una función de predicción y $\delta\vec{u}_k$ representa perturbaciones aleatorias con distribución W. La segunda ecuación es el proceso de medición. En esta $\vec{z}_k$ es la información que percibe un sensor, $\vec{h}(\vec{u}_k)$ es una función que vincula las observaciones con el estado de la planta y $\delta\vec{z}_k$ modela errores aleatorios con distribución V.

El filtro de partículas puede operar con modelos no lineales y perturbaciones de cualquier tipo. El estado y las mediciones pueden ser magnitudes escalares o vectoriales de dimensión arbitraria. Esto permite describir con precisión una amplia variedad de plantas.

El algoritmo genera un conjunto de hipótesis sobre el estado del sistema y les asocia una puntuación que refleja su consistencia con las mediciones. Formalmente:

$$I_k = \{(\vec{u}_k^r, \omega_k^r)\} : r \in \{1, \dots, N\}\}$$

Donde I_k es un conjunto de N elementos, cada uno compuesto por una conjetura sobre el estado de la planta $\vec{u}_k^r$ en un instante k y su evaluación ω_k^r. El filtro opera en 3 etapas que se repiten iterativamente:

1. **Predicción**: se generan hipótesis del próximo estado usando la descripción actual y el modelo de comportamiento:
$$I'_{k+1} = \{(\vec{u}_{k+1}^{r\prime}, \omega_k^r)\} : r \in \{1, \dots, N\}\}$$
$$\vec{u}_{k+1}^{r\prime} = \vec{g}(\vec{u}_k^r) + \delta\vec{u}_{k+1}^r \quad , \delta\vec{u}_{k+1}^r \sim W$$
Donde la primera expresión es el conjunto de postulados sobre el estado futuro de la planta y la segunda es el modelo de comportamiento para elaborar predicciones.
2. **Evaluación**: se califica la consistencia de las hipótesis con las mediciones:
$$I''_{k+1} = \{(\vec{u}_{k+1}^{r\prime}, \omega_{k+1}^{r\prime})\} : r \in \{1, \dots, N\}\}$$
$$\omega_{k+1}^{r\prime} = \omega_k^r \, p(\vec{z}_{k+1} | \vec{u}_{k+1}^{r\prime})$$
La primera expresión es el conjunto de conjeturas con los pesos actualizados y la segunda es la calificación de los estados. Las nuevas evaluaciones son proporcionales a las anteriores y a la probabilidad de medir $\vec{z}_{k+1}$ dado el estado $\vec{u}_{k+1}^{r\prime}$.
3. **Selección**: se extraen N elementos al azar desde el conjunto de hipótesis. La probabilidad utilizada para seleccionar cada conjetura es proporcional a su puntuación, lo que, en general, hace que las partículas con alta valoración sean escogidas varias veces. Los postulados resultantes son calificados con un valor constante. El proceso se condensa en las siguientes ecuaciones:
$$I_{k+1} = \{(\vec{u}_{k+1}^r, \omega_{k+1}^r)\} : r \in \{1, \dots, N\}\}$$
$$p(\vec{u}_{k+1}^r, \omega_{k+1}^r) \propto \omega_{k+1}^{r\prime}$$
$$\omega_{k+1}^r = \frac{1}{N}$$
Donde, la primera expresión es el conjunto de conjeturas escogidas, la segunda es la probabilidad de selección y la tercera son las nuevas evaluaciones.

El algoritmo se puede usar para seguir cuerpos móviles empleando información ruidosa. Para esto, la ubicación se establece como variable de estado, la dinámica se describe con un modelo cinemático y las mediciones se obtienen desde sensores, que pueden ser escogidos a conveniencia. El método puede prevenir fallas causadas por oclusiones breves, efectuando estimaciones con el modelo de movimiento. Sin embargo, cuando no hay mediciones por un tiempo prolongado o el modelo tiene alta incertidumbre, el procedimiento puede entregar resultados erróneos.

3.4.3. Otros procedimientos

Existen numerosos métodos para estimar el estado de una planta. Entre ellos hay varias versiones del filtro de partículas que apuntan a aumentar la eficiencia, mejorar resultados o describir sistemas diferentes. Otro procedimiento ampliamente utilizado es el filtro de Kalman [31], que se enfoca en plantas lineales con perturbaciones gaussianas:

$$\vec{u}_k = A\vec{u}_{k-1} + B\delta\vec{u}_k \quad , \delta\vec{u}_k \sim n(0,1)$$

$$\vec{z}_k = C\vec{u}_k + D\delta\vec{z}_k \quad , \delta\vec{z}_k \sim n(0,1)$$

La primera ecuación indica que el estado siguiente es una combinación lineal de la situación previa y de las perturbaciones. La segunda señala que la medición es una suma ponderada del estado y los errores. En ambas, las desviaciones son aleatorias con distribución normal.

Los sistemas lineales con ruido gaussiano son resueltos eficientemente por el filtro de Kalman. Sin embargo, el desempeño se degrada cuando el comportamiento no es lineal o las perturbaciones no obedecen distribuciones normales. En estas situaciones, una mejor elección es el filtro de partículas, que puede funcionar con menos restricciones en el comportamiento de la planta y es más tolerante a desviaciones.

3.5. Software

3.5.1. C++

Es un lenguaje de programación multipropósito [32], desarrollado a principios de los 80’s para soportar proyectos de gran envergadura y generar código máquina eficiente. Hoy en día, es ampliamente utilizado en informática y está presente en una gran variedad de equipos y sistemas operativos. Algunas aplicaciones incluyen programación de sistemas, drivers, entretenimiento, entre otros.

El lenguaje es compilado y puede utilizar objetos. La versión estándar incluye librerías que proveen operaciones de lectura/escritura de archivos, contenedores de datos, herramientas para manipular información, etc. También incorpora plantillas, que permiten elaborar códigos genéricos para procesar varios tipos de datos.

3.5.2. OpenCV

Librería para procesamiento de imágenes escrita en C++ [33], que puede ser usada en varios lenguajes de programación, sistemas operativos y plataformas de hardware. Su desarrollo

comenzó en 1999 para fomentar la utilización de aplicaciones visuales. Es de código abierto y puede ser empleada libremente.

La biblioteca provee procedimientos para manipular imágenes y videos. También incorpora herramientas de inteligencia artificial. Permite realizar tareas de adquisición y escritura de imágenes, transformaciones, reconocimiento y seguimiento de objetos, etc. Los métodos de OpenCV están altamente optimizados y utilizan recursos de hardware para acelerar la ejecución de los programas.

3.5.3. Qt

Librería escrita en C++ que facilita la creación de aplicaciones con interfaz de usuario o línea de comandos [34]. Soporta múltiples lenguajes de programación, sistemas operativos y tipos de hardware. Su desarrollo se inicia en 1991 y ha continuado ininterrumpido hasta la fecha. Está disponible gratuitamente para aplicaciones no comerciales.

Proporciona métodos para crear interfaces de usuario e interactuar con el cliente. Permite generar ventanas, botones, cuadros de texto, etc. También posibilita el despliegue de contenido audiovisual, animaciones, entre otros.

Capítulo 4: Implementación

La construcción de una plataforma para rastrear una esquina y detectar gestos de clic con una webcam demanda la implementación de varios subsistemas. En primer lugar, es necesario un componente que pueda diferenciar el cuerpo de interés de los demás objetos en la escena. Luego, se requieren mecanismos para localizar vértices, sus desplazamientos y reconocer agitaciones. Por último, es necesaria una interfaz para que el usuario pueda utilizar la plataforma. Para que el sistema sea útil, su funcionamiento debe ser eficiente y confiable, lo que se consigue con algoritmos de baja complejidad y tolerantes a varios ambientes.

4.1. Descripción general

La plataforma procesa las imágenes obtenidas desde una webcam para controlar un cursor y dibujar trayectorias. Su funcionamiento se organiza en tres bloques: contornos móviles, puntos de interés e interfaz de usuario (Ilustración 24). En el primero se determinan los bordes externos de los objetos que se desplazan. En el segundo se localizan y rastrean esquinas y se reconoce el gesto de clic. En el último se despliegan opciones de configuración, cursores y trazados. Estos componentes permiten que el usuario controle un ordenador moviendo cualquier objeto con vértices.

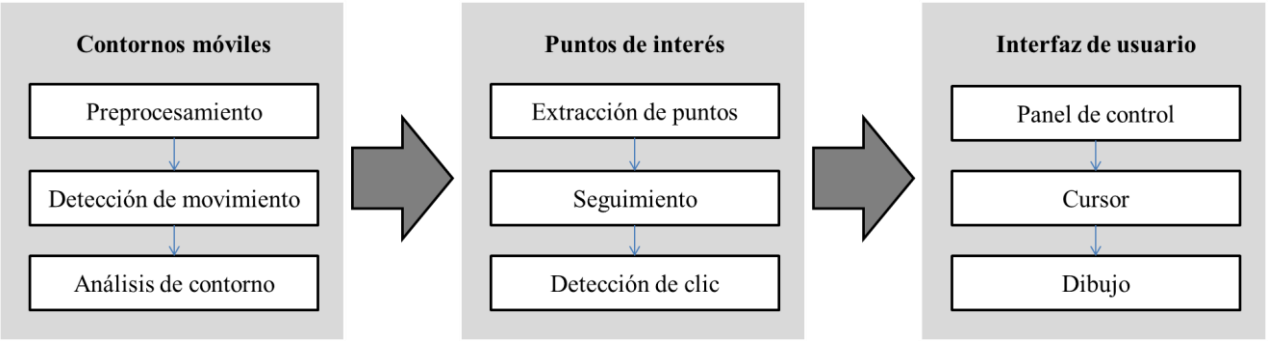

Ilustración 24: Diagrama de bloques de implementación.

Los cuerpos móviles son detectados con doble diferencia de cuadros, lo que permite delimitar los bordes con precisión en escenarios complejos. El ruido de las imágenes puede generar falsas detecciones, lo que se corrige con operaciones de difusión, binarización y erosión. El algoritmo fragmenta los cuerpos debido a que no registra los sectores uniformes, lo que se remedia extrapolando las regiones con orificios y reparando las fronteras. El resultado es el contorno externo de los objetos móviles sin estructuras interiores ni transiciones abruptas.

Las esquinas son extraídas desde el borde exterior. Esto se realiza midiendo la distancia de cada punto al centro de masa y seleccionando las ubicaciones más distantes. El procedimiento puede arrojar detecciones en zonas poco pronunciadas o varias marcas asociadas a una misma punta, lo que es verificado y corregido. Estos algoritmos entregan los vértices de los objetos, que en el caso de una mano corresponden a los extremos de los dedos.

Los desplazamientos de una esquina son identificados con un filtro de partículas. El algoritmo genera conjeturas sobre la posición, rapidez y dirección de movimiento del vértice, las que son seleccionadas según su consistencia con las puntas localizadas en las imágenes. El procedimiento incorpora mecanismos para discriminar el punto de interés entre varias alternativas y estimar su evolución cuando no hay información por períodos breves. Estas rutinas generan la trayectoria de una esquina y predicen su ubicación.

Los gestos de clic son vibraciones producidas por el usuario en la vecindad de la esquina. Estas se manifiestan como cambios luminosos intermitentes, que son monitoreados por la plataforma. Las pulsaciones se comprueban solamente cuando los objetos están en reposo para prevenir detecciones accidentales durante desplazamientos. De este modo, cada vez que el usuario se detiene y agita temporalmente su mano sin realizar un desplazamiento neto, el programa lo notifica como un clic.

La trayectoria de una esquina y las pulsaciones son utilizadas para controlar un cursor –como un mouse- y dibujar con la mano. Los parámetros del programa son accesibles al usuario a través de un panel de control que permite ajustar el uso de procesador, precisión, sensibilidad, entre otros aspectos. La plataforma está programada para el sistema operativo Windows 8 con el lenguaje C++ y las librerías OpenCV y Qt.

4.2. Módulos

4.2.1. Contornos móviles

Preprocesamiento

Los cuadros entregados por la webcam son modificados para disminuir el ruido y facilitar el trabajo. Primero, las imágenes se transforman a escala de grises para reducir la cantidad de información a procesar. Luego, se aplica una operación de difusión de 5x5 que reduce fuertemente las fluctuaciones de la cámara, pero puede alterar los bordes (Ilustración 25). El resultado entrega una imagen que puede ser procesada con facilidad y que simplifica el reconocimiento de objetos.

Ilustración 25: Preprocesamiento. Imágenes (izq.) original, (cen.) monocromática y (der.) suavizada.

Detección de movimiento

Las regiones ocupadas por objetos móviles se identifican con doble diferencia de cuadros, lo que es combinado con operaciones de dilatación y erosión para mejorar los resultados. Primero, se

resta un par de imágenes para obtener las variaciones luminosas (Ilustración 26). Posteriormente, se aplica un umbral para discriminar zonas activas, con un valor de 5 en la escala de 0 a 255. Luego, se efectúan erosiones para eliminar ruido y dilataciones para conectar segmentos, cada una con dos iteraciones usando un núcleo de 3x3 (Ilustración 27). Finalmente, se intersectan dos máscaras de movimiento consecutivas, lo que entrega la ubicación de los cuerpos móviles con precisión (Ilustración 28).

Ilustración 26: Diferencia de cuadros. Imágenes (izq.) anterior, (cen.) posterior y (der.) resta.

Ilustración 27: (Izq.) Binarización. (Der.) Operaciones morfológicas.

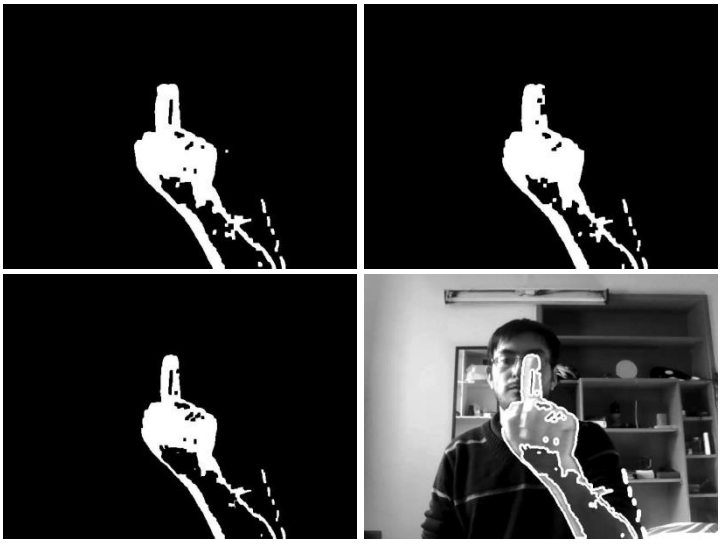

Ilustración 28: Intersección de cuadros consecutivos. (Arriba) Diferencia de cuadros anterior y posterior. (Abajo) Intersección y superposición con imagen original.

Los procedimientos filtran el ruido mediante cuatro mecanismos. Primero, la operación de difusión efectuada en la etapa de preprocesamiento atenúa las fluctuaciones luminosas. Luego, la binarización rechaza las perturbaciones que no exceden un umbral. Posteriormente, la operación de erosión elimina pequeñas regiones aisladas. Por último, la intersección de cuadros descarta

puntos que cambian de lugar aleatoriamente. Los efectos de omitir algunos de estos pasos se muestran en la Ilustración 29.

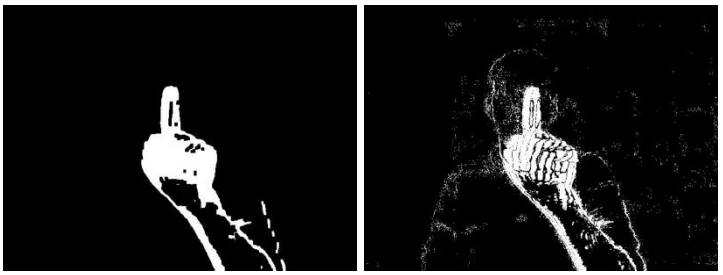

Ilustración 29: Doble diferencia de cuadros con (izq.) y sin (der.) difusión, erosión ni dilatación.

Análisis de contorno

Los objetos se reconstruyen a partir de los fragmentos encontrados durante la detección de movimiento. En primer lugar, se calculan las distancias mínimas entre cada par de marcas utilizando sus contornos (Ilustración 30). Luego, se vinculan los sectores en que la separación no excede un valor umbral, de lo que resultan cuerpos de mayor tamaño. Posteriormente, se determina el centro de masa de cada entidad y se extraen los puntos más alejados en cada dirección, lo que produce el borde exterior (Ilustración 31). Este procedimiento puede crear orificios apuntando hacia el centro en las regiones no detectadas durante la doble diferencia de cuadros, los que son sustituidos por rectas tangentes a la figura. De este modo, se obtiene un borde suave y consistente con la geometría de los objetos (Ilustración 32).

Ilustración 30: (Izq.) Regiones móviles. (Der.) Contornos de objeto.

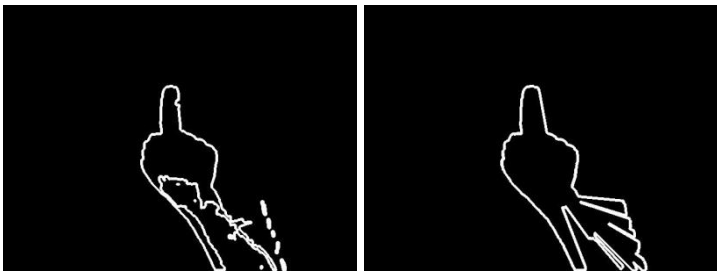

Ilustración 31: (Izq.) Contornos de brazo. (Der.) Borde externo.

Ilustración 32: Corrección de discontinuidades. (Izq.) Contorno original. (Cen.) Borde corregido. (Der.) Superposición en imagen original.

La implementación utiliza una distancia umbral de 60 píxeles para discriminar si dos manchas forman parte de un mismo cuerpo. El borde exterior se determina buscando los puntos más distantes en 1440 direcciones equiespaciadas con origen en el centro de masa. En la corrección de discontinuidades, se rechaza cualquier orificio de al menos 10 píxeles de profundidad, que apunte hacia el centro con un error máximo de 5° y con un arco inferior a 10°.

4.2.2. Puntos de interés

Extracción

Los contornos se caracterizan mediante esquinas, puesto que son fáciles de procesar y útiles en tareas de reconocimiento. Para su detección, se calcula la distancia del borde al centro de masa y se seleccionan las posiciones que están más alejadas respecto a los elementos adyacentes. El método puede arrojar falsos positivos en discontinuidades, segmentos planos o cerca de los vértices. Esto se corrige comprobando si el contorno desciende al menos un 20% al centro de masa, lo que permite descartar las posiciones ubicadas en rugosidades y curvas planas. Las marcas duplicadas se eliminan midiendo la distancia entre los candidatos a esquinas y seleccionando el punto más alejado del origen cuando la separación es menor que 50 píxeles. La Ilustración 33 muestra un ejemplo donde se registran las posiciones inicialmente seleccionadas y las corregidas.

Ilustración 33: Detección de extremos. (Izq.) Máximos locales. (Cen.) Selección. (Der.) Superposición en imagen original.

Seguimiento

La trayectoria de una esquina se determina con un filtro de partículas (Ilustración 34), el que se ha escogido por su facilidad de implementación y flexibilidad. El algoritmo utiliza un modelo de movimiento para describir el comportamiento de una punta y un modelo de medición para corregir las estimaciones con los datos de las imágenes. También incorpora modificaciones para operar cuando la detección de esquinas falla o se localizan varios vértices simultáneamente. Los

parámetros del programa están ajustados para conseguir un buen desempeño con un uso moderado de procesador.

Ilustración 34: (Izq.) Localización y (der.) trayectoria calculados con filtro de partículas.

Modelo de movimiento

El comportamiento de cualquier objeto físico se rige por las leyes de Newton. Estas señalan que la velocidad de un cuerpo cambia gradualmente al aplicar una fuerza finita, por lo que la rapidez es prácticamente constante en un intervalo temporal pequeño. En períodos prolongados, el comportamiento puede ser complejo e impredecible, lo que puede representarse por una sucesión de deslizamientos rectilíneos uniformes que fluctúan aleatoriamente de un instante a otro. En el caso de una esquina moviéndose en una imagen, la dinámica se puede describir como:

$$\vec{u}_{i+1} = A\,\vec{u}_i + \delta\vec{u}_i, \quad \vec{u}_i = \begin{pmatrix} x_i \\ y_i \\ v_i^x \\ v_i^y \end{pmatrix}, \quad A = \begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Donde $\vec{u}_i$ es el estado en el instante i, $\vec{u}_{i+1}$ es el estado siguiente, (x_i, y_i) son las coordenadas, (v_i^x, v_i^y) son las velocidades, A es la matriz de transición, Δt es el intervalo temporal y $\delta\vec{u}_i$ es el vector de perturbaciones. En la implementación, el ruido ha sido modelado con una distribución gaussiana de media $\vec{\mu}$ y covarianza Σ, la cual puede ser ajustada para representar una gran cantidad de perturbaciones:

$$p(\delta\vec{u}_i) = \frac{1}{\sqrt{(2\pi)^4|\Sigma|}} \exp\left(-\frac{1}{2}(\delta\vec{u}_i - \vec{\mu})^T \Sigma^{-1}(\delta\,\vec{u}_i - \vec{\mu})\right)$$

Modelo de medición

El programa estima la posición y velocidad de una esquina utilizando los vértices localizados en las imágenes. Estos contienen errores introducidos por la cámara de video y los métodos de detección, lo que se puede expresar formalmente como:

$$\vec{z}_i = C\,\vec{u}_i + \delta\vec{z}_i, \quad \vec{z}_i = \begin{pmatrix} x_i' \\ y_i' \end{pmatrix}, \quad \vec{u}_i = \begin{pmatrix} x_i \\ y_i \\ v_i^x \\ v_i^y \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Donde $\vec{z}_i = (x_i', y_i')$ son las observaciones, C es la matriz de medición y $\delta\vec{z}_i$ es el vector de perturbaciones. El ruido es descrito con una distribución de Gauss con media $\vec{v}$ y covarianza T:

$$p(\delta\vec{z}_i) = \frac{1}{\sqrt{(2\pi)^4|\mathrm{T}|}}\exp\left(-\frac{1}{2}(\delta\vec{z}_i - \vec{v})^T\mathrm{T}^{-1}(\delta\,\vec{z}_i - \vec{v})\right)$$

Adaptaciones

El filtro de partículas puede fallar cuando no se detectan esquinas, debido a que necesita mediciones para funcionar. Esto puede suceder en dos tipos de situaciones, las que deben ser distinguidas y tratadas por separado. Un caso se produce cuando los vértices no son detectados por periodos breves, lo que se resuelve haciendo que el algoritmo itere sin calificar las hipótesis hasta que registre nueva información. El otro escenario ocurre cuando los objetos se detienen, lo que se soluciona anulando la velocidad de las partículas y ubicándolas en la última posición conocida hasta que encuentre una esquina cercana. Estas situaciones se diferencian midiendo el tiempo en que no se registran observaciones, que es pequeño para omisiones transitorias y largo para frenados.

Durante la operación del programa se pueden detectar múltiples esquinas el mismo tiempo. En este caso, el filtro de partículas convencional es incapaz de discriminar cuál punto corresponde al objeto que está siguiendo. Este problema es resuelto evaluando cada hipótesis con todos los vértices y seleccionando la puntuación más alta, lo que favorece las ubicaciones que calzan mejor con las conjeturas. Matemáticamente:

$$\omega_{i+1}^r = \max\left(\left\{\omega_i^r p(\,\vec{z}_{i+1}^n|\vec{u}_{i+1}^r) : n \in \{1,\dots,M\}\right\}\right)$$

Donde $\omega_{i+1}^{r\prime}$ la calificación de la conjetura $\vec{u}_{i+1}^r$, ω_i^r es la evaluación en el instante anterior, $\vec{z}_{i+1}^n$ es una de las M esquinas registradas por la plataforma y $p\left(\,\vec{z}_{i+1}^n|\vec{u}_{i+1}^r\right)$ es la probabilidad condicional de medir $\vec{z}_{i+1}^n$ dado $\vec{u}_{i+1}^r$.

Parámetros

El filtro de partículas requiere un conjunto de parámetros para funcionar. Por una parte, los modelos de movimiento y medición están sujetos a perturbaciones gaussianas que se caracterizan por un valor promedio y una covarianza. Estas cantidades se determinan experimentalmente, para lo que se emplean videos en que una persona se ubica a 50cm de la cámara y mueve su dedo índice a 50cm/s siguiendo trayectorias conocidas (Ilustración 35). Por otra parte, el filtro de partículas tiene una configuración interna que establece su comportamiento y uso de procesador, la que se obtiene mediante ensayo y error.

Ilustración 35: Escena de video de calibración.

- **Modelo de movimiento**: Las perturbaciones que afectan el movimiento se obtienen mediante la diferencia entre el estado en el instante $i+1$ y la estimación hecha con los datos del período anterior i:

$$\delta\vec{u}_i = \vec{u}_{i+1} - A\,\vec{u}_i$$

Las posiciones el dedo son proporcionadas con el video, mientras que las velocidades están ausentes y deben ser estimadas. Esto se hace empleando las coordenadas de los cuadros anterior y posterior al instante i:

$$v_i^x = \frac{x_{i+1} - x_{i-1}}{2\Delta t}, \quad v_i^y = \frac{y_{i+1} - y_{i-1}}{2\Delta t}$$

La Ilustración 36 muestra los histogramas de las desviaciones de posición y velocidad. Visualmente, las barras siguen un patrón similar a una campana gaussiana, lo que indica que el error se puede aproximar con una distribución normal. La Tabla 1 presenta los índices de confiabilidad para el test de Jarque-Bera que verifica gaussianidad. La media y covarianza del modelo de perturbaciones son:

$$p(\delta\vec{u}_i) = \frac{1}{\sqrt{(2\pi)^4|\Sigma|}}\exp\left(-\frac{1}{2}(\delta\vec{u}_i - \vec{\mu})^T\Sigma^{-1}(\delta\,\vec{u}_i - \vec{\mu})\right)$$

$$\vec{\mu} = \begin{pmatrix} 0.05 \\ 0.06 \\ 0.11 \\ 0.10 \end{pmatrix}, \Sigma = \begin{pmatrix} 3.45 & -0.41 & 1.41 & 0.01 \\ -0.41 & 3.14 & -0.32 & 1.31 \\ 1.41 & -0.32 & 2.85 & -0.27 \\ 0.01 & 1.31 & -0.27 & 2.64 \end{pmatrix}$$

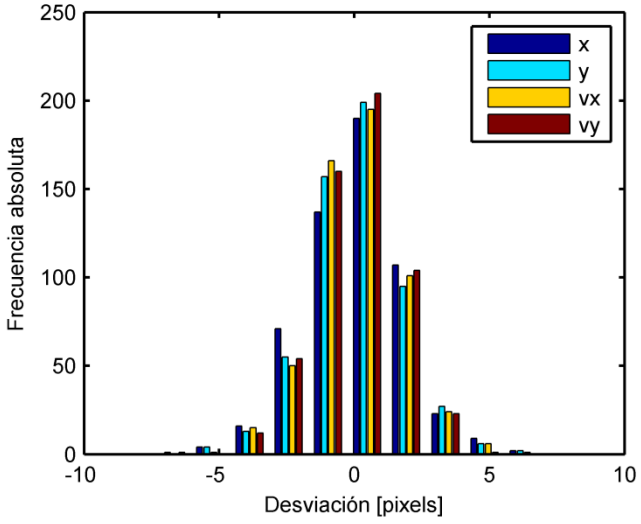

Ilustración 36: Histograma de error de predicción de estado.

Variable	Nivel de significancia
x	0.03047
y	0.00093
v^x	0.01326
v^y	0.00942

Tabla 1: Test de Jarque-Bera para perturbaciones en el estado.

- **Modelo de medición**: Los errores de medición son la diferencia entre las coordenadas medidas y las reales:

$$\delta\vec{z}_i = \vec{z}_i - C\,\vec{u}_i$$

Esta expresión requiere la ubicación de la esquina y la posición observada. Lo primero se obtiene desde el video y lo segundo desde los cálculos del programa. La desviación se evalúa para muchos instantes y los resultados se utilizan para obtener el promedio y la covarianza del modelo de ruido, lo que produce:

$$p(\delta\vec{z}_i) = \frac{1}{\sqrt{(2\pi)^4|\mathrm{T}|}}\exp\left(-\frac{1}{2}(\delta\vec{z}_i - \vec{v})^T\mathrm{T}^{-1}(\delta\,\vec{z}_i - \vec{v})\right)$$

$$\vec{v} = \begin{pmatrix}-2.91\\-4.19\end{pmatrix}, \mathrm{T} = \begin{pmatrix}38.04 & -7.99\\-7.99 & 52.64\end{pmatrix}$$

La Ilustración 37 muestra el histograma de las desviaciones. La distribución tiene un máximo global y decrece uniformemente, lo que calza cualitativamente con una campana de Gauss. Los valores del test de Jarque-Bera se presentan en la Tabla 2.

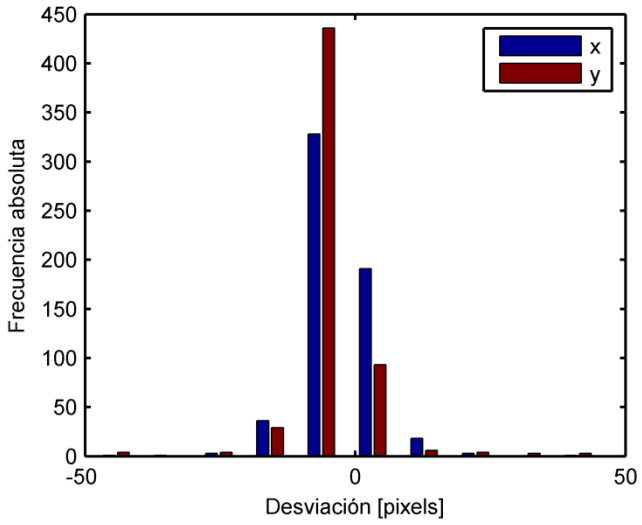

Ilustración 37: Histograma de error de medición.

Variable	Nivel de significancia
x'	0.00153
y'	0.00067

Tabla 2: Test de Jarque-Bera para errores de medición.

- **Filtro de partículas**: El filtro de partículas funciona con 1000 hipótesis, debido a que el desempeño del algoritmo mejora poco al utilizar un mayor número de conjeturas y el uso de procesador crece rápidamente. Las esquinas que están a más de 100 píxeles de la posición del

filtro son descartadas, debido a que es poco probable que estén relacionadas con el objeto de interés y pueden hacer que el programa pierda el rastro de la punta. En los casos en que no hay detecciones válidas, el programa itera 5 ciclos consecutivos y se detiene en la última ubicación conocida, a menos que registre nueva información.

Detección de clic

El gesto de clic se define como una secuencia de 3 eventos que ocurren sobre el punto de interés: detención, sacudida y detención. Su reconocimiento se efectúa en dos etapas: detección de movimiento y análisis temporal.

La primera fase determina si hay movimiento cerca de la esquina. Para ello, se extrae un cuadrado desde la máscara del objeto móvil con centro en la última posición del vértice. Luego, se cuentan los puntos en que hay cambios luminosos. Si esta cantidad supera un umbral, entonces hay actividad. De lo contrario, la región está detenida. La Ilustración 38 presenta un diagrama de flujo que resume el sistema.

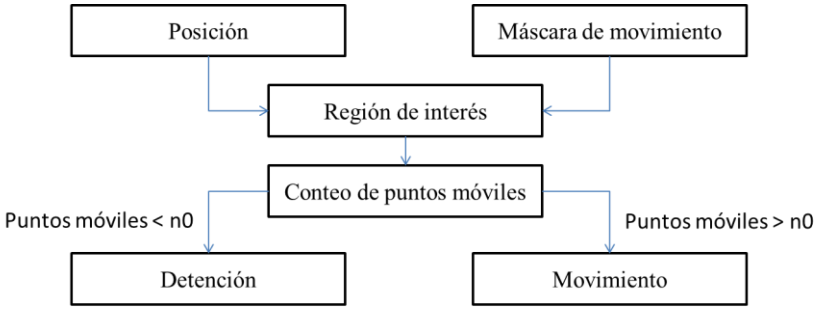

Ilustración 38: Diagrama de flujo para identificación de estado de movimiento.

La segunda parte examina la duración de las detenciones y movimientos para detectar pulsaciones. Primero, se cuentan los ciclos consecutivos en que el punto ha estado quieto. Si se detectan cambios y si el período de inactividad supera un umbral, entonces es posible que se haya iniciado un gesto de clic. Para confirmarlo, se registra la duración de la transición. Si esta es muy larga, significa que el usuario realiza un desplazamiento regular. De lo contrario, se produce otra detención y si no hay actividad por un tiempo, entonces se ha concluido un clic. El diagrama de flujo para estos acontecimientos se muestra en la Ilustración 39.

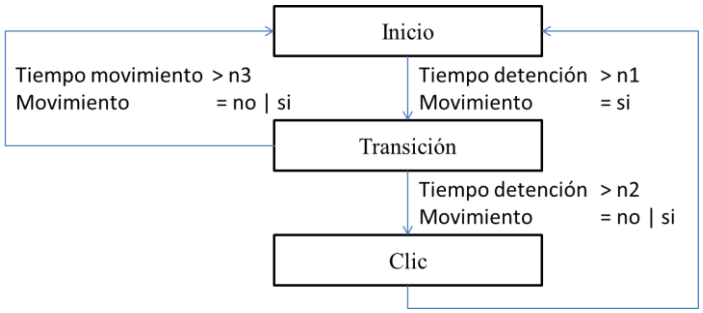

Ilustración 39: Diagrama de flujo para detección de clic.

En la implementación, la región de análisis tiene un tamaño de 50x50 píxeles y se requieren al menos n0=980 puntos para que haya movimiento. El usuario debe estar en reposo al menos n1=5

ciclos para iniciar el gesto. La sacudida debe durar menos de n3=15 iteraciones para que no sea considerada un desplazamiento. El clic concluye cuando no hay actividad durante n2=2 períodos.

Interacción entre clic y seguimiento

Cuando los sistemas de clic y seguimiento operan simultáneamente pueden interferir entre sí. Por ejemplo, si se pierde el rastro del punto de interés por algunos ciclos o las mediciones son intermitentes, el componente que detecta pulsaciones puede arrojar falsos positivos. Por el contrario, si el usuario mueve la mano para seleccionar un elemento, la operación puede fallar debido a que el puntero se desplaza por acción de la unidad de movimiento.

Estos problemas son resueltos alternando el funcionamiento de los módulos de clic y seguimiento. La activación de cada componente se regula según el comportamiento del usuario (Ilustración 40). Cuando el cursor llega al lugar deseado, la persona frena la mano, por lo que disminuyen los cambios luminosos y las mediciones desaparecen. Esto señala que solamente el módulo para detectar pulsaciones debe trabajar. Si la persona desea mover el cursor, desplaza su mano, lo que hace que el sistema vuelva a detectar esquinas consistentes con el modelo cinemático por varios ciclos consecutivos. Esto indica que el programa tiene que enfocarse en rastrear el cursor. De este modo, es posible hacer clics y movimientos con bajo riesgo de confusión.

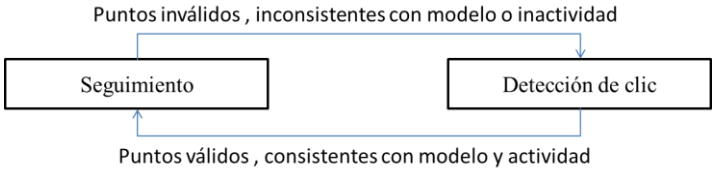

Ilustración 40: Alternación entre módulos de seguimiento y detección de clic.

En la implementación, las mediciones se consideran inválidas cuando están ubicadas a más de 100 píxeles del cursor. Una sucesión de puntos obedece el modelo si las predicciones discrepan en menos de 50 píxeles. La ausencia de actividad se produce cuando no hay cambios luminosos durante 9 ciclos en una vecindad de 100x100 centrada en la última posición.

4.2.3. Interfaz de usuario

Control de cursor

Las esquinas se muestran en la pantalla como círculos semitransparentes de 40 píxeles de diámetro para que el usuario sepa en qué región del monitor se localizan sus gestos. La ubicación del filtro de partículas se despliega con un disco opaco de 20 píxeles que cambia de color verde a azul cuando hay una pulsación (Ilustración 41 e Ilustración 42).

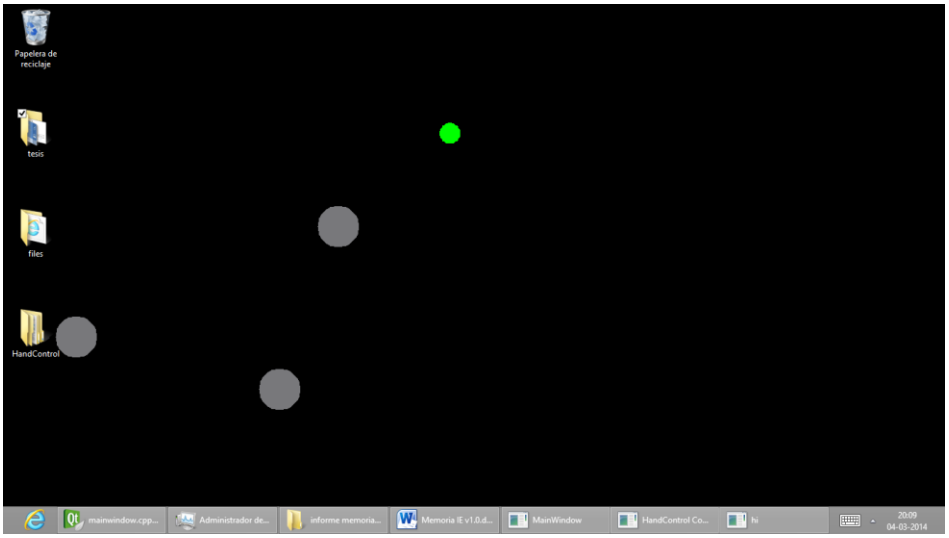

Ilustración 41: Puntos característicos y cursor en escritorio de Windows.

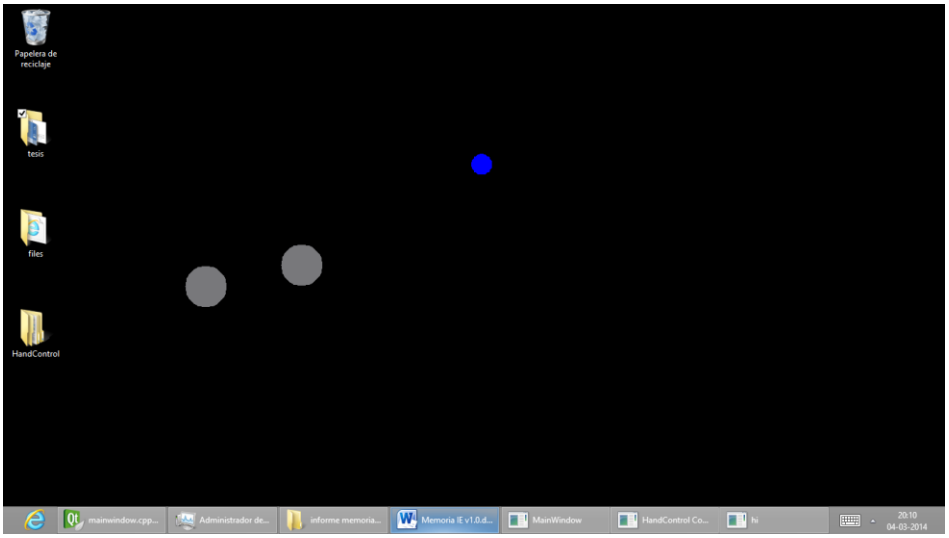

Ilustración 42: Cursor presionado.

Panel de control

La plataforma se puede configurar a través de un panel de control. En este se presentan 5 perfiles de funcionamiento: equilibrado, precisión, sensibilidad, económico y personalizado (Tabla 3 e Ilustración 43). El modo equilibrado entrega un buen desempeño con un uso de procesador moderado. La opción de precisión aumenta la resolución de las imágenes para incrementar la exactitud con que se identifican los objetos. El perfil de sensibilidad disminuye los umbrales de detección para funcionar mejor con movimientos lentos y baja iluminación. La alternativa económica reduce la resolución de los cuadros y la frecuencia de actualización para aminorar el número de operaciones. Por último, el modo personalizado permite modificar cualquier parámetro del programa (Ilustración 44).

Modo	Resolución de imagen	Cuadros por segundo	Difusión	Umbral de movimiento	Hipótesis filtro de partículas
Equilibrado	640x480	30	5x5	5	1000
Precisión	1024x768	30	3x3	5	1000
Sensibilidad	640x480	24	9x9	3	1000
Económico	320x240	15	3x3	5	100
Personalizado	Modificable	Modificable	Modificable	Modificable	Modificable

Tabla 3: Modos de funcionamiento.

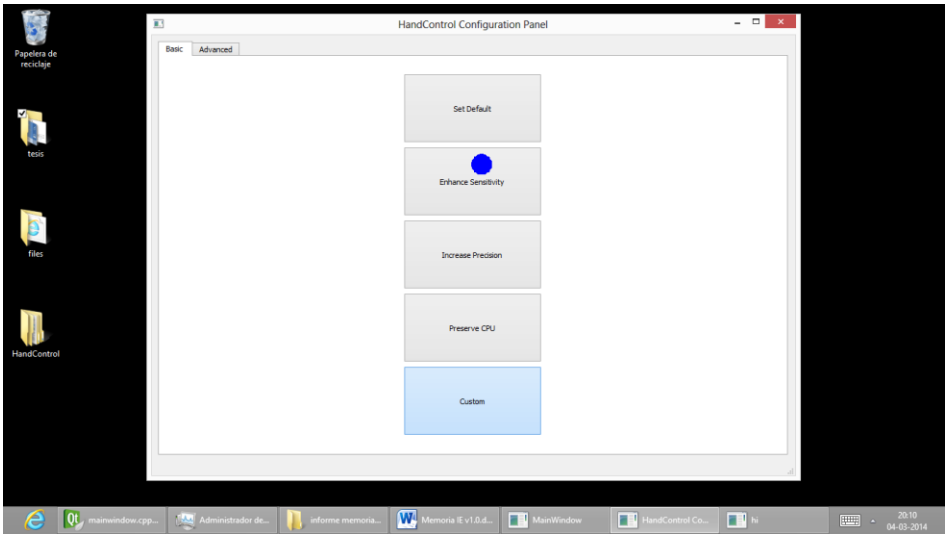

Ilustración 43: Opciones básicas de configuración.

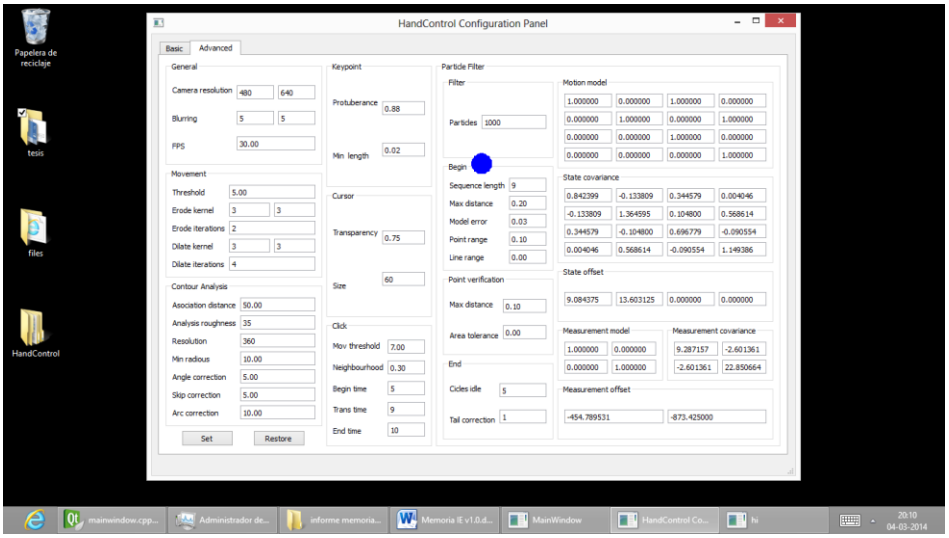

Ilustración 44: Opciones avanzadas de configuración.

Dibujo

La plataforma incorpora una interfaz de dibujo que permite seleccionar el grosor y color de los trazados. La aplicación incluye opciones para guardar trabajos, limpiar la pantalla y cerrar la ventana. Una vista del programa se muestra en la Ilustración 45.

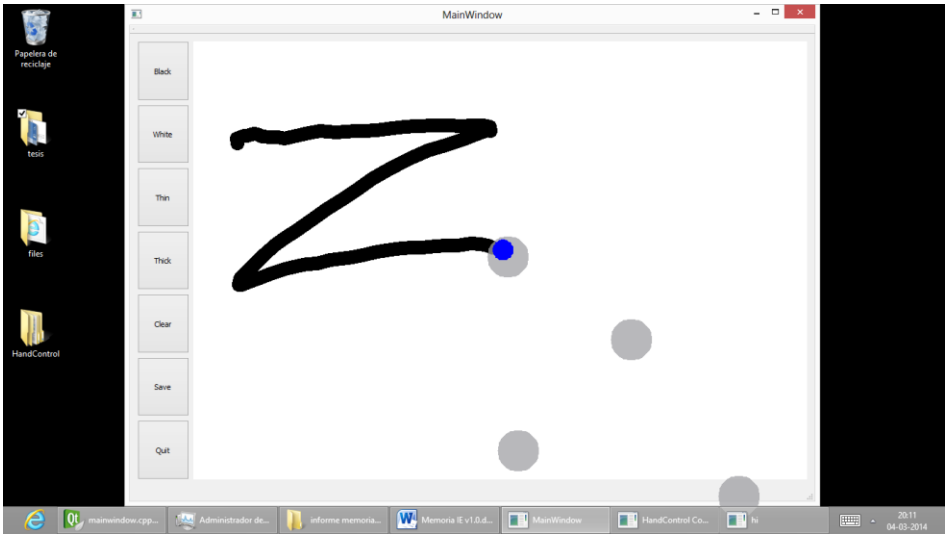

Ilustración 45: Interfaz de dibujo.

Capítulo 5: Desempeño

El desempeño de la plataforma se determina mediante un conjunto de pruebas. En primer lugar, se mide la capacidad de identificar y seguir esquinas en varias situaciones. Luego, se evalúan los errores al concluir un desplazamiento. Después, se verifica la habilidad para discriminar pulsaciones, movimientos y detenciones. Finalmente, se registra el uso de procesador, se hace una comparación con un sistema que utiliza detección de piel y se resumen los resultados. Estos exámenes son útiles para evaluar la efectividad de los algoritmos e identificar métodos para mejorar la plataforma.

6.1. Detección y seguimiento

La capacidad de identificar y rastrear un dedo en varias configuraciones ambientales es evaluada con 11 videos. Cada uno dura 10 segundos, muestra 30fps y tiene resolución de 640x480. Las grabaciones son realizadas en lugares con varios tipos de iluminación y fondos, donde una persona efectúa movimientos con diversas velocidades, ubicaciones y posturas. Uno de los registros tiene las características especificadas en la Tabla 4, mientras que el resto se diferencia en uno de los atributos de la Tabla 5. Los resultados entregan información sobre el desempeño del programa y permiten especificar las situaciones en que es utilizable.

Distancia	50cm
Iluminación	Normal
Fondo	Heterogéneo
Velocidad	~40cm/s
Postura	Normal

Tabla 4: Condiciones de utilización típicas.

Distancia	100cm	150cm
Iluminación	Baja	Muy baja
Fondo	Uniforme	Contraluz
Velocidad	~20cm/s	~100cm/s
Postura	Incómoda	Muy incómoda

Tabla 5: Configuraciones usadas en las pruebas.

El comportamiento de la plataforma se caracteriza mediante un conjunto de indicadores. La curva de detección permite conocer la proporción de localizaciones exitosas en función del umbral de separación. El error promedio indica la discrepancia típica entre las estimaciones y la ubicación real del dedo. La desviación estándar y covarianza precisan la variabilidad de las fluctuaciones. El porcentaje de puntos con separación menor que 20 píxeles permite saber si los resultados están usualmente cerca de la ubicación deseada. Estos índices son aplicados tanto para la detección de esquinas como para el seguimiento en cada uno de los videos de prueba.

6.1.1. Condiciones normales

La plataforma está calibrada para operar en un conjunto de condiciones predeterminadas. El recinto tiene iluminación natural, carece de objetos móviles y el fondo es heterogéneo. El usuario

mueve su mano a 50cm de la cámara, con el dedo índice levantado y manteniendo la cabeza y tronco fijos. Un ejemplo de esta configuración se muestra en la Ilustración 46. El funcionamiento del programa en este ambiente es el principal indicador de su efectividad, por lo que es importante verificarlo.

Ilustración 46: Escena en condiciones normales.

Detección de puntos característicos

El dedo índice es ubicado correctamente la mayor parte del tiempo. La Ilustración 47 muestra que los contornos y esquinas detectados son consistentes con el brazo y el extremo del dedo. La Ilustración 48 señala que la punta es localizada con desviación inferior a 20 píxeles en el 95.25% de los cuadros. Los datos de la Tabla 6 indican que las mediciones están desplazadas de la posición real, lo que es atribuible a las distorsiones introducidas por las operaciones de suavizado, erosión y dilatación. Por otra parte, la variabilidad es mayor en el eje perpendicular a la dirección del dedo, lo que se debe a que este es redondeado en el extremo. De estos antecedentes, se concluye que el programa localiza las esquinas con un leve corrimiento y con mayor precisión en la vertical.

Ilustración 47: Bordes y puntos característicos.

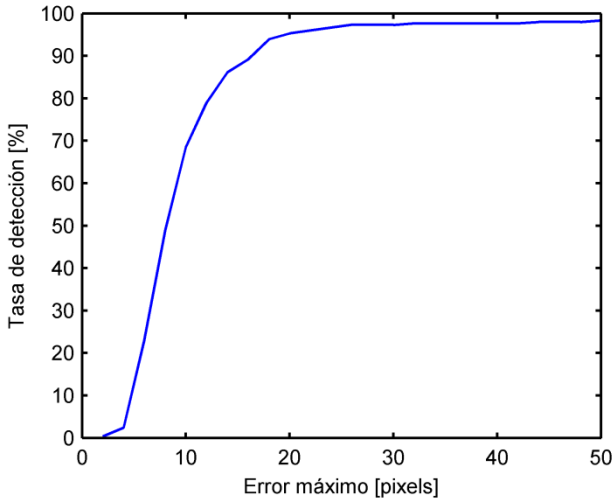

Ilustración 48: Tasa de detección de dedo índice en función del error máximo.

Detecciones correctas	95.25%
Error promedio en eje X [píxeles]	-3.47
Error promedio en eje Y [píxeles]	-5.77
Covarianza XX [píxeles^2]	28.58
Covarianza XY,YX [píxeles^2]	-8.96
Covarianza YY [píxeles^2]	11.70

Tabla 6: Estadísticas de desviaciones inferiores a 20 píxeles.

Seguimiento

El algoritmo rastrea correctamente la ubicación del dedo en casi todo el recorrido. La Ilustración 49 muestra que el trayecto de la punta y la estimación filtro de partículas tienen formas similares. La Ilustración 50 indica que el 99.66% del recorrido difiere en menos 20 píxeles. El error promedio es 4.75 píxeles, la desviación estándar es 3.59 píxeles y el alejamiento sobrepasa los 20 píxeles solo una vez (Tabla 7). Estos antecedentes confirman que el programa sigue la ubicación del dedo con fluctuaciones pequeñas en casi todos los cuadros.

Ilustración 49: Trayectorias (izq.) real, (cen.) detectada y (der.) superposición.

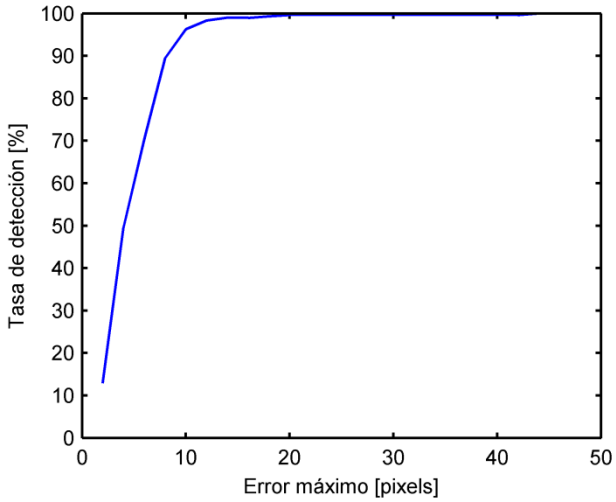

Ilustración 50: Tasa de detección de dedo índice en función del error máximo durante seguimiento.

Detecciones menores que 20 píxeles	99.66%
Desviación máxima [píxeles]	43.94
Desviación promedio [píxeles]	4.75
Desviación estándar [píxeles]	3.59

Tabla 7: Estadísticas de seguimiento.

Los resultados indican que la unidad de seguimiento mejora las estimaciones hechas por el componente de detección de esquinas. Por ejemplo, la localización con error menor que 20 píxeles ocurre en el 99.66% de los puntos durante la fase de rastreo, mientras que en la etapa previa solamente en el 95.25%. La explicación es que el filtro de partículas incorpora un modelo de movimiento que permite verificar la consistencia de los datos, corregir errores y extrapolar cuando la ubicación de esquinas falla. De este modo, el módulo de rastreo contribuye a aumentar la confiabilidad de las estimaciones.

El programa determina exitosamente la trayectoria del dedo en el entorno para el que fue diseñado. La detección de esquinas comete errores menores y los resultados son perfeccionados por el módulo de rastreo. La precisión de las estimaciones se puede aumentar utilizando una cámara que capture imágenes de mejor calidad, lo que permitiría prescindir de las operaciones que controlan el nivel de ruido y distorsionan los objetos.

6.1.2. Distancia a la cámara

La distancia del usuario a la cámara puede afectar el funcionamiento del programa. Esto se evalúa con 3 videos en que la separación es de 50cm, 100cm y 150cm, donde los movimientos y escenarios son idénticos a los utilizados en condiciones normales. Ejemplos de cada secuencia se presentan en la Ilustración 51. Los resultados permiten evaluar la influencia de la ubicación de la persona en el desempeño.

Ilustración 51: Escenas a (izq.) 50cm, (cen.) 100cm y (der.) 150cm.

Detección de puntos característicos

El programa mejora levemente su desempeño al aumentar la distancia. La Ilustración 52 muestra que los contornos y las esquinas son visualmente correctos en cada configuración. Las curvas de la Ilustración 53 indican que las tasas de detección crecen con la separación cuando el umbral es menor que 20 píxeles y sus valores terminales sobrepasan el 90%. Los resultados estadísticos de la Tabla 8 señalan que el error promedio y la covarianza descienden con la distancia. La explicación de estos resultados es que los movimientos lejanos parecen lentos, por lo que las imágenes son menos borrosas y las puntas son localizadas con mayor precisión.

Ilustración 52: Bordes y extremos a (izq.) 50cm, (cen.) 100cm y (der.) 150cm.

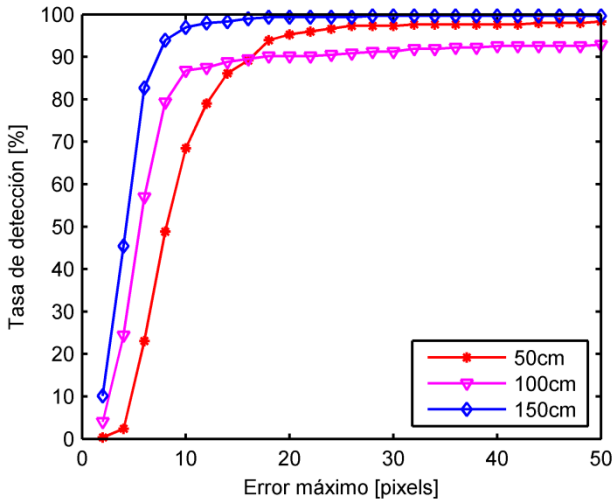

Ilustración 53: Tasa de detección en función del error máximo a múltiples distancias.

Distancia	50cm	100cm	150cm
Detecciones correctas	95.25%	90.17%	99.32%
Error promedio en eje X [píxeles]	-3.47	-0.36	0.21
Error promedio en eje Y [píxeles]	-5.77	-3.24	-1.90
Covarianza XX [píxeles^2]	28.58	14.08	8.67
Covarianza XY,YX [píxeles^2]	-8.96	0.16	2.03
Covarianza YY [píxeles^2]	11.70	10.38	11.14

Tabla 8: Estadísticas de desviaciones menores que 20 píxeles a múltiples distancias.

Seguimiento

La punta del dedo es rastreada correctamente en todas las distancias. La Ilustración 54 muestra que las trayectorias estimadas son similares a las verdaderas (ambas están superpuestas en la misma imagen). Las curvas de la Ilustración 55 indican que al menos el 96% de los recorridos tiene error inferior a 20 píxeles. La fluctuación promedio en cada configuración es inferior a 6 píxeles, la desviación estándar no supera 8 píxeles y, en el peor de los casos, el error asciende a 65.41 píxeles (Tabla 9). Por lo tanto, la habilidad de seguir un punto mantiene un mismo nivel independientemente de la ubicación del usuario.

Ilustración 54: Trayectorias real y estimada a (izq.) 50cm, (cen.) 100cm y (der.) 150cm.

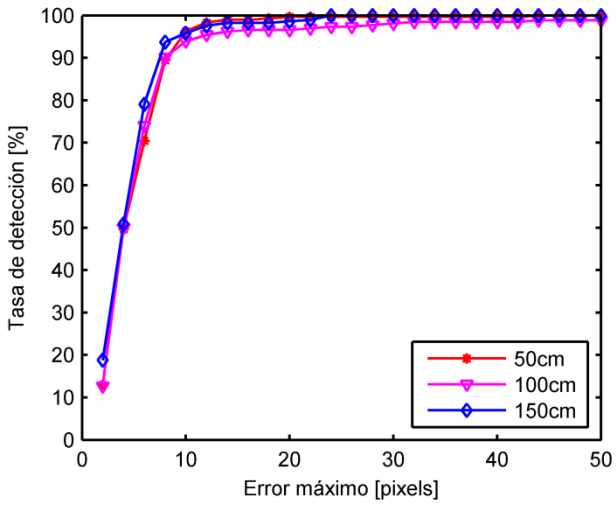

Ilustración 55: Tasa de detección en función del error máximo a múltiples distancias durante seguimiento.

Distancia	50cm	100cm	150cm
Detecciones menores que 20 píxeles	99.66%	96.56%	98.61%
Desviación máxima [píxeles]	43.94	65.41	23.53
Desviación promedio [píxeles]	4.75	5.60	4.41
Desviación estándar [píxeles]	3.59	7.62	3.25

Tabla 9: Estadísticas de seguimiento a múltiples distancias.

El desempeño del sistema de seguimiento es uniforme para las distancias verificadas, a pesar de que las mediciones encontradas en la etapa anterior no tienen la misma regularidad. Esto se debe a que el filtro de partículas corrige los datos y empareja los niveles de ruido. Sin embargo, la detección de movimiento a mayor distancia puede ser afectada por la resolución de las imágenes, lo que dificulta la discriminación de desplazamientos pequeños.

En síntesis, la plataforma funciona bien en un amplio rango de distancias y la precisión aumenta a mayor separación. Esto último es una señal de que a mayor velocidad aparente, los objetos se ven borrosos. La respuesta de la plataforma puede mejorar con una cámara que tenga mayor resolución y que registre los cuerpos móviles con menor distorsión.

6.1.3. Intensidad luminosa

La iluminación puede variar según la hora del día y el lugar, entre otros factores. Los efectos del nivel se luz se determinan con 3 grabaciones con intensidad luminosa normal, baja y muy baja. En cada situación, se realizan los mismos tipos de desplazamientos, a 50cm de la cámara, en lugares idénticos y con fondo estático. En la Ilustración 56 se presentan tomas de cada una de las secuencias. Los resultados permiten evaluar la sensibilidad de la plataforma a la iluminación.

Ilustración 56: Movimiento con iluminación (izq.) muy tenue, (cen.) baja y (der.) normal.

Detección de puntos característicos

La localización de la punta del dedo es correcta con luz normal o tenue, pero falla con frecuencia cuando hay mayor oscuridad, debido a que los contornos no son detectados (Ilustración 57). En iluminación muy baja, el 68% de puntas son ubicadas con error inferior a 20 píxeles (Ilustración 58), mientras que en los otros escenarios la detección supera el 95%. Dentro de las esquinas desviadas en menos de 20 píxeles, se observa que el corrimiento disminuye y que la covarianza vertical aumenta cuando se reduce la luz, lo que es atribuible a alteraciones en la delimitación de los bordes (Tabla 10). Por lo tanto, los resultados tienen alteraciones pequeñas al reducir la iluminación, pero hay un punto a partir del cual la plataforma comienza a fallar.

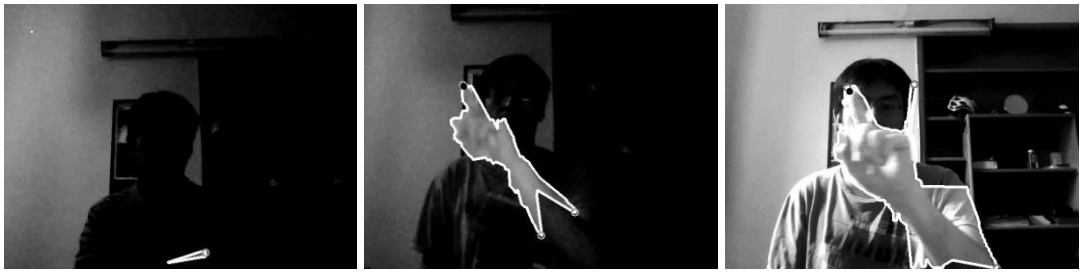

Ilustración 57: Contornos y puntos característicos con iluminación (izq.) muy tenue, (cen.) baja y (der.) normal.

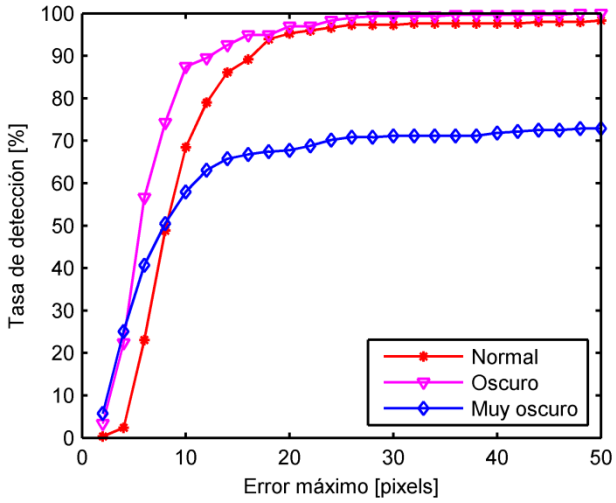

Ilustración 58: Tasa de detección en función del error máximo en varias iluminaciones.

Iluminación	Normal	Oscura	Muy oscura
Detecciones correctas	95.25%	96.95%	67.80%
Error promedio en eje X [píxeles]	-3.47	-0.94	1.77
Error promedio en eje Y [píxeles]	-5.77	-3.04	-1.62
Covarianza XX [píxeles^2]	28.58	23.41	19.25
Covarianza XY,YX [píxeles^2]	-8.96	-2.65	-3.75
Covarianza YY [píxeles^2]	11.70	15.93	21.47

Tabla 10: Estadísticas de desviaciones menores que 20 píxeles en varias iluminaciones.

Seguimiento

Las trayectorias son detectadas correctamente en luz normal o tenue, pero hay errores cuando la iluminación es la más baja (Ilustración 59). En los ambientes normal y oscuro, las trayectorias tienen desviación inferior a 20 píxeles en al menos el 98% de los puntos (Ilustración 60) y las fluctuaciones son pequeñas (Tabla 11). En el escenario muy oscuro, alrededor del 40% de las esquinas no son localizadas y los errores son un orden de magnitud superior. Esto indica que el programa tolera varios niveles de luz, pero hay un extremo en que su desempeño comienza a declinar.

Ilustración 59: Datos estimados y reales en luz (izq.) muy baja, (cen.) baja y (der.) normal.

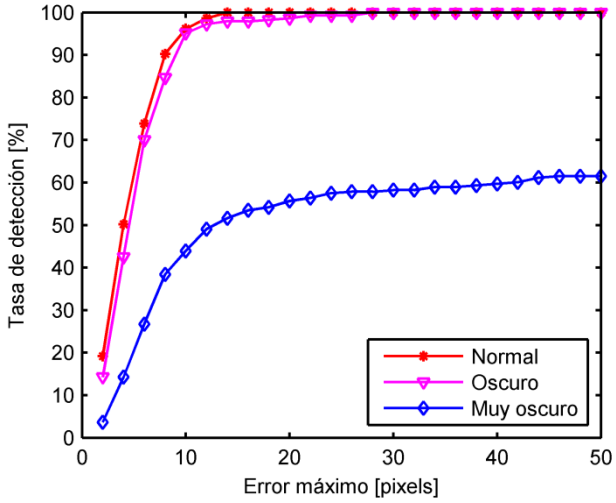

Ilustración 60: Tasa de detección en función del error máximo en varias iluminaciones durante seguimiento.

Iluminación	Normal	Oscura	Muy oscura
Detecciones menores que 20 píxeles	99.66%	98.62%	55.68%
Desviación máxima [píxeles]	43.94	27.80	141.44
Desviación promedio [píxeles]	4.75	5.13	42.50
Desviación estándar [píxeles]	3.59	3.61	44.81

Tabla 11: Estadísticas de seguimiento en varias iluminaciones.

Los resultados muestran que el algoritmo tiene un buen desempeño para luz normal y tenue. Sin embargo, los objetos no se distinguen cuando hay iluminación insuficiente. En este caso, el filtro de partículas empeora las estimaciones hechas en la etapa de detección, debido a que el rastro se pierde rápidamente y el algoritmo es incapaz de recuperar la punta del dedo. El desempeño se puede mejorar utilizando una cámara más sensible o iluminación artificial.

6.1.4. Fondo

La capacidad de funcionar en varios entornos es un requisito básico para cualquier aplicación de procesamiento de imágenes de uso general. Esto se evalúa en tres escenarios comunes (Ilustración 61): heterogéneo, uniforme y contraluz. Los resultados ayudan a especificar en qué tipos de lugares la plataforma opera correctamente e identificar fallas que pueden ocurrir durante uso cotidiano.

Ilustración 61: Movimiento en fondos (izq.) normal, (cen.) uniforme y (der.) contraluz.

Detección de puntos característicos

El programa detecta correctamente las esquinas en los fondos heterogéneo y uniforme, pero falla cuando hay una fuente de radiación (Ilustración 62). Esto se debe a que la mano refleja luz hacia el cuerpo del usuario, por lo que su movimiento genera cambios luminosos en los alrededores que el programa asocia erróneamente al objeto de interés. La tasa de detección para un umbral de 20 píxeles es de 71% cuando hay emisores de luz, mientras que en los otros casos excede el 95% (Ilustración 63). La covarianza de los errores se incrementa levemente cuando hay un objeto radiante frente a la cámara (Tabla 12), lo que indica que los contornos están más deformados. De esto se desprende que el programa funciona bien independiente de los cuerpos en el fondo, siempre que no emitan luz.

Ilustración 62: Bordes y puntos característicos en fondos (izq.) normal, (cen.) uniforme y (der.) contraluz.

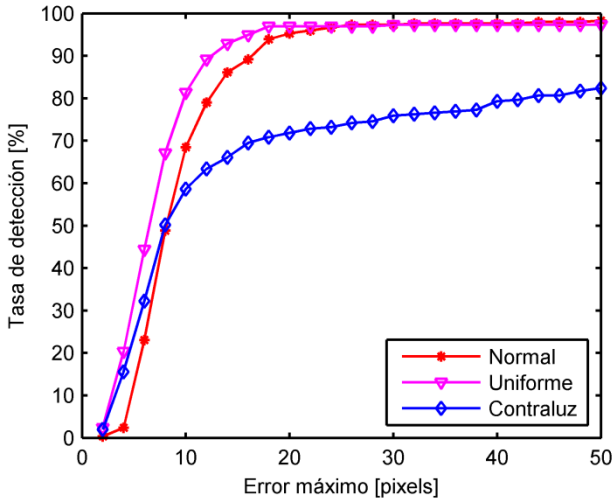

Ilustración 63: Tasa de detección en función del error máximo en varios fondos.

Fondo	Normal	Uniforme	Alto contraste
Detecciones correctas	95.25%	96.95%	71.86%
Error promedio en eje X [píxeles]	-3.47	4.01	-1.07
Error promedio en eje Y [píxeles]	-5.77	-2.86	-2.16
Covarianza XX [píxeles^2]	28.58	21.72	37.85
Covarianza XY,YX [píxeles^2]	-8.96	2.76	0.55
Covarianza YY [píxeles^2]	11.70	10.12	18.52

Tabla 12: Estadísticas de desviaciones menores que 20 píxeles en varios fondos.

Seguimiento

El programa determina correctamente las trayectorias en los casos con fondo heterogéneo y uniforme, pero falla cuando hay una fuente luminosa en el fondo (Ilustración 64). En la peor configuración, la tasa de seguimiento bordea el 50% (Ilustración 65) y los errores son un orden de magnitud superiores respecto a las otras situaciones (Tabla 13). Por lo tanto, la plataforma mantiene un bien desempeño independientemente de los objetos en el fondo, siempre que estos no sean radiantes.

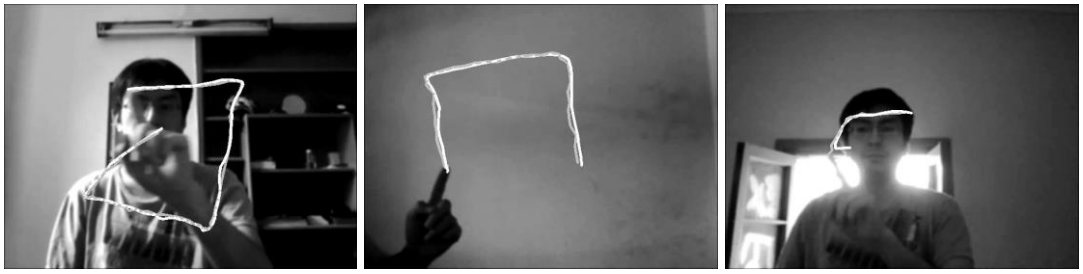

Ilustración 64: Datos reales y estimados en fondos (izq.) normal, (cen.) uniforme y (der.) contraluz.

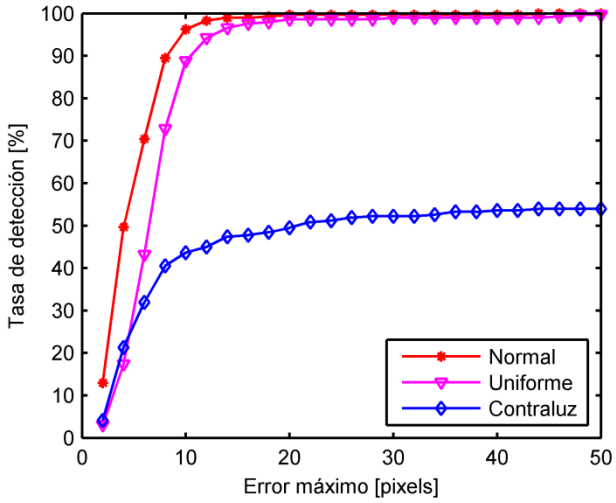

Ilustración 65: Tasa de detección en función de error máximo en varios fondos durante seguimiento.

Fondo	Normal	Uniforme	Alto contraste
Detecciones menores que 20 píxeles	99.66%	98.64%	49.48%
Desviación máxima [píxeles]	43.94	28.75	166.54
Desviación promedio [píxeles]	4.75	6.72	57.28
Desviación estándar [píxeles]	3.59	3.18	57.28

Tabla 13: Estadísticas de seguimiento en varios fondos.

En resumen, la plataforma es independiente de los objetos en el fondo, pero es intolerante a fuentes de radiación y a cambios luminosos que no corresponden al movimiento de la mano. Esto es una limitación inherente del programa y su resolución requiere identificar o desarrollar técnicas para discriminar el origen de las variaciones de luz. Una posibilidad consiste en utilizar un sensor de profundidad para reconocer y descartar las regiones ubicadas detrás del dedo, independientemente de los cambios luminosos en el entorno.

6.1.5. Velocidad

El programa debe seguir la trayectoria de un dedo que se puede mover a varias velocidades. La capacidad de hacerlo se determina con 3 videos en que el usuario realiza desplazamientos lentos (~20cm/s), normales (~40cm/s) y rápidos (~100cm/s). Ejemplos de las secuencias se muestran en la Ilustración 66. Los resultados permiten tener una noción del rango de velocidades en que puede operar el algoritmo y determinar vías para expandirlo.

Ilustración 66: Movimiento a velocidad (izq.) lenta, (cen.) normal y (der.) rápida.

Detección de puntos característicos

Los contornos y esquinas son localizados exitosamente en todos los regímenes (Ilustración 67). Sin embargo, a medida que la velocidad aumenta, la tasa de detección se reduce (Ilustración 68) y los errores se incrementan (Tabla 14). Por lo tanto los contornos y esquinas se ubican correctamente en todas las situaciones, pero las desviaciones son mayores durante desplazamientos rápidos.

Ilustración 67: Bordes y puntos característicos a velocidad (izq.) lenta, (cen.) normal y (der.) rápida.

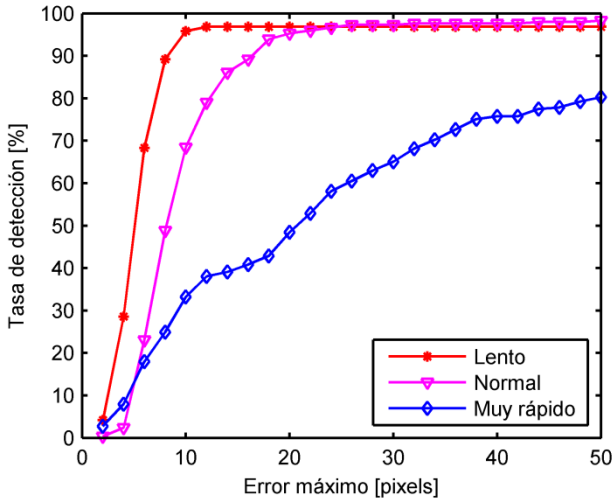

Ilustración 68: Tasa de detección en función del error máximo para múltiples velocidades.

Velocidad	Lento	Normal	Muy rápido
Detecciones correctas	96.86%	95.25%	48.44%
Error promedio en eje X [píxeles]	0.88	-3.47	-0.28
Error promedio en eje Y [píxeles]	-2.81	-5.77	-3.41
Covarianza XX [píxeles^2]	15.08	28.58	61.87
Covarianza XY,YX [píxeles^2]	-0.38	-8.96	-8.99
Covarianza YY [píxeles^2]	4.38	11.70	29.80

Tabla 14: Estadísticas de desviaciones menores que 20 píxeles para múltiples velocidades.

Seguimiento

Las trayectorias son determinadas correctamente a velocidades baja y normal, pero el rastro se pierde inmediatamente cuando el movimiento es rápido (Ilustración 69). La tasa de detección para un umbral de 20 píxeles es de 10.9% para desplazamientos bruscos, mientras que en los otros casos excede el 99% (Ilustración 70). La variabilidad aumenta con la velocidad, sobre todo en el régimen más extremo (Tabla 15). Por lo tanto, el seguimiento es preciso en movimientos lentos o normales, pero es inefectivo para transiciones aceleradas.

Ilustración 69: Trayectorias real y estimada a velocidades (izq.) lenta, (cen.) normal y (der.) rápida.

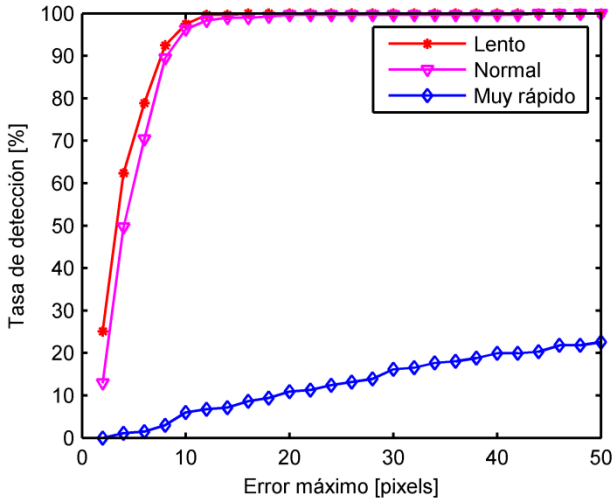

Ilustración 70: Tasa de detección en función del error máximo para múltiples velocidades durante seguimiento.

Velocidad	Lenta	Normal	Muy rápido
Detecciones menores que 20 píxeles	100.00%	99.66%	10.90%
Desviación máxima [píxeles]	14.22	43.94	397.25
Desviación promedio [píxeles]	3.88	4.75	132.37
Desviación estándar [píxeles]	2.57	3.59	90.46

Tabla 15: Estadísticas de seguimiento para múltiples velocidades.

Los resultados muestran que los desplazamientos bruscos no pueden ser rastreados por el programa. Esto se debe a que la cámara registra los objetos móviles con distorsiones que se acentúan con la velocidad, lo que aumenta las desviaciones en la localización de esquinas. Por otro lado, el filtro de partículas está calibrado para movimientos suaves, por lo que no está preparado para seguir cambios abruptos. Esto hace que la plataforma pierda el rastro de los objetos rápidos inmediatamente.

En conclusión, el programa puede seguir objetos a diversas velocidades, pero tiene un límite en que empieza a fallar. Las causas se deben a distorsiones de la cámara y a la configuración del filtro de partículas. Los resultados se pueden mejorar empleando una cámara de alta velocidad y calibrando el algoritmo de seguimiento para rastrear movimientos rápidos.

6.1.6. Postura del usuario

Los movimientos de cabeza y tronco pueden afectar el funcionamiento del programa. Los efectos se analizan con tres videos en que el usuario agita la mano en posiciones vertical, horizontal e invertida. Las posturas requieren movimientos corporales y pueden causar fatiga. En la Ilustración 71 se presentan cuadros de cada situación.

Ilustración 71: Movimiento en poses (izq.) cómoda, (cen.) incómoda y (der.) muy incómoda.

Detección de puntos característicos

La forma y esquina del dedo solo son localizadas correctamente cuando el usuario utiliza una postura cómoda. En los otros casos, hay confusiones con la cabeza y el tronco (Ilustración 72). La tasa de detección disminuye a medida que la incomodidad aumenta (Ilustración 73). Los errores de los puntos exitosamente identificados tienen características similares, pero la información es más precisa en la dirección paralela al dedo (Tabla 16). Por lo tanto, la detección de esquinas funciona solo en la postura más agradable y la exactitud depende de la orientación de la mano.

Ilustración 72: Contornos y puntos característicos en poses (izq.) cómoda, (cen.) incómoda y (der.) muy incómoda.

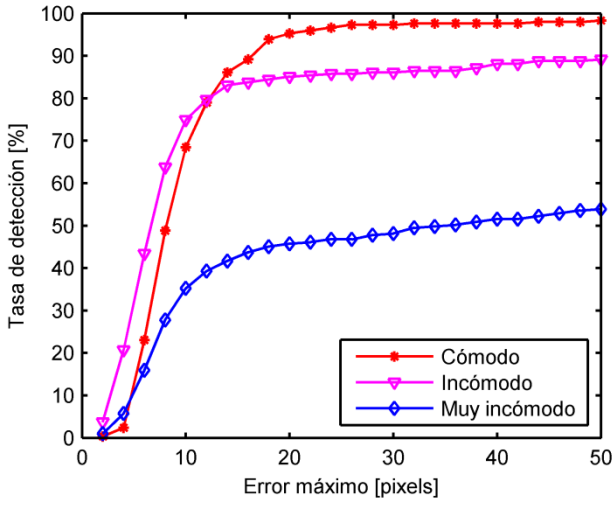

Ilustración 73: Tasa de detección en función del error máximo para varias posturas.

Postura	Normal	Incómoda	Muy incómoda
Detecciones correctas	95.25%	85.08%	45.76%
Error promedio en eje X [píxeles]	-3.47	-2.51	-5.70
Error promedio en eje Y [píxeles]	-5.77	-1.44	-1.53
Covarianza XX [píxeles^2]	28.58	9.64	22.94
Covarianza XY,YX [píxeles^2]	-8.96	1.37	1.84
Covarianza YY [píxeles^2]	11.70	31.18	15.49

Tabla 16: Estadísticas de desviaciones menores que 20 píxeles para varias posturas.

Seguimiento

Las trayectorias se estiman correctamente cuando se utiliza la postura normal y falla en los otros casos (Ilustración 74). La tasa de detección se incrementa con el nivel de comodidad (Ilustración 75). La magnitud de las desviaciones aumenta al menos 10 veces cuando la persona no utiliza una postura natural (Tabla 17). Los resultados indican que el algoritmo es incapaz de rastrear movimientos cuando la postura es incómoda.

Ilustración 74: Trayectorias real y estimada en poses (izq.) cómoda, (cen.) incómoda y (der.) muy incómoda.

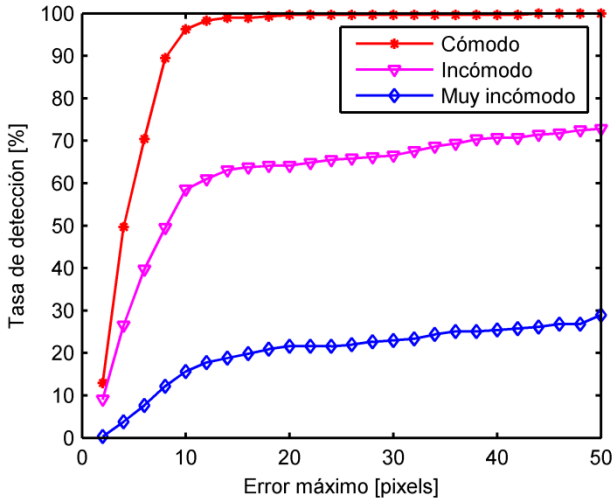

Ilustración 75: Tasa de detección en función del error máximo para varias posturas durante seguimiento.

Postura	Normal	Incómoda	Muy incómoda
Detecciones menores que 20 píxeles	99.66%	64.11%	21.60%
Desviación máxima [píxeles]	43.94	352.06	436.72
Desviación promedio [píxeles]	4.75	64.95	119.16
Desviación estándar [píxeles]	3.59	105.64	92.06

Tabla 17: Estadísticas de seguimiento para varias posturas.

Los resultados muestran dos hechos. Por una parte, el programa falla en poses incómodas. Esto se explica por los cambios luminosos generados por los movimientos de la cabeza y tronco, los cuales se confunden con el objeto de interés. Por otro lado, la magnitud de la covarianzas depende de la orientación de la mano. Esto se debe a que la punta del dedo es curva, lo que dificulta encontrar la ubicación del extremo en la dirección tangente. Así, cuando la mano está alineada con la vertical, la coordenada en el eje Y se determina con exactitud. De lo contrario, cuando la mano es paralela a la horizontal, la posición en el eje X es más confiable.

En resumen, la plataforma solamente funciona cuando los movimientos corporales son pequeños y la precisión de las coordenadas depende de la orientación del dedo. La confiabilidad se puede mejorar con dos medidas. Una consiste en incorporar técnicas para discriminar la mano de los objetos del fondo. Por ejemplo, se puede extraer la mano por su color o cercanía a la cámara, de modo que las variaciones luminosas detrás no afectan al programa. La otra es mejorar la ubicación de la punta, lo que se puede hacer proyectando las coordenadas sobre el eje central del dedo.

6.2. Detenciones

La plataforma reconoce la ubicación de la mano utilizando cambios luminosos que se generan por desplazamientos. Cuando el usuario se detiene o mueve muy lento, la punta del dedo se hace invisible para el programa, lo que impide medir su posición. En estos casos, el algoritmo supone que el cursor permanece en la última ubicación conocida. Sin embargo, esta conjetura podría ser incorrecta, puesto que los cambios luminosos disminuyen gradualmente durante el frenado, lo que puede deformar los contornos antes de que la mano esté completamente quieta. Debido a esto, es necesario evaluar la precisión con que el dedo se identifica durante una detención.

Las desviaciones se registran con 50 videos en que la persona desplaza la mano hasta un punto conocido. Las grabaciones están hechas en condiciones de uso normal, duran 3 segundos y muestran 30fps. El programa procesa cada registro y entrega la posición final del puntero, lo que posteriormente es comparado con la ubicación real del dedo índice.

Los errores al final de las trayectorias son mayores que en el recorrido. La Ilustración 76 muestra que la tasa de detección es menor en la posición terminal que en los puntos transitorios para cualquier umbral. De acuerdo a la Tabla 18, el corrimiento y las fluctuaciones tienen el doble de magnitud en la ubicación de reposo que en las de movimiento.

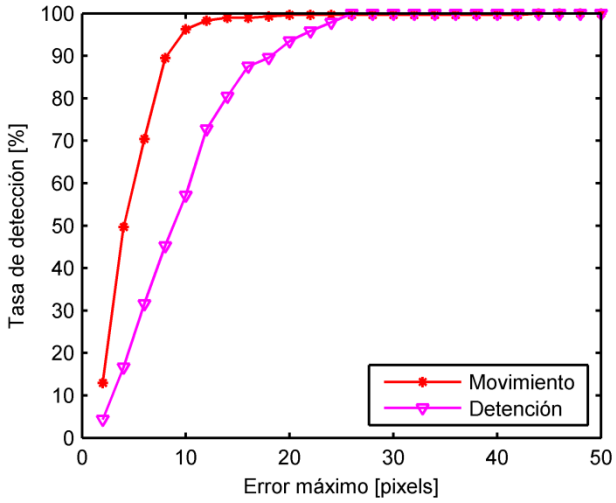

Ilustración 76: Tasa de detección en función del error máximo durante desplazamiento normal y detención.

Estado	Movimiento	Detención
Detecciones menores que 20 píxeles	99.66%	93.75%
Desviación máxima [píxeles]	43.94	37.56
Desviación promedio [píxeles]	4.75	9.62
Desviación estándar [píxeles]	3.59	6.51

Tabla 18: Estadísticas de seguimiento normal y detención.

En conclusión, los movimientos del usuario son registrados con precisión, pero los puntos de detención tienen el doble de incertidumbre. Esto tiene repercusiones en algunas aplicaciones. Por ejemplo, si la plataforma es utilizada para controlar un mouse, la selección de un objeto podría requerir varios intentos. En cambio, en un software de dibujo, los recorridos cobran mayor importancia, por lo que en este caso el programa es más confiable. Un camino para mejorar el desempeño en detenciones consiste en utilizar técnicas basadas en color para registrar la punta en ausencia de movimiento.

6.3. Clic, desplazamiento y reposo

El módulo de seguimiento se detiene cuando el usuario mantiene su mano en reposo. En estos casos la plataforma verifica continuamente si se generan clics, inicia un desplazamiento o mantiene la detención. Las fallas en la discriminación de estos eventos podrían causar pulsaciones y movimientos indeseados durante el uso del programa, lo que es perjudicial para la experiencia del usuario. Por lo tanto, es necesario examinar la habilidad del programa para realizar estas tareas correctamente.

El desempeño se verifica con videos en los que una persona está en reposo, hace clic o inicia un desplazamiento. En cada gesto, se utilizan 50 secuencias grabadas en condiciones de uso normal. Cada una dura 3 segundos, muestra 30fps y tiene resolución de 640x480. El algoritmo estima la acción realizada en cada video y el resultado se compara con la operación realmente efectuada.

Los resultados se resumen en la matriz de confusión de la Tabla 19. En esta, los encabezados de las filas se refieren a los gestos realizados por la persona y los de las columnas indican la acción reconocida por el programa. Las celdas indican la frecuencia relativa con que un gesto se identifica en cada tipo de grabación. Por ejemplo, la plataforma determina incorrectamente que hay reposo en el 8% de los videos en que el usuario acelera. De acuerdo a la tabla, el sistema identifica los gestos de clic en el 84% de los casos, desplazamiento en el 82% y reposo en el 92%.

Las fallas se deben a limitaciones del módulo de detección de clics para seguir movimientos demasiado rápidos o lentos. El algoritmo analiza secuencias temporales por un período pequeño y considera solamente la información contenida dentro de una vecindad alrededor de la esquina, lo que produce errores cada vez que los movimientos exceden los límites de duración o localización. De este modo, cuando hay desplazamientos veloces, una aceleración puede ser interpretada erróneamente como pulsación y un clic se puede confundir con un movimiento. En el otro extremo, cuando hay cambios suaves, una pulsación se puede interpretar como un movimiento o los gestos pueden pasar desapercibidos.

	Clic	Aceleración	Reposo
Clic	84%	10%	6%
Aceleración	10%	82%	8%
Reposo	8%	0%	92%

Tabla 19: Matriz de confusión para clic, desplazamiento y reposo.

En síntesis, la habilidad de distinguir correctamente lo que una persona hace cuando empieza a moverse es moderada. Ocasionalmente, cuando el usuario efectúa una pulsación, el cursor puede cambiar de posición o permanecer inactivo. Una aceleración puede gatillar clics o no ser detectada. En reposo se puede generar una pulsación errónea. Estos efectos, en el peor de los casos, pueden ocurrir con probabilidad de 18% cada 3s, por lo que puede ser necesario repetir un gesto de vez en cuando. Los resultados se pueden mejorar con un rediseño del sistema de detección de clics que considere movimientos a velocidad extremadamente rápida o lenta.

6.4. Uso de procesador

La plataforma realiza un conjunto de operaciones sobre los cuadros que recibe de una webcam para determinar el comportamiento de la mano. Esto se efectúa en un procesador (CPU), el que usualmente es responsable de la ejecución de múltiples programas en paralelo. Cuando las tareas demandan mayor capacidad que la disponible, el ordenador puede funcionar intermitentemente, lo que afecta seriamente el confort del usuario. Debido a esto, es importante hacer un diagnóstico de los requerimientos de la plataforma e identificar posibilidades de optimización.

El uso de CPU se determina con el Monitor de Rendimiento del sistema operativo Windows 8, en un notebook con las características especificadas en la Tabla 20. Las mediciones son realizadas en los estados de reposo y seguimiento en condiciones normales. Las pruebas duran 1.6 minutos y en este periodo no hay aplicaciones funcionando en paralelo. Los resultados entregan información sobre el tiempo de uso de un núcleo virtual del procesador.

Modelo de PC	SVF15215CLB
Modelo de procesador	Intel Core i5-3337U
Núcleos físicos/virtuales	2/4
Velocidad de reloj normal/máxima	1.8GHz/2.7GHz
Caché	3MB
Puntuación PassMark	3612

Tabla 20: Especificaciones del equipo de prueba.

Los resultados en reposo y seguimiento se muestran en la Ilustración 77 e Ilustración 78 respectivamente. Durante inmovilidad, el sistema emplea en promedio el 28% de un núcleo virtual, con oscilaciones entre 17.18% y 43.72%. Cuando hay movimiento, el uso medio es 65.46% y varía entre 53.03% y 81.24%. En todo momento, la plataforma se ejecuta a 30 cuadros por segundo, a pesar de que el consumo de recursos puede fluctuar según las acciones del usuario.

El aumento del uso de procesador durante el movimiento se debe a dos factores. Por una parte, cuando las imágenes registran cambios luminosos, se efectúan operaciones de extracción y análisis de contorno. Estas requieren una cantidad de cómputos proporcional al cuadrado del perímetro de los bordes, lo que puede variar ampliamente durante el uso del programa. Por otra parte, en este régimen se activa el sistema de seguimiento, que utiliza un filtro de partículas con 1000 hipótesis. Este algoritmo es poco eficiente y requiere un número considerable de operaciones para funcionar.

La plataforma se ejecuta holgadamente en un núcleo virtual del procesador. Si se considera que la CPU soporta hasta 4 procesos simultáneos, se puede concluir que el programa no obstruye el funcionamiento de aplicaciones comunes y que la experiencia del usuario se ve inalterada. Por otra parte, la plataforma puede funcionar en sistemas menos potentes o más antiguos, pero las mediciones efectuadas son insuficientes para establecer los requerimientos mínimos. Se puede conseguir mayor eficiencia optimizando los algoritmos para procesar contornos y el filtro de partículas.

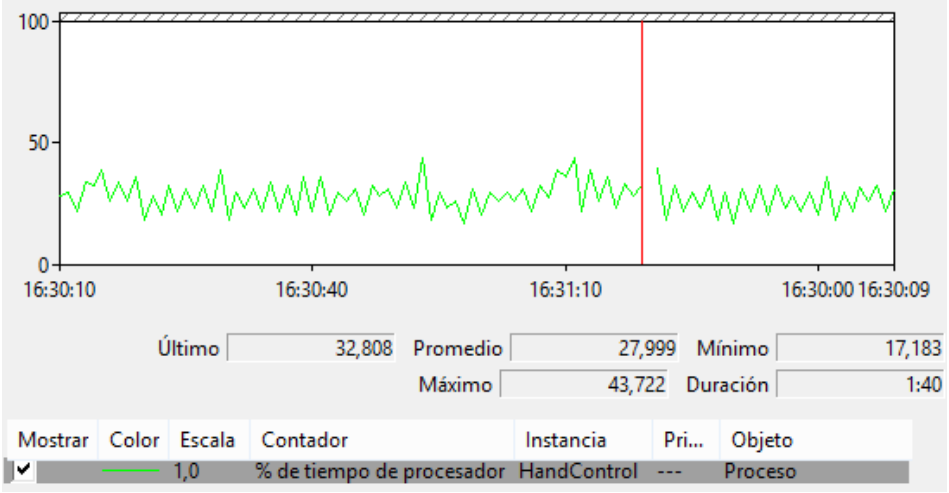

Ilustración 77: Uso de un núcleo lógico del procesador Intel Core i5-3337U con plataforma en modo equilibrado durante reposo.

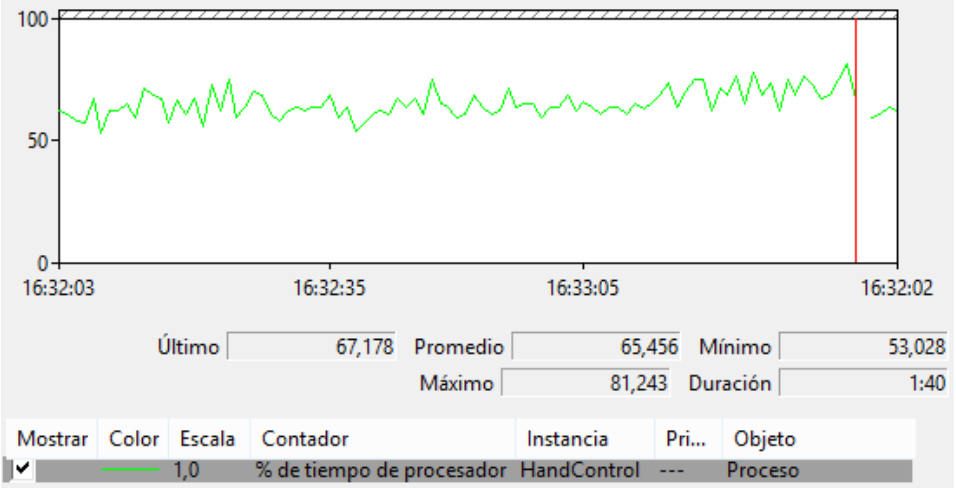

Ilustración 78: Uso de un núcleo lógico del procesador Intel Core i5-3337U con plataforma en modo equilibrado durante seguimiento.

6.5. Comparación

Existen varios trabajos orientados a reconocer gestos utilizando una webcam. La mayoría de ellos reconocen los objetos por su aspecto, pero este tipo de métodos puede fallar cuando el cuerpo de interés es similar a su entorno. La plataforma descrita en el presente documento utiliza una aproximación basada en cambios luminosos, pensada para abordar las dificultades reportadas en los otros trabajos. Ambas aproximaciones tienen diferencias, pero no son excluyentes, por lo que su integración podría mejorarlas mutuamente.

Para realizar una comparación, es necesario tomar como referencia al menos un ejemplo puntual. El trabajo [35] es un sistema de reconocimiento de gestos que utiliza detección de color y movimiento. Este rastrea el centro de masa de una mano e identifica señas dibujadas en el aire. Los resultados indican que el seguimiento es exitoso en el 91.3% de las pruebas de la base de datos ALON-EASY. La mayoría de los errores se deben a confusiones con regiones de color similar al de la mano, como los brazos y la cara.

El planteamiento de [35] difiere en varios aspectos a la aproximación de este documento. Un enfoque discrimina la mano utilizando su color, mientras que el otro lo hace mediante cambios luminosos. Un trabajo sigue la trayectoria de un puño, en contraste con el otro que rastrea un dedo. Las bases de datos para evaluación son diferentes (Ilustración 79) y la tasa de detección es mayor que 90% en cada caso. Las causas de falla son disímiles: el método basado en color confunde objetos parecidos, mientras que el que utiliza cambios luminosos es intolerante a variaciones en el fondo. Por lo tanto, ambos procedimientos funcionan en ámbitos diferentes y tienen niveles de eficacia similares en sus respectivos entornos.

Ilustración 79: Imágenes de base de datos (izq.) ALON-EASY y (der.) local.

En conclusión, los métodos difieren en un conjunto de características. Sin embargo, estas diferencias no son excluyentes, por lo que, en principio, es posible desarrollar una plataforma que combine ambos enfoques para que colaboren en un mismo objetivo. Este nuevo planteamiento, podría corregir las deficiencias de cada aproximación. Por ejemplo, cuando hay movimiento en el fondo, se puede utilizar la información entregada por el componente de detección de color, que es más confiable en este caso. A la inversa, cuando hay objetos similares, se puede emplear la detección de cambios luminosos, que es más certera en esta situación. El único caso en que ambos métodos pueden fallar ocurre cuando se mueven objetos similares, como cuando la cara se desplaza detrás de la mano, lo que requiere nuevas indagaciones para su resolución.

6.6. Resumen

La detección y seguimiento de una esquina se realizan con un error promedio de 5 píxeles en las situaciones marcadas en la Tabla 21. La identificación del punto de detención tiene un error promedio de 10 píxeles. Las pulsaciones son discriminadas correctamente en el 82% de los intentos. Durante un desplazamiento, el programa utiliza el 65% de un núcleo lógico del procesador Intel Core i5-3337u, lo que equivale a un 16% de la capacidad total. La plataforma distingue correctamente los objetos que tienen color similar, lo que es una ventaja respecto a los sistemas de detección de piel, pero puede fallar cuando hay cambios en el fondo o zonas radiantes.

	Valor 1	Valor 2	Valor 3
Distancia	50cm	100cm	150cm
Iluminación	Normal	Baja	Muy baja
Fondo	Uniforme	Heterogéneo	Contraluz
Velocidad	~20cm/s	~40cm/s	~100cm/s
Postura	Normal	Incómoda	Muy incómoda

Tabla 21: Desempeño en función de las condiciones ambientales. Las celdas oscuras indican buena evaluación.

La plataforma es capaz de localizar la punta del dedo y pulsaciones con bajo error y eficientemente en varios entornos. El desempeño es independiente de la presencia de cuerpos parecidos la escena, pero es afectado por cambios luminosos en el fondo. La confiabilidad se puede mejorar utilizando colores y rediseñando la unidad de detección de clics. También puede ser útil una cámara de mejor calidad o un sensor de profundidad. El uso de procesador se puede reducir con técnicas más eficientes para procesar contornos y optimizaciones en el filtro de partículas.

Capítulo 6: Conclusiones

La plataforma determina la trayectoria de una esquina y gestos de clic analizando variaciones luminosas en imágenes obtenidas con una webcam. El recorrido es calculado con un error promedio de 5 píxeles en varios tipos de iluminación, fondos, posiciones del usuario y velocidades. Los puntos de detención son localizados con una desviación media de 10 píxeles. Las pulsaciones son reconocidas exitosamente en el 82% de los intentos. El uso de procesador para un chip Intel Core i5-3337u es 16%. El desempeño se degrada cuando hay movimiento en el fondo u objetos radiantes.

La detección de cambios luminosos distingue objetos de apariencia similar, lo que es una causa de falla común en los métodos basados en color. Sin embargo, la plataforma es intolerante a cambios en el fondo, lo que no sucede en el otro caso. Por otra parte, ambas aproximaciones tienen desempeños similares en sus entornos de utilización. Por lo tanto, ambas las técnicas son complementarias y se pueden combinar para operar en una mayor variedad de situaciones.

La utilización de sensores infrarrojos puede aumentar el desempeño del sistema. Por ejemplo, un dispositivo para medir profundidad permite discriminar el movimiento en el fondo, lo que corrige la mayoría de las limitaciones de la plataforma. En general, la información combinada de múltiples sensores puede generar estimaciones más precisas que un dispositivo aislado, por lo que se recomienda emplearlos si es posible.

El estudio tiene varias limitaciones. Las pruebas se realizaron en solo un ordenador, lo que es insuficiente para asegurar si la plataforma se comportará igual en otros dispositivos o configuraciones. Por otra parte, se utilizaron imágenes de 640x480 en todas las evaluaciones, por lo que se desconoce el desempeño en resoluciones distintas. Tampoco se comprobó el funcionamiento del programa con la CPU operando a máxima capacidad, lo que puede afectar la cantidad imágenes procesadas por segundo y la calidad del seguimiento. Estos exámenes pueden servir para mejorar el sistema y facilitar su utilización por parte de terceros.

La plataforma es inadecuada para aplicaciones reales, debido a que falla cuando hay movimiento en el fondo, como sucede habitualmente cuando se desplaza la cabeza o el tronco. Sin embargo, esta deficiencia puede ser abordada de varias maneras y su corrección permitiría realizar sistemas de entretenimiento con novedosos mecanismos de interacción para ordenadores personales o, en una perspectiva más futurista, dotar robots con la capacidad de reconocer indicaciones con las manos.

Bibliografía

[1] C. Breazeal, C. Kidd, A. Thomaz, G. Hoffman y M. Berlin, «Effects of nonverbal communication on efficiency and robustness in human-robot teamwork,» de *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.

[2] N. Mavridis, «A Review of Verbal and Non-Verbal Human-Robot Interactive Communication,» Enero 2014. [En línea]. Available: http://arxiv.org/abs/1401.4994. [Último acceso: 27 Mayo 2014].

[3] M. Kinjal, N. Shah, M. Kirit, R. Rathod, M. Shardul y J. Agravat, «A survey on Human Computer Interaction Mechanism Using Finger Tracking,» *International Journal of Computer Trends and Technology*, vol. 7, nº 3, pp. 174-177, 2014.

[4] A. S. Ghotkar y G. K. Kharate, «Hand Segmentation Techniques to Hand Gesture Recognition for,» *International Journal of Human Computer Interaction,* vol. 3, nº 1, pp. 15-25, 2012.

[5] G. R. S. Murthy y R. S. Jadon, «A review of vision based hand gestures recognition,» *International Journal of Information Technology and Knowledge Management,* vol. 2, nº 2, pp. 405-410, 2009.

[6] J. P. Wachs, M. Kölsch, H. Stern y Y. Edan, «Vision-based hand-gesture applications,» *Communications of the ACM* , vol. 52, nº 2, pp. 60-71, 2011.

[7] K. Yasuoka y M. Yasuoka, «On the prehistory of QWERTY,» *Zinbun,* vol. 42, pp. 161-174, 2011.

[8] B. Myers, «A Brief History of Human Computer Interaction Technology,» *ACM interactions,* vol. 5, nº 2, pp. 44-54, 1998.

[9] J. Cannan y H. Hu, «Human Machine Interaction(HMI): A Survey,» de *School of Computer Science & Electronic Engineering, University of Essex*, 2010.

[10] F. Ion, «From touch displays to the Surface: A brief history of touchscreen technology,» Arstechnica, 2013 Abril 4. [En línea]. Available: http://arstechnica.com/gadgets/2013/04/from-touch-displays-to-the-surface-a-brief-history-of-touchscreen-technology/3/. [Último acceso: 7 Marzo 2014].

[11] A. Carrillo, J. M. Cejudo, F. Domínguez y E. Rodríguez, «Graphics tablet technology in second year thermal engineering teaching,» *Journal of Technology and Science Education,* vol. 3, nº 3, 2013.

[12] W. Carlson, «The emergence of Computer Graphics technology,» The Ohio State University, 2003. [En línea]. Available: http://design.osu.edu/carlson/history/lesson2.html. [Último acceso: 7 Marzo 2014].

[13] B. Juang y L. Rabiner, «Automatic speech recognition - A brief history of the technology development,» *Elsevier Encyclopedia of Language and Linguistics*, 2005.

[14] C. A. Mark, «Fifty Years of Moore's Law,» *IEEE Transactions on Semiconductor Manufacturing*, vol. 24, nº 2, pp. 202-207, 2011.

[15] L. Motion, «Leap Motion,» [En línea]. Available: https://www.leapmotion.com/. [Último acceso: 7 Marzo 2014].

[16] Flutter, «Flutter,» [En línea]. Available: https://flutterapp.com/. [Último acceso: 7 Marzo 2014].

[17] Microsoft, «Kinect for Windows,» [En línea]. Available: http://www.microsoft.com/en-us/kinectforwindows/. [Último acceso: 7 Marzo 2014].

[18] NeuroTechnology, «NPointer,» [En línea]. Available: http://www.neurotechnology.com/npointer.html. [Último acceso: 7 Marzo 2014].

[19] Crea Software, «Enable Viacam,» [En línea]. Available: http://eviacam.sourceforge.net/index_es.php. [Último acceso: 7 Marzo 2014].

[20] PointGrab, «PointGrab,» [En línea]. Available: http://www.pointgrab.com/. [Último acceso: 7 Marzo 2014].

[21] A. Sears y J. A. Jacko, The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications, Hillsdale, NJ, USA: L. Erlbaum Associates Inc, 2007.

[22] European Agency for Safety and Health at Work , «The human-machine interface as a emerging risk,» European Agency for Safety and Health at Work, 4 Octubre 2009. [En línea]. Available: https://osha.europa.eu/en/publications/literature_reviews/HMI_emerging_risk. [Último acceso: 7 Marzo 2014].

[23] S. Kim y R. Casper, «Applications of Convolution in Image Processing with MATLAB,» [En línea]. Available: http://www.math.washington.edu/~wcasper/math326/projects/sung_kim.pdf. [Último acceso: 7 Marzo 2014].

[24] E. Dougherty y R. Lotufo, Hands-on Morphological Image Processing, SPIE Press , 2003.

[25] J. S. O. Heikkila, «A real-time system for monitoring of cyclists and pedestrians,» de *Second IEEE Workshop on Visual Surveillance*, Fort Collins, Colorado, 1999.

[26] E. Rosten y T. Drummond, «Machine learning for high speed corner detection,» de *9th Euproean Conference on Computer Vision*, Graz, 2006.

[27] R. C. Gonzalez y R. E. Woods, «Representation and Description,» de *Digital Image Processing*, Prentice Hall, 2008, pp. 817-882.

[28] D. Lowe, «Distinctive image features from scale-invariant keypoints,» *International Journal of Computer Vision*, vol. 60, nº 2 , pp. 91-110, 2004.

[29] M. Muja y D. G. Lowe, «Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration,» de *International Conference on Computer Vision Theory and Applications*, Lisboa, 2009.

[30] M. Arulampalam, S. Maskell, N. Gordon y T. Clapp, «A tutorial on particle filters for online nonlinear/non-gaussian Bayesian tracking,» *IEEE Transactions on Signal Processing*, vol. 50, nº 2, pp. 174-188, 2002.

[31] R. Kalman, «A New Approach to Linear Filtering and Prediction,» *Transaction of the ASME—Journal of Basic Engineering*, vol. 82, pp. 35-45, 1960.

[32] B. Stroustrup, The C++ Programming Language, Addison-Wesley, 1997.

[33] OpenCV, «The OpenCV Reference Manual,» 08 11 2011. [En línea]. Available: docs.opencv.org/opencv2refman.pdf. [Último acceso: 7 Marzo 2014].

[34] Digia, «Documentación de Proyecto Qt,» [En línea]. Available: http://qt-project.org/doc/. [Último acceso: 7 Marzo 2014].

[35] J. Lee, Reconocimiento robusto y tiempo real de gestos, Memoria de Ingeniería Civil en Computación. Universidad de Chile, 2010.